Sub-coloring and Hypo-coloring Interval Graphs*

Rajiv Gandhi[1], Bradford Greening, Jr.[1], Sriram Pemmaraju[2], and Rajiv Raman[3]

[1] Department of Computer Science, Rutgers University-Camden, Camden, NJ 08102.
E-mail: rajivg@camden.rutgers.edu.
[2] Department of Computer Science, University of Iowa, Iowa City, Iowa 52242.
E-mail: sriram@cs.uiowa.edu.
[3] Max-Planck Institute for Informatik, Saarbrücken, Germany.
E-mail: rraman@mpi-inf.mpg.de.

Abstract. In this paper, we study the *sub-coloring* and *hypo-coloring* problems on interval graphs. These problems have applications in job scheduling and distributed computing and can be used as "subroutines" for other combinatorial optimization problems. In the sub-coloring problem, given a graph G, we want to partition the vertices of G into minimum number of sub-color classes, where each sub-color class induces a union of disjoint cliques in G. In the hypo-coloring problem, given a graph G, and integral weights on vertices, we want to find a partition of the vertices of G into sub-color classes such that the sum of the weights of the heaviest cliques in each sub-color class is minimized. We present a "forbidden subgraph" characterization of graphs with sub-chromatic number k and use this to derive a a 3-approximation algorithm for sub-coloring interval graphs. For the hypo-coloring problem on interval graphs, we first show that it is NP-complete and then via reduction to the max-coloring problem, show how to obtain an $O(\log n)$-approximation algorithm for it.

1 Introduction

Given a graph $G = (V, E)$, a *k-sub-coloring* of G is a partition of V into *sub-color classes* $V_1, V_2, \ldots, V_k$; a subset $V_i \subseteq V$ is called a sub-color class if it induces a union of disjoint cliques in G. Figure 1(a) shows a 2-sub-coloring of a graph, with the black vertices forming one sub-color class and the white vertices the other. The smallest k for which a graph has a k-sub-coloring is called the *sub-chromatic number* of G, and is denoted $\chi_s(G)$. The *sub-coloring problem* [1, 4, 5, 9, 25] seeks to find a partition of vertices of G into the smallest number of sub-color classes. Clearly, any proper coloring of G is also a sub-coloring, since any proper color class can be viewed as the disjoint union of size-1 cliques; hence, $\chi_s(G) \leq \chi(G)$. Of course, the sub-chromatic number can be much smaller than the chromatic number (e.g., consider a large clique). Figure 1(b) shows a graph G with $\chi(G) = \chi_s(G) = 3$. To see that $\chi_s(G) \geq 3$, observe that each of the 2-paths induced by $\{b, d, e\}$ and $\{c, f, g\}$ require 2 sub-color classes. Furthermore, if the subgraph induced by $\{b, c, d, e, f, g\}$ is colored using 2 sub-colors, then vertex a cannot be added to either of the sub-color classes because each of the sub-color classes will contain at least one vertex from $\{b, d, e\}$ and at least one vertex from $\{c, f, g\}$. We call the graph shown in Figure 1(b) a *binary clique of order 3*, denoted $BC(3)$. Later (in Section 2) we define the family, $BC(k)$, $k \geq 1$, of order k binary cliques and show that the presence of an induced binary clique is an obstacle to having a small sub-chromatic number, in the sense that $\chi_s(BC(k)) \geq k$.

Given a graph $G = (V, E)$, and a vertex weight function $w : V \to \mathbb{N}$, the *hypo-coloring problem* [7] seeks to find a partition of the vertices of G into sub-color classes such that the sum of the weights of the heaviest cliques in each sub-color class is minimized. In other words, if $V_1, V_2, \ldots, V_k$ are the sub-color classes of a hypo-coloring solution, then the cost of the solution is $\sum_{i=1}^{k} \max_{K \subseteq V_i} w(K)$, where each K is a clique in the sub-color class V_i and $w(K)$ is the sum of the weights of vertices in K. Figure 1(a) shows a hypo-coloring of a vertex-weighted graph with cost $17 + 11 = 28$.

* Part of this work was done when the first author was visiting the University of Iowa. Research partially supported by Rutgers University Research Council Grant and by NSF award CCF-0830569.

Fig. 1. (a) shows a 2-sub-coloring of a graph whose chromatic number is 3. If the numbers next to the vertices are taken to be vertex-weights then the white-black coloring is a hypo-coloring of cost 17 + 11 = 28. (b) shows a graph whose chromatic number and sub-chromatic number are both 3. This is an example of $BC(3)$, a binary clique of order 3. The binary string vertex-labels are useful in defining the family of binary cliques (see Section 2).

