Improved Bounds for Sum Multicoloring and Scheduling Dependent Jobs with Minsum Criteria

Rajiv Gandhi* Magnús M. Halldórsson† Guy Kortsarz† Hadas Shachnai‡

Abstract

We consider a general class of scheduling problems where a set of *dependent* jobs needs to be scheduled (preemptively or non-preemptively) on a set of machines so as to minimize the weighted sum of completion times. The dependencies among the jobs are formed as an *arbitrary* conflict graph. An input to our problems can be modeled as an instance of the *sum multicoloring* (SMC) problem: Given a graph and the number of colors required by each vertex, find a proper multicoloring which minimizes the sum over all vertices of the largest color assigned to each vertex. In the *preemptive* case (pSMC), each vertex can receive an arbitrary subset of colors; in the *non-preemptive* case (npSMC), the colors assigned to each vertex need to be contiguous. SMC is known to be no easier than classic graph coloring, even in the case of unit color requirements.

Building on the framework of Queyranne and Sviridenko (*J. of Scheduling, 5:287-305, 2002*), we present a general technique for reducing the sum multicoloring problem to classical graph multicoloring. Using the technique, we improve the best known results for pSMC and npSMC on several fundamental classes of graphs, including line graphs, $(k+1)$-claw free graphs and perfect graphs. In particular, we obtain the first constant factor approximation ratio for npSMC on interval graphs, on which our problems have numerous applications. We also improve the results of Kim (*SODA 2003, 97–98*) for npSMC of line graphs and for resource-constrained scheduling.

*Department of Computer Science, Rutgers University, Camden, NJ 08102. E-mail: {rajivg,guyk}@camden.rutgers.edu.

†Department of Computer Science, University of Iceland, IS-107 Reykjavik, Iceland. E-mail: mmh@hi.is.

‡Department of Computer Science, The Technion, Haifa 32000, Israel. E-mail: hadas@cs.technion.ac.il. Part of this work was done while the author was on leave at Bell Laboratories, Lucent Technologies, 600 Mountain Ave., Murray Hill, NJ 07974.

1 Introduction

We consider a general class of problems in which jobs that utilize non-sharable resources need to be scheduled (preemptively or non-preemptively) on multiple machines. Scheduling any job j depends on whether another job sharing resources with j is being scheduled. The dependencies among the jobs are modeled by an *arbitrary* conflict graph, in which the vertices represent the jobs, and an edge between two vertices means that the corresponding jobs cannot be scheduled simultaneously. Then the problem of scheduling dependent jobs can be formulated as a coloring problem: a proper coloring of the conflict graph partitions the set of jobs to subsets of non-conflicting jobs. Thus, when all jobs have the same (unit) execution time, we get a graph *coloring* problem, and when the execution times are arbitrary, we get a graph *multicoloring* problem.

In this work, we focus on the *sum of completion times* measure. For unit-length jobs, this is known as the *chromatic sum* or *sum coloring* (SC) of the conflict graph. Let $G = (V, E)$ be the conflict graph. Given a coloring ψ of G, the sum coloring of ψ is given by $\mathtt{SC}(G, \psi) = \sum_v \psi(v)$. The minimum chromatic sum of G is given by $\mathtt{SC}(G) = \min_\psi \mathtt{SC}(G, \psi)$. In the weighted case, each vertex v has a weight, w_v, and we need to minimize $\sum_v w_v \psi(v)$ over all proper colorings.

An instance of a multicoloring problem is a pair (G, x), where $G = (V, E)$ is a graph, and x is a vector of *color requirements* (or *lengths*) of the vertices. A *multicoloring* of G is an assignment $\psi : V \to 2^{\mathbf{N}}$, such that each vertex $v \in V$ is assigned a set of x_v distinct colors, and adjacent vertices receive non-intersecting sets of colors. Denote by $f_v(\psi) = \max_{i \in \psi(v)} i$ the largest color assigned to v by a multicoloring ψ. The *sum multicoloring* (SMC) of ψ on G is $\mathtt{SMC}(G, \psi) = \sum_{v \in V} f_v(\psi)$. The SMC problem is to find a multicoloring ψ, such that $\mathtt{SMC}(G, \psi)$ is minimized. In the weighted case, we want to minimize $\sum_{v \in V} w_v f_v(\psi)$, over all proper multicolorings ψ. When all the color requirements are equal to 1, the problem reduces to SC. A multicoloring, ψ, is called *non-preemptive* if the colors assigned to each vertex v are contiguous, i.e., if for any $v \in V$, $(\max_{i \in \psi(v)} i) - (\min_{i \in \psi(v)} i) + 1 = x_v$. We denote this version of the problem by npSMC; the preemptive problem, where each vertex v can receive *any* set of x_v colors, is denoted by pSMC.

Scheduling dependent jobs, and the resulting variants of the sum (multi)coloring problem, have numerous applications, in particular on interval graphs. The following practical scenarios yield instances of our problems on this natural subclass of graphs.

Session scheduling on a path: In a path network, pairs of nodes need to communicate, for which they need use of the intervening path. If two paths intersect, the corresponding sessions cannot be held simultaneously. In this case, it would be natural to expect the sessions (i.e., "jobs") to be of different lengths, leading to the sum multicoloring problem on interval graphs.

Storage allocation: Storage allocation in a warehouse involves minimizing the total distance traveled by a robot [W97]. Goods are checked in and out at known times; thus, goods that are not in the warehouse at the same time can share the same location. We represent each of the goods by an interval on the line, which gives the time interval in which it is available at the warehouse. Numbering the storage locations by their distance from the counter, the total distance corresponds to sum coloring the intervals formed by the goods.

VLSI design: In the wire-minimization problem [NSS99], terminals lie on a single vertical line (each terminal is represented by an interval on this line), and with unit spacings are vertical bus

lanes. Pairs of terminals are to be connected via horizontal wires on each side to a vertical lane, with non-overlapping pair utilizing the same lane. With the vertical segments fixed, the wire cost corresponds to the total length of horizontal segments. Numbering the lanes in increasing order of distance from the terminal line, lane assignment to a terminal corresponds to coloring the terminal's interval by an integer. The wire-minimization problem then corresponds to sum coloring an interval graph.

