Sum Multi-Coloring of Graphs

(Extended Abstract)

Amotz Bar-Noy[*] Magnús M. Halldórsson[†] Guy Kortsarz[‡]

Ravit Salman[§] Hadas Shachnai[¶]

Abstract

Scheduling dependent jobs on multiple machines is modeled by the graph *multi-coloring* problem. In this paper, we consider the problem of minimizing the average completion time of all jobs. This is equivalent to the following coloring problem: Given a graph and the number of colors required by each vertex, find a multi-coloring which minimizes the sum of the largest colors assigned to the vertices. We call this problem the *sum multi-coloring* problem. In the special case where all jobs have the same (unit) execution times, the problem reduces to the known *sum coloring* problem.

We study three variants of the problem that appear in many applications. In the preemptive case, a vertex can be assigned any set of colors. In the non-preemptive case, this set of colors has to be *contiguous*. In the third variant, called the co-scheduling problem, the vertices are colored in batches, where each batch is an independent set in the graph that starts with the same color.

This paper reports a comprehensive study of the sum multi-coloring problem. We extend most of the known results for the sum coloring problem into the three variants of the sum multi-coloring problem. We concentrate on finding approximation algorithms since the special case of the sum coloring problem is already hard to solve and hard to approximate.

In particular, for the preemptive sum multi-coloring problem and for the co-scheduling problem, we present an $O(\rho)$-approximation polynomial time algorithms for all graphs for which the maximum independent set can be approximated within a factor $O(\rho)$ by a polynomial time algorithm. For the preemptive case, we describe in addition constant factor approximation algorithms for bipartite graphs, bounded degree graphs and line graphs. For the non-preemptive sum multi-coloring problem, we present a $\min\{O(\rho \log p), O(\rho \log n)\}$-approximation algorithm for all graphs for which the maximum independent set can be approximated within a factor $O(\rho)$ by a polynomial time algorithm, where p is the maximum number of colors required by any vertex and n is the number of vertices in the graph. In addition, we show constant approximations e.g. for bipartite graphs and planar graphs. Finally, we show lower bounds for any algorithm that is based on iteratively finding maximum independent sets – a procedure used by our approximation algorithms.

[*]Dept. of EE, Tel-Aviv University, Tel-Aviv 69978, Israel. E-mail: amotz@eng.tau.ac.il.

[†]Science Institute, University of Iceland, IS-107 Reykjavik, Iceland. E-mail: mmh@hi.is.

[‡]Dept. of Computer Science, Open University, Ramat Aviv, Israel. E-mail: guyk@tavor.openu.ac.il.

[§]Dept. of Mathematics, Technion, Haifa 32000, Israel. E-mail: maravit@tx.technion.ac.il.

[¶]Dept. of Computer Science, Technion, Haifa 32000, Israel. E-mail: hadas@cs.technion.ac.il.

1 Introduction

In multi-processor systems certain resources may not be shared concurrently by some sets of jobs. A fundamental problem in distributed computing is to efficiently schedule jobs that are competing for such resources. The scheduler has to satisfy the following two conditions: (i) *mutual exclusion*: no two conflicting jobs are executed simultaneously. (ii) *no starvation*: the request of any job to run is eventually granted. The problem is well-known in its abstracted form as the *dining/drinking philosophers* problem (see, e.g., [L81, NS93, SP88]).

Scheduling dependent jobs on multiple machines is modeled as a graph *coloring* problem, when all jobs have the same (unit) execution times, and as graph *multi-coloring* for arbitrary execution times. The vertices of the graph represent the jobs and an edge in the graph between two vertices represents a dependency between the two corresponding jobs that forbids scheduling these jobs at the same time. More formally, for a weighted undirected simple graph $G = (V, E)$ with n vertices, let the *length* of a vertex v be a positive integer denoted by $x(v)$ and called the *color requirement* of v. A multi-coloring of the vertices of G is a mapping into the power set of the positive integers, $\Psi : V \mapsto 2^N$. Each vertex v is assigned a set S_v of $x(v)$ distinct numbers (colors) and adjacent vertices are assigned with disjoint sets of colors.

The traditional optimization goal is to minimize the total number of colors assigned to G. In the setting of a job system, this is equivalent to finding a schedule in which the time when *all* the jobs have been completed is minimized. Such an optimization goal favors the system. However, from the point of view of the jobs themselves, an important goal is to minimize the average completion time of the jobs. Formally, given a multi-coloring Ψ, define by $f_\Psi(v)$ the maximum color assigned to v by Ψ. Then minimizing the average completion time is equivalent to minimizing the sum of all these numbers. In the *sum multi-coloring* (SMC) problem, we look for a multi-coloring Ψ that minimizes $\sum_{v \in V} f_\Psi(v)$. When all the color requirements are equal to 1 the problem is reduced to the *sum coloring* (SC) problem. In this paper we study the following three variants of the sum multi-coloring problem:

- In the *preemptive* (p-SMC) problem, each vertex may get any set of colors.
- In the *non-preemptive* (np-SMC) problem, the set of colors assigned to each vertex has to be contiguous.
- In the *co-scheduling* (co-SMC) problem, vertices are colored in batches. Each batch is an independent set, and the scheduler may start coloring the next batch only when it has completed coloring all the vertices in the previous batch.