Our Contribution. This paper studies the approximability of sub-coloring and hypo-coloring on interval graphs. On the positive side, we present (in Section 2) a 3-approximation algorithm for sub-coloring interval graphs. This is the first constant-factor approximation algorithm for the problem. To obtain this algorithm, we define a family of "obstacles," $BC(k)$, $k \geq 1$, binary cliques of order k, and show that $\chi_s(G) \geq k^*$ for any graph G, where k^* is the largest k for which G contains an induced $BC(k)$. Furthermore, for interval graphs, we show that this lower bound is tight to within a constant factor. In other words, if for an interval graph G, k^* is the largest k for which G contains an induced $BC(k)$, then we show that $\chi_s(G) \leq 3 \cdot k^*$. This proof is constructive and leads to our approximation algorithm. This overall approach seems promising for obtaining small-sized sub-colorings of other classes of graphs such as chordal graphs and disk graphs. We also present an $O(\log n)$-approximation algorithm for the hypo-coloring problem, via reduction to the max-coloring problem [23]. Specifically, we show that for any graph G, an optimal solution to max-coloring is an $O(\log n)$-approximation for the hypo-coloring problem. Since the problem of max-coloring interval graphs admits a 2-approximation [23], we get an $O(\log n)$-approximate solution for hypo-coloring interval graphs. In fact, we get an $O(\log n)$-approximation for hypo-coloring on a variety of graph classes including perfect graphs, unit disk graphs, circle graphs, etc., all of which admit constant-factor approximation algorithms for the max-coloring problem [22]. On the negative side, we show (in Section 3) that hypo-coloring on interval graphs is NP-complete.

It is worth noting here that the complexity status of sub-coloring on interval graphs is unknown. Results due to Broersma et al. [5] imply that there is an $n^{O(\log n)}$-time algorithm for the sub-coloring problem on interval graphs. The existence of a subexponential time algorithm for the problem makes it unlikely that sub-coloring on interval graphs is NP-complete. Given this, it is worth further exploring the possibility that the problem has a polynomial-time algorithm.

1.1 Applications

The sub-coloring and hypo-coloring problems have a variety of applications to job scheduling, distributed computing, and combinatorial optimization. We sketch some of these applications next.

Job scheduling. Motivated by applications in *batch scheduling*, in this paper, we study the sub-coloring and hypo-coloring problems on *interval graphs*. An interval graph is an intersection graph of intervals on the real line [13] and these graphs are typically used to model problems in job scheduling and resource allocation [6, 23]. Consider a batch scheduling environment, where we have a collection of jobs $\mathcal{J} = \{J_1, \cdots, J_n\}$. Jobs have processing times, given by $w : \mathcal{J} \to \mathbb{N}$. We are also given a conflict graph on the jobs, $G = (\mathcal{J}, E)$ such that in any batch, only non-conflicting jobs or equivalently, a set of pairwise non-adjacent jobs can be scheduled on an arbitrary number of parallel machines. In a batch schedule the jobs in the same batch are run simultaneously. The next batch of jobs start running when all the jobs in the current batch complete execution. This problem has been abstracted as the *max-coloring* problem [23]. Now suppose that in a batch we have one job with a *large* processing time, and other jobs with very *small* processing times, then all the machines except for the machine processing this large job are idle until the batch completes. We can get an improved schedule if in each batch the jobs form a union of disjoint cliques. Thus, each clique can be run on the same machine sequentially, while the next set of jobs is scheduled

once this batch of jobs completes. Such schedules can be seen as batch schedules with a kind of *backfilling* [24, 18].

Distributed Computing. The sub-coloring problem on general graphs is also motivated by the *network decomposition* problem in distributed computing [2, 20]. A vertex partition $V_1, V_2, \ldots, V_k$ of a graph $G = (V, E)$ induces a *cluster graph* with vertex set $\{1, 2, \ldots, k\}$ and edges $\{i, j\}$ iff there is an edge in G between some $u \in V_i$ and $v \in V_j$. The network decomposition problem seeks to find a vertex partition $V_1, V_2, \ldots, V_k$ of G such that each *cluster* $G[V_i]$ has small diameter and the cluster graph has small chromatic number. A $(c(n), d(n))$-network decomposition is one in which the cluster graph chromatic number is bounded above by $c(n)$ and the diameter of each cluster is bounded about by $d(n)$. For example, Awerbuch et al. [2] present a deterministic, distributed algorithm running in $O(n^{\epsilon(n)})$ time for computing a $(n^{\epsilon(n)}, n^{\epsilon(n)})$-network decomposition, where $\epsilon(n) = \sqrt{\log \log n}/\sqrt{\log n}$. Using this they obtain the first, deterministic, sublinear distributed algorithms for several classical problems in distributed computing. If we restrict the diameter of each V_i to 1, i.e., a clique, then the network decomposition problem is equivalent to the sub-coloring problem. This is because each proper color class of the cluster graph is a disjoint union of cliques from the input graph. Lately, this approach to designing fast distributed algorithms has become very popular in wireless networks [15]. This motivates the study of sub-coloring on geometric intersection graphs such as *unit disk graphs (UDGs)* or *disk graphs*, that are commonly used to model wireless networks. It is easy to see that the sub-chromatic number of UDGs is bounded above by a constant; in fact, a constant-size sub-coloring of a UDG can be efficiently computed even without geometric representation of the UDG [21]. However, the problem seems to be open for disk graphs.