Other applications of sum (multi)coloring include traffic intersection control, session scheduling in local-area networks and compiler design (a comprehensive survey appears in [BHK$^+$00]). Instances of SMC on line graphs and, more generally, on $(k+1)$-claw free graphs, are derived mainly from applications that involve resource constrained scheduling. Our results apply also to permutation graphs, which model, e.g., train scheduling problems.

1.1 Our Results

We present (in Section 2.1) a general technique for reducing SMC to the classic graph (multi)coloring problem. Using the technique, we improve the best known results for pSMC and npSMC on several fundamental classes of graphs, including line graphs, $(k+1)$-claw free graphs and perfect graphs. In particular, we obtain the first constant factor approximation ratio for npSMC on interval graphs. Our improved bound of 7.682 for npSMC of line graphs is achieved by a simple greedy algorithm (see in Section 3.1). The previous best ratio of 10, achieved by an algorithm of Kim [K-03], involved solving an LP with an exponential number of constraints.

While our main focus is on minimizing the sum of completion times of the *jobs*, our technique can be applied to other minsum optimization problems, such as *resource constrained scheduling (RCS)*. In RCS, we have a set of jobs, each requesting up to k resources; jobs that need to utilize the same resource cannot be processed simultaneously. We say that a resource has completion time i if the last job utilizing this resource completes at time i. Our goal is to find a non-preemptive schedule that minimizes the sum of completion times of all the *resources*. We show (in Section 4) that our technique yields an approximation ratio of $2e \cdot k \approx 5.437k$. This improves the best ratio known of $8k-7$ given in [K-03], for any $k \geq 3$.

For simplicity, in formulating our results it is implicitly assumed that the number of machines is "unbounded". The technique can, however, be applied in a system with any given number of machines, with slightly weaker performance ratios. Due to space constraints, we relegate this description to Appendix A. Also, we formulate our results for the unweighted case, and show (in Section 4) how to generalize the results for the weighted versions of the problems.

Relation to Min-sum Set Cover Our results include an approximation ratio of 3.591 for sum coloring of perfect graphs. This improvement upon the previous ratio of 4 (of [BBH$^+$98]) is of particular interest, due to the relation of SC to the *min-sum set cover* problem. The input to min-sum set cover consists of a universe $\mathcal{U}$ and a collection of subsets $\mathcal{S} = \{S_i\},\ S_i \subseteq \mathcal{U}$. A feasible solution is an ordered sub-collection of subsets $\mathcal{S}' = \{S'_1, S'_2, \ldots\}$, such that $\bigcup_i S'_i = \mathcal{U}$. We say that $u \in \mathcal{U}$ has *cover time* i if S'_i is the first subset in the order of $\mathcal{S}'$ to include u. The goal is to minimize the sum of cover times over all the elements of $\mathcal{U}$. Feige *et al.* [FLT-02] showed that min-sum set cover admits a 4-approximation and that, unless P=NP, for any constant $\epsilon > 0$, there is no $(4-\epsilon)$-approximation. Observe that SC is a special case of min-sum set cover, in which $\mathcal{S}$ is the

collection of all independent sets in G. Hence, our 3.591-approximation implies that the min-sum set cover problem in its full generality is provably harder to approximate than SC on perfect graphs.

Techniques Our general approximation technique builds on the framework of Queyranne and Sviridenko [QS-02] for scheduling jobs with release times on parallel machines. As in [QS-02], we divide the time line into intervals of geometrically increasing size (see also [HSW-96, HSSW-97]), using randomized starting points (as introduced in [CP$^+$96]), and approximate the classic makespan problem on each block. Note, however, that the results in [QS-02] do not apply to *arbitrary* conflict graphs. The class of problems studied in [QS-01, QS-02] include shop scheduling (open shop and job shop) and entail a different optimization criteria than SMC. (As shown in [GHKS04], open shop scheduling is in fact a special case of the data migration problem [K-03].)

1.2 Related Work

The SC problem was introduced in [K89] and the SMC problems in [BHK$^+$00]. Table 1 summarizes the known results for SC, pSMC and npSMC in various classes of graphs. New bounds given in this paper are shown in boldface. In each of these entries, we give in parenthesis the previous best known bound for the problem. Entries marked with $\cdot$ follow by inference, either by using containment of graph classes (interval graphs are perfect), or by SC being a special case of SMC. When omitted, [BBH$^+$98] is the references for SC and [BHK$^+$00] for SMC. Also, in the table below, c represents some constant.

	SC		**SMC**	
	u.b.	*l.b.*	pSMC	npSMC
General graphs	$\cdot$	$n^{1-\epsilon}$	$n/\log^2 n$	$n/\log n$
Perfect graphs	**3.591** (4)	$c > 1$ [BK98]	**5.436** (16)	$O(\log n)$
Interval graphs	1.796 [HKS03]	$c > 1$ [G01]	**5.436** (7.184)	**11.273** ($O(\log n)$)
Bipartite graphs	27/26 [G$^+$02]	$c > 1$ [BK98]	1.5	2.8
Planar graphs	PTAS [HK02]	NPC [HK02]	PTAS [HK02]	PTAS [HK02]
Trees	1 [K89]		PTAS [HKP$^+$03]	1 [HKP$^+$03]
$k+1$-claw free	k		k	**1.796k^2+.5** ($4k^2$–$2k$) [HKS03]
k-sets	k		k	**3.591k+.5** ($6k$–2) [K-03]
Line graphs	2	NPC	2	**7.682** (10) [K-03]

Table 1: Known results for sum (multi-)coloring problems

There is a wide literature on parallel machine scheduling with the objective of minimizing the sum of completion times. These works generally deal with scheduling *independent* jobs, or allow for *precedence constraints* which are directed dependencies. The undirected conflict graphs considered here require quite different treatment.

Some work has been done on resource-constrained scheduling. Kubale [K-96] studied the complexity of scheduling biprocessor tasks. They also investigate special classes of graphs, and showed that npSMC of line graphs of trees is NP-hard in the weak sense. Afrati et al. [AB$^+$00] gave a polynomial time approximation scheme for the problem that we consider, minimizing sum of completion times of dedicated tasks. However, their method applies only to the case where the total

number of processors is a fixed constant. Coffman et al. [CG$^+$85] analyzed the makespan version of npSMC of line graphs, which arises in the *file transfer problem*. They showed that a class of greedy algorithms yields a 2-approximation and gave a $(2+\epsilon)$-approximation for a version with more general resource constraints. Kim [K-03] gave an LP formulation of the npSMC problem on line graphs and intersection graphs of k-sets,[1] improving the earlier bounds of [HKS03]. The paper presents also a ratio of $8k-7$ for the RCS problem with k resources.

