“Rent-or-Buy” Scheduling and Cost Coloring Problems

Takuro Fukunaga[1], Magnús M. Halldórsson[2], and Hiroshi Nagamochi[1]

[1] Dept. of Applied Math. and Physics, Graduate School of Informatics, Kyoto University.
[2] Dept. of Computer Science, University of Iceland.

Abstract. We study several *cost coloring problems*, where we are given a graph and a cost function on the independent sets and are to find a coloring that minimizes the function costs of the color classes. The “Rent-or-Buy” scheduling/coloring problem (RBC) is one that captures e.g., job scheduling situations involving resource constraints where one can either pay a full fixed price for a color class (representing e.g., a server), or a small per-item charge for each vertex in the class (corresponding to jobs that are either not served, or are farmed out to an outside agency). We give exact and approximation algorithms for RBC and three other cost coloring problems (including the previously studied Probabilistic coloring problem), both on interval and on perfect graphs. The techniques rely heavily on the computation of maximum weight induced k-colorable subgraphs (k-MCS). We give a novel bicriteria approximation for k-MCS in perfect graphs, and extend the known exact algorithm for interval graphs to some problem extensions.

1 Introduction

Consider the following scheduling scenario. You are given a collection of jobs, some of which exclusive access to a specialized resource, e.g., a brain scanner. The jobs have all been fixed, with known start and end times, and you have to supply all those that ask with a brain scanner. You know that the minimum number of scanners needed is exactly the largest number χ of jobs that will be concurrently operating, But while you could go out to buy χ scanners, you also have the option to rent them at a fixed price per job. The task is then to decide which jobs to buy a scanner for and which ones to rent one for.

We can formulate this more generally as a graph coloring problem, where jobs are nodes in the graph and edges corresponds to the use of a non-sharable resource. More generally, we may assume that each job i requires a quantity w_i of a given non-sharable resource (in the example above, it may correspond to the rent being a function of length of the job). We obtain the following problem:

Rent-or-Buy Coloring Problem (RBC):
Given: Graph $G = (V, E)$, with vertex weights $w_v \in (0, 1]$.
Find: A proper vertex coloring C consisting of color classes $I_1, I_2, \ldots, I_t$.

Minimize: $f(C) = \sum_i f(I_i)$, where $f(I) = \min(w(I), 1)$ and $w(I) = \sum_{v \in I} w_v$.

In scheduling applications where jobs represent time intervals, the corresponding graph is an interval graph. In ordinary graph coloring, we "pay" one unit for each color that we start to use. The idea behind Rent-or-Buy coloring is that one may often be able to take care of the smaller color classes cheaper, e.g., by paying some elementwise "fine".

We consider more generally *cost coloring problems*, where we have some non-negative cost function $f : 2^V \mapsto \mathbf{R}^+$ on the independent sets of the graph. We will assume anywhere in the paper that the cost of a coloring C is the sum of the costs of the color classes, i.e. $f(C) = \sum_{I \in C} f(I)$.

Intuitively, this corresponds to a scheduling scenario where the cost of a resource is some function of the usage of the resource, when we view each color as a (copy of a) resource. This can apply to many of the innumerable applications of graph colorings. For instance, the cost of a classroom in a timetabling application is not really a unit; different classrooms may have different costs, depending on size, and depending on the amount of use. The cost of a frequency in frequency allocation may depend on time- or space-limitations of the usage. The cost of fulfilling server requests, e.g., for bandwidth allocation in networks, may depend on the willingness to deploy servers, outsource some of the traffic (at a volume-dependent cost), or to pay the indirect cost of refusing service.

The cost coloring framework is very general, which leads us to consider which types of cost functions are natural and of practical interest. First, we usually assume the function to be *monotone*, in that if you request more of a resource, it won't cost less. Second, most reasonable cost coloring functions have the property that they depend only on the combined weight of the set, not the distribution of the weights nor on which particular vertices participate in the set. We call such functions *separable*, when the costs can be represented by a single-variable function, i.e., abusing notation, when $f(I) = f(w(I))$, for any independent set I. We focus on separable functions here, with one exception.

Third, as a consumer, one normally expects there to be an incentive to buy in large quantities; i.e., that the *residual* unit cost goes down with request size. This corresponds to the cost function being *concave*; a separable function f is concave if $f(x) + f(y) \leq 2f((x + y)/2)$, for any $x, y \in \mathbf{R}$. Not all real-life costs obey concavity strictly speaking, as the large-quantity incentives can involve a sequence of thresholds, but tend to nearly-concave.

Our objective in this paper is to address some of the most basic cost coloring problems. The very most basic one would be the ordinary Graph Coloring problem, which has the trivially monotone, concave and separable cost function $f(I) = 1$. We shall be treating, in addition to RBC, the following natural problems. Recall that the cost of a coloring is the sum of the osts of the color classes.

