2000

Coloring clique hypergraphs

Hoifung Poon
West Virginia University

Coloring Clique Hypergraphs

Hoifung Poon

Thesis submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Computer Science

Elaine Eschen, Ph.D., Chair
Hong-Jian Lai, Ph.D.
Frances L. Van Scoy, Ph.D.

Department of Computer Science

Morgantown, West Virginia
2000

Keywords: clique hypergraph, coloring, extendible, chordal

ABSTRACT

Coloring Clique Hypergraphs

Hoifung Poon

Let $G = (V, E)$ be a simple graph. The *clique hypergraph* of G, denoted as $CH(G)$, has V as its set of vertices, and the maximal cliques as its hyperedges. Let S_k be a set of k colors. A map $c : V \mapsto S_k$ is a *proper k-coloring* for $CH(G)$ if any maximal clique of G with at least two vertices receives at least two distinct colors. Let $W \subset V$, and let $s \geq 1$. We say that G is (W, s)-*extendible* if any assignment on W with at most s colors can be extended to a proper s-coloring of $CH(G)$. We prove that the clique hypergraphs of chordal and comparability graphs are bicolorable and that the clique hypergraphs of circular-arc graphs are 3-colorable. Our main result is the characterization of $(W, 2)$-extendibility for chordal graphs in the case when $|W| = 2$.

Contents

1 Introduction

Let $G = (V, E)$ be a simple graph, that is, G has no loops or multiple edges. The **clique hypergraph** of G, denoted as $CH(G)$, is defined as an ordered pair (V, MC), where V denotes the set of vertices and MC denotes the set of maximal cliques of G. Here a clique is simply a complete graph and a clique is **maximal** if it is not properly contained in any other clique of the graph G. Thus in the clique hypergraph of G, the hyperedges are all the maximal cliques. The **independent-set hypergraph** of G is defined similarly, with hyperedges being the maximal independent sets.

Let S_k denote a set of k colors. A **proper k-coloring of the clique hypergraph of G** is a map $c : V(G) \mapsto S_k$, such that for every maximal clique M in G with at least two vertices, the vertices in M receive at least two distinct colors.

The question of coloring clique hypergraphs was raised by Duffus **et al.** in [4]. In that paper this question was posed together with the question of coloring independent-set hypergraphs. Perfect graphs were mentioned as of special interest, for the complement of a perfect graph is still perfect and so the two problems become one in this case. Recall that a graph is **perfect** if the maximum clique size is equal to the chromatic number for every induced subgraph. Further information on perfect graphs can be found in [7].

In a recent extended abstract [9], J. Kratochvil and Zs. Tuza investigated the complexity aspect of the problem. They proved that it is NP-complete to determine the bicolorability of clique hypergraphs of perfect graphs. They also showed that for planar graphs bicolorability can be decided in polynomial time.

In this paper, we first study this problem for some special graph classes. For example, we prove the bicolorability of clique hypergraphs of **chordal** and **comparability** graphs, both of which are subclasses of perfect graphs. It should be noted that the second result was mentioned but not proved in [4]. We also prove that the clique hypergraph of a circular-arc graph is 3-colorable, which is the best possible.

We spend the major portion of our paper investigating a new aspect of this problem, coloring extension. Coloring extension is a very useful tool for coloring problems. It helps not only in theoretic investigation, but also in designing efficient coloring algorithms. We introduce a general definition for clique hypergraph coloring extensions and investigate this problem for chordal graphs. We prove a coloring extension theorem for chordal graphs. We also look at the algorithmic aspects of the problem and provide efficient algorithms for all the coloring results we prove.

The rest of this paper is organized as follows. The remaining part of this section establishes the notation and terminology that is used throughout this paper. Readers are referred to [1] for undefined graph theoretical terms, and [7] for background knowledge about special graph classes. In Section 2, we give the definitions as well as major properties of some special graph classes, which are needed in the following sections. In Section 3, we prove that the clique hypergraphs of chordal graphs and comparability graphs are bicolorable. We also prove that the clique hypergraphs of circular-arc graphs are 3-colorable. Section 4 is devoted to our main result, an Extension Theorem. There we define a new concept called (W, s)**-extendibility** for clique hypergraph coloring. In Section 5 we provide efficient algorithms for our coloring results as well as our result on the extension problem. We conclude this paper in Section 6 with some open problems.

In this paper, graphs are assumed to be simple and finite, with nonempty vertex set. Let G be a graph, $V(G)$ and $E(G)$ denote respectively the vertex set and the edge set of G. The parameter G may be dropped if free of ambiguity. Let $A \subseteq V(G)$. $G[A]$ denotes the induced subgraph of G generated by the vertices in A. For subsets V_1, V_2 of V, we denote by $[V_1, V_2]$ the set of edges with one end in V_1 and the other in V_2. An edge of G is a cut-edge of G if its deletion increases the number of connected components.

We denote $uv \in E$ as an undirected edge connecting u, v. Vertex u is called a **neighbor** of vertex v, and vice versa, when $uv \in E$. For $v \in V$, $N(v)$ denotes the **open neighborhood** of v, that is, the set of all neighbors of v. On the other hand, $N[v]$ denotes the **closed neighborhood** of v, which consists of all v's neighbors as well as v. If G is equipped with an orientation D, we denote $(x, y) \in D$ as a directed edge from x to y in this orientation. An **in-neighbor** of a vertex v in D is a vertex u such that $(u, v) \in D$; an **out-neighbor** of v is a vertex w such that $(v, w) \in D$. The **in-degree** of v is the number of its in-neighbors, and the **out-degree** of v is the number of its out-neighbors.

2 Preliminaries

In this section we introduce several graph classes and state some related properties that we use in following sections. For more information, see [2, 15, 7]. We first introduce chordal graphs, then comparability graphs. At last we introduce a general model for producing special graph classes, the intersection graphs, from which we derive the definitions of circular-arc graphs and interval graphs.

A graph is **chordal** if it contains no chordless cycle of length greater than three. Let G be a graph. A **simplicial vertex** of G is a vertex whose neighborhood induces a clique. An ordering of the vertices $S = (v_1, \cdots, v_n)$ is called a **perfect elimination scheme** if for every $1 \leq i \leq n$, v_i is a simplicial vertex in $G[v_i, \cdots, v_n]$.

A chordal graph is also called **triangulated**. Fulkerson and Gross characterized chordal graphs by means of perfect elimination scheme as follows:

Proposition 1 (Fulkerson and Gross, [5]) A graph G is chordal if and only if there exists a perfect elimination scheme for G.

Dirac gave the following structural characterization of chordal graphs:

Proposition 2 (Dirac, [3]) Every chordal graph has a simplicial vertex. Moreover, if G is not a clique, then it has two nonadjacent simplicial vertices.

Note that if v is a simplicial vertex of G, then $N[v]$ induces a maximal clique. In this case, $G[N[v]]$ is actually the only maximal clique containing v. If $S = (v_1, \cdots, v_n)$ is a perfect elimination scheme of G, then all maximal cliques of G are of the form $N[v_i] \cap \{v_i, \cdots, v_n\}$ [5]. By the characterization stated in Proposition 1, one can deduce that a chordal graph must be perfect, for example, see [8].