2 Sum Multicoloring via Makespan Approximations

In this section we describe and analyze our main approximation technique. Later, we show how to obtain our results by applying the general technique to specific classes of graphs, and to the different variants of the sum multicoloring problem that we consider here.

2.1 Algorithms and Implementation

Our technique uses two components: (i) a lower bound, f_v^*, on the completion time of the vertex v in an optimal solution, for any $v \in V$; a parameter $d \geq 1$, which indicates how well the lower bound captures the optimal value; (ii) a (makespan) multicoloring algorithm $\mathcal{A}$ with performance ratio ρ, for some $\rho \geq 1$.

Given the f_v^* values, the algorithm schema, ALG, breaks the time line (or the color sequence $1, 2, \ldots$) into intervals. We use in the partition two parameters: α, chosen uniformly at random from $[0, 1)$, and a constant $\beta > 1$ (to be optimized). Let $c_k = \beta^{\alpha+k}$, for $k = 0, 1, \ldots, L$, where $c_L \geq \max_v x_v$. The intervals induce a partition of the graph into *blocks* $V_\ell = \{v \in V : c_{\ell-1} < f_v^* \leq c_\ell\}$, $\ell = 1, \ldots, L$, of vertices whose completion times (f_v^*) fall in the respective interval. We then apply the makespan multicoloring algorithm on each block in sequence. We show that when this is possible, our algorithm attains a ratio of $d \cdot e\rho \approx 2.718d\rho$ for pSMC, $1.796d\rho + 0.5$ for npSMC, and $1.796d\rho$ for SC.

The lower bounds, f_v^*, can be obtained either by solving a linear program, or by using an approximation algorithm for the preemptive sum multicoloring problem. This results in two algorithms described below. As shown in Section 2.2, we can unify the analyses of the two algorithms, once we guarantee that each satisfies certain properties.

LP-based Algorithm One way to obtain the f_v^* values is by solving the LP relaxation of an integer programming formulation of the problem. (Such LP relaxations have been used in the past in scheduling *independent* jobs; see, e.g., [W-85, Q-93, S-96].) Before we describe our LP-based algorithm, we give some underlying properties of this algorithm. Let OPT be the cost of an optimal solution, and $OPT^* = \sum_v f_v^*$ the total of the lower bounds f_v^*. Also, we denote by $\omega(H, x)$ the maximum weight of any clique in a subgraph H, i.e., largest sum of color requirements. For a subset U of vertices, let $x(U) = \sum_{u \in U} x_u$.

We require that the following properties be satisfied:

(P1) $OPT^* \leq OPT$.
(P2a) $\max_{v \in V_\ell} f_v^* \geq \omega(V_\ell, x)/d$, for some $d \geq 1$, for all $1 \leq \ell \leq L$.
(P2b) There is a multicoloring algorithm, $\mathcal{A}$, that approximates the makespan of any

[1] We give the precise definition in Section 3.1.

graph in the given graph class within a ρ factor of the weighted clique size, and in particular,

$$\mathcal{A}(V_\ell, x) \leq \rho \cdot \omega(V_\ell, x), \text{ for } \ell = 1, 2, \ldots, L. \tag{1}$$

We formulate sum multicoloring with an integer program that uses *linear ordering* variables (see, e.g., [P-80, HSSW-97]). For any edge $uv \in E$, there is a variable $\delta_{uv} \in \{0, 1\}$, such that $\delta_{uv} = 1$ if u precedes v in the schedule, and 0 otherwise. Let $N(v)$ denote the set of neighbors of v in G, and $C_1, \ldots, C_{N_v}$ denote the maximal cliques in $N(v)$. The constraints (2) follow from the requirement that the vertices in any clique C are assigned *disjoint* sets of colors; thus the completion time f_v of a vertex v in a clique C is at least the sum of the color requirements of the vertices in C that completed before v plus that of v itself.

$$\begin{aligned} (LP) \qquad & \text{minimize } \sum_{v \in V} f_v \\ \text{subject to:} \quad & \forall v \in V,\ 1 \leq r \leq N_v \ : \ f_v \geq x_v + \sum_{u \in C_r} x_u \delta_{uv} \\ & \forall uv \in E \ : \ \delta_{uv} + \delta_{vu} = 1 \end{aligned} \tag{2}$$

In the linear relaxation of LP, we allow f_v to take non-integral values ≥ 1. We denote by f_v^* the value of f_v in an optimal LP solution. Note that the program is equally valid for the preemptive and non-preemptive variants.

The next lemma shows that the above LP formulation satisfies property (P2a) with $d = 2$. It is based on a result of [K-03] (Lemma 2.3), attributed to [HSSW-97]. The proof is given in appendix.

Lemma 2.1 *For any* $1 \leq \ell \leq L$, $\max_{v \in V_\ell} f_v^* \geq \dfrac{\omega(V_\ell, x)}{2}$.

In particular, since $\max_{v \in V_\ell} f_v^* \leq c_\ell$, this implies that $c_\ell \geq \omega(V_\ell, x)/2$ for $\ell = 1, \ldots, L$. We now summarize the steps of the LP-based algorithm with parameters $\beta, \alpha > 1$.

Algorithm $\mathsf{ALG}_{\mathrm{LP}}$

(i) Solve the linear program LP to obtain the f_v^* values.

(ii) Partition the vertices in the graph to the blocks $V_1, V_2, \ldots$ by their f_v^* values.

(iii) Color the blocks in sequence using a non-preemptive multicoloring algorithm $\mathcal{A}$ which satisfies Property (P2b); that is, suppose that the last color used for the block V_ℓ is col_ℓ, then $\mathcal{A}$ starts coloring the block $V_{\ell+1}$ with $col_\ell + 1$.