Two-tiered rents with opening costs (TTR): This is a generalization of RBC with two residual costs, c_1 and c_2. Once the weight of the class reaches a certain threshold, the per-item cost changes to the second cost. Additionally, we allow a fixed charge c (less than 1) for the non-zero use of any color, which can represent a cost for "opening" or initiating the use of that resource. The cost function f for each color class I is $f(I) = c + \min(c_1 \cdot w(I), T) + \max(c_2 \cdot (w(I) - T), 0)$. We are not aware of previous work on TTC or RBC.

Threshold colorings: Suppose we have two modes of servicing (independent) sets, depending on their size. E.g., we can either schedule a group by renting a taxi, at a fixed price, or by renting a bus (that will definitely fit all), at a higher fixed price. We seek as before a minimum cost schedule of everyone, taking conflicts into account. The cost function $f(w(I))$ is now constant c_1, when $w(I)$ is at most the threshold T, and a larger constant c_2, when $w(I)$ is above the threshold.

A special case is when the above-threshold cost is too high to be ever cost-effective, e.g. n-fold the below-threshold cost. We have then the *bounded-coloring* problem, which models the case of scheduling conflicting unit-size jobs with bounded number of machines. It is NP-hard on bipartite and interval graphs [3].

Probabilistic coloring: In the *Probabilistic coloring* problem [18], we are given a graph G with independent vertex probabilities $p_v \in [0,1]$ and are to find a coloring where the cost $f(I)$ of a color class is the cumulative probability $f(I) = P(I) = 1 - \prod_{v \in I}(1 - p_v)$. This was proposed to model robustness in optimization, where one is presented *a priori* with a supergraph of what will be used in the future. The cost of the coloring is then the expectation of the number of colors actually used. This cost function is both concave and monotone, but not separable. Probabilistic coloring is NP-hard in bipartite graphs [18], split graphs [5], and interval graphs [12, 4], but solvable in co-bipartite graphs [18], and co-interval graphs [13]. It admits a $\sqrt{\rho_{GC} n}$-approximation, where ρ_{GC} is the approximability of Graph Coloring, a 3/2-factor in bipartite graphs [18], and 2-approximation in split graphs [5].

Our Results and Techniques We can observe that applying ordinary coloring will not give good approximations for these cost coloring problems, nor does the usual approach of repeatedly coloring maximum independent sets. Instead, we make a strong link to the problem of finding a maximum (weight) induced k-colorable subgraph (k-MCS). RBC is in fact solved exactly by finding a maximum k-MCS, for the right choice of k. For approximation, we present in Section 2 a novel bicriteria approximation for k-MCS on perfect graphs, which allows us to approximate RBC in Section 3 within a factor of 2.

In order to solve TTR, we modify the flow reduction of Arkin and Silverberg [1] for weighted k-MCS in interval graphs to give an $O(n^2 \log n)$-time algorithm to solve the following extension: given an interval graph and integers k and h, find a maximum weight k-colorable subgraph whose removal leaves a h-colorable subgraph. This allows us to solve TTR also optimally in interval graphs.

We then show in Section 4 that Probabilistic colorings are always within a factor of $e/(e-1)$ from related RBC colorings. This gives then a complete characterization of Probabilistic coloring, within constant factors, and improved approximations for several classes of graphs.

As a third simple and natural cost function, we consider in Section 5 the approximability of Threshold colorings. These are perhaps the simplest non-concave but separable cost functions. Here we utilize another extension of the k-MCS problems, one where a particular subset is constrained to be in the resulting k-colorable subgraph, which we show to be solvable on interval graphs. We get a 2-approximation for interval graph and 3.7-approximation on perfect graphs.

Related work Entropy coloring is a problem from information theory involving the separable cost measure $f(I) = w(I)\ln(1/w(I))$. It models transmission rate with side information, and has applications in digital compression [4]. It is NP-hard on interval graphs, hard to approximate within a $\Omega(n)$-factor (its value is always at most $\log n$) [4], but polynomially solvable on co-interval graphs [13] and co-bipartite graphs [4]

Gijswijt, Jost and Queyranne [13] recently introduced a general framework for cost coloring problems that they call *value-polymatroidal*. It contains monotone problems where moving vertices from a smaller class to a bigger class does not increase the total cost, i.e., when $f(I \cup \{v\}) + f(J) \leq f(I) + f(J \cup \{v\})$, for any independent sets I, J with $f(I) \leq f(J)$. This class includes all the problems treated in this paper, except Threshold coloring. It also includes the *max coloring* problem [7, 19], which has the non-separable, monotone cost function $f(I) = \max_{v \in I} w_v$. They give a polynomial time algorithm for all such problems on co-interval graphs (complements of interval graphs).