Combinatorial Optimization. For certain types of graph optimization problems, computing an optimal or a "near" optimal solution directly may be difficult, but the problem may be approached by solving it separately on simple subgraphs such as a clique or a disjoint union of cliques, and then combining the solutions. Specifically, for certain problems, computing a sub-coloring, solving the problem separately on disjoint sub-color classes, and then picking the best of these solutions directly yields an approximation factor of $\alpha \cdot \chi_s(G))$, where α is the approximation factor within which we can solve the problem on each sub-color class. An example of this is the *maximum feasible subsystem* problem on interval matrices [8]. Here, we are given an infeasible linear system $l \leq Ax \leq u$, where A is an *interval matrix* and the objective is to find the largest feasible subsystem. Elbassioni et al. [8] present a constant factor bi-criteria approximation algorithm for the case in which A represents a disjoint union of cliques. When A represents an arbitrary interval graph G, it is possible to start with a sub-coloring of G and solve the problem in each sub-color class. This immediately gives us a $O(\chi_s(G))$-approximation algorithm, which in the worst case translates to an $O(\log n)$-approximation algorithm for n-vertex interval graphs. This approach of using sub-coloring as a preprocessing step in a combinatorial optimization problem has also used by Bansal et al. [3].

1.2 Related Work

The sub-chromatic number of a graph was introduced by Mynhardt and Broere [4, 19], and studied as *spot-coloring* by Hartman [14]. Achlioptas [1] proved that F-free coloring[4] is NP-hard even when F is any graph with at least 3 vertices. By setting $F = P_3$ we get that the sub-coloring problem on general graphs is NP-hard. Fiala et al. [9] showed that F-free coloring is NP-hard even for triangle-free planar graphs with maximum degree 4, while giving polynomial time algorithms for sub-coloring on cographs and graphs of bounded tree-width. Stacho [25] has shown that sub-coloring

[4] For a graph $G = (V, E)$, an F-free coloring is a partition of the vertex set of G such that in each color class, the vertices do not have F as an induced subgraph.

on chordal graphs is NP-complete. Broersma et al. [5] study algorithmic and combinatorial aspects of sub-coloring on various classes of graphs. Specifically, they show that when G is chordal $\chi_s(G)$ is $\Theta(\log n)$. They also show that for any constant r, there is a polynomial time algorithm to compute a sub-coloring of interval graphs that have sub-coloring $\leq r$. However, they do not consider the problem of obtaining an approximation algorithm for sub-coloring on general interval graphs.

Motivated by the problem of batch scheduling conflicting jobs, de Werra et al. [7] introduced the hypo-coloring problem. The authors give a polynomial time algorithm for graphs with maximum degree 2, and for forests with bounded maximum degree. They also show that the problem is NP-hard on bipartite graphs and triangle-free planar graphs.

A problem that seems related to hypo-coloring is the *max-coloring problem*. Given a graph $G = (V, E)$ and a weight function $w : V \to \mathbb{N}$ the problem is to find a proper vertex coloring $C_1, C_2, \ldots, C_k$ of G that minimizes $\sum_{i=1}^{k} \max_{v \in C_i} w(v)$. Note that the special case of this problem in which $w(v) = 1$ for all $v \in V$ is simply the problem of coloring graphs using fewest colors. Pemmaraju et al. [23] show that the max-coloring problem on interval graphs is NP-complete and give a 2-approximation algorithm. In Section 3 we study the relation between the optimal solutions for max-coloring and hypo-coloring and provide a reduction from hypo-coloring to max-coloring that leads to an $O(\log n)$-approximation to hypo-coloring.

2 A 3-Approximation for Sub-coloring Interval Graphs

This section presents an algorithm that takes as input an interval graph $G = (V, E)$ and returns a partition $S_1, S_2, \ldots, S_k$ into sub-color classes such that $k \leq 3 \cdot \chi_s(G)$. We start by first establishing a lower bound on $\chi_s(G)$, for any graph G. A *complete binary tree of order* k, $k \geq 1$, denoted $CBT(k)$, is a rooted tree with vertex set $\{0,1\}^{k-1}$ and edge set $\{\{\alpha, \alpha 0\}, \{\alpha, \alpha 1\} \mid \alpha \in \{0,1\}^{k-2}\}$. The *root* of the tree is the vertex labeled ε, the empty string. A *binary clique of order* k, denoted $BC(k)$, is obtained from $CBT(k)$ by adding edges $\{\alpha, \beta\}$ where β is a strict prefix of α. Figure 1(b) shows $BC(3)$ along with the binary string labels for the vertices. The edges $\{a, d\}$, $\{a, e\}$, $\{a, f\}$, and $\{a, g\}$ were added in going from $CBT(3)$ to $BC(3)$.