The preemptive version corresponds to the scheduling approach commonly used in modern operating systems [SG98]. Jobs may be interrupted during their execution and resumed at later time. The non-preemptive version captures the execution model adopted in real-time systems where scheduled jobs must run to completion. The co-scheduling approach is used in some distributed operating systems [T95]. In such systems, the scheduler identifies subsets of *cooperating processes* that can benefit from running at the same time interval (e.g., since the processes in the set communicate frequently with each other). Then each subset is executed simultaneously on several processors until *all* the processes in the subset complete.

1.1 Related work

The sum coloring problem was introduced in [K89]. Further research can be found in [KKK89, KS89, TEA$^+$89]. Recent research concentrated on finding approximation algorithms and proving hardness results [BBH+98, BK98]. The paper [BBH+98] addressed general graphs and bounded degree graphs, and the paper [BK98] studied bipartite graphs.

A "good" sum coloring solution tries to color as many as possible vertices as "early" as possible. This suggests the following natural MAXIS heuristic for the sum coloring problem proposed in [BBH+98]:

MAXIS: Choose a maximum independent set in the graph, color all of its vertices with the next available color, and iterate until all vertices are colored.

This procedure is shown to produce a 4-approximation algorithm. However, the algorithm is polynomial only for those graphs for which a maximum independent set can be found in a polynomial time. In addition, this heuristic provides a $O(\rho)$ approximation polynomial time algorithm if the maximum independent can be solved by an $O(\rho)$ approximation polynomial time algorithm. We note that coloring the graph with minimum number of colors does not always help the sum coloring problem. For instance, while bipartite graphs can be colored with two colors, there exist bipartite graphs (in fact, trees) for which the optimal sum coloring uses $\Omega(\log n)$ colors [KS89].

In [MPT94] the problem of on-line non-clairvoyant scheduling of jobs was studied. Approximation algorithms were presented for minimizing the average completion times, when the jobs are *independent*. For the case in which there are some dependencies among the jobs, the paper discussed algorithms for minimizing the overall completion time of the schedule. The problem of scheduling dependent jobs, where the objective is minimizing the average completion time, remained open. This problem is the focus of our work.

We conclude this subsection with the known hardness results for the sum coloring problem that carry over to the sum multi-coloring problems. Sum coloring cannot be approximated in general within $n^{1-\epsilon}$, for any $\epsilon > 0$, unless $NP = ZPP$ [FK96, BBH+98]. Also, it is NP-hard to approximate within some factor $c > 1$ on bipartite graphs [BK98].

1.2 Results

This paper reports a comprehensive study of the sum multi-coloring problem. We extend most of the known results for the sum coloring problem to the three variants of the sum multi-coloring problem. We concentrate on finding approximation algorithms since the sum coloring problem is already hard to solve and hard to approximate and therefore this is the case with the sum multi-coloring problem. We detail below our main results for the three variants of the sum multi-coloring problem, and summarize further results.

1.2.1 The preemptive sum multi-coloring problem

We present an efficient $O(1)$-approximation algorithm for the p-SMC problem on graphs for which the maximum independent set can be found in polynomial time (e.g. perfect graphs). In graphs for which the maximum independent set can be approximated within a factor $O(\rho)$, our algorithm is an $O(\rho)$ approximation algorithm.

We also study special families of graphs. For bipartite graphs, we describe a 1.5-approximation algorithm. (The best ratio known for the SC problem is 10/9 [BK98].) Next we present a $(\Delta+2)/3$ approximation algorithm for graphs of maximum degree Δ and a 2-approximation for line graphs, generalizing the same bounds obtained for the sum coloring case [BBH+98]. The algorithm works by sorting the vertices in non-decreasing order of color requirements and then coloring the vertices in order using the first available colors. Using an arbitrary order, the algorithm is exactly $\Delta + 1$-approximation on bounded-degree graphs and $\Omega(\sqrt{p})$-approximation even on line graphs of stars, where p is the maximum color requirement in the graph.

Finally, when the scheduling graph is created by conflicts of resources and each job uses at most k resources, the resulting graph is an intersection graph of a set system where each set is of size at most k. We can generalize the argument for line graphs to give a k-approximation to the p-SMC problem on such graphs.