Applying an Approximation Algorithm for `pSMC` An alternative way of obtaining the infeasible solution, f_v^*, is to use the preemptive solution when solving the non-preemptive problem. In this case, we replace (P2a) and (P2b) by the following properties.

(P2a′) There is a d-approximation algorithm for `pSMC`, for some $d \geq 1$.
(P2b′) There is non-preemptive multicoloring algorithm, $\mathcal{A}$, that approximates the makespan of any graph in the given graph class within a ρ factor of the number of colors used by a preemptive multicoloring, and in particular,

$$\mathcal{A}(V_\ell, x) \leq \rho \cdot \mathtt{pMC}(V_\ell, x), \text{ for } \ell = 1, 2, \ldots, L. \tag{3}$$

We now summarize the steps of the algorithm based on the approximation for pSMC. The algorithm gets as parameters the values $\beta, \alpha > 1$.

Algorithm $\mathsf{ALG}_{\text{PRE}}$

(i) Apply to G a d-approximation algorithm for pSMC. Let f_v^{pre} be the completion time of $v \in V$. Set for any $v \in V$, $f_v^* = f_v^{pre}/d$,

(ii) Partition the vertices in the graph to the blocks $V_1, V_2, \ldots$ by their f_v^* values.

(iii) Color the blocks in sequence using a non-preemptive multicoloring algorithm $\mathcal{A}$ which satisfies Property (P2b′).

2.2 Analysis

We use in the analysis the following notation. Recall that the *(multi)chromatic number* $\chi(G)$ of a graph G is the minimal number of colors required for (multi)coloring the vertices in G properly. In scheduling terms, this is the minimal total length (or *makespan*) of any legal schedule. We use the notation pMC, npMC for the preemptive and non-preemptive versions of this problem, respectively. Let ℓ_v denote the block into which v falls (ℓ_v is a function of α). Let t_ℓ denote the number of colors used by the multicoloring algorithm $\mathcal{A}$ on block ℓ. If we apply algorithm ALG_{LP}, then by properties (P2a) and (P2b),

$$t_\ell \leq \rho\omega(V_\ell, x) \leq \rho d c_\ell. \tag{4}$$

Similarly, if we use $\mathsf{ALG}_{\text{PRE}}$, we have that $t_l \leq \rho \cdot \texttt{pMC}(V_\ell, x) \leq \rho \max_{v \in V_\ell} f_v^{pre} = \rho d \max_{v \in V_\ell} f_v^* \leq \rho d c_\ell$. We proceed to analyze our algorithm schema, ALG, without making any assumptions on the algorithm used (i.e., the analysis applies for both ALG_{LP} and $\mathsf{ALG}_{\text{PRE}}$).

Denote by $\tilde{f}_v$ the last color (completion time) of a vertex v by our algorithm schema ALG. This color is the sum of the makespans of the colorings of the previous blocks, plus the completion time f'_v of v within the current block, i.e. $\tilde{f}_v = \sum_{r=1}^{\ell-1} t_r + f'_v$.

Bound for pSMC We first consider a general scenario, that captures, e.g., the preemptive case. We trivially bound the last color of $v \in V_\ell$ under $\mathcal{A}$ by the total number of colors used, i.e., $f'_v \leq t_\ell$. Hence, we get for each vertex independently that

$$\tilde{f}_v \leq \sum_{r=1}^{\ell} t_r \leq \frac{d \cdot \rho \beta^{\alpha+\ell+1}}{\beta - 1}, \tag{5}$$

and

$$\mathsf{ALG}(V, x) = \sum_{v \in V} \tilde{f}_v \leq d \cdot \rho \sum_{v \in V} \frac{\beta^{\alpha+\ell_v+1}}{\beta - 1} = d \cdot \rho \cdot \frac{\beta}{\beta - 1} \sum_{v \in V} c_{\ell_v}, \tag{6}$$

where ℓ_v is the block in which v is colored and c_ℓ is the largest color in block ℓ.

We now select α uniformly at random from $[0, 1)$. Then ℓ_v and c_ℓ are also random variables. The proof of the following lemma is given in the Appendix.

Lemma 2.2 *For any $\beta > 1$ and $v \in V$, $\mathbf{E}[c_{\ell_v}] = \frac{\beta-1}{\ln \beta} f_v^*$, where the expectation is over the random choices of α.*

Recall that $OPT^* = \sum_v f_v^*$. Combining (6) with Lemma 2.2 we get that

$$\mathbf{E}[\mathsf{ALG}(V,x)] \leq d\rho\frac{\beta}{\beta-1}\sum_{v\in V}\frac{\beta-1}{\ln\beta}f_v^* \leq d\rho\frac{\beta}{\ln\beta}OPT^*.$$

The function $f(\beta) = \beta/\ln\beta$ is minimized when $\beta = e \approx 2.718$. This gives the following.
Theorem 2.3 *There is a $(d \cdot e\rho)$-approximation algorithm for* `pSMC`.

Bound for `npSMC` In the non-preemptive case, we may use the schedule output by algorithm $\mathcal{A}$ for V_ℓ either directly or reversed. In the reverse order, any vertex v, whose last color is f_v, is colored with $(t_\ell - f_v + 1), (t_\ell - f_v + 2), \ldots, (t_\ell - f_v + x_v)$. By selecting the order that yields the better weighted average completion time, we may assume that on average, each job is at least half-way through completion at the half-way mark for V_ℓ. That is, on average, for any vertex $v \in V_\ell$, $f'_v \leq (t_\ell + x_v)/2$. Thus, we have

$$\begin{aligned}
\tilde{f}_v &\leq \sum_{r=1}^{\ell-1} t_r + \frac{t_\ell}{2} + \frac{x_v}{2} \\
&\leq d\cdot\rho\left(\frac{\beta^{\alpha+\ell}}{2} + \sum_{r=0}^{\ell-1}\beta^{\alpha+r}\right) + \frac{x_v}{2} \qquad (7)\\
&\leq d\cdot\rho\beta^{\alpha+\ell}\left(\frac{1}{2} + \frac{1}{\beta-1}\right) + \frac{x_v}{2} \\
&= d\cdot\rho\cdot c_\ell\left(\frac{\beta+1}{2(\beta-1)}\right) + \frac{x_v}{2} \qquad (8)
\end{aligned}$$

Combining (8) with Lemma 2.2 we have

$$\begin{aligned}
\mathbf{E}[\mathsf{ALG}(V,x)] &= \sum_{v\in V}\mathbf{E}[\tilde{f}_v] \leq d\cdot\rho\frac{\beta+1}{2(\beta-1)}\sum_v \mathbf{E}[c_{\ell_v}] + \frac{x(V)}{2} \\
&= d\cdot\rho\frac{\beta+1}{2\ln\beta}OPT^* + \frac{x(V)}{2}
\end{aligned}$$

The function $f(\beta) = (\beta+1)/\ln\beta$ is minimized when $\beta = \gamma \approx 3.59112$, for a ratio of $d\gamma\rho/2 + 0.5$.