Rose, Tarjan and Lueker [13] used a well-known linear time algorithm, Lexicographic Breadth-First-Search(LexBFS), to efficiently recognize a chordal graph and produce for it a perfect elimination scheme. For more detail of this algorithm, see [13] or [7]. The following notable consequence of Proposition 2 was exploited by the algorithm of Rose, Tarjan and Lueker, and is needed in later sections:

Proposition 3 Let G be a chordal graph and v be an arbitrary vertex in G. Then a perfect elimination scheme can be generated by LexBFS with v being the last vertex in the scheme.

An orientation D of a graph G is **transitive** if for any $a, b, c \in V(G)$, $(a, b) \in D$ and $(b, c) \in D$ imply that $(a, c) \in D$. An undirected graph is called a **comparability graph** if it has a transitive orientation. In a directed graph, a **source** is a vertex whose in-degree is 0, and a **sink** is a vertex whose out-degree is 0. We need the following:

Proposition 4 Let H be a simple graph with a transitive orientation D. Then there must exist a source x in H. Note that by definition, this means that there does not exist any vertex $y \in V(H)$ with $(y, x) \in D(H)$.

Proof: By way of contradiction, suppose that H has no source. Let $p_0 \in V(H)$ be a vertex in H. p_0 is not a source, thus there must exist some $p_1 \in V(H)$, such that $(p_1, p_0) \in D$. Now that p_1 is not a source in H, thus there again exists some $p_2 \in V(H)$, such that $(p_2, p_1) \in D$. This process can continue by induction and we can obtain a walk $p_0 p_1 p_2 \cdots$ where $(p_{i+1}, p_i) \in D$ for any i. Since the graph is finite, some vertex must repeat in the walk and so there must exist a directed cycle $p_{j+1}(= p_i)p_j \cdots p_i$. By the transitivity of the orientation, we must have $(p_i, p_i) \in D$, hence H has a loop, contrary to the assumption that H is simple. $\square$

In [7] it is shown that comparability graphs are perfect and can be recognized in polynomial time. Recently Spinrad and McConnell discovered a linear time algorithm for recognition, see [15].

Let $\mathcal{F}$ be a family of nonempty sets. The **intersection graph** of $\mathcal{F}$ is obtained by representing each set in $\mathcal{F}$ by a distinct vertex and connecting two vertices by an edge if and only if their corresponding sets intersect. A graph is called a **circular-arc graph** if it is the intersection graph of a family of arcs on a circle. Such a family of arcs comprise a **circular-arc model** for the original graph. Another example of intersection graphs is the **interval graph**, which is the intersection graph of a family of intervals on a line. Again, these intervals form an **interval model** for the given graph. Gilmore and Hoffman [6] established the following characterization for interval graphs:

Proposition 5 A graph G is an interval graph if and only if G is chordal and its complement $\bar{G}$ is a comparability graph.

From the circular-arc model one can deduce the following well-known fact:

Proposition 6 Let G be a circular-arc graph and v be a vertex in G. Then $G - N[v]$ is an interval graph.

Proof: By definition we have a circular-arc model for G, in which we denote C as the circle and $\mathcal{A}$ as the set of arcs corresponding to the vertices of G. Let $A \in \mathcal{A}$ be the arc associated with v. Then $C - A$ is a curve which is homeomorphic to a line in topological sense. Also note that any arc associated with a vertex in $G - N[v]$ must not intersect A, thus it must be an arc contained in $C - A$. Hence $C - A$ and all arcs associated with vertices in $G - N[v]$ comprise an interval model for $G - N[v]$ and so $G - N[v]$ is an interval graph. $\square$

3 Some results on coloring clique hypergraphs

Theorem 1 The clique hypergraph of a chordal graph is bicolorable.

Proof: Let G be a chordal graph. We use induction on $|V(G)|$. If $|V(G)| \leq 2$, the conclusion is obvious. Assume that the conclusion holds for all chordal graphs G with $|V(G)| \leq k$ where $k \geq 2$.

Let G be a chordal graph with $|V(G)| = k + 1$. By Proposition 2 there exists a simplicial vertex v in G.

Denote $G' = G - v$. Then G' is still chordal and $|V(G')| = k$. So by the inductive hypothesis, $CH(G')$ is bicolorable. Hence there exists a proper 2-coloring $c : V(G') \mapsto \{1, 2\}$ for $CH(G')$.

If v is an isolated vertex, c can be extended to G by assigning to v an arbitrary color.

Otherwise $N(v) \neq \emptyset$ and so there exists $w \in V(G')$ such that $vw \in E(G)$.

Denote $d \in \{1, 2\}$ as the other color different from $c(w)$. We extend c to G by defining $c(v) = d$. We show that this is a proper 2-coloring for $CH(G)$.

Let M be a maximal clique in G of cardinality at least 2. If M is contained in G', then M is also maximal in G'. Thus the vertices in M must receive at least two different colors by the choice of c. Otherwise v is in M, and so $M = G[N[v]]$ since v is simplicial. By our way of extending c, M must receive two different colors, $c(v)$ and $c(w)$. This proves that c so extended is a proper bicoloring of $CH(G)$.

Thus the proposition is still true when $|V(G)| = k+1$. By induction, the hypergraph of a chordal graph is bicolorable. □

Note that K_2 is a chordal graph and a proper coloring for its clique hypergraph must color the vertices differently. Thus this result is the best possible.

Theorem 2 The clique hypergraph of a comparability graph is bicolorable.
Proof: Let G be a comparability graph. By definition G has a transitive orientation D. We color all the sources with 1, and color the other vertices with 2. We show that this is a proper bicoloring for the clique hypergraph of G.

Let K be a maximal clique of G. We show that K has exactly one source of G. Note that every two vertices in K are adjacent, so K can contain at most one source. Hence it suffices to prove that K must contain a source of G.

Now that $K = G[V(K)]$ is an induced subgraph of G, so K is also a comparability graph, for the restriction of D to K is still a transitive orientation. By Proposition 4, K has a source x. We claim that x is also a source in G.

Let $y \in V(K)$ with $y \neq x$. Since K is a clique, x must be adjacent to y. Hence in the orientation D, there must exist a directed edge between x and y. Now x is a source in K, thus $(y, x) \notin D$. Therefore $(x, y) \in D$, that is, there exists a directed edge from x to any other vertex in K.

Suppose that x is not a source of G. Then there must exist some $z \in V(G)$, such that $(z,x) \in D$. Hence $z \notin V(K)$. Also by transitivity we have $(z,y) \in D$ for any $y \in V(K)$ with $y \neq x$. Thus z is adjacent to every vertex in K and so $K \cup \{z\}$ is a clique with one more vertex than K. This is contrary to the maximality of K. Hence we prove that K must contain exactly one source of G. $\square$

Note that K_2 is a comparability graph and a proper coloring for its clique hypergraph must color the vertices differently. Thus this result is best possible.

Theorem 3 The clique hypergraph of a circular-arc graph is 3-colorable.

Proof: Let G be a circular-arc graph. Let $v \in V(G)$. Consider $G' = G - N[v]$. By Proposition 6, G' is an interval graph. Thus by Proposition 5, G' is chordal. By Theorem 1, $CH(G')$ is 2-colorable. Let $c : V(G') \mapsto \{1,2\}$ be such a proper 2-coloring for the clique hypergraph of G'. We extend it to G by defining $c(v) = 1$, and $c(w) = 3$ for any $w \in N(v)$.

We show that this is a proper 3-coloring of $CH(G)$. Let M be a maximal clique in G with $|V(M)| \geq 2$.