In a companion paper [12], we study separable cost coloring problems, and give approximation algorithms on perfect graphs. In particular, we show that concave separable functions admit a *robust* approximation, in that there is an algorithm that given a graph, produces a coloring that *simultaneously* approximates any concave function on perfect graphs. We also show how to modify these colorings to approximate (in a function-specifical way, necessarily) any monotone separable cost function. In comparison, our results here are more specialized, but the approximation factors are better (e.g., 2 for RBC on perfect graphs vs. 6 for any concave function, and 3.7 for Threshold coloring vs. 12 for any monotone separable function).

Some other types of coloring problems with weights have been considered. In the optimal chromatic cost problem (OCCP) [17], the cost of a color class is linear in its size, but each class has a different multiplier specific. The sum coloring problem [15] is a special case where the multipliers are the natural numbers. These fall outside of our framework, which assumes that all colors are equal.

Notation Let $G = (V, E)$ be a graph given with vertex weights w_v. Let n denote the number of vertices. For a subset $S \subset V$, $G[S]$ denotes the subgraph of G induced by S. For a set S, let $w(S) = \sum_{v \in S} w_v$, and let $w(G) = w(V)$.

A coloring is a partition of V into independent sets. A *k-subgraph* is an induced k-colorable subgraph. We may overload the notation and refer to a vertex subset $S \subset V$ as a k-subgraph if $G[S]$ is k-colorable. k-*MCS* refers to the problem of finding a k-subgraph of maximum total weight, and *Graph Coloring* refers to the classical vertex coloring problem, using the minimum number $\chi(G)$ of colors.

2 Maximum *k*-Colorable Subgraph Problem

Our approach is heavily based on finding large induced subgraphs with small chromatic number. Yannakakis and Gavril [21] showed that integer programming formulation for weighted k-MCS on interval graphs satisfies total unimodularity property, and is therefore polynomially solvable. Arkin and Silverberg [1] gave a $O(n^2 \log n)$ time solution via reduction to minimum cost flow. It was also shown in [21] that k-MCS was polynomially solvable in chordal graphs for fixed k, but NP-hard for k unbounded, even on the subclass of split graphs. Frank [11] showed that unweighted k-MCS is solvable on comparability graphs. Saha and Pal [20] showed weighted k-MCS solvable in permutation graphs. Further, it is easy to compute on bipartite graphs, as we need either find a maximum weight IS or a two-coloring, and can also be computed in $f(k)n$-time on partial k-trees.

2.1 Approximation of Maximum *k*-Subgraphs

The solution of the max k-subgraph problem is an important component of approximation algorithms for numerous coloring problems, e.g., sum coloring [15], sum multi-coloring, batch sum coloring [6], and co-coloring [10]. One would hope to replace the subroutine by an approximation algorithm, for graph classes where k-MCS is NP-hard. However, there are different types of approximations possible. Let W be the weight of an optimal k-subgraph.

Primal: Find a k-subgraph of weight at least cW, for c largest possible.
Dual: Find a $d \cdot k$-subgraph of weight at least W, for d smallest possible.
Complementary: Find a subgraph T such that $V \setminus T$ induces a k-subgraph, and the weight of at most f times that of a minimum such subgraph, for f smallest possible.

The primal approximation does not suffice here, or in the abovementioned problems. For instance, suppose we are given a 3-colorable graph G with all $w_v = 0.2$. Then a 1.1-approximate 3-colorable subgraph still leaves $0.1n$ vertices uncolored, for RBC cost of $0.02n = \Omega(n)$, while the optimal solution has cost 3. Instead, we need an approximation of the dual objective, which has unfortunately proved difficult.

We develop here a *bicriteria* approximation in terms of the dual and the complementary measures. We say that a vertex set S is a (t, s)-*approximation* to k-MCS if it is a tk-subgraph and $w(V \setminus S) \leq s \cdot w(V \setminus S^*)$, where S^* is a maximum k-subgraph. Namely, it gives a subgraph that requires t times as many colors, and leaves behind up to s times the weight left by the optimal solution.

Theorem 1. *There is an algorithm that, given a perfect graph G and integers k and t, yields a $(t, \frac{t}{t-1})$-approximation to k-MCS.*

Proof. Let an s-clique refer to an unweighted clique, i.e. a set of s mutually adjacent vertices. Consider the following strategy:

```
Let G' = G and w'_v = w_v for each vertex v.
i ← 1
while there exists a t·k+1-clique C_i in G' do
   Let w_i = min_{u∈C_i} w'_u.
   Let w'_v ← w'_v − w_i, for each v ∈ C_i.
   Remove all vertices v with w'_v = 0 from G'.
   i ← i + 1
od
Let S be the remaining set of vertices in G'.
Output G[S].
```

Note that since there exists no $tk+1$-clique in $G[S]$, and G is perfect, the resulting subgraph $G[S]$ is tk-colorable, establishing the first part of the claim.