1.2.2 The non-preemptive sum multi-coloring problem

We show that the pure MAXIS algorithm has a relatively poor ratio for the SMC problems. We therefore introduce a version of MAXIS that is sensitive to color requirements. Let $t = \min\{\log p, \log n, r\}$ where r is the number of different color-requirements in the graph. We show this algorithm to be an $O(t)$-approximation algorithm on graphs for which the maximum independent set can be found in polynomial time. The result also translates to an $O(\rho \cdot t)$-approximation algorithm, when the maximum independent set can be approximated within a factor $O(\rho)$.

Again we address special families of graphs. We describe a 2.796-approximation algorithm for bipartite graphs, and a $1.55k + 1$-approximation algorithm when a k-coloring of the graph is given. The algorithms are randomized, but can be de-randomized. This implies, e.g., an $O(k)$-approximation on graphs of treewidth k and a 7.2-approximation factor for planar graphs. For bounded-degree graphs, we show that the sorted greedy algorithm yields a $\Delta + 1$-approximation.

1.2.3 The co-scheduling problem

We give an $O(1)$ ($O(\rho)$) approximation algorithm for the co-SMC problem on graphs in which the maximum independent set can be found in polynomial time (approximated within $O(\rho)$ factor). This, in fact, is achieved by the same algorithm as used for the np-SMC problem.

We explore the relationship among the three models and give a construction that indicates why it is harder to approximate the non-preemptive sum coloring. Namely, while finding independent sets iteratively suffices to approximate both the p-SMC and the co-SMC problems, any such solutions must be $\Omega(\log p)$ off for the np-SMC problem.

1.2.4 Performance bounds for the MAXIS algorithm

An immediate application of the MAXIS algorithm for the sum coloring problem appears in the $O(1)-$approximation algorithm for the p-SMC problem. Furthermore, most of our algorithms reduce to MAXIS for the SC problem if the color requirements are uniform. It is therefore natural to find the exact performance of MAXIS for the sum coloring problem. In [BBH$^+$98] it was shown that this algorithm yields a 4-approximation to the sum coloring problem. In this paper, we give a (rather non-trivial) construction that shows that MAXIS cannot achieve an approximation factor better than 4.

As for the multi-coloring versions, we note that it is usually preferable to color many vertices early, even if it means that more colors will be needed in the end. Thus, the MAXIS algorithm is a natural candidate heuristic also for the sum multi-coloring problems. However, we show that MAXIS is only an $\Omega(\sqrt{p})$ approximation algorithm, where p is the largest color requirement in the graph. For the np-SMC problem, its performance cannot even be bounded in terms of p.

1.2.5 Further results

In this subsection we report more results of our study, whose detailed description has been omitted.

Planar graphs: We show that already the sum coloring problem is NP-hard on planar graphs, implying a similar result for the sum multi-coloring versions. We complement this hardness result by designing an algorithm for the sum coloring problem on planar graphs the approximation ratio of which is $1+\varepsilon$ for any $\varepsilon > 0$.

Extensions to the weighted case: In the weighted sum multi-coloring problem, each vertex v is associated with a weight $w(v)$ and the goal is to minimize $\sum_{v \in V} w(v) \cdot f_\Psi(v)$. Most of our results apply also for this optimization goal, in particular the approximations of all three variants on general graphs. In this extended abstract we describe the results only in the unweighted model to simplify the presentation. Though, some of the generalizations are straightforward.

Extensions to On-line Scheduling of Dependent Jobs: The above results for graph multi-coloring immediately apply to off-line scheduling of dependent jobs, that is, when all jobs are released at the same time. The following results apply also to the on-line setting, where the jobs can be released at arbitrary (unknown) times, and the scheduler is *non-clairvoyant*, i.e., it does not know the total execution time of a job until it completes running that job. The first-fit algorithm is $(\Delta+1)$-competitive for bounded degree graphs. For a set of jobs, in which the ratio between the maximum running time and the minimum running time is q, MAXIS is $\Theta(q)$-competitive. Note that in this case any on-line algorithm has a competitive ratio $\Omega(q)$ [MPT94], even when the jobs are independent. Thus, MAXIS is the best possible (to within a constant factor). In fact, it can be shown, that any on-line deterministic algorithm is doomed to a poor performance, even for the sum coloring problem on interval graphs. Specifically, any deterministic algorithm that on-line colors an interval graph is at least ω-competitive, where ω is the size of the maximum clique in the interval graph. A matching on-line $O(\omega)-$ratio algorithm is derived via an algorithm of Kierstead [K91].

1.3 Extended abstract organization:

Due to space limitations, this extended abstract presents only the main results where some of the proofs are relegated to the Appendix. Section 2 describes some notation and definitions. Section 3 presents approximation algorithms for the preemptive sum multi-coloring problem; Section 4 describes the results for the non-preemptive sum multi-coloring problem; and Section 5 discusses the co-scheduling problem. Appendix A provides some of the proofs and Appendix B shows the performance bounds for the MAXIS algorithm.