If $v \in V(M)$, then $M \subseteq G[N[v]]$. So M receives colors 1 and 3 since $|V(M)| \geq 2$.

Otherwise $v \notin V(M)$. Then $V(M) \cap V(G') \neq \emptyset$ by the maximality of M. If $V(M) \subseteq V(G')$, M is also a maximal clique in G' and so M receives two distinct colors 1 and 2. Otherwise $V(M) \cap N(v) \neq \emptyset$, in which case $c(V(M)) \cap \{1,2\} \neq \emptyset$ and $3 \in c(V(M))$. So M receives at least two different colors.

This proves that the given coloring is a proper 3-coloring for the clique hypergraph of G. $\square$

This result is best possible. We will use the term **cycle** to mean a chordless cycle. We show that any odd cycle C_{2k+1} with $k \geq 2$ is a circular-arc graph and the chromatic number of its clique hypergraph is 3.

First note that a cycle is a circular-arc graph. Also note that if a cycle has more than 3 vertices, then each of its edges is a maximal clique and vice versa. Hence for a cycle with 4 or more vertices, the proper coloring for its clique hypergraph coincides with a proper coloring for the cycle itself.

It is a well-known fact that the chromatic number of an odd cycle is 3. Therefore we conclude that the chromatic number for the clique hypergraph of an odd cycle with more than 3 vertices must be 3. This indicates that the bound in Theorem 3 is the best possible.

4 An Extension Theorem

For coloring problems it is often quite helpful to consider coloring extensions. In general, an extension problem assumes the existence of a partial coloring and ponders whether it can be extended to a full coloring. This type of consideration is useful for inductive arguments. For example, if G is a graph and G_1, G_2 are subgraphs of G such that $G = G_1 \cup G_2$ and $W = V(G_1) \cap V(G_2)$. If we know that any partial coloring on W with certain constraints can be extended to one of the G_i's, then the coloring problem on G may be reduced to that of a proper subgraph of G and we might be able to use induction.

The above consideration leads to our investigation on the extension problem with respect to clique hypergraph coloring. We introduce the following very general definition.

Definition Let G be a graph, $W \subseteq V(G)$, and let $s \geq 1$. We say that G is (W, s)**-extendible** if any assignment on the vertices in W with at most

s distinct colors can be extended to a proper s-coloring of the clique hypergraph of G.

There are three parameters in this definition, G, W and s. In further investigations, one might restrict G to be in a special graph class, or one might pose certain constraints on W, for example, cardinality bound or structural requirement.

In this section we look at the special case when G is chordal and $|W| = 2$. Note that $|W|$ must be not less than 2 for the problem to be nontrivial. If $|W| = 1$, G is (W, s)-extendible if and only if $CH(G)$ is s-colorable. Also note that, by Theorem 1, the clique hypergraph of a chordal graph is bicolorable. So naturally we would set $s = 2$. Therefore we consider the following problem:

Is a chordal graph G $(\{a, b\}, 2)$**-extendible for distinct** $a, b \in V(G)$ **?**

Our investigation leads to the discovery of certain kind of "forbidden structure", which can be formulated into the following definitions.

Definition A **forbidden path** of G is a path in G in which each edge is a maximal clique of G.

Definition Let G be a graph and $a, b \in V(G)$ be two distinct vertices of G. The graph G is (a, b)**-path essential** if there exists a path $P = ap_1p_2 \cdots p_kb$ in G, and if $V(G)$ can be partitioned into $k + 2$ mutually disjoint sets $A_0, A_1, \cdots, A_{k+1}$, so that $a \in A_0, b \in A_{k+1}, p_i \in A_i$, and

$$E(G) = E(P) \cup \bigcup_{0}^{k+1} E(G[A_i])$$

Before we state and prove our Extension Theorem, let us first derive some observations and lemmas.

Observation 1 If G has a forbidden (a,b)-path, then G is not $(\{a,b\},2)$-extendible.

Proof: Let $P = p_0 \cdots p_{k+1}$ be a forbidden (a,b)-path with $p_0 = a, p_{k+1} = b$. Thus $p_i p_{i+1}$ is a maximal clique of G. Let c be a proper 2-coloring of the clique hypergraph of G. Then by definition, $c(p_i) \neq c(p_{i+1})$. Hence c must color the vertices in P alternatively with 1 and 2. Thus the color of a is associated with the color of b by the parity of the path length. That is, if the path is of odd length, then a, b must be assigned with different colors; if the path is of even length, then a, b must be assigned with the same color. Therefore there must exist some coloring assignment on $\{a,b\}$ that can not be extended to a proper 2-coloring for graph G. Hence G is not $(\{a,b\},2)$-extendible. $\square$

Observation 2 If G is (a,b)-path essential and if P is an (a,b)-path as in the definition, then each of the following holds:

(i) Each edge in P is a cut-edge of G.
(ii) P is the unique (a,b)-path in G.
(iii) P is a forbidden path of G.
(iv) For any $v \notin V(P)$, $|N(v) \cap V(P)| \leq 1$.

Proof: Denote $P = p_0 \cdots p_{k+1}$ with $a = p_0, b = p_{k+1}$, and let the A_i's denote the vertex partitions, as in the definition of (a,b)-path essential.

Let $0 \leq j \leq k$. Denote $V_1 = \bigcup_0^j A_i$ and $V_2 = \bigcup_{j+1}^{k+1} A_i$. Then V_1, V_2 form a partition of V and by definition $[V_1, V_2] = \{p_j p_{j+1}\}$. Hence $p_j p_{j+1}$ is a cut-edge of G. This proves (i).

Now for any j the deletion of $p_j p_{j+1}$ separates a, b into different components. Thus any (a,b)-path must include all the edges in P. But P is already an (a,b)-path. Therefore P is the unique (a,b)-path in G. This proves (ii).

If for some j, p_j and p_{j+1} have a common neighbor q, then p_j and p_{j+1} are in a 3-cycle $qp_jp_{j+1}q$ and so p_jp_{j+1} is not a cut-edge of G, contrary to (i). Therefore p_j and p_{j+1} have no common neighbor and hence p_jp_{j+1} must be a maximal clique of G. This proves (iii).

Now consider (iv). Assume that $v \in A_j$ for some j. Since $v \notin V(P)$, any edge e incident with v must not be in P. Thus e must be in $G[A_j]$. Since $A_j \cap V(P) = \{p_j\}$, so $N(v) \cap V(P) \subseteq \{p_j\}$. Hence $|N(v) \cap V(P)| \leq 1$. This finishes the proof. □

Observation 3 If G is (a, b)-path essential, then G is not $(\{a, b\}, 2)$-extendible.

Proof: This is clear by Observation 1 and Observation 2. □

Observation 4 Let G be a connected graph and $P = p_0 \cdots p_{k+1}$ be a path in G. If each component of $G - E(P)$ contains exactly one vertex in $V(P)$, then G is (p_0, p_{k+1})-path essential.

Proof: Denote $G' = G - E(P)$. By the assumption of the proposition, there exists a one-to-one and onto mapping between the set of components of G' and $V(P)$. Let H_i be the component of G' that contains p_i. Denote $A_i = V(H_i)$. Then $p_i \in A_i$. Since H_i's are components of G', $V(G') = V(G)$, so A_i's are mutually disjoint and they form a partition of $V(G)$. There exists no edge between different components in G', hence $E(G') = \bigcup_0^{k+1} E(H_i) = \bigcup_0^{k+1} E(G[A_i])$ and so

$$E(G) = E(P) \cup E(G') = E(P) \cup \bigcup_{0}^{k+1} E(G[A_i])$$

Therefore G is (p_0, p_{k+1})-path essential. □

From now on, we always use the numbers 1 and 2 as the two colors. Note that if $t \in \{1, 2\}$, then $3 - t$ is exactly the other color in $\{1, 2\}$ different from t. Prior to continuing, we state some assumptions and notation which we use throughout the following lemmas without mentioning them explicitly.