Since $w(S) \geq w'(S)$, the weight outside S is at most the weight deleted in the loop iterations, or

$$w(G \setminus S) = w(G) - w(S) \leq w(G) - w'(S). \tag{1}$$

The weight reduced from the cliques in G' in each round are evenly spread over the $t \cdot k + 1$ vertices; thus, at most $1/t$-fraction can belong to any k-subgraph, including a maximum weight k-subgraph S^*. Hence, at least a $(t-1)/t$-fraction of the weight comes from outside S^*. Thus,

$$w(G) - w'(S) = \sum_i w_i(C_i) \leq \frac{t}{t-1}\left[w(G) - w(S^*)\right] = \frac{t}{t-1} w(G \setminus S^*).$$

Combined with (1), we have the second part of the claim.

This is a tight bound for this approach, as can be seen by adding to any k-colorable graph a collection of $t \cdot k + 1$-cliques, along with a single $t \cdot k$-clique.

A generalization of this argument can be useful in some cases. It suffices to change only the loop condition of the algorithm of the previous proof to read "while the approximation algorithm finds a $2k$-clique". In particular, we obtain a $(4, 2)$-approximation for circular arc graphs, and $(2k, 2)$-approximation of intersection graphs of k-hypergraphs (ones with maximum edge size k).

Theorem 2. *Let $\mathcal{G}$ be a hereditary class of graphs. Suppose there is an algorithm that given number s and a graph in $\mathcal{G}$ either returns a clique of size s or a coloring of size ρs. Then, there is a $(2\rho, 2)$-approximation of k-MCS in $\mathcal{G}$.*

Repeatedly finding large independent sets is a natural approach. While it does not give a constant factor approximation, it can be used to get some non-trivial bounds for hard classes of graphs. The following lemma is a slight strengthening of an argument made numerous times before (see, e.g., [14]).

Lemma 1. *Suppose the maximum independent set (MIS) problem can be approximated within a factor of* ρ *on a hereditary class of graphs. Then, there is a* $(\rho \log n, 1)$*-approximation of* k*-MCS. Further, if* $\rho = n^{\Omega(1)}$*, then there is a* $(O(\rho), 1)$*-approximation.*

3 Rent-or-Buy Coloring (and TTR)

It can be quickly verified that ordinary colorings can be far off the mark under the Rent-or-Buy measure. An optimal coloring can leave all colors balanced, for a unit cost per color, while by using more colors, we may only need a single large color class, with the rest in small, cheap classes.

Another approach was used for *max coloring*, where the vertex set was first partitioned into weight classes [19]. However, this would reduce to ordinary coloring in the case of uniform weights, which again would not be sufficient. Thus, a different approach is needed for RBC.

3.1 Exact Algorithms for Interval graphs

The following result shows that RBC is closely related to a well-known optimization problem. A proof of a more general result is given in Lemma 2.

Theorem 3. *Let* G *be a graph, and suppose we can compute a maximum weighted* k*-colorable subgraph in* G*, for any* k*. Then, we can solve RBC in polynomial time.*

Corollary 1. *RBC is polynomially solvable on interval, comparability, and bipartite graphs, as well as partial* k*-trees.*

We now give an alternative flow formulation of k-MCS problem on interval graphs, which allows for additional constraints on the remaining subgraph. We call a vertex set $S \subset V$ a (k, h)-*subgraph* if it is a k-subgraph and $V \setminus S$ is an h-subgraph. The (k, h)-MCS problem is that of finding a maximum weight (k, h)-subgraph. Observe that a maximum weight k-subgraph is also a maximum weight (k, h)-subgraph, for some $0 \leq h < k$.

Theorem 4. *Let* G *be an interval graph and* k *and* h *be given. Then, a maximum weight* (k, h)*-subgraph can be computed in time* $O((k + h)n \log n)$*.*

Proof. We modify the construction of [1]. Recall that an interval graph can be represented as a linearly ordered set of maximal cliques $C_1, \ldots, C_t$ of sizes $q_1, q_2, \ldots, q_t$. Let R be $k + h$. We assume that $q_i \leq R$ for every $i = 1, \ldots, t$ since otherwise G has no (k, h)-subgraph.