2 Definitions and Notations

For a given undirected graph $G=(V,E)$, let n denote the number of vertices. The color requirement of each vertex $v \in V$ (referred also as the length) is given by the mapping $x : V \rightarrow N$. For a given mapping x, we denote by $\mathcal{S}(G) = \sum_v x(v)$ the sum of the color requirements of the vertices in G. We denote by p the maximum color requirement in G, that is $p = \max_{v \in V} x(v)$. An *independent set* in G is a subset I of V such that any two vertices in I are non-adjacent.

A *multi-coloring* of G is a an assignment $\Psi : V \rightarrow 2^N$, such that each vertex $v \in V$ is assigned $x(v)$ distinct colors and the set of vertices colored by any color i is independent. Denote by C_i the independent set that consists of vertices with $i \in \Psi(v)$, by $c_1^\Psi(v), \ldots, c_{x(v)}^\Psi(v)$ the collection of $x(v)$ colors assigned to v, and by $f_\Psi(v) = c_{x(v)}^\Psi(v)$ the largest color assigned to v. Given a graph $G(V,E)$, a mapping $x : V \rightarrow \mathbf{N}$, and a multi coloring of G, $\Psi : V \rightarrow \mathbf{2^N}$, the multi-color sum of G

with respect to Ψ is

$$\mathrm{SMC}(G, \Psi) = \sum_{v \in V} f_\Psi(v) \ .$$

A multi-coloring Ψ is *contiguous* (non-preemptive), if for any $v \in V$, the colors assigned to v satisfy $c^\Psi_{i+1}(v) = c^\Psi_i(v) + 1$ for $1 \le i < x(v)$. In the context of scheduling, this means that all the jobs are processed without interruption. A multi-coloring Ψ solves the co-scheduling problem, if the set of vertices can be partitioned into k disjoint independent sets $V = I_1 \cup \cdots \cup I_k$ with the following two properties: (i) $c^\Psi_1(v) = c^\Psi_1(v')$ for any $v, v' \in I_j$, for $1 \le j \le k$. (ii) $c^\Psi_{x(v)}(v) < c^\Psi_1(v')$ for all $v \in I_j$ and $v' \in I_{j+1}$ for $1 \le j < k$. In the context of scheduling, this means scheduling to completion all the jobs corresponding to I_1, and only then starting to process the jobs in $V \setminus I_1$.

The *minimum multicolor sum* of a graph G, denoted by $\mathrm{pSMC}(G)$, is the minimum $\mathrm{SMC}(G, \Psi)$ over all multi-colorings Ψ. We denote the minimum contiguous multi-color sum of G by $\mathrm{npSMC}(G)$. The minimum multi-color sum of G for the co-scheduling problem is denoted by $\mathrm{coSMC}(G)$. It holds that for any graph G:

$$\mathcal{S}(G) \le \mathrm{pSMC}(G) \le \mathrm{npSMC}(G) \le \mathrm{coSMC}(G) \ .$$

Let P be one of the three sum multi-coloring problems. We say that an algorithm A approximates P by a factor β if for all graphs G (belonging to a certain family of graphs) we have that

$$\frac{\mathrm{SMC}(G, A(G))}{\mathrm{SMC}(G, OPT(G))} \le \beta \ ,$$

where $A(G)$ and $OPT(G)$ are the sum multi-coloring of G produced by algorithm A and the best solution for P on G. In some cases, we compute β by bounding the ratio between $\mathrm{SMC}(G, \Psi)$ and $\mathcal{S}(G)$ when it is the only lower bound we know for $OPT(G)$. Furthermore, we sometimes bound the ratio between $f_A(v)$ and $x(v)$ for all vertices $v \in V$ to get the bound for $\frac{\mathrm{SMC}(G,A(G))}{\mathcal{S}(G)}$.

Finally, we would like to comment on our results that are based on algorithm MAXIS. The algorithms run in polynomial time only for graphs in which finding a (weighted) maximum independent set can be done in polynomial time (e.g., perfect graphs). An additional factor of $O(\rho)$ is added when finding a (weighted) maximum independent set could be approximated by a factor of $O(\rho)$ by a polynomial time algorithm. Given the known hardness results, such a relationship is unavoidable. In the remainder, let ρ refer to the approximation ratio of the independent set algorithm used as a subroutine in MAXIS (with $\rho = 1$ if the algorithm is exact).

3 The preemptive sum multi-coloring problem

In the preemptive version of the sum multi-coloring problem, a vertex may get any set of $x(v)$ colors without restrictions like in the other two versions. Our first result is an $O(1)$-approximation for the p-SMC problem on general graphs. This is an extension of the result for the sum-coloring problem in the sense that it establishes a connection between sum multi-coloring and maximum weighted independent sets. Then we address several families of graphs: bipartite graphs, bounded degree graphs, and line graphs.