Lemma 1. *If a graph G (not necessarily an interval graph) contains $BC(k)$ as an induced subgraph, then $\chi_s(G) \geq k$.*

Proof. The proof is by induction on k. The base case ($k = 1$) is trivially true and we assume as induction hypothesis that $\chi_s(BC(k-1)) \geq k-1$. Let $G = BC(k)$ and let v be the root of $BC(k)$. $G - v$ consists of two disjoint copies of $BC(k-1)$ — call these H_1 and H_2. From the induction hypothesis and the fact that H_1 is an induced subgraph of G, we conclude that $\chi_s(G) \geq k-1$. Now suppose that there is sub-coloring of G using $k-1$ sub-colors. Consider the sub-color class C in this sub-coloring, that contains the vertex v. In order for C to be a sub-color class, $C - v$ must belong completely to H_1 or completely to H_2. This means that one of H_1 or H_2 has a sub-coloring using $k-2$ or fewer colors, a contradiction. Thus $\chi_s(G) \geq k$.

The 3-approximation algorithm that we will present next has two main phases. Suppose that G is the input interval graph and A is the set of intervals corresponding to this graph. In the first phase (partitioning phase), we partition the intervals in A into subsets $S_1, S_2, \ldots, S_k$ and show that G contains an induced binary clique of order k, implying via Lemma 1 that $\chi_s(G)$ is lower bounded by the size of the partition. In the second phase (coloring phase) of the algorithm we sub-color each S_i using at most 3 colors, and using a different set of colors for each subclass. This yields a sub-coloring with $3k \leq 3\chi_s(G)$ colors.

2.1 Partitioning Phase

Our partitioning procedure takes as input a collection A of intervals. An interval I in A is said to be *internal* if it completely contains (in the geometric sense) two disjoint intervals I_1 and I_2. Any interval that is not internal is called *external*. The partitioning algorithm (shown in Figure 2) simply peels of "layers" of external intervals. Note that every non-empty collection of intervals has a non-empty subset of external intervals.

```
PARTITION(A)
1   A_0 ← A; k ← 0
2   while (A_k ≠ ∅) do
3         k ← k + 1
4         S_k ← intervals that are external in A_{k−1}
5         A_k ← A_{k−1} \ S_k
6   return S_1, S_2, ..., S_k
```

Fig. 2. Partitioning algorithm that takes a set A of intervals as input and returns a partition $S_1, S_2, \ldots, S_k$.

Lemma 2. *If* PARTITION*(A) returns a $S_1, S_2, \ldots, S_k$, then G, the interval graph corresponding to A, contains an induced $BC(k)$.*

Proof. We will prove by induction that for any j, $1 \leq j \leq k$, and any interval $I \in S_j$, I is the root of a binary clique H of order j contained entirely within the intervals in $B_j := S_1 \cup S_2 \cup \cdots S_j$ and furthermore all intervals in H are entirely contained (in the geometric sense) in I. The base case ($j = 1$) is trivially true. Consider any interval $I \in S_j$. There are two disjoint intervals I_1 and I_2 in S_{j-1} that are completely contained in I. Otherwise, I would have been a member of S_{j-1}. By the inductive hypothesis, I_1 is the root of H_1, a binary clique of order $j-1$ and I_2 is the root of H_2, a binary clique of order $j-1$. Furthermore, all intervals in H_i, $i = 1, 2$ are contained in I_i. This implies that H_1 and H_2 are disjoint and also that interval I has edges to all the intervals in H_1 and in H_2. The graph induced by I and the intervals in H_1 and H_2 is a binary clique or order j, with root I, and with all intervals contained within I.

2.2 Coloring Phase

After partitioning A into subsets $S_1, S_2, \ldots, S_k$, we "color" the intervals in each subset S_i as follows. For the rest of this subsection let S denote an arbitrary S_i. We start by choosing the left-most maximal clique in S; call this M_1. Let $I_1 \in M_1$ be the interval with the right-most right endpoint, and let N_1 be the set of intervals not in M_1 that are completely contained within I_1. We then remove intervals in $M_1 \cup N_1$ from S and if $S \neq \emptyset$, we repeat the process and compute M_2 and N_2 and so on. Once this process terminates, for some $k \geq 1$, we have partitioned S into subsets $M_1, M_2, \ldots, M_k$ and $N_1, N_2, \ldots, N_k$. We then "color" all intervals in $N_1 \cup N_2 \cup \cdots \cup N_k$ using color C_2 and alternately "color" the intervals in $M_1, M_2, \ldots, M_k$, using color C_1 and C_0. The pseudocode for this algorithm is given in Figure 4. See Figure 3 for an illustration of this algorithm.

2.3 Analysis

Observe that after the 3COLOR algorithm finishes processing a subset S, each interval in S is assigned to exactly one of 3 subsets, C_0, C_1, or C_2. We now show that each C_j, $j = 0, 1, 2$, is a union of disjoint cliques. This suffices to prove that C_0, C_1, C_2 is a valid 3-sub-coloring of S.