In the following lemmas, we always assume that G is chordal. We denote a, b as two distinct vertices in G. By Proposition 1 in Section 2, there exists a perfect elimination scheme $S = (v_1, \cdots, v_n)$ for G.

Denote $S_R(v)$ as all the vertices in G that follow v in S and denote $N_R(v)$ as all the neighbors of v that follow v in S. We call a neighbor of v that follows v as a **right neighbor** of v. A **left neighbor** of v is defined similarly. Denote $S_R[v] = S_R(v) \cup \{v\}$ and $N_R[v] = N_R(v) \cup \{v\}$.

We are going to prove an extension theorem. This theorem completely characterizes the $(W, 2)$-extendibility for chordal graph G with $|W| = 2$. Our investigation started by pondering the number of right neighbors of a in scheme S. We came to the conclusion that it must be 1 if G is not $(\{a, b\}, 2)$-extendible. Then we pondered the same question for this only right neighbor of a and so on. The consideration leads to Lemmas 2—4. We then assumed the existence of that (a, b)-path and thought about the possibility for G to still be extendible. This gives rise to Lemmas 5—7. Up to this point we have obtained a pretty detailed picture of the possible structure when G is not extendible. Further investigation leads to Lemmas 8—9 as well as our Extension Theorem, which neatly characterizes the situation and implies an efficient algorithm for recognition of $(\{a, b\}, 2)$-extendibility.

Lemma 1 Let M be a maximal clique in G and v be the leftmost vertex of M in the scheme S, then $M = G[N_R[v]]$.

Proof: Since v is the leftmost one among all vertices of M, $V(M) \subseteq S_R[v]$. Note that M is a clique and so v is adjacent to all the other vertices in M, hence we have $V(M) \subseteq S_R[v] \cap N[v] = N_R[v]$. On the other hand, S is a perfect elimination scheme and so $G[N_R[v]]$ is a clique. Thus we have $M = G[N_R[v]]$ by the maximality of M. □

Now in Lemmas 2—7, we assume that the perfect elimination scheme S is chosen so that b is the last vertex in S, that is, $v_n = b$. We do not state this explicitly in Lemmas 2—7. And we observe that this is feasible by Proposition 3 in Section 2.

Lemma 2 Let $p_0 \cdots p_l$ be a path for some $l \geq 0$, where $p_0 = a$, $p_s \neq b$ for $s = 1, 2, \cdots, l$, and such that $N_R(p_i) = \{p_{i+1}\}$ for $i < l$. If $|N_R(p_l)| \geq 2$, G is $(\{a, b\}, 2)$-extendible.

Proof: Let w_1, w_2 be two distinguished distinct right neighbors of p_l in the scheme. We may assume that w_2 follows w_1. Suppose that we have assigned colors $d_1, d_2 \in \{1, 2\}$ to a and b, respectively. Define:

$$d = \begin{cases} d_1 & \text{if } l \text{ is even;} \\ 3 - d_1 & \text{if } l \text{ is odd.} \end{cases}$$

Now consider the following algorithm:

```
1.  Color v_n(= b) with color d_2;
2.  for i := n - 1 downto 1 do
        case v_i ≠ p_l, w_1 :
            if v_i has no right neighbors
               then Color v_i with an arbitrary color;
               else Color v_i differently from one of its right
               neighbors;
        case v_i = w_1 :
            Color w_1 differently from w_2;
        case v_i = p_l :
            Color p_l with d
 end.
```

We show that this yields a proper 2-coloring for $CH(G)$. First note that the coloring so generated has a property that, for any vertex v, either $N_R(v) = \emptyset$ or there exists some $v' \in N_R(v)$ such that v and v' receive different colors. This is obviously true for all vertices other than p_l. For p_l, note that both w_1 and w_2 are its neighbors and they receive different colors. So whatever the color p_l receives, this color differs from either that of w_1 or that of w_2. Also note that $N_R(p_i) = \{p_{i+1}\}$. Hence the algorithm assigns colors to $p_l, p_{l-1}, \cdots, p_0$ alternating between 1 and 2. Thus by our choice of d, which is assigned to p_l, the color assigned to $a(= p_0)$ is d_1.

Let M be a maximal clique in G. Let $v \in V(M)$ be its leftmost vertex in the scheme. Then $M = G[N_R[v]]$ by Lemma 1. If M has more than one vertex, then $N_R(v) \neq \emptyset$, so there exists $v' \in N_R(v)$ such that v and v' receive different colors. Note that v' is also in M. Thus the proof is finished. □

Lemma 3 Let $p_0 \cdots p_l$ for some $l \geq 0$ be a path in G, where $p_0 = a$, $p_s \neq b$ for $s = 1, 2, \cdots, l$, and such that $N_R(p_i) = \{p_{i+1}\}$ for $i < l$. If $N_R(p_l) = \emptyset$, G is $(\{a, b\}, 2)$-extendible.

Proof: Suppose that we have preassigned colors $d_1, d_2 \in \{1, 2\}$ to a and b, respectively. Define

$$d = \begin{cases} d_1 & \text{if } l \text{ is even;} \\ 3 - d_1 & \text{if } l \text{ is odd.} \end{cases}$$

Consider the following algorithm.

```
1.  Color v_n(= b) with color d_2;
2.  for i := n − 1 to 1 do
        if v_i = p_l
           then Color p_l with d;
           else
               if v_i has no right neighbors
                  then Color v_i with an arbitrary color ;
                  else Color v_i differently from one of its right
                  neighbors.
  end.
```

Now we show that this yields a proper 2-coloring for $CH(G)$. First note that $N_R(p_l) = \emptyset$. Thus the coloring so generated has a property that, for any vertex v, either $N_R(v) = \emptyset$ or there exists some $v' \in N_R(v)$ such that v and v' receive different colors. Also note that $N_R(p_i) = \{p_{i+1}\}$ and so the algorithm assign colors to $p_l, p_{l-1}, \cdots, p_0$ alternatively with 1 and 2. Thus by our choice of d, which is assigned to p_l, the color assigned to $a(= p_0)$ is d_1.

Let M be a maximal clique in G. Let $v \in V(M)$ be its leftmost vertex in the scheme. Then $M = G[N_R[v]]$ by Lemma 1. If M has more than one vertex, $N_R(v) \neq \emptyset$, so there exists $v' \in N_R(v)$ such that v and v' receive different colors. Note that v' is also in M. Thus the proof is finished. $\square$

Lemma 4 If G is not $(\{a, b\}, 2)$-extendible, then there exists an (a, b)-path $P = p_0 p_1 \cdots p_{k+1}$ with $p_0 = a, p_{k+1} = b$, and such that $N_R(p_i) = \{p_{i+1}\}$ for $i = 0, 1, \cdots, k$.

Proof: Note that p_0 is a path and $p_0 = a$, $p_0 \neq b$. By Lemma 2 and Lemma 3, $|N_R(p_0)| = 1$. Assume that p_1 is that only right neighbor of p_0.