Note that the above schema can be derandomized, by partitioning the interval $(0, 1]$ to smaller intervals; we can then search for the best value for α in these intervals, to within desired precision. We summarize in the next result.
Theorem 2.4 *There is a $(d\gamma\rho/2 + 0.5)$-approximation algorithm for* `npSMC`*, where $\gamma \approx 3.59112$.*

Deterministic and simultaneous approximation If we make do without randomization, we can still obtain reasonable bounds that translate to simultaneous approximations of makespan and weighted completion time.

By the definition of V_ℓ, $f_v^* > \beta^{\alpha+\ell-1}$. Then, from (5) we obtain, for each vertex v, a bound of

$$\frac{\tilde{f}_v}{f_v^*} \leq d\cdot\rho\frac{\beta^2}{\beta-1}\ .$$

This is optimized when $\beta = 2$,
Theorem 2.5 *There is an algorithm that approximates simultaneously* `pSMC` *(*`npSMC`*) and* `pMC` *(*`npMC`*), to within factor $4d\rho$.*

Sum coloring approximation When the graph has unit color requirements, we get the SC problem. For this case, we obtain a slight improvement. The proof of the next theorem is given in the Appendix.

Theorem 2.6 *There is a $(d\gamma\rho/2)$-approximation algorithm for* SC*, where $\gamma \approx 3.59112$.*

3 Approximating Sum Multicoloring

We now apply our technique to the npSMC problem on several classes of graphs. We use both the preemptive approximation and the LP-based algorithm.

3.1 Approximating npSMC

Line graphs Here we can apply both the LP and the preemptive relaxations with equal performance ratio, but the latter is both combinatorial and more efficient. A greedy 2-approximation algorithm for pSMC on line graphs is presented in [BHK$^+$00] (that holds also in the weighted case). Thus, we can apply algorithm $\mathsf{ALG}_{\text{PRE}}$, with $d = 2$.

For non-preemptive multicoloring line graphs, we use the greedy algorithm of [CG$^+$85] that schedules each job as early as possible (i.e. colors each vertex with the smallest possible colors), breaking ties arbitrarily. This ensures that each vertex is always waiting for a neighbor until it is scheduled to completion. The completion time of a vertex is then at most the sum of the lengths of its neighbors, which is at most twice the length of the larger clique involving the vertex (see [CG$^+$85]). Thus, in this case, we have $\rho = 2$. Now, using Theorem 2.4, we get a performance bound for line graphs.

Theorem 3.1 *There is a* 7.683*-approximation algorithm for* npSMC *on line graphs.*

This improves on the recent factor of 10 by Kim [K-03] and the factor of 12 obtained by a combinatorial (greedy) algorithm in [HKS03]. Observe that the non-preemptive algorithms are all measured in terms of the preemptive optimum.

Intersection graphs of k-sets Resource-bounded scheduling when each job uses at most k resources is modeled by graphs that are intersection graphs of sets of size at most k. For each resource r, the vertices using that resource form a clique C_r. Then, for any $v \in V$, $N(v)$ can be partitioned into at most k maximal cliques.

We can extend the LP-based strategy for line graphs to intersection graphs of k sets. In this case, the non-preemptive greedy multicoloring algorithm of [CG$^+$85] uses at most $k\omega$ colors, where ω is the maximal size of any of the resource cliques. Thus, it suffices to consider only cliques induced by individual resource, and not those cliques formed by interplay of a collection of resources. In other words, the clique constraints in LP need only involve the resource-cliques, therefore the number of constraints in polynomial. Hence, we obtain a non-preemptive solution with $d = 2$ and $\rho = k$, and by Theorem 2.4, we get

Theorem 3.2 *There is a* $(3.591k + 0.5)$*-approximation for* npSMC *on intersection graphs of k-sets.*

This improves on the ratio of $6k - 2$ of [K-03].

$(k+1)$-claw free graphs The combinatorial strategy for line graphs can be generalized for $(k+1)$-claw free graphs, albeit with a worse ratio function than for LP-based algorithm for intersection

graphs of k-sets. The sorted greedy algorithm of [BHK$^+$00] yields a ratio of k for pSMC in $(k+1)$-claw free graphs, resulting in a preemptive relaxation with $d = k$ in our schema. Also, as above, the makespan algorithm has performance ratio $\rho = k$. Thus, we get

Theorem 3.3 *There is a combinatorial* $(1.796k^2 + 0.5)$*-approximation for* npSMC *on* $(k+1)$*-claw free graphs.*

Interval graphs The npMC problem on interval graphs is better known as *dynamic storage allocation*. Gergov gave an algorithm that uses at most $3\omega(G)$ colors [G-99]. The number of maximal cliques in an interval graph is at most n. Thus, LP has a polynomial number of constraints and we can use it to obtain a multicoloring satisfying (P1) and (P2a), with $d = 2$. We can also use the approximation of the preemptive solution of [HKS03] as a relaxation with $d = 7.184$. Applying Theorem 2.4, we obtain the first constant approximation factor for this problem.