Fig. 3. The coloring algorithm identifies $M_1 = \{a, b\}$ – this is the leftmost maximal clique. I_1 is b and therefore $N_1 = \{c\}$. Deleting $\{a, b, c\}$, makes $\{d, e, f\}$ the next leftmost maximal clique. Hence, $M_2 = \{d, e, f\}$, $I_2 = e$, and $N_2 = \{g\}$. As a result, $C_0 = \{d, e, f\}$, $C_1 = \{a, b\}$, and $C_2 = \{c, g\}$.

```
3Color(S)
1    k ← 0
2    C_0 ← C_1 ← C_2 ← ∅
3    while (S ≠ ∅) do
4         k ← k + 1
5         M_k ← leftmost maximal clique
6         I_k ← interval in M_k with the rightmost right endpoint
7         N_k ← intervals not in M_k that are completely contained in I_k
8         S ← S \ (M_k ∪ N_k)
9         if k is even then
10           C_0 ← C_0 ∪ M_k
11        else
12           C_1 ← C_1 ∪ M_k
13        C_2 ← C_2 ∪ N_k
14   return C_0, C_1, C_2
```

Fig. 4. Sub-Coloring algorithm that takes a subset S of intervals produced by the partition algorithm and computes a 3-sub-coloring of S.

Lemma 3. *C_2 is a union of disjoint cliques.*

Proof. Let N_x and N_y be any two particular but arbitrary cliques in C_2. Without loss of generality, let $x < y$. Note that N_x is the set of intervals in C_2 which are completely contained in interval I_x and N_y is the set of intervals in C_2 which are completely contained in interval I_y. If I_x and I_y do not overlap, then clearly there are no overlaps between intervals in N_x and intervals in N_y. However, if I_x and I_y do overlap, then it must be that $y = x + 1$. Now assume for contradiction that interval $J \in N_x$ overlaps with interval $K \in N_y$. Since J is completely contained in I_x, and K overlaps J, clearly K also overlaps I_x. This means that K belongs to M_y (the leftmost maximal clique after all intervals in M_x are removed), and hence cannot belong to N_y, a contradiction. □

Lemma 4. *C_1 (C_0, respectively) is a union of disjoint cliques.*

Proof. Let M_x and M_y be any two particular but arbitrary cliques in C_1 (C_0, respectively) and assume for contradiction that there are intervals in M_x which overlap intervals in M_y. Without loss of generality assume that $x < y$. Let $J \in M_x$ be an interval which overlaps with an interval $K \in M_y$. Recall that $I_x \in M_x$ is the interval having rightmost right endpoint in M_x, therefore I_x also overlaps K. This implies that M_x and M_y are consecutive maximal cliques, that is, $M_y = M_{x+1}$ and thus the algorithm would not have assigned both M_x and M_y to C_1, (C_0, respectively) a contradiction. □

Lemmas 3 and 4 lead to the following corollary.

Corollary 1. *The algorithm* 3Color(S) *computes a 3-sub-coloring of S.*

This corollary, along with Lemmas 1 and 2 leads to the following result.

Theorem 1. *There is a 3-approximation algorithm for the sub-coloring problem on interval graphs.*

Partitioning interval graphs into proper interval graphs. An interval graph G is called *proper interval graph* if there is an interval representation of G such that no interval properly contains another. Gardi [10] posed as an open problem, the problem of obtaining a constant-factor approximation to partitioning an interval graph into the smallest number of proper interval graphs. We note here that the sub-coloring approximation algorithm immediately yields a 9-approximation to this problem.

Theorem 2. *There is a 9-approximation algorithm for partitioning an interval graph into fewest number of proper interval graphs.*

Proof. Let k be the minimum number of proper interval graphs into which an input interval graph G can be partitioned. Clearly $k \leq \chi_s(G)$. It can also be easily seen that a proper interval graph has sub-chromatic number at most 3; this implies that $\chi_s(G) \leq 3k$. Our algorithm computes a partition of G into at most $3\chi_s(G)$ sub-colors (each of which may be viewed as a proper interval graph). The theorem follows.

3 Hypo-coloring of Interval Graphs

In this section we study the computational complexity of the hypo-coloring problem on interval graphs. We first show that it is NP-complete and then present an $O(\log n)$-approximation algorithm for it via a reduction to the max-coloring problem.

3.1 NP-Completeness

The NP-completeness of hypo-coloring on interval graphs is shown by using a reduction from Coloring Circular-Arc Graphs [11]. Our proof is heavily influenced by the NP-completeness proof of minimum sum coloring on interval graphs by D. Marx [16] and the proof of NP-completeness of max-coloring on interval graphs by Pemmaraju et al. [23].

Coloring Circular-Arc Graphs
INPUT: A circular-arc graph $G = (V, E)$, and a number $k \in \mathbb{N}$.
QUESTION: Does G have a coloring of cost at most k?

We may assume that a circular arc representation of G is given to us since there is a polynomial time algorithm for recognizing a given graph G as a circular arc graph and returning a circular arc representation if G is indeed a circular arc graph [17]. Also, without loss of generality, we can assume that there exists a point on the circle that is contained in precisely k circular arcs. If not, consider a sector that does not contain any end-points of any of the arcs. If this sector is contained in $\ell < k$ arcs, then we can introduce $k - \ell$ arcs that are contained only in this sector, without changing the k-colorability of the graph. If the sector was contained in at least $k+1$ arcs, then G is not k-colorable, and the reduction can be trivially completed. In the following proof, we view the hypo-coloring problem as a decision problem in which we are given an additional input, a positive integer W, and asked if the given instance has a hypo-coloring of cost at most W.