If $p_1 = b$, we are done. Otherwise by Lemma 2 and Lemma 3, again we have $|N_R(p_1)| = 1$. Assume that p_2 is that only right neighbor of p_1.

By induction, we obtain a growing path $p_0 p_1 \cdots$ with $N_R(p_i) = \{p_{i+1}\}$. Since the graph is finite, this path must end by reaching b. Thus it is exactly the path asserted by the proposition. This finishes the proof. $\square$

Lemma 5 Let $P = p_0 p_1 p_2 \cdots p_{k+1}$ be a path in G with $N_R(p_i) = \{p_{i+1}\}$ for $i = 0, 1, \cdots, k$. Then for any vertex $v \notin V(P)$, exactly one of the following must hold:

(i) $|N_R(v) \cap V(P)| \leq 1$

(ii) v is a left neighbor of exactly two consecutive vertices in P.

Proof: Suppose that $|N_R(v) \cap V(P)| \geq 2$. Then there exist $0 \leq s < t \leq k+1$ with $p_s, p_t \in N_R(v)$. We may assume that s, t are chosen so that s is the minimum and t is the maximum. Now p_s, p_t are right neighbors of v in a perfect elimination scheme. Hence p_s and p_t must be adjacent in G. Since $N_R(p_i) = \{p_{i+1}\}$ for any i, by induction, p_t follows p_s. Thus p_t is a right neighbor of p_s. But p_s has only one right neighbor p_{s+1}, so $p_t = p_{s+1}$. Therefore $N_R(v) = \{p_s, p_{s+1}\}$ by the extremality of s, t. This finishes the proof. □

Lemma 6 Let $P = p_0p_1p_2 \cdots p_{k+1}$ be a path in G with $p_0 = a, p_{k+1} = b$ and $N_R(p_i) = \{p_{i+1}\}$ for $i = 0, 1, \cdots, k$. Assume that there exists a vertex $u \in V(G) - V(P)$ such that $p_l, p_{l+1} \in N_R(u)$ for some $0 \le l \le k$. Then G is $(\{a, b\}, 2)$-extendible.

Proof: Fix an l as in the assumption of the proposition. Define:

$$d = \begin{cases} d_1 & \text{if } l \text{ is even;} \\ 3 - d_1 & \text{if } l \text{ is odd.} \end{cases}$$

Consider the following algorithm.

```
1.  Color v_n(= b) with color d_2;
2.  for i := n − 1 to 1 do
        case v_i ∉ V(P) :
            if v_i has no right neighbors
              then Color v_i with an arbitrary color;
              else Color v_i differently from one of its right
              neighbors.
        case v_i = p_m, m ≠ l, m ≤ k :
            Color v_i differently from p_{m+1};
        case v_i = p_l :
            color p_l with d;
  end.
```

We show that this does produce a proper coloring for $CH(G)$. First note that this algorithm has a property that, for any vertex $v \neq p_l$, either $N_R(v) =$

$\emptyset$ or there exists some $v' \in N_R(v)$ such that v and v' receive different colors. Also note that the algorithm assigns colors to $p_l, p_{l-1}, \cdots, p_0$ alternatively with 1 and 2. Thus by our choice of d, which is assigned to p_l, the color assigned to $a(= p_0)$ is d_1.

Let M be a maximal clique in G and $v \in V(M)$ be the leftmost vertex in the scheme. Then $M = G[N_R[v]]$.

Note that u is adjacent to both p_l and p_{l+1}, thus $up_lp_{l+1}u$ is a clique that contains $p_lp_{l+1} = G[N_R[p_l]]$, thus $G[N_R[p_l]]$ is not a maximal clique. This indicates that $v \neq p_l$.

Now if M has more than one vertex, then $N_R(v) \neq \emptyset$, so there exists $v' \in N_R(v)$ such that v and v' receive different colors. Note that v' is also in M. This finishes the proof. □

Lemma 7 If G is not $(\{a, b\}, 2)$-extendible, then there exists an (a, b)-path $P = p_0p_1 \cdots p_{k+1}$ with $p_0 = a, p_{k+1} = b$, and such that $N_R(p_i) = \{p_{i+1}\}$ for $i = 0, 1, \cdots, k$. Furthermore, for any vertex $v \notin V(P)$, $|N(v) \cap V(P)| \leq 1$.

Proof: The first conclusion is clear by Lemma 4. Let v be a vertex so that $v \notin V(P)$. Since we have chosen $p_{k+1} = b$ to be the last vertex in the scheme, $N_R(p_{k+1}) = \emptyset$. Thus $v \notin N_R(p_i)$ for $0 \leq i \leq k+1$. Hence $N(v) \cap V(P) = N_R(v) \cap V(P)$. Now that G is not $(\{a, b\}, 2)$-extendible, by Lemma 5 and Lemma 6, $|N_R(v) \cap V(P)| \leq 1$, and so $|N(v) \cap V(P)| \leq 1$. □

Lemma 8 Suppose that G is not $(\{a, b\}, 2)$-extendible. Then each of the following holds:

(i) a, b must be in the same component of G, say C.

(ii) G is (a, b)-path essential if and only if C is (a, b)-path essential.

Proof: Note that each component of G is still chordal and can be considered independently as far as coloring is concerned.

If a, b are in different components of G, we can assign colors to a, b in advance and then bicolor each component properly. Hence G must be

$(\{a, b\}, 2)$-extendible, contrary to our assumption. So a, b must be in the same component of G. This proves (i).

If G is (a, b)-path essential, then there exists a path $P = ap_1p_2 \cdots p_kb$ in G together with a partition of $V(G)$, $A_0, \cdots, A_{k+1}$, as in the definition. Now P must be contained in C since C is a component. Let $B_i = A_i \cap V(C)$. Then B_i's form a partition of $V(C)$. Note that C is a component, thus C is also an induced subgraph of G. Hence we have

$$E(C) = E(P) \cup \bigcup_{0}^{k+1} E(G[B_i]) = E(P) \cup \bigcup_{0}^{k+1} E(C[B_i])$$

Therefore C is (a, b)-path essential.

The other way around is similar. Assume that C is (a, b)-path essential and we have a partition of $V(C)$, B_i's, as in the definition. Then we let $A_0 = B_0 \cup V(G - C)$, and let $A_i = B_i$ for $i \geq 1$. This is a partition of $V(G)$. Comparing with the definition, it is easy to observe that G is also (a, b)-path essential. □

The following Lemma is needed in our proof and it may also be useful in future research work. Note that for this Lemma, we do not place any restriction on the perfect elimination scheme S as in Lemmas 2—7. In particular, p_{k+1} need not be the last one in S.

Lemma 9 Let G be a connected chordal graph and S be an arbitrary perfect elimination scheme of G. Let $P = p_0p_1 \cdots p_{k+1}$ be a path in G, such that in S we have $N_R(p_i) = \{p_{i+1}\}$ for $i = 0, 1, \cdots, k$. Assume that for any vertex $v \notin V(P)$, $|N(v) \cap V(P)| \leq 1$. Then each of the following holds:

(i) Each component of $G - E(P)$ contains exactly one vertex in P.

(ii) G is (p_0, p_{k+1})-path essential.

Proof: In view of Observation 4, it suffices to show (i). Let $G' = G - E(P)$.

By the assumption of the proposition, $N_R(p_i) = \{p_{i+1}\}$ for $0 \leq i \leq k$. So $G[V(P)] = E(P)$. Now by definition of G', $E(G') \cap E(P) = \emptyset$. Thus

$G'[V(P)]$ must be an independent set.