Theorem 3.4 *There is an* 11.273*-approximation and a combinatorial* 38.7*-approximation for* npSMC *on interval graphs.*

3.2 Approximating pSMC

Perfect graphs On perfect graphs, LP can be solved in polynomial time, even though the number of constraints may be exponential, because there is a polynomial time separation algorithm: given a solution $\mathbf{f}$ for LP, we can test in polynomial time whether all the constraints are satisfied. For a vertex $v \in V$, we set, for each neighbor $u \in N(v)$, $x'_u = x_u \delta_{uv}$. We can now find the maximum weight clique in $N(v)$ with respect to $\mathbf{x}'$, since any subgraph of G is perfect. Then, we can test in polynomial time whether f_v satisfies the constraint (2) by checking whether the inequality holds for this maximum weight clique. (For more details, see e.g., [Q-93].) The solution for LP yields a multicoloring ψ^* that satisfies (P1) and (P2), with $d = 2$. The multicoloring problem pMC on perfect graphs is solvable in polynomial time, within arbitrary desired precision, as shown in [GLS-93], yielding our $\rho = 1 + O(1/n)$. Applying Theorems 2.3 and 2.6, we improve on the previous best factors of 16 for pSMC [BHK$^+$00] and 4 for SC [BBH$^+$98].

Theorem 3.5 *There is a* $2e \approx 5.436$*-approximation for* pSMC *and a* 3.592*-approximation for* SC *on perfect graphs.*

4 Extensions

Weights Note that vertex weights can be added in our LP formulation, to get the fractional values f^*_v that satisfy (P1) and (P2) for the weighted minsum objective. We then apply as before for each block ℓ the makespan algorithm $\mathcal{A}$.

Release times Our technique can be applied also in the case where each job J_j has a release time, r_j. In this case, in the LP formulation we add for any vertex v the constraint $f_v \geq r_v + x_v$. This ensures that, for any $v \in V_\ell$, $r_v \leq c_\ell$. Hence, when applying the makespan algorithm, $\mathcal{A}$, we start scheduling the vertices in V_ℓ at $\max(\sum_{r=1}^{\ell-1} t_r, \beta^{\alpha+\ell})$. This is attained by taking $\beta = 2$, which slightly increases the performance bounds that we obtained for ALG, both in the preemptive and the non-preemptive case.

Theorem 4.1 ALG *attains a ratio of* $d\rho 1.5/\ln 2 \approx 2.16 d\rho$ *for* npSMC *and* $d\rho 2/\ln 2 \approx 2.89 d\rho$ *for* pSMC *instances with release times.*

Resource Constrained Scheduling Recall that in RCS, the resources are represented as cliques in our conflict graph G. Let $\mathcal{C}$ denote the set of maximal cliques in G, then RCS can be formulated as the following linear program.

$$
\begin{array}{lll}
(LP-RCS) & \text{minimize} \quad \sum_{\hat{C}\in\mathcal{C}} f_{\hat{C}} & \\
\text{subject to:} & \forall \hat{C}\in\mathcal{C},\ \forall v\in\hat{C} & :\ f_v \geq x_v + \sum_{u\in\hat{C}} x_u\delta_{uv} \\
& \forall \hat{C}\in\mathcal{C},\ \forall v\in\hat{C} & :\ f_C \geq f_v \qquad (9) \\
& \forall uv\in E & :\ \delta_{uv}+\delta_{vu}=1 \qquad (10)
\end{array}
$$

This corresponds to only the last vertex of each clique contributing to the objective function in the `npSMC` problem. Our analysis in the preemptive case was separate for each vertex, bounding the cost for the vertex only by the last color used in that block. Thus, we obtain an approximation ratio of $2e \cdot k$ for RCS. This improves on the previous ratio of $8k-7$ presented by Kim [K-03], for any $k \geq 3$. For $k=2$, the ratio of 10.45 is worse than the best known approximation ratio of 5.055 [GHKS04], but is achieved by a polynomial-size linear program.

Acknowledgments. We thank Moses Charikar and Chandra Chekuri for helpful comments and suggestions.

References

[AB+00] F. Afrati, E. Bampis, A. Fishkin, K. Jansen, and C. Kenyon. Scheduling to minimize the average completion time of dedicated tasks. *FSTTCS '00*.

[BBH+98] A. Bar-Noy, M. Bellare, M. M. Halldórsson, H. Shachnai, T. Tamir. On chromatic sums and distributed resource allocation. *Inf. Comp.* **140**:183–202, 1998.

[BHK+00] A. Bar-Noy, M. M. Halldórsson, G. Kortsarz, H. Shachnai, and R. Salman. Sum multicoloring of graphs. *J. Algorithms* **37**(2):422–450, 2000.

[BK98] A. Bar-Noy and G. Kortsarz. The minimum color-sum of bipartite graphs. *J. Algorithms*, **28**:339–365, 1998.

[CP+96] S. Chakrabarti, C. A. Phillips, A. S. Schulz, D. B. Shmoys, C. Stein and J. Wein. Improved scheduling algorithms for minsum criteria. *ICALP '96*, 875–886.

[CG+85] E. G. Coffman, Jr., M. R. Garey, D. S. Johnson and A. S. LaPaugh. Scheduling file transfers. *SIAM J. Comput.* **14**:744–780, 1985.

[FLT-02] U. Feige, L. Lovász, P. Tetali. Approximating min-sum set cover. *APPROX'02*, 94–107.

[G-99] J. Gergov. Algorithms for compile-time memory allocation. *SODA'99*.

[G+02] K. Giaro, R. Janczewski, M. Kubale and M. Malafiejski. A 27/26-approximation algorithm for the chromatic sum coloring of bipartite graphs. *APPROX '02*, 131–145.

[G01] M. Gonen. Coloring Problems on Interval Graphs and Trees. M.Sc. thesis, The Open Univ., Tel-Aviv, 2001.

[GHKS04] R. Gandhi, M. M. Halldórsson, G. Kortsarz and H. Shachnai, Approximating non-preemptive open-shop scheduling and related problems. *ICALP '04*.

[GLS-93] M. Grötschel, L. Lovász and A. Schrijver. Geometric Algorithms and Combinatorial Optimization. Springer-Verlag, 1993.

[HSW-96] L. A. Hall, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line algorithms. *SODA'96*, 142–151. Jan 1996.

[HSSW-97] L. A. Hall, A. Schulz, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. *Math. Operations Research* 22:513–544, 1997.

[HK02] M. M. Halldórsson and G. Kortsarz. Tools for multicoloring with applications to planar graphs and partial k-trees. *J. Algorithms* 42(2), 334–366, 2002.