Construct a directed network $H = (V, E)$ with vertices $v_0, \ldots, v_t$. There is an edge (v_{i-1}, v_i) of capacity $R - q_i$ and weight 0, for each $i = 1, \ldots, t$. We call these *dummy edges*, and let E_1 denote the set of these in H. Also, for each vertex v of weight w_v that is contained in cliques $C_j, C_{j+1}, \ldots C_{j+\ell}$, add an edge to H

from v_{j-1} to $v_{j+\ell}$ of capacity 1 and weight w_v. We call these edges *subgraph edges*, and let E_2 denote the set of these in H. This completes the construction. Observe that subgraph edges used by a 1-flow from v_0 to v_t in H correspond to vertices in an independent set of G. Hence a k-flow in H gives a k-subgraph of the same weight in G.

Now we show that a k-flow exists in H, and after removing the k-flow, H still has an h-flow. This implies that we can obtain a maximum weight (k, h)-subgraph of G by computing a maximum weight k-flow in H.

Let $\delta^+(v_i)$ (resp., $\delta^-(v_i)$) denote the set of edges in H leaving (resp., entering) v_i. For a set F of edges, let $c(F)$ denote the sum of capacities of those in F. In H, subgraph edges in $\delta^+(v_i)$ correspond to vertices v in G such that $v \notin C_i$ and $v \in C_{i+1}$. Similarly, subgraph edges in $\delta^-(v_i)$ correspond to vertices v in G such that $v \in C_i$ and $v \notin C_{i+1}$. Hence $c(\delta^+(v_i) \cap E_2) - c(\delta^-(v_i) \cap E_2) = q_{i+1} - q_i$ holds for each $i = 1, \ldots, t-1$. Since $c(\delta^+(v_i) \cap E_1) = R - q_{i+1}$ and $c(\delta^-(v_i) \cap E_1) = R - q_i$, we can observe that $c(\delta^-(v_i)) = c(\delta^+(v_i))$ for each $i = 1, \ldots, t-1$. By the construction of H, $c(\delta^+(v_0)) = c(\delta^-(v_t)) = R$ also hold. Therefore, we can observe that H has a k-flow from v_0 to v_t. After removing a k-flow from H, $c(\delta^-(v_i)) = c(\delta^+(v_i))$ still holds for each $i = 1, \ldots, t-1$, and $c(\delta^+(v_0)) = c(\delta^-(v_t)) = R - k = h$. Hence we can still push an h-flow.

The number of vertices and edges in H is linear in n, the number vertices in G. Each flow increase can be obtained in the time required for a shortest-path computation in the residual graph [16].

Observe that in the time spent to compute the flow, we actually obtain a series of values (k_j, h_j) for each $k_j + h_j = R$. Also, observe that a maximum weight (k, h)-subgraph problem is solvable in bipartite graphs, since in this case trivially $k = 1$.

Theorem 5. *TTR is polynomially solvable on interval and bipartite graphs.*

Proof. Observe that the two-tiered rent cost of an independent set without opening costs can be viewed as selecting the smaller of two linear functions:

$$f(I) = c_2 w(I) + \min((c_1 - c_2) \cdot w(I), y_0),$$

where $y_0 = (c_1 - c_2)T$. Thus, the cost of the coloring C can be represented as $c_2 w(G) + \sum_{I \in C} \min((c_1 - c_2) \cdot w(I), y_0)$. Thus, it is equivalent to RBC after scaling the weights by a factor of $y_0/(c_1 - c_2)$, and adding $c_2 \cdot w(G)$ to the objective function. The addition of constant terms to the objective function does not affect the optimization of the problem.

With opening costs, we want also to minimize the number of colors used on the non-full color classes. We therefore seek a k-subgraph, with the right value of k, whose remaining graph can be colored with few colors. Hence, it suffices to try all maximum (k, h)-subgraphs, for all k and h.

3.2 Approximation of Perfect Graphs

Lemma 2. *Suppose we have a (t, t)-approximation algorithm for k-MCS. Then, we can approximate RBC within a factor of t.*

Proof. Let k' be the number of colors with weight at least 1 in an optimal RBC coloring and S^* be the set of vertices in those colors. The cost of the optimal solution is then $k' + w(V \setminus S^*)$.

Let S be a (t,t)-approximate solution to k'-MCS. Then, if we color S using at most $t \cdot k'$ colors, and the remaining vertices arbitrarily, we get a coloring of cost at most $t \cdot k' + w(V \setminus S) \leq t(k' + w(V \setminus S^*))$. Thus, we have a performance ratio of t.

By Theorem 1, we get a 2-approximation of RBC, but it applies more generally to TTR.

Corollary 2. *TTR, with non-negative costs, is* 2*-approximable on perfect graphs, even with opening costs.*

Proof. Recall from Theorem 5 that TTR without opening costs is equivalent to RBC after scaling. With opening costs, we want also to minimize the number of colors used on the non-full color classes. The subgraph found in Theorem 1 is trivially $\chi(G)$-colorable, and if we color the remaining graph optimally, we use at most $2\chi(G)$ colors in total. Thus, our opening costs are at most twice that of any coloring.