3.1 General graphs

We first recall the *weighted* sum coloring problem that is closely related to the sum coloring problem. In the weighted version, each vertex v has a weight $w(v)$, and we need to give each vertex v a (single) color $\Psi(v)$. The goal is to minimize: $\sum_{v \in V} w(v)\Psi(v)$. The weighted version also admits a 4−approximation, via the weighted MAXIS algorithm, selecting an independent set with maximum *weight* in each round [BBH$^+$98].

We now define another optimization goal. We denote by $\mathrm{AV}_\Psi(v)$ the average color assigned to v by Ψ, namely, $\mathrm{AV}_\Psi(v) = (\sum_{i \in \Psi(v)} i)/x(v)$. Let $\mathrm{SA}_\Psi(G) = \sum_v \mathrm{AV}_\Psi(v)$ denote the sum of averages of Ψ. Let $\mathrm{SA}^*(G)$ be the minimum possible average sum. Clearly, $f_\Psi(v) \geq \mathrm{SA}_\Psi(v)$ for each v and Ψ. In fact, $f_\Psi(v) \geq \mathrm{AV}_\Psi(v) + (x(v)-1)/2$. Hence,

$$\mathrm{pSMC}(G) \geq \mathrm{SA}^*(G) + \frac{\mathcal{S}(G) - n}{2} \ . \tag{1}$$

In order to approximate pSMC, we describe an algorithm that approximates $\mathrm{SA}^*(G)$. The idea is the following: (i) Make $x(v)$ copies of each vertex v into a clique, with each copy of v adjacent to all copies of neighbors of v in G. (ii) Give each copy of v a weight $1/x(v)$. Let $\tilde{G}$ denote the resulting weighted graph. (iii) Find in $\tilde{G}$ an approximately minimum weighted sum coloring, using weighted MAXIS. In the above procedure, each vertex v is assigned $x(v)$ different colors via its copies. If the colors assigned to v are $c_1(v), \ldots, c_{x(v)}(v)$, the weighted contribution of v to the color-sum in $\tilde{G}$ is $(\sum_i c_i(v))/x(v)$. This coincides with the definition of the average color of v in pSMC. Hence, MAXIS on $\tilde{G}$ produces a 4−approximation for $\mathrm{SA}^*(G)$.

Since $x(v)$ is not necessarily polynomially bounded in n, this is not a polynomial reduction as stated. Instead, we can simulate the execution of MAXIS on $\tilde{G}$ by omitting all the extra copies of each vertex, since an independent set can contain at most one copy of a vertex. We are led to the following observation.

Observation 3.1 *Suppose that a multi-coloring Ψ of a graph G has the property that each color i is an independent set of weight within a ρ factor from optimal on the remaining graph (the subgraph induced by the vertices colored with i or later color). Then, $SA_\Psi(G) \leq 4\rho \cdot SA^*(G)$. If further, Ψ is contiguous, then* $\mathrm{SMC}(G, \Psi) \leq 4\rho \cdot \mathrm{pSMC}(G)$.

Proof: In a contiguous coloring, the final color differs from the average color by exactly half the color requirement of the vertex: $f_\Psi(v) = AV_\Psi(v) + (x(v)-1)/2$. Thus, by the first part of the observation and by Inequality 1 we have

$$\mathrm{SMC}(G,\Psi) = SA_\Psi(G) + \frac{\mathcal{S}(G) - n}{2} \leq 4\rho \cdot SA^*(G) + \frac{\mathcal{S}(G) - n}{2} \leq 4\rho \cdot \mathrm{pSMC}(G) \ . \qquad \Box$$

We are now ready to state the algorithm. Let P denote the above algorithm that approximates $SA^*(G)$. Intuitively, we take the coloring Ψ produced by P and form a new coloring Ψ'. If a vertex was colored i under Ψ it will be colored $2i-1$ and $2i$ under Ψ', until fully colored. Let C denote the number of colors used by P on G.

Algorithm 1 *Algorithm* Dbl

```
$\Psi \leftarrow P(G)$
Let $C_i$ denote $\{v \in G : i \in \Psi(v)\}$, for $i = 1, 2, \ldots, C$
$\Psi'(v) \leftarrow \emptyset$, for each $v \in V(G)$
for $i \leftarrow 1$ to $C$
  for each $v \in C_i$
    if $|\Psi'(v)| < x(v)$ then $\Psi'(v) \leftarrow \Psi'(v) \cup \{2i-1\}$
      if $|\Psi'(v)| < x(v)$ then $\Psi'(v) \leftarrow \Psi'(v) \cup \{2i\}$
```

It is not hard to verify that Algorithm Dbl produces a legal multi-coloring. We now prove the following theorem regarding the approximation ratio of the algorithm.