[HKP+03] M. M. Halldórsson, G. Kortsarz, A. Proskurowski, R. Salman, H. Shachnai, and J. A. Telle. Multicoloring trees. *Inf. Computation* 180(2):113–129, 2003.

[HKS03] M. M. Halldórsson, G. Kortsarz, H. Shachnai. Sum coloring interval and k-claw free graphs with application to scheduling dependent jobs. *Algorithmica* 37:187–209, 2003.

[J-97] K. Jansen. The optimum cost chromatic partition problem. *CIAC '97*, 25–36.

[K-03] Y. A. Kim. Data migration to minimize the average completion time, *SODA'03*.

[K-96] M. Kubale. Preemptive versus non preemptive scheduling of biprocessor tasks on dedicated processors. *European J. Operational Research* **94**:242–251, 1996.

[K89] E. Kubicka. The chromatic sum of a graph. PhD thesis, Western Michigan, 1989.

[NSS99] S. Nicoloso, M. Sarrafzadeh and X. Song. On the sum coloring problem on interval graphs. *Algorithmica* **23**:109–126,1999.

[P-80] C. N. Potts. An algorithm for the single machine sequencing problem with precedence constraints, *Math. Prog. Stud.* 13, 78–87, 1980.

[Q-93] M. Queyranne. Structure of a simple scheduling polyhedron. *Math. Prog.* 58:263–285, 1993.

[QS-01] M. Queyranne, M. Sviridenko. A $2+\epsilon$-approximation algorithm for generalized preemptive open shop problem with minsum objective. *J.Alg.* 45:202-212, 2002.

[QS-02] M. Queyranne, M. Sviridenko. Approximation algorithms for shop scheduling problems with minsum objective. *J. Scheduling* 5:287-305, 2002.

[S-96] A. S. Schulz. Scheduling to minimize total weighted completion time: Performance guarantees of LP-based heuristics and lower bounds. *IPCO '96*, 301–315.

[W97] G. Woeginger. Private communication, 1997.

[W-85] L. Wolsey. Mixed Integer Programming Formulations for Production Planning and Scheduling Problems. Invited talk at *12th ISMP*, MIT, 1985.

A Scheduling Dependent Jobs on Parallel Machines

In the following we describe how our technique can be applied for scheduling a set of n dependent jobs on m identical machines. As before, we get as input the conflict graph of the jobs, G. Our problem of minimizing the sum of completion times can be formulated as the following integer program.

$$(LP(m)) \qquad \text{minimize} \sum_{v \in V} f_v$$

$$\text{subject to:} \qquad \forall S \subseteq V \qquad : \sum_{v \in S} x_v f_v \geq \frac{(x(S))^2 + \sum_{v \in S} x_v^2}{2m} \tag{11}$$

$$\forall v \in V,\ 1 \leq r \leq N_v \quad : f_v \geq x_v + \sum_{u \in C_r} x_u \delta_{uv} \tag{12}$$

$$\forall uv \in E \qquad : \delta_{uv} + \delta_{vu} = 1$$

In the linear programming relaxation we allow $f_v \geq 1$. For a subset of vertices $S \subseteq V$ and a vertex $v \in S$, we denote by $P_v(S)$ the set of vertices in S whose coloring is completed no later than f_v^* in ψ^*; that is, $P_v(S) = \{u \in S \mid f_u^* \leq f_v^*\}$. The above program satisfies the next lemma, due to [HSSW-97]).

Lemma A.1 *For any $v \in V$ and a subset of vertices $S \subseteq V$, $f_v^* \geq \dfrac{x(P_v(S))}{2m}$.*

We note that on the classes of graphs that we study $LP(m)$ can be solved in polynomial time. This follows from the fact that, given an optimal solution, we can use the separation algorithm of [Q-93] to test whether all the constraints in (11) are satisfied; the other set of constraints may be either of polynomial size, or exponential, in which case we apply the separation algorithm described in Section 3.2. We now describe our algorithm, ALG, distinguishing between the preemptive and non-preemptive case.

Preemptive scheduling In the preemptive case, we solve $LP(m)$ and partition the time axis as before, to the intervals $(\beta^{\alpha+\ell-1}, \beta^{\alpha+\ell}]$. For any $\ell \geq 1$, we multicolor V_ℓ using a ρ-approximation algorithm for the pMC problem; that is, we initially assume that we have *unbounded* number of machines; then, we 'fix' the preemptive schedule of V_ℓ, by partitioning each color class I_g to $\lfloor |I_g|/m \rfloor$ sets of size m, and at most one set of size smaller than m. By that, we ensure that at most m jobs are processed at any given time. Let t_ℓ be the total number of colors used after we fix the schedule of V_ℓ. Now, we schedule V_ℓ after $V_{\ell-1}$, in the next t_ℓ time units.

In analyzing our algorithm, we first upper bound the number of colors used by $\mathcal{A}$ after we fix the schedule of V_ℓ. Using (1), we have that

$$t_\ell = \mathcal{A}(V_\ell, x) \leq \rho\ \omega(V_\ell, x) + \frac{x(V_\ell)}{m} \leq d \cdot \rho \beta^{\alpha+\ell} + \frac{x(V_\ell)}{m}. \tag{13}$$

Let $V_\ell^- = \cup_{r=1}^{\ell} V_r$ denote the set of jobs scheduled up to (and including) the ℓ-th block. Recall that $\tilde{f}_v$ is the completion time of v under ALG. Then,

$$\sum_{v \in V_\ell} \tilde{f}_v = \sum_{v \in V_\ell} \sum_{r=1}^{\ell} t_r \leq \sum_{v \in V_\ell} \frac{d \cdot \rho \beta^{\alpha+\ell+1}}{\beta - 1} + \frac{x(V_\ell^-)}{m} \tag{14}$$

By Lemma A.1,

$$\frac{x(V_\ell^-)}{m} \leq 2 \cdot f^*_{V_\ell} \leq 2\beta_{\alpha+\ell}.$$

Hence, overall we get that

$$\mathsf{ALG}(V,x) = \sum_{v\in V} \tilde{f}_v \leq d \cdot \rho \sum_{\ell\geq 1} \frac{\beta^{\alpha+\ell+1}|V_\ell|}{\beta-1} + 2\sum_{\ell\geq 1}\beta^{\alpha+\ell} = (d\cdot\rho\frac{\beta}{\beta-1}+2)\sum_{\ell\geq 1}\beta^{\alpha+\ell} \tag{15}$$

Randomizing on α we have that

$$\mathbf{E}[\mathsf{ALG}(V,x)] \leq OPT^*(d\cdot\rho\frac{\beta}{\ln\beta} + \frac{2(\beta-1)}{\ln\beta}),$$

and taking $\beta = e$ we get the next result.