3.3 Hardness and Approximation of General and Split Graphs

For general graphs, we can obtain a bound using Lemma 1, which matches the best approximation factor known for the ordinary graph coloring problem [14].

Corollary 3. *Let ρ_{IS} be the best possible approximation ratio of MIS on general graphs. Then, RBC and TTC can be approximated within a factor of $O(\rho_{IS})$. In particular, they can be approximated within $O(n(\log\log n)^2/\log^3 n)$ [8].*

RBC is clearly equivalent to Graph Coloring when all $w_v = 1$. Therefore, as a more general problem, it inherits all the hardness characteristics. However, one may still ask how hard the problem is for other vertex weights. For instance, the problem is trivial when $w(G) \leq 1$, since any coloring has then the same cost. From the results of Feige and Kilian [9], that were derandomized by Zuckerman [22], we have the following.

Observation 6 *RBC is NP-hard to approximate within a $\min(n, w(G)/n^\epsilon)$-factor, and is trivially $w(G)$-approximable.*

Essentially the same reduction from X3C (exact 3-set cover) as used on related problems [5, 13] shows the hardness of RBC on split graphs, a subclass of chordal graphs.

Theorem 7. *RBC is strongly NP-hard on split graphs with uniform weights.*

Proof. Let $X = \{s_1, s_2, \ldots, s_{3m}\}$ be a finite set and $T = \{e_1, \ldots, e_n\}$ be a set of triples, that is an input to X3C. Form a graph with vertex set $X \cup T$, where X is independent, T is a clique, and (s_i, e_j) is an edge iff $s_i \notin e_j$. Assign each vertex the weight $w = 1 - 1/(2n)$. Then, any coloring of cost less than n uses only the minimum n colors, with each e_i in a diferent class. The cost of such a coloring is $n - (n-t)/(2n)$, where t is the number of colors that contain more than one vertex. Thus, the minimum cost of an RBC coloring is $n-(n-m)/(2n)$ iff (X, T) admits a cover with m sets iff (X, T) admits an exact cover.

This is complemented with a polynomial time approximation scheme (PTAS).

Theorem 8. *RBC admits a PTAS on split graphs.*

Proof. Let (U, V, E) be a split graph with independent set U and clique V. Let $\epsilon > 0$ be given and let $k = 1/\epsilon$. Initially, assign each node in V to a different color. Try for each subset $S \subset V$ of size at most k the following: For each node $u \in N(S) = \{u \in U : \exists v \in S, (u, v) \notin E\}$, assign u to the color of some non-neighbor in S. Color the rest of U in a separate color.

Consider an optimal RBC coloring C, and let S^* be the set of nodes from U that participates in color classes of C of cost 1 (i.e., of weight at least 1). If $|S^*| \leq k$, then our solution is optimal when we try $S = S^*$. Otherwise, it holds for any $S \subset S^*$ of cardinality k that the cost of C is $OPT \geq |S^*| + w(U \setminus S^*) \geq k + w(U \setminus S)$. For such a set S, the cost of the algorithm is at most

$$|S| + w(U \setminus S) + w(V \setminus N(S)) \leq (k+1) + w(U \setminus S) \leq (1+1/k)OPT = (1+\epsilon)OPT.$$

Hence, one of the solutions found is within $1 + \epsilon$ factor of an optimal RBC coloring of the graph.

4 Probabilistic Coloring Problem

One of the useful features of Rent-or-Buy is that its colorings closely approximate Probabilistic colorings. This is helpful, since RBC is much more amenable to computation.

Theorem 9. *Let C be a coloring of a graph G with vertex weights $p_v \in (0, 1]$. Let $f_{RB}(C)$ ($f_{Pr}(C)$) be the cost of C under the Rent-or-Buy measure (the probabilistic coloring measure), respectively. Then, $f_{RB}(C) \geq f_{Pr}(C) \geq (1-1/e)f_{RB}(C)$.*

Proof. The first inequality follows from the definitions of the measures, that $P(I) \leq W(I)$, since by inclusion-exclusion it holds that $\prod_v (1-p_v) \geq 1 - \sum_v p_v$.

Let I be a color class under C. The cost $P(I)$ of I under the probabilistic measure is given by

$$P(I) = 1 - \prod_{v \in I}(1 - p_v) \geq 1 - \prod_{v \in I} e^{-p_v} = 1 - e^{-w(I)}.$$

If $w(I) \geq 1$, then $f_{RB}(I) = 1$ and we have that $P(I) \geq 1 - e^{-1} = 1 - 1/e$. Otherwise, $f_{RB}(I) = w(I)$. Observe that the function $(1 - e^{-x})/x$ is decreasing in the interval $(0, 1]$. Hence, the ratio is maximized for $w(I) = 1$. Since the ratio holds for each color class individually, it also holds for the sum of the color classes.