Theorem 3.2 *Assume the conditions of Observation 3.1. Then, Algorithm* Dbl *approximates p-SMC within a factor of* 16ρ.

Proof: Let m_v be the median color assigned to v by P, i.e. the color $c^{\Psi}_{\lceil x(v)/2\rceil}$. Note that $m_v \leq 2 \cdot AV_{\Psi}(v)$, since less than half the elements of a set of natural numbers can be larger than twice its average. Also, Dbl uses at most $2 \cdot m_v$ colors on v. Hence, $f_{\Psi'}(v) \leq 4 \cdot AV_{\Psi}(v)$, and

$$\mathrm{SMC}(G, \mathsf{Dbl}) = \sum_{v \in V} f_{\Psi'}(v) \leq 4 \cdot SA_{\Psi}(G) \ . \tag{2}$$

Therefore, by Inequality (2), Observation 3.1, and Inequality (1), respectively:

$$\mathrm{SMC}(G, \mathsf{Dbl}) \leq 4 \cdot SA_{\Psi}(G) \leq 16\rho \cdot \mathrm{SA}^*(G) \leq 16\rho \cdot \mathrm{pSMC}(G) \ .$$

□

3.2 Bipartite Graphs

We first consider graphs that can be colored with k colors. I.e., the set of vertices V can be partitioned into k disjoint independent sets $V = V_1 \cup \cdots \cup V_k$. Consider the following **Round-Robin** algorithm: For $1 \leq i \leq k$ and $h \geq 0$, at round $t = k \cdot h + i$ give the color t to all the vertices of V_i that still need a color. It is not hard to see that $f(v) \leq k \cdot x(v)$ for all $v \in V$. Hence, the Round-Robin algorithm is a k-approximation algorithm. In particular, it is a 4-approximation algorithm for planar graphs and a $\Delta + 1$-approximation algorithm for graphs with maximum degree Δ. For bipartite graphs, Round-Robin is a 2-approximation algorithm. In this subsection we give a non-trivial algorithm whose approximation factor is at most $1.5 - 1/(2n)$.

We need the following definitions and notations. Let G be a bipartite graph $G(V_1, V_2, E)$ with n vertices, such that edges connect vertices in V_1 with vertices in V_2. We denote by $\alpha(G)$ the size of a maximum independent set in G. We use the term *processing* an independent set $W \subseteq V$ to mean assigning the next available color to all the vertices of W. Suppose that the first i colors assigned by a multi-coloring Ψ are distributed among the vertices. The *reduced graph* of G is the graph for which the $x(v)$ values are decreased accordingly with the colors assigned so far, deleting vertices that were fully colored. Finally, let $\gamma(n) = (2n^2)/(3n-1)$.

Informally, the algorithm distinguishes the following two cases: If the size of the maximum independent set in the current reduced graph is "large," the algorithm chooses to process a maximum independent set. Otherwise, if the maximum independent set is "small," the algorithm works in a fashion similar to Round-Robin. Once a vertex (or a collection of vertices) is (are) assigned their required number of colors, the algorithm re-evaluates the situation.

Algorithm 2 BC

```
While some vertices remain do
    1. Let $\tilde{G}$ be the current reduced graph. Let $\tilde{n}$ be the number of vertices in $\tilde{G}$.
    2. If $\alpha(\tilde{G}) \leq \gamma(\tilde{n})$ do
        (a) Let $m$ be the minimum $x(v)$ in $\tilde{G}$. Assume without loss of generality that $V_1$ contains
            at least as much vertices $v$ with $x(v) = m$ as $V_2$.
        (b) Give the next $m$ colors to the remaining vertices in $V_1$, and after that, give the next
            $m$ colors to the remaining vertices in $V_2$.
    3. else $(\alpha(\tilde{G}) > \gamma(\tilde{n}))$
        (a) Choose a maximum independent set $I \subset V$ of size $\alpha(\tilde{G})$. Let $m$ be the minimum
            $x(v)$ value in $I$.
        (b) Give the next $m$ colors to all the vertices in $I$.
```

The algorithm runs in polynomial time, since finding a maximum independent set in a bipartite graph can be performed in polynomial time using flow techniques (cf., [GJ79]) and since in each iteration at least one vertex is deleted. We prove the following theorem in Appendix A.

Theorem 3.3 *Algorithm* BC *approximates p-SMC on bipartite graphs within a factor of* 1.5.

3.3 Bounded degree graphs and line graphs

A natural algorithm for the multi-coloring problem is the *greedy* (First-Fit) algorithm. It processes the vertices in an arbitrary order, assigning to each vertex $v \in V$ the set S_v of the smallest $x(v)$ colors with which none of its preceding neighbors have been colored. This method has the advantage of being *on-line*, processing requests as they arrive. Let Δ denote the maximum degree of G. The following theorem states the performance of Algorithm Greedy. The proof is omitted.