Theorem A.2 *There is a $(d\rho + 2(e-1))$-approximation algorithm for the preemptive minsum of completion times of dependent jobs on m parallel machines, where d is as given in (P2a), ρ is the approximation ratio of algorithm A for preemptive makespan, and $e \approx 2.718$ is the base of the natural logarithm.*

Non-preemptive scheduling In the non-preemptive case, it may not be possible to 'fix' the schedule of V_ℓ, i.e., transform a schedule with 'unbounded' number of machines to one that uses at most m machines at any time. Thus, when scheduling the jobs in V_ℓ, we assume that $\mathcal{A}$ is an approximation algorithm for the makespan problem on m machines.

Theorem A.3 *Let $\mathcal{A}$ be a ρ-approximation algorithm for the non-preemptive makespan problem on m parallel machines; then, ALG achieves the approximation ratio $(d\gamma\rho/2 + 0.5)$ to the non-preemptive minsum of completion times, where d is as given in (P2a), and $\gamma \approx 3.59112$.*

Note that we apply here ALG with possible reverse of the schedule. We decide on reversing the schedule for each machine separately.

B Some proofs

Proof of Lemma 2.1 Let C be a clique in G. Let f_v be the completion time of $v \in C$ in the solution for LP. Indeed, $C \setminus \{v\} \subseteq N(v)$. From LP, we get that

$$\begin{aligned}
\sum_{v\in C} x_v f_v &\geq \sum_{v\in C} x_v(x_v + \sum_{u\in C, u\neq v} x_u\delta_{uv}) \\
&= \sum_{v\in C} x_v^2 + \sum_{u,v\in C}(x_v x_u \delta_{uv} + x_v x_u \delta_{vu}) \\
&\geq \frac{\sum_{v\in C} x_v^2 + (\sum_{u\in C} x_u)^2}{2}
\end{aligned} \tag{16}$$

Now, let C_ℓ be a maximum weight clique in V_ℓ, and let v_ℓ be the vertex in C_ℓ with the largest completion time in V_ℓ, $f^*_{v_\ell}$. From (16), we have that $\sum_{u\in C_\ell} x_u f_u \geq x(C_\ell)^2/2 = \omega(V_\ell, x)^2/2$. We also have that $\sum_{u\in C_\ell} x_u f_u \leq f^*_{v_\ell}\sum_{u\in C_\ell} x_u = f^*_{v_\ell} x(C_\ell) = f^*_{v_\ell}\omega(V_\ell, x)$.

Proof of Lemma 2.2 By the definition of ℓ_v, $c_{\ell_v-1} = \beta^{\alpha+\ell_v-1} < f_v^* \le \beta^{\alpha+\ell_v} = c_{\ell_v}$. Let us write $f_v^* = \beta^x$, i.e. $x = \log_\beta f_v^*$. Let $y_v = \ell_v + \alpha - x$ and note that y_v is in the range $[0,1)$. We may write $y_v = (\alpha - x) \bmod 1$. The values f_v^* and x are fixed and independent of α. Thus, when α is chosen uniformly at random from $[0,1)$, y_v is also uniformly distributed in $[0,1)$. The random variable β^{y_v} then has expected value

$$\mathbf{E}[\beta^{y_v}] = \int_0^1 \beta^t dt = \frac{\beta-1}{\ln\beta}.$$

Hence,

$$\mathbf{E}[c_{\ell_v}] = \mathbf{E}[\beta^{\ell_v+\alpha}] = \mathbf{E}[\beta^{\ell_v+\alpha-x}] \cdot \beta^x = \frac{\beta-1}{\ln\beta} f_v^*. \tag{17}$$

Proof of Theorem 2.6 Continuing from (7), we have

$$\begin{aligned}
\sum_{v\in V_\ell} \tilde{f}_v &\le d\rho|V_\ell|\left(\frac{\beta^{\alpha+\ell}}{2} + \sum_{r=0}^{\ell-1}\beta^{\alpha+r}\right) + \frac{1}{2}\sum_{v\in V_\ell} x_v \\
&= d\rho|V_\ell|\left(\frac{\beta^{\alpha+\ell}}{2} + \beta^\alpha\frac{\beta^\ell-1}{\beta-1}\right) + \frac{1}{2}\sum_{v\in V_\ell} x_v \\
&= d\rho|V_\ell|\left(\beta^{\alpha+\ell}\frac{\beta+1}{2(\beta-1)} - \frac{\beta^\alpha}{\beta-1}\right) + \frac{1}{2}\sum_{v\in V_\ell} x_v.
\end{aligned}$$

Thus,

$$\mathsf{ALG}(V,x) = \sum_{v\in V}\tilde{f}_v \le d\rho\sum_{\ell\ge 1}|V_\ell|\left[\frac{\beta+1}{2(\beta-1)}\beta^{\alpha+\ell} - \frac{\beta^\alpha}{\alpha-1}\right] + \frac{|V|}{2}\ .$$

Hence, applying Lemma 2.1, we have

$$\begin{aligned}
\mathbf{E}[\mathsf{ALG}(V,x)] &= \sum_{v\in V}\mathbf{E}[\tilde{f}_v] \le d\rho\sum_v\left[\mathbf{E}[c_{\ell_v}]\cdot\frac{\beta+1}{2(\beta-1)} - \frac{1}{\beta-1}\right] + \frac{|V|}{2} \\
&= d\rho\cdot\frac{\beta+1}{2\ln\beta}OPT^* - \frac{d\rho|V|}{\beta-1} + \frac{|V|}{2} \le d\rho\cdot\frac{\beta+1}{2\ln\beta}OPT^*.
\end{aligned}$$

The last inequality follows from the fact that $\frac{\rho d}{\beta-1} > 1/2$, since $\rho \ge 1$, $\beta < 5$, and in the cases we have studied, $d \ge 2$.