Denote H as a component of G'. Since G is connected, H must contain at least one vertex in P. Denote $m = min\{0 \leq i \leq k+1 : p_i \in V(H) \cap V(P)\}$.

If $m = k+1$, then $V(P) \cap V(H) = \{p_m\}$ by the choice of m. Hence in this case, H contains exactly one vertex in P, namely, p_m.

Otherwise $m \leq k$. We show that for any $x \in V(H)$, either $x \notin V(P)$ or $x = p_m$.

Note that H is a component, hence the distance between any two vertices in H is finite. We use induction on $d = d_H(x, p_m)$ and prove in addition that, if $xy \in E(H)$ and $d_H(x, p_m) = d_H(y, p_m) + 1$, then $y \in N_R(x)$, that is, x precedes y in S. Note that this implies that for any $x \in V(H)$, either $x = p_m$ or x precedes p_m in S.

If $d = 0$, $v = p_m$, the conclusion is obvious.

If $d = 1$, $x \in N_H(p_m) \subset N(p_m)$. Note that by a previous argument, in the subgraph H, p_m is not adjacent to any other vertex in P. Therefore $x \notin V(P)$. Since $m \leq k$, $N_R(p_m) = \{p_{m+1}\}$. But $x \neq p_{m+1}$, so $x \notin N_R(p_m)$. Thus $p_m \in N_R(x)$ since x and p_m are adjacent and so x precedes p_m in S. Now if $xy \in E(H)$ and $d_H(x, p_m) = d_H(y, p_m) + 1$. Then $d_H(y, p_m) = 0$ and so $y = p_m$, thus $y \in N_R(x)$.

Now suppose that the conclusion holds for $d = l \geq 1$ and we proceed to the inductive step by assuming that $x \in V(H)$ is such that $d_H(x, p_m) = l+1$.

Let $y \in N_H(x)$ be such that $d_H(y, p_m) = l \geq 1$. Let $z \in N_H(y)$ with $d_H(z, p_m) = l - 1$. The existence of y and z is clear since $l \geq 1$.

By the inductive hypothesis, $y \notin V(P)$ and $z \in N_R(y)$. Also y precedes p_m for $l \geq 1 > 0$. Since $d_H(x, p_m) = l + 1 > d_H(z, p_m) + 1$, $xz \notin E(H)$.

If y precedes x in the scheme, both x and z are its right neighbors and so $xz \in E(G)$. But $xz \notin E(H)$, hence the only possibility is that $xz \in E(P)$ and so $x, z \in V(P)$. Note that $y \notin V(P)$ and now y is adjacent to both x and z, we reach a contradiction to the assumption that $|N(y) \cap V(P)| \leq 1$.

Therefore x precedes y in the scheme and so $y \in N_R(x)$. Now x precedes

y and y precedes p_m, so x precedes p_m.

If $x \in V(P)$, then $x = p_j$ for some $j < m$. But $x \in H$. This is contrary to the minimality of m. Thus $x \notin V(P)$.

Hence the conclusion still holds for $d = l + 1$. By induction this proves that each component H in G' contains exactly one vertex in P. So the proof is finished. □

Remark: In general, if G is not connected, (i) may not hold. But (ii) still holds given the same assumptions except for the connectivity of G.

To see this, note that p_0, p_{k+1} are connected by a path P, so they must be in the same component of G, say C. Note that C is an induced subgraph of G, so the restriction of S on $V(C)$ is a perfect elimination scheme for C. Now all assumptions in Lemma 9 holds for C, including the connectivity since C is a component. Hence C must be (p_0, p_{k+1})-path essential. By Lemma 8, G is also (p_0, p_{k+1})-path essential. This concludes our remark.

Now we are in the position to state and prove our Extension Theorem.

Extension Theorem Let G be a simple chordal graph and $a, b \in V(G)$ be two distinct vertices. Then G is $(\{a, b\}, 2)$-extendible if and only if G is not (a, b)-path essential.

Proof: We prove the contrapositive statement, that G is (a, b)-path essential if and only if G is not $(\{a, b\}, 2)$-extendible.

By Observation 3, it suffices to show that if G is not $(\{a, b\}, 2)$-extendible, then G is (a, b)-path essential. In view of Lemma 8, we may assume that G is connected.

By Lemma 7, there exists a path $P = p_0 \cdots p_{k+1}$ with $p_0 = a, p_{k+1} = b$ and $N_R(p_i) = \{p_{i+1}\}$ for $0 \leq i \leq k$, and such that for any vertex $v \notin V(P)$, $|N(v) \cap V(P)| \leq 1$. Hence by Lemma 9, G is (a, b)-path essential. This finishes the proof of the Extension Theorem. □

The following fact may come in handy in future investigations.

Corollary Let G be a simple 2-edge-connected chordal graph. Then G is $(W, 2)$-extendible for any subset $W \subseteq V(G)$ of cardinality 2.

Proof: Denote $W = \{a, b\}$. Then $a, b \in V(G)$ are two distinct vertices of G. If G is not $(W, 2)$-extendible, G must be (a, b)-path essential by the Extension Theorem. Thus by Observation 2, there is a unique (a, b)-path and any edge in this path is a cut-edge of G. This is contrary to our assumption that G is 2-edge-connected. Hence by contradiction we show that G must be $(W, 2)$-extendible. □

5 Algorithms

From now on, for a given graph, we denote n as the number of vertices and denote m as the number of edges. We use pseudo-Pascal code to describe our algorithms. In particular $!=$ denotes inequality and $==$ denotes equality.

Algorithm 1: Bicolor the clique hypergraph of a chordal graph.

Let G be a chordal graph. Let $(v_1, \cdots, v_n)$ be a perfect elimination scheme for G. We color G as follows:

```
1.  Color v_n with 1.
2.  for i := n-1 to 1 do
        Search v_i's adjacency list for the first vertex v_m
```

```
such that m > i;
        if no such vertex is found
           then color v_i with an arbitrary color;
           else color v_i differently from v_m.
  end.
```

Complexity: This algorithm requires $O(n+m)$ time since for each vertex we check at most as many vertices as its neighbors.

Proof of correctness: We show that the resulting coloring is a 2-coloring for the clique hypergraph of G.

Let K be a maximal clique. Let v_k be a vertex in K such that k is minimum. Then $K = G[A]$ by Lemma 1. Now if $|V(K)| > 1$ then there exists some $v_m \in N(v_k)$ such that $m > k$ since we choose k to be the minimum in K. Hence by the algorithm, v_k is colored differently from one of such v_m. Note that such v_m is also in K. This concludes the proof. □

Algorithm 2: Bicolor the clique hypergraph of a comparability graph.

Let G be a comparability graph. We color it as follows:

```
1.  Find a transitive orientation for G.
2.  Find all the sources in G and color them with 1.
3.  Color the rest of the vertices with 2.
```

Complexity: Suppose that an orientation is given and each vertex has an in-list and an out-list, which contain respectively its in-neighbors and out-neighbors. Note that the in-lists can easily be computed in $O(n+m)$ time from the out-lists and vice versa. Then we can count for each vertex its in-degree and the total time is $O(n+m)$. Thus we can find all the sources in linear time by noting that a source is just a vertex with in-degree 0. Hence step 2 is linear once step 1 is finished. Step 3 is obviously linear. Hence step 1 is the bottleneck step for this algorithm.