Theorem 3. *Hypo-coloring interval graphs is NP-Complete.*

Proof. The problem is clearly in NP. To show NP-hardness, we reduce Coloring Circular-Arc Graphs to hypo-coloring interval graphs. Given a circular-arc graph G and parameter k, let r be a ray from the center of the circle that passes through k arcs of G. We construct an interval graph H from G by splitting the arcs intersecting r. More formally, let $I = \{I_1, \cdots, I_k\}$ be the arcs intersecting r. We replace each arc $I_i \in I$ by two arcs I'_i and I''_i, that start and end respectively at r. This gives us an interval graph and we can assume that the intervals $I' = \{I'_i \mid i = 1, 2, \ldots, k\}$ form the leftmost intervals, and the intervals $I'' = \{I''_i \mid i = 1, 2, \ldots, k\}$ form the rightmost intervals.

We set the left end-points of the intervals I'_i such that $l(I'_i) < l(I'_j)$ whenever $i < j$ and we set $r(I''_i) < r(I''_j)$ whenever $i < j$. Here $l(I)$ and $r(I)$ respectively denote the left and right end-points of an interval I. Further, we add two sets of intervals $L = \{L_1, \cdots, L_k\}$ and $R = \{R_1, \cdots, R_k\}$, such that $r(L_i) = l(I'_i)$ and $l(R_i) = r(I''_i)$ for all $i = 1, \cdots, k$. The weights of the intervals are defined as follows. We let $w(L_i) = w(R_i) = 1 + i \cdot \epsilon$, for $\epsilon = \frac{1}{k+1}$), $w(I) = 1$, $\forall I \notin L \cup R$. This gives us the interval graph H. Note that scaling the weights by a factor of $k+1$ will guarantee that all vertex-weights in H are integral.

Suppose G can be colored with k colors, this gives us a hypo-coloring of H of cost $C := k + k(k+1)\epsilon/2$ as follows. To each interval in H, assign a color c if it's corresponding interval in G is assigned a color c. In such a coloring, the intervals I'_i and I''_i are assigned the same color for each $i = 1, \cdots k$, and such a coloring allows us to assign the same color to the intervals L_i and R_i for each i.

On the other hand, suppose there is a hypo-coloring of cost C or less. We first show that such a coloring must in fact be a proper coloring of H. To see this, consider just the subgraph of H induced by the intervals in $L \cup I'$. A hypo-coloring of this subgraph with cost at most C must itself be proper because any hypo-color class with a clique of size larger than 1 will force us to place one of the intervals in $L \cup I'$ in a new color class, incurring a cost of at least $C + 1$. Further, in such a proper coloring, if L_i is not placed in the same color class as R_i for each $i = 1, \cdots, k$, the coloring has cost at least $C + \epsilon$. Hence, a hypo-coloring of cost C or less must be proper, and place L_i and R_i in the same color class, for each i, which is only possible if I'_i and I''_i are placed in the same color class for each $i = 1, \cdots, k$, and this yields a proper coloring of the circular-arc graph G. □

3.2 An $O(\log n)$-approximation for Hypo-coloring

In this section we show that an optimal solution to the max-coloring problem on any graph G is an $O(\log n)$-approximation to the hypo-coloring problem with input G. Since there is a 2-approximation algorithm for max-coloring on interval graphs [23] this implies that there is an $O(\log n)$-approximate solution for hypo-coloring interval graphs.

Theorem 4. *Given any graph G, an optimal max-coloring of G is an $O(\log n)$-approximation for hypo-coloring of G.*

Proof. Let OPT_H be an optimal hypo-coloring solution. We will prove the claim by showing that there is a feasible maxcoloring solution whose cost is $O(\log n)OPT_H$. Let $S_1, S_2, \ldots, S_k$ be the k color classes in OPT_H and let the cliques in S_i be given by $S_i^1, S_i^2, \ldots, S_i^{p_i}$. Let m_i be the maximum number of vertices in any clique in S_i. In other words, $m_i = \max_{1 \le j \le p_i} |S_i^j|$. Consider the max-coloring solution in which there are color classes $C_i^1, C_i^2, \ldots, C_i^{m_i}$ for each S_i in OPT_H. For $1 \le x \le m_i$, the color class C_i^x is formed by including the x^{th} heaviest vertices from each of the cliques in S_i. Since the cliques are disjoint, so are the x^{th} heaviest vertices from each of the cliques. Consider the heaviest vertex $v \in C_i^x$ and let v belong to clique S_i^y. Since v is the xth heaviest vertex in S_i^y there must be $x-1$ vertices in S_i^y of weight at least $w(v)$ that are placed in $C_i^1, C_i^2, \ldots, C_i^{x-1}$. Hence,

$$w(C_i^x) \le \frac{W_i}{x}$$

where W_i is the weight of the heaviest clique in S_i. Thus $C_i^1, C_i^2, \ldots, C_i^{m_i}$ is a feasible maxcoloring solution for input S_i with cost

$$\le W_i + \frac{W_i}{2} + \ldots + \frac{W_i}{m_i} = W_i H_{m_i} \le W_i(\ln m_i + 1).$$

Thus the total cost of our maxcoloring solution for G becomes

$$\begin{aligned}
&\leq W_1(\ln m_1 + 1) + W_2(\ln m_2 + 1) + \ldots + W_k(\ln m_k + 1) \\
&\leq W_1(\ln m + 1) + W_2(\ln m + 1) + \ldots + W_k(\ln m + 1) \\
&= O(\log m)(W_1 + W_2 + \ldots + W_k) \\
&= O(\log m)OPT_H
\end{aligned}$$

where m is the number of vertices in the largest clique among all batches $S_1, S_2, \ldots, S_k$. Since $m \leq n$, we have obtained a solution to maxcoloring of cost $O(\log n)OPT_H$. □

Note that the above analysis is tight by the example in Figure 5.