These bounds are best possible. An independent set of weight 1 can consist of a single node of weight 1, or n nodes of weight $1/n$ each. In both cases, the RBC cost is the same, while the probabilistic measure results in cost of 1, in the former case, and $1 - 1/e + O(1/n)$, in the latter case.

Theorem 9 immediately implies that RBC and Probabilistic coloring have the same approximation behavior, within this factor of 1.582.

Corollary 4. *If RBC is ρ-approximable on a graph G, then Probabilistic coloring is approximable within a factor of $\rho \cdot \frac{e}{e-1} \leq 1.582\rho$ on G.*

Combining this with our bounds on RBC of Corollaries 1 and 2 and , 3, and Theorem 8, we obtain the following improved bounds on Probabilistic coloring.

Theorem 10. *Probabilistic coloring is approximable within* 1.582 *on interval and comparability graphs,* 3.164 *on perfect graphs,* 1.583 *on split graphs,* $n^{0.2111}$ *on 3-colorable graphs [2], and* $O(n(\log \log n)^2/\log^3 n)$ *on general graphs.*

5 Threshold Coloring

We note that neither finding an ordinary coloring nor repeatedly finding a maximum independent set leads in general to constant factor approximation. Instead, one can treat the two costs separately. We use the solution of another MCS-variant to obtain a better approximation.

Let *Pre-(k,h)-MCS* be a variation of (k, h)-MCS where we are additionally given a set P that is required to be a part of the k-subgraph solution. This is similar to precoloring-extension problems, except the actual colors of the nodes in P are not prespecified.

Lemma 3. *Pre-(k,h)-MCS can be solved in $O(\chi(G) n \log n)$ time on an interval graph G.*

Proof. Use the construction of Theorem 4, and change only the weight of the edges corresponding to vertices in P to a small enough value. The flow paths will now never use those edges, if at all possible.

Recall that we can compute the solutions for all values of k and h in only $\chi(G)$ flow computation, for a total time of $O(\chi^2(G) n \log n)$.

Theorem 11. *Threshold coloring can be approximated within a factor of 2 on interval graphs.*

Proof. For each k and h, try the following approach, and use the one of smallest cost. Compute, using the above lemma, the optimal (k, h)-subgraph containing, as a preselected set P, the set of nodes with weight at least the threshold T. Then, color the remaining h-colorable subgraph, and divide each class of size s into $\lceil s/T \rceil$ classes of size at most T each.

Suppose the optimal solution used k_0 color classes of the larger cost c_2, leaving a h_0-colorable subgraph of size L to be covered with classes of cost c_1. Then, the cost of that solution is at least $k_0 \cdot c_2 + \max(h_0, \lceil L/T \rceil) \cdot c_1$. For these values of $k = k_0$ and $h = h_0$, our solution uses k_0 expensive classes, and colors a subgraph of total weight at most L with the inexpensive classes. At most h_0 of those classes are less than full, and at most $\lfloor L/T \rfloor$ full. Hence, the total cost of the algorithm solution is at most $k_0 \cdot c_2 + (h_0 + \lfloor L/T \rceil) \cdot c_1$, or at most twice the optimal.

Theorem 12. *Threshold coloring can be approximated within a factor of $\rho \leq 3.7$ on perfect graphs.*

The proof is deferred to the appendix for lack of space.

References

1. E. M. Arkin and E. B. Silverberg. Scheduling jobs with fixed start and end times. *Disc. Applied Math.* 18:1–8, 1987.
2. S. Arora, E. Chlamtac. New approximation guarantee for graph coloring. *STOC*, 2006.
3. H. Bodlaender, K. Jansen. Restrictions of graph partition problems. Part I. *Theoretical Computer Science* 148:93–109, 1995.
4. J. Cardinal, S. Fiorini, G. Joret. Minimum entropy coloring. ISAAC 2005: 819–828. 2006.
5. F. Della Croce, B. Escoffier, C. Murat, and V. Th. Paschos. Probabilistic coloring of bipartite and split graphs. ICCSA 2005: 202–211.
6. L. Epstein, M. M. Halldórsson, A. Levin, and H. Shachnai. Weighted Sum Coloring in Batch Scheduling of Conflicting Jobs. In Proc. of *APPROX-RANDOM* 2006, 116–127.
7. B. Escoffier, J. Monnot, V. Th. Paschos. Weighted Coloring: Further complexity and approximability results. *Inf. Process. Lett.* 97(3): 98–103, 2006.
8. U. Feige. Approximating Maximum Clique by Removing Subgraphs. *SIAM J. Discrete Math.* 18(2): 219–225, 2004.
9. U. Feige and J. Kilian. Zero knowledge and the chromatic number. *JCSS*, 57:187–199, October 1998.
10. F. V. Fomin, D. Kratsch, J.-C. Novelli. Approximating minimum cocolorings. *Inf. Process. Lett.* 84(5): 285-290, 2002.
11. A. Frank. On chain and antichain families of a partially ordered set. *Journal of Combinatorial Theory Series B*, 29:176–184, 1980.
12. T. Fukunaga, M. Halldórsson, H. Nagamochi. Robust cost colorings. Manuscript, 2007.
13. D. Gijswijt, V. Jost, and M. Queyranne. Clique partitioning of interval graphs with submodular costs on the cliques. EGRES Technical Report 2006-14, `www.cs.elte.hu/egres`, 2006.