J. Spinrad and R. McConnell give an $O(n+m)$ time algorithm for finding the transitive orientation [10], although it is quite complex. This task can also be done easily in $O(n^2)$ or $O(n + m \log n)$ time. For more detail, see [15, 11, 12, 14].

Correctness Proof: The correctness of this algorithm is clear from the proof of Theorem 2, where we proved that the clique hypergraph of a comparability graph is bicolorable. □

Algorithm 3: Color the clique hypergraph of a circular-arc graph with at most 3 colors.

Let G be a circular-arc graph. We use the numbers 1,2,3 as the three colors. We color G as follows. One thing to note is that this algorithm does not need to assume a given circular-arc model.

1. `Pick an arbitrary vertex` $v \in V(G)$;
2. `Obtain a new graph` $G' = G - N[v]$;
3. `Color` G' `with colors 1 and 2;`
4. `Color` v `with 1;`
5. `Color the neighbors of` v `with 3.`

Complexity: We give detail for step 3. Since G is a circular-arc graph, G' must be an interval graph and hence must be chordal. Thus the clique hypergraph of G' is bicolorable by Theorem 1. By Algorithm 1, such a proper bicoloring can be obtained in linear time. Therefore step 3 can be done in linear time. It is clear that the other steps are also linear. Hence the complexity of this algorithm is $O(n+m)$.

Correctness Proof: The correctness of this algorithm is clear from the proof of Theorem 3. □

Algorithm 4: Let G be a chordal graph and $a, b \in V(G)$ be two distinct vertices in G. Determine whether G is $(\{a,b\},2)$-extendible.

Let $v \in V(G)$, and denote the degree of v in G by $d(v)$.

```
1.  Find a perfect elimination scheme for G in which b is
the last vertex;
2.  Relabel the vertices of G by this scheme;
(Thus i denotes the vertex at the
i-th place in the scheme.)
3.  Sort the adjacency lists by vertex labels;
4.  p[0] := a,  i := 0;
5.  repeat
          case p[i] has no right neighbors :
               Halt and return("Extendible!");
          case p[i] has two or more right neighbors :
               Halt and return("Extendible!");
          otherwise :
               p[i+1] :=  the only right neighbor of p[i];
               i := i + 1;
 until p[i] = b;
6.  k := i − 1;
7.  for i := 0 to k do
          Denote the adjacency lists of p[i] and p[i+1] as
```

```
            (a_l^i) and (a_l^{i+1}), respectively;
        s := 1, t := 1;
        repeat
            case a_s^i = a_t^{i+1} :
                halt and return("Extendible!");
            case a_s^i > a_t^{i+1} :
                t := t + 1;
            case a_s^i < a_t^{i+1} :
                s := s + 1;
        until s > d(p[i]) or t > d(p[i + 1]);
    end;
8.  return("Not Extendible!").
```

Complexity: We give details for steps 1,3,5, and 7. The other steps are obviously linear.

Step 1 can be done in linear time as mentioned in Section 2.

Step 3 can be done in linear time as follows. We keep a copy of the original adjacency lists and initialize a new one with each list set to empty at first. Then we traverse the vertices in the order of their labels. For each vertex we add it to the new list of each of its neighbors. The total execution time is $O(n + m)$ and we use $O(n + m)$ extra storage.

Step 5 can be implemented as follows: for each $p[i]$, traverse its adjacency list and find its first right neighbor, that is, a neighbor with a larger label. If we find one, then continue until either reaching the end of the list or encountering the second right neighbor. Thus, we need only to exhaust the adjacency list for each p[i] in the worst case. In total, this amounts to $O(n + m)$ time.

Step 7 is also linear. For any given i, each time in its `repeat` block, we either reach a halting condition and stop executing, or we must increase by

1 the value of s or t. By the ending condition (see the `until` line), we need to execute less than $d(p[i]) + d(p[i+1])$ times for each i. Hence in total we need no more than $\sum_0^k (d(p[i]) + d(p[i+1]))$ times. Note that

$$\sum_0^k (d(p[i]) + d(p[i+1])) \leq \sum_0^{k+1} 2d(p[i]) \leq \sum_{v \in V(G)} 2d(v) = 4m$$

Hence step 7 is also linear.

Thus, Algorithm 4 is in $O(n+m)$ time.

Correctness Proof: By step 1, b is the last vertex in the scheme that we found. Hence $N_R(b) = \emptyset$. Note that this algorithm returns "Not Extendible" if and only if step 8 is executed. It is clear from the algorithm that step 8 be executed if and only if:

(i) There exists a path $P = p_0 \cdots p_{k+1}$ in G such that $p_0 = a, p_{k+1} = b$ and $N_R(p_i) = p_{i+1}$ for $0 \leq i \leq k$.

(ii) For each $0 \leq i \leq k$, p_i and p_{i+1} have no common neighbors.

By Lemma 5, (i) and (ii) together indicate that for any $v \notin V(P)$, $|N_R(v) \cap V(P)| \leq 1$. Since $N_R(p_i) = p_{i+1}$ for $0 \leq i \leq k$ and $N_R(p_{k+1}) = N_R(b) = \emptyset$, $N(v) \cap V(P) = N_R(v) \cap V(P)$ and so $|N(v) \cap V(P)| \leq 1$. By the remark following Lemma 9, this implies that G is (a, b)-path essential. Furthermore by our Extension Theorem, this implies that G is not $(\{a, b\}, 2)$-extendible.

On the other hand, if the execution returns "Extendible", then it must be that some halting condition is satisfied.

If this takes place in step 5, there exists a path $p_0 \cdots p_l$ in G where $p_0 = a$ and $N_R(p_i) = \{p_{i+1}\}$ for $i < l$, $p_s \neq b$ for $s = 1, 2, \cdots, l$. Furthermore either p_l has no right neighbors or it has two or more right neighbors. By Lemma 2 and Lemma 3, this implies that G is $(\{a, b\}, 2)$-extendible.

Otherwise the halting condition is reached in step 7. This means that step 5 has been passed and so there exists a path $p_0 \cdots p_{k+1}$ in G where $p_0 = a, p_{k+1} = b$, and such that $N_R(p_i) = \{p_{i+1}\}$ for $0 \leq i \leq k$. Furthermore for some i, p_i and p_{i+1} have common neighbor v. The only neighbors of p_{i+1} in P are p_i and p_{i+2}, but neither of these is a neighbor of p_i. Therefore $v \notin V(P)$. So we can also conclude that v is not a right neighbor of either p_i or p_{i+1}. Hence $p_i, p_{i+1} \in N_R(v)$ and so by Lemma 6, G is $(\{a,b\},2)$-extendible.

Hence if the execution returns "Extendible", then G is $(\{a,b\},2)$-extendible.

Thus we prove that, if "Not Extendible" is returned, G is really not extendible; if "Extendible" is returned, G is really extendible. Hence Algorithm 4 does determine the $(\{a,b\},2)$-extendibility of G. This finishes the proof. □

Algorithm 5: Let G be a chordal graph and $a, b \in V(G)$ be two distinct vertices in G. Suppose that we have assigned colors $x_1, x_2 \in \{1,2\}$ to a and b, respectively. Determine whether this assignment can be extended to a proper 2-coloring for the clique hypergraph of G.

First note that the problem here is different from that in Algorithm 4, for a particular partial coloring may be extendible even when in general the graph is not extendible.