Fig. 5. An instance G showing a logarithmic gap between the optimum hypo-coloring and optimum max-coloring. This consists of n disjoint cliques $Q_1, \cdots, Q_n$. $|Q_i| = i$, and $w(v) = W/i$ for each $v \in Q_i$. Hence, each clique has a weight W. Clearly, OPT_{hc} for G is W since we can pack all intervals in one sub-color class. On the other hand OPT_{mc} requires n color classes, $C_1, \ldots, C_n$, where C_i has a weight of W/i. Thus, OPT_{mc} is $\Theta(W \log n)$ for this instance.

4 Open Questions

The results in this paper raise a number of open questions. Two of these problems, that interest us the most are:

- Obtaining constant-factor approximation algorithms for sub-coloring on more general classes of graphs such as disk graphs, chordal graphs, etc.
- Obtaining a constant-factor approximation for hypo-coloring on interval graphs or alternately showing that such an approximation is not possible.

References

1. D. Achlioptas. The complexity of g-free colorability. *Discrete Math*, pages 21–30, 1997.
2. Baruch Awerbuch, Andrew V. Goldberg, Michael Luby, and Serge A. Plotkin. Network decomposition and locality in distributed computation. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 364–369, 1989.
3. Nikhil Bansal, Zachary Friggstad, Rohit Khandekar, and Mohammad R. Salavatipour. A logarithmic approximation for unsplittable flow on line graphs. In *Proceedings of the 20th ACM-SIAM Symposium on Discrete Algoxrithms (SODA)*, pages 702–709, 2009.
4. Izak Broere and Christina M. Mynhardt. Generalized colorings of outerplanar and planar graphs. In *Proceedings of the 10th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 151–161, 1984.
5. Hajo Broersma, Fedor V. Fomin, Jaroslav Nesetril, and Gerhard J. Woeginger. More about subcolorings. *Computing*, 69:187–203, 2002.
6. A.L. Buchsbaum, H. Karloff, C. Kenyon, N. Reingold, and M. Thorup. OPT versus LOAD in dynamic storage allocation. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, 2003.
7. D. deWerra, M. Demange, J. Monnot, and V.Th.Paschos. A hypocoloring model for batch scheduling. *Discrete Applied Mathematics*, 146:3–25, 2005.
8. Khaled M. Elbassioni, Rajiv Raman, Saurabh Ray, and René Sitters. On the approximability of the maximum feasible subsystem problem with 0/1-coefficients. In *Proceedings of the 20th ACM-SIAM Symposium on Discrete algorithms (SODA)*, pages 1210–1219, 2009.

9. J. Fiala, K. Jansen, V.B.Le, and E.Seidel. Graph subcoloring:complexity and algorithms. *SIAM Journal on Discrete Mathematics*, 16:635–650, 2003.
10. F. Gardi. On partitioning interval and circular-arc graphs into proper interval subgraphs with applications. In *Latin American Theoretical Informatics Symposium (LATIN)*, pages 129–140, 2004.
11. M. R. Garey, D. S. Johnson, G. L. Miller, and C. H. Papadimitriou. The complexity of coloring circular arcs and chords. *SIAM J. Algebraic and Discrete Methods*, 1:216–227, 1980.
12. M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the theory of NP-completeness*. W.H. Freeman and Company, San Fransisco, 1979.
13. M.C. Golumbic. *Algorithmic graph theory and perfect graphs*. Academic Press, NY, 1980.
14. Chirs Hartman. *Extremal Problems in Graph Theory*. 1997.
15. Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. On the locality of bounded growth. In *Proceedings of the 24th Annual ACM symposium on Principles of Distributed Computing (PODC)*, pages 60–68, New York, NY, USA, 2005. ACM.
16. D. Marx. A short proof of the NP-completeness of minimum sum interval coloring. *Operations Research Letters*, 33(4):382–384, 2005.
17. Ross M. McConnell. Linear-time recognition of circular-arc graphs. *Algorithmica*, 37(2):93–147, 2003.
18. A.W. Mu'alem and D.G. Feitelson. Utilization, predictability, workloads, and user runtime estimates in scheduling the ibm sp2 with backfilling. *IEEE Trans. Parallel and Distributed Syst.*, 12(6):529–543, 2001.
19. Christina M. Mynhardt and Izak Broere. Generalized colorings of graphs. In *Proceedings of the 11th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 583–594, 1985.
20. Alessandro Panconesi and Aravind Srinivasan. On the complexity of distributed network decomposition. *J. Algorithms*, 20(2):356–374, 1996.
21. Sriram V. Pemmaraju and Imran A. Pirwani. Good quality virtual realization of unit ball graphs. In *Proceedings of the 15th Annual European Symposium on Algorithms (ESA)*, pages 311–322, 2007.
22. Sriram V. Pemmaraju and Rajiv Raman. Approximation algorithms for the max-coloring problem. In *Proceedings of the 32nd International Colloquium on Automata Languages and Programming (ICALP)*, pages 1064–1075, 2005.
23. S.V. Pemmaraju, R. Raman, and K. Varadarajan. Buffer minimization using max-coloring. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 562–571, 2004.
24. E. Shmueli and D.G. Feitelson. Backfilling with look ahead to optimize the packing of parallel jobs. *J. Parallel and Distributed Computing*, 65(9):1090–1107, 1995.
25. Juraj Stacho. *Complexity of Generalized Colourings of Chordal Graphs*. 2008.