Theorem 3.4 Greedy *approximates p-SMC within a factor of* $\Delta + 1$. *Furthermore, there exist graphs for which this bound is tight.*

In order to get a better approximation factor, we consider a modified version of Greedy. The sorted greedy (SG) algorithm orders the vertices in a non-decreasing order by color requirements, and then applies Greedy. This slight change improves the approximation ratio by a factor of nearly 3.

Theorem 3.5 SG *provides a* $\frac{\Delta+2}{3}$*-approximation to* $\mathrm{pSMC}(G)$*, and that is tight.*

SG also has good approximation ratio for line graphs and intersection graphs of k-uniform hypergraphs.

Theorem 3.6 SG *provides a* $2 - 4/(\Delta + 4)$*-approximation to* $\mathrm{npSMC}(G)$ *on line-graphs. More generally, it provides a* $k(1 - 2(k-1)/(\Delta + 2k))$ *approximation ratio on intersection graphs of* k*-uniform hyper-graphs.*

The proofs of theorems 3.5 and 3.6 are given in Appendix A.

4 The non-preemptive sum multi-coloring problem

In the non-preemptive version of the sum multi-coloring problem the set of colors assigned to any vertex must be contiguous. Due to this restriction, we could not provide a constant approximation algorithm. Our algorithm has a logarithmic approximation factor. We describe our results for general graphs, bipartite graphs, and k-colorable graphs. Some of the proofs appear in Appendix A.

4.1 General graphs

Let r be the number of different color-requirements (lengths) in the graph. The following is a $\rho \cdot \min\{O(\log p), O(\log n), r\}$-approximation algorithm for the np-SMC problem.

Algorithm 3 *SameLengthIS*

```
Let $Small \leftarrow \{v \mid x(v) \le p/n^2\}$.
Color first the vertices of $Small$, arbitrarily but fully and non-preemptively.
Let $V'$ denote $V \setminus Small$.
$G' \leftarrow (V', E', x', w)$, where
   $x'(v) \leftarrow 2^{\lceil \log x(v) \rceil}$, for each $v \in V'$,
   $w(v) \leftarrow 1/x'(v)$, the weight of each $v \in V'$, and
   $E' \leftarrow E \cup \{(u,v) : x'(u) \ne x'(v)\}$
Apply weighted MAXIS to $G'$, coloring the vertices fully in each step.
```

Note that it suffices that MAXIS be able to work on induced subgraphs of G, since we can restrict attention to the subgraphs induced by vertices of the same length.

Theorem 4.1 *Assuming the maximum independent set can be approximated within ratio ρ, Algorithm* SameLengthIS *approximates* npSMC *within a factor of* $\min\{8\rho\log p, 16\rho\log n + 1.5\}$.

Observe that the algorithm produces a valid coloring, G' is a super-graph of G, and that it is contiguous since jobs are executed to completion. Now, consider the contribution of the vertices in $Small$. The color-sum of these vertices is bounded by $p/n^2(1+2+\ldots+n) < p/2$. We need also to include the delay incurred, namely the fact that the least color of the vertices of V' is not 1. Note, however, that the maximum color used for a vertex in $Small$ is no larger than p/n, hence the total "delay" for the vertices of V' is no larger than p. In total, the vertices of $Small$ cause an increase by a 1.5 factor when $p \geq n^2$. The theorem follows from the following two lemmas.

Lemma 4.2 $\mathrm{SMC}(G, \mathsf{SameLengthIS}) \leq 4\rho \cdot \mathrm{pSMC}(G')$.

Proof: Since the weights of G' are upper bounds on the weights of G, the cost of the coloring on G' is an upper bound of its cost on G. Observe that each color class is an independent set of maximum weight among independent sets in the current remaining graph. Thus, the coloring approximates p-SMC of G', within a factor of 4ρ, by Observation 3.1. □

Lemma 4.3 *Let* $z = \log(\min\{p, n^2\})$. *Then,* $\mathrm{pSMC}(G') \leq 2z \cdot \mathrm{pSMC}(G)$.

Proof: Any independent set of G is partitioned into at most z independent sets in G', one for each length class. In addition, the rounding up of the lengths at most doubles the cost of the coloring. □

Let r be the number of different color requirements. Clearly, a similar algorithm yields an $O(r)$ approximation. Also, the factor $\log p$ can be replaced by $\log q$, where q is the ratio between the largest to the smallest color requirement.

4.2 Bipartite graphs and k-colorable graphs

In this subsection we deal with the non-preemptive sum multi-coloring problem, for bipartite graphs and graphs where a k-coloring is given. The proof of the following theorem is given in Appendix A.