14. M. M. Halldórsson. A still better performance guarantee for approximate graph coloring. *Inform. Process. Lett.*, 45:19–23, 25 January 1993.
15. M. M. Halldórsson, G. Kortsarz, and H. Shachnai. Sum coloring interval and k-claw free graphs with application to scheduling dependent jobs. *Algorithmica* **37**:187–209, 2003.
16. M. Iri. Network Flow, Transportation, and Scheduling: Theory and Algorithms. Academic Press, 1969.
17. K. Jansen. Approximation Results for the Optimum Cost Chromatic Partition Problem. J. Algorithms 34:54–89, 2000.
18. C. Murat, V. Th. Paschos. On the probabilistic minimum coloring and minimum k-coloring. *Disc. Appl. Math.*, 154:564–586, 2006.
19. S. V. Pemmaraju and R. Raman. Approximation Algorithms for the Max-coloring Problem. In Proc. of *ICALP* 2005, 1064–1075.
20. A. Saha, M. Pal. Maximum weight k-independent set problem on permutation graphs. *Int. J. Comput. Math.* 80(12): 1477-1487, 2003.
21. M. Yannakakis and F. Gavril. The maximum k-colorable subgraph problem for chordal graphs. *Information Processing Letters*, 24(2):133–137, 1987.
22. D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on the Theory of Computing*, pages 681–690, 2006.

Proof of Theorem 12

Proof. Let us denote by $R = c_2/c_1$ the ratio between the two costs. For simplicity, let us scale the costs so that $c_1 = 1$. Observe that if $R \leq 3.7$, then using an optimal graph coloring yields an R-approximation for Threshold coloring. Thus, we assume that $R \geq 3.7$.

We first find an optimal graph coloring of the subgraph induced by vertices of weight at least the threshold T. Since the optimal solution needs also to color these vertices in classes of the same cost, the cost of these classes is at most OPT, the cost of the optimal solution.

On the remaining graph G', we try for each value of k the following approach and retain the cheapest solution. Let $t = 2.4293$. Find a $(t, t/(t-1))$-approximate k-MCS by Theorem 1, and color the $t \cdot k$-subgraph with classes of cost R. Then, find an optimal graph coloring of the remaining subgraph, and divide each color into the fewest possible classes of size at most T.

Suppose the optimal solution used k_0 color classes of the larger cost R, leaving a subgraph of size L to be covered with classes of cost 1. That subgraph required at least $\chi(G) - k_0$ colors, and also needed at least $\lceil L/T \rceil$ classes of the small size. Hence, the cost of the optimal solution is $OPT \geq k_0 \cdot R + \max(\chi(G) - k_0, \lceil L/T \rceil)$. For this value of $k = k_0$, our solution used $t \cdot k_0$ expensive classes, and colored a subgraph of total weight at most $t/(t-1) \cdot L$ with the inexpensive classes. At most $\chi(G)$ of those classes were less than full, and at most $\lfloor t/(t-1) \cdot L/T \rfloor$ were full. Hence, the cost of the algorithm's solution is at most

$$OPT + t \cdot k_0 \cdot R + t/(t-1) \cdot L/T + \chi(G).$$

Rewrite this as the sum of three terms: OPT, $t/(t-1) \cdot (k_0 \cdot R + L/T)$, and $[t - t/(t-1)]R \cdot k_0 + \chi(G)$. The first two terms are at most $1 + t/(t-1) \leq 2.6997$ times OPT. We can also verify as follows that the last term is at most $(R-1) \cdot k_0 + \chi(G) \leq OPT$. Namely, $[t - t/(t-1)] \cdot R \leq R - 1$ is equivalent to $R \cdot (1 - t + t/(t-1)) \geq 1$, which holds for $R \geq 3.7$ since $1 - t + t/(t-1) > 1/3.6991$.