5 Appendix

5.1 Hypo-coloring, Max-coloring and Dynamic Storage Allocation

The *dynamic storage allocation (DSA)* problem [6], also known as the interval coloring problem. Formally, an instance of this problem consists of an interval graph $G = (V, E)$ and a weight function $w : V \to \mathbb{N}$. A feasible solution to this problem is an assignment of an interval $I(v)$ to each vertex v such that $|I(v)| = w(v)$ and $I(u) \cap I(v) = \emptyset$ if u and v are adjacent vertices. The goal is to minimize $|\cup_{v \in V} I(v)|$. The DSA problem is NP-complete for interval graphs [12], and Buchsbaum et al. [6] give a $(2+\epsilon)$-approximation algorithm for interval graphs. Here we explore how the optimal solution to hypo-coloring relates to that of DSA and max-coloring. Let $OPT_{dsa}(I)$, $OPT_{hc}(I)$ and $OPT_{mc}(I)$ denote costs of optimal solutions to DSA, hypo-coloring and max-coloring respectively for an instance I. First note that any solution to max-coloring is a feasible solution to hypo-coloring and any solution to hypo-coloring is a feasible solution to DSA. This implies the following proposition.

Proposition 1. *For any instances I of a weighted interval graph,*

$$OPT_{dsa}(I) \leq OPT_{hc}(I) \leq OPT_{mc}(I).$$

Note that the example in Figure 5 (which was used in [23] to show that $OPT_{mc}= \Omega(\log n) \cdot OPT_{dsa}$) also shows that $OPT_{hc}= \Omega(\log n) \cdot OPT_{dsa}$. Now, consider the graph G shown in Figure 6. This graph is a union of n disjoint graphs $G_1, \cdots, G_n$. G_1 is a single interval. of weight n. G_i, $i = 2, \cdots, n$ is constructed recursively from G_{i-1} by taking two disjoint copies of G_{i-1} and adding an interval spanning both copies of G_{i-1}. Each interval in G_i has a weight n/i. The maximum weight clique in each G_i has a weight of n, and it is easy to see that $OPT_{dsa}= n$. However, we now show that $OPT_{hc}= \Theta(n \log n)$.

Fig. 6. Example showing the gap between the optimum dynamic storage allocation and optimum hypo-coloring.

Lemma 5. *For the graph G shown in Figure 6, OPT_{hc}=$OPT_{mc}= \Theta(n \log n)$ and $OPT_{dsa}= \Theta(n)$*

Proof. Since the weight of the heaviest clique in each G_i equals n, for $i = 1, \cdots, n$, $OPT_{dsa} \geq n$. To see that $OPT_{dsa}= n$, consider the following interval coloring. For G_1, assign the interval $[0, n]$. Assume we have constructed an interval coloring for G_j, $j = 1, \cdots, i-1$, each of which has a height n. To construct an interval coloring for G_i, note that G_i consists of two disjoint copies of G_{i-1}. Using the same interval coloring as for G_{i-1} for the two disjoint copies in G_i, we obtain a coloring of height $(i-1)n/i$, since the weight of each interval in the copy of G_{i-1} in G_i has weight n/i. This leaves enough space to pack the one remaining interval of weight n/i to obtain a coloring of height n.

Now we consider an optimal max-coloring on G. Note first that $\chi(G_i) = i$, and hence OPT_{mc} uses at least i color classes, each of weight n/i. We can reuse the colors of G_i for the first i color classes of G_j for all $j > i$. The weight of color class j in this coloring is dominated by the weight of an interval from G_j, whose weight is n/j. Hence, OPT_{mc} has a cost $n + n/2 + \cdots + 1 = n \cdot H_n \leq n \log n + 1$.

For the hypo-coloring problem on G, note that $\chi_s(G_i) = i$. Thus, an optimal hypo-coloring of G_i uses at least i colors, and color class consists j of disjoint cliques, each of size 1, and has weight n/j. The coloring produced is hence, exactly the same as that for the max-coloring problem on G, and hence has weight $n \log n + 1$. □