If G is $(\{a,b\},2)$-extendible, then any partial coloring on a, b can be extended. Otherwise by our Extension Theorem, G is (a,b)-path essential. In this case, the color of a is uniquely determined by that of b: if the (a,b)-path is even, a must be colored the same as b; otherwise they must be colored differently.

The following algorithm first decides whether G is (a,b)-path essential as in Algorithm 4. The difference here is that when we reach a condition that can conclude about the extendibility, we mark the point for later coloring, whereas in Algorithm 4, we just halt and report.

On the other hand, if we find that the graph is (a,b)-path essential, we then look at the parity of the path length together with the partial coloring on a, b. If they agree as in the previous discussion, then we conclude the extendibility in this particular case. Otherwise we report that the partial coloring can not be extended.

Finally, if it is extendible, we modify Algorithm 1 to generate an extension.

```
1.  Find a perfect elimination scheme for G in which b is
the last vertex;
2.  Relabel the vertices of G by this scheme;
(Thus i denotes the vertex
         at the i-th place in the scheme.)
3.  Sort the adjacency lists by vertex labels;
4.  isExtendible:= false, isMarked:= false;
5.  mark:= 0, w_1 := -1, w_2 := -1; x:= 0;
6.  p[0] := a, i := 0;
7.  repeat
         case p[i] has no right neighbors :
              isExtendible:= true, isMarked:= true, mark:= p[i];
              if i is even
                 then x:= x_1;
                 else x:= 3 - x_1;
         case p[i] has two or more right neighbors :
              Assign to w_1, w_2 the first two right neighbors of
p[i], respectively;
              isExtendible:= true, isMarked:= true, mark:= p[i];
              if i is even
                 then x:= x_1;
                 else x:= 3 - x_1;
         otherwise :
              p[i+1] := the only right neighbor of p[i];
              i := i + 1;
```

```
 until (p[i] = b) or (isExtendible == true);
8.  if (isExtendible == true)
        then skip to step 11; 9.  k := i - 1;
10.  for i := 0 to k do
        Denote the adjacency lists of p[i] and p[i+1] as
            (a_l^i) and (a_l^{i+1}), respectively;
        s := 1, t := 1;
        repeat
            case a_s^i = a_t^{i+1} :
                isExtendible:= true, isMarked:= true, mark:= p[i];
                if i is even
                   then x:= x_1;
                   else x:= 3 - x_1;
            case a_s^i > a_t^{i+1} :
                t := t + 1;
            case a_s^i < a_t^{i+1} :
                s := s + 1;
        until s > d(p[i]) or t > d(p[i+1]) or (isExtendible == true);
 end;
11.  len:= k + 1;
12.  if (len is even) and (x_1 == x_2) isExtendible==true;
13.  if (len is odd) and (x_1 != x_2) isExtendible==true;
14.  if (isExtendible==false) halt and return("Not Extendible!");
15.  Otherwise color G as following:
16.  Color vertex n(=b) with x_2;
17.  for v := n - 1 to 1 do
        case (isMarked==true) and (v==mark) :
            Color v with x;
        case v == w_1 :
            Color v differently from w_2;
```

```
        Otherwise :
              if v has no right neighbor
                    then Color v with an arbitrary color;
                    else
                          Find a right neighbor v' of v;
                          Color v differently from v';
end;
```

Complexity: The coloring part added in the end is obviously linear. Thus in comparison with Algorithm 4, it is clear that Algorithm 5 is also in $O(n+m)$ time.

Correctness Proof: Suppose that G is (a,b)-path essential. If the (a,b)-path has even length while $x_1 != x_2$, then by our previous discussion, no proper 2-coloring of G exists with x_1, x_2 assigned respectively to a, b. The same conclusion holds if the (a,b)-path has odd length while $x_1 == x_2$. Hence it is clear that when the algorithm reports "Not Extendible", it is really true that no extension exists.

On the other hand, if the above case does not happen, then we are able to extend the partial coloring using the last part in the algorithm. To verify the correctness of this part, readers are referred to the algorithms in Lemma 2, Lemma 3 and Lemma 6 for more details and explanation. □

6 Open problems

Many interesting problems remain still open for bicoloring clique hypergraphs. For example, we have shown the clique hypergraph bicolorability for two subclasses of perfect graphs, that is, chordal graphs and comparability graphs. Naturally we may turn to other subclasses of perfect graphs, say, weakly chordal graphs, and consider the same problem. This kind of investigation can help locate the NP-completeness nature of the problem for perfect graphs. On the other hand, we have proved that the clique hypergraph of a circular-arc graph is 3-colorable. This naturally prompts the following question:

Is it NP-complete to determine the clique hypergraph bicolorability for circular-arc graphs ?

If the answer is **yes**, then we know that there must exist some circular-arc graph other than a cycle whose clique hypergraph requires 3 colors.

In this paper we also introduce the definition for coloring extension for clique hypergraphs, see Section 4. We also obtain a result about the extendibility for chordal graphs and 2-sets, see the Extension Theorem. The general definition gives rise to a vast set of new problems. For example, we may want to characterize the extendibility for partial coloring on 3-sets in chordal graphs. We may also ask the similar question for comparability graph or circular-arc graphs or any other graph class for which we can get a bound of color numbers for their clique hypergraphs. A lot of investigations are needed to follow this track.

References

[1] J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*, American Elsevier, New York, 1976.

[2] A. Brandstaedt and V. Le and J. Spinrad, *Graph classes: A survey*, SIAM 1999.

[3] G. Dirac, *On rigid circuit graphs*, Abh. Math. Sem. Univ. Hamburg, 25(1961), 71-76.

[4] D. Duffus and B. Sands and N. Sauer and R. Woodrow, *Two-colouring all two-element maximal antichains*, J. Combinatorial Theory Ser. A, 57(1991), 109-116.

[5] D. Fulkerson and O. Gross, *Incidence matrices and interval graphs*, Pacific J. Math., 15(1965), 835-855.

[6] P. Gilmore and A. Hoffman, *A characterization of comparability and interval graphs*, Canad. J. Math., 16(1964), 539-548.

[7] M. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, 1980.

[8] T. Kloks, *Treewidth — Computations and Approximations*, *Lecture notes in computer science*, 842, Edited by G. Goos and J. Hartmanis, Springer-Verlag, 1994.

[9] J. Kratochvil and Zs. Tuza, *On the complexity of bicoloring clique hypergraphs of graphs(Extended Abstract)*, Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), San Francisco, CA, January 9-11, 2000, p. 40-41.

[10] R.M. McConnell and J. Spinrad, *Linear-time modular decomposition and efficient transitive orientation of comparability graphs*, in *5th ACM-SIAM Symposium on Discrete Algorithms(1994)*, 536-545.

[11] R.M. McConnell and J. Spinrad, *Linear-time transitive orientation*, in *8th ACM-SIAM Symposium on Discrete Algorithms(1997)*, 19-25.

[12] J.H. Muller and J. Spinrad, *Incremental modular decomposition*, J. Assoc. Comput. Mach., 36(1989), 1-19.

[13] D. Rose and R. Tarjan and G. Lueker, *Algorithmic aspects on vertex elimination on graphs*, SIAM Journal on Computing, 5(1976), 266-283.

[14] J. Spinrad, *Two-dimensional partial orders*, Ph.D. thesis, Dept. of EECS, Princeton University, N.J., 1982.

[15] J. Spinrad, *Efficient Graph Representations*, manuscript.