Theorem 4.4 *If G is bipartite, then we can approximate* $\mathrm{npSMC}(G)$ *within a* 2.796 *factor. If a k-coloring of a graph G is given, then we can approximate* $\mathrm{npSMC}(G)$ *within a* $1.55k + 1$ *factor.*

It is interesting to note that we can also obtain a contiguous coloring Ψ with a strong property: In the bipartite case, each vertex v can be completed by step $5 \cdot x(v)$ and it can be shown that 5 is the smallest such factor possible. This, for instance, implies that we approximate the makespan and the average completion time simultaneously within an absolute factor 5. For a graph where a k-coloring is given, each vertex v is completed by step $(ek+1)x(v)$.

5 The Co-Scheduling problem

Recall the definition of the co-scheduling problem. In this version, the multi-coloring first chooses an independent set I_1. It assigns colors to all the vertices in I_1 using the first x_1 colors where $x_1 = \max_{v\in I_1} x(v)$. Then, the multi-coloring cannot use the first x_1 colors for other vertices and therefore has to use colors x_1+1, x_2+2 etc. to color the rest of the vertices. The goal is to minimize the sum of all $f_\Psi(v)$.

We first show that Algorithm SameLengthIS of the previous section has a better performance for the co-SMC problem. It approximates the co-SMC problem within a constant factor as opposed to the logarithmic factor it achieves for the np-SMC problem. Then we construct a graph H for which coSMC$(H) = \Omega(\log p) \cdot$ npSMC(H). Such a graph indicates that this discrepancy in the performance is inherent.

5.1 General graphs

Theorem 5.1 SameLengthIS *approximates the Co-scheduling problem within a factor of $O(\rho)$, assuming the maximum independent set can be approximated within ratio $O(\rho)$.*

Proof: First, note that the algorithm colors in rounds, and thus produces a valid co-schedule. Now, each color class is an independent set of weight within ρ factor of maximum, among independent sets in the current remaining graph. Thus by Observation 3.1, the coloring approximates p-SMC of G', and thus also the coSMC, within a factor of 4ρ. Applying Lemma 4.2,

$$\mathrm{SMC}(G, \mathsf{SameLengthIS}) \leq \mathrm{SMC}(G', \mathsf{SameLengthIS}) \leq 4\rho \cdot \mathrm{pSMC}(G') \leq 4\rho \cdot \mathrm{coSMC}(G') \ .$$

We now relate coSMC(G') to coSMC(G). The optimal algorithm A_{OPT} may execute vertices of G of different length simultaneously. However, we can *simulate* A_{OPT} via an algorithm that at each round, schedules only jobs that belong to the same class in SameLengthIS (and therefore have roughly the same x value). The simulation is performed by going over the vertices according to their different lengths in increasing order of lengths. A single round of A_{OPT}, whose longest job is of length X, translates to separate rounds of total length at most $1+2+4+\cdots+X/2+X = 2X-1$. Hence, we get a simulation that at most doubles the length of the schedule, and each job is executed within double the time. Also, recall that rounding the lengths of the jobs to the nearest power of two at most doubles the length of the optimal schedule. Hence, coSMC$(G') \leq 4 \cdot$ coSMC(G), from which the theorem follows. □

5.2 A construction separating co-SMC and np-SMC

We have given an $O(1)$-approximations to the p-SMC problem and to the coSMC problem, while only a $\min\{O(\log p), O(\log n)\}$-approximation for the npSMC problem. It would be curious to know the precise relationships among these three models. We give a partial answer by constructing a graph H for which coSMC$(H) = \Omega(\log p) \cdot$ npSMC(H).

Consider the following graph with vertices $V = \{v_{i,j,k}\}$ for $i = 0, \ldots, \log p$, $j = 1, \ldots, 2^{\log p - i + 1}$, and $k = 1, \ldots, 2^i$. The vertex $v_{i,j,k}$ has a length 2^i. The edges are $E = \{(v_{i,j,k}, v_{i,j',k'}) : k \neq k'\}$. The graph has p vertices each of length $\ell = 2^i$, arranged in completely connected independent sets of size p/ℓ. Vertices of different lengths are not adjacent. Consider the straightforward non-preemptive coloring where each clique is colored independently. Observe that the sum of the lengths of vertices in each clique is p, thus the average of the completion times of vertices in the clique is $p/2 + x(v)/2$. The number of vertices of each length is p and the total sum of the lengths $\mathcal{S}(G) = p^2 \sum_{i=0}^{(\log p)/2} 1/4^i = (4/3)p^2$. Hence, the sum of this contiguous multi-coloring is $p \log p(p/2) + (2/3)p^2 = O(p^2 \log p)$.

On the other hand, any independent set contains at most 2^i vertices from each length class i. Hence, in any independent set of length ℓ in a co-schedule, there are at most 2ℓ vertices, or amortized 2 per step. If we ignore the first $2p$ vertices colored, the average completion time of the $\Omega(p(\log p))$ vertices is at least $(p-1)(\log p)/4$. The sum of the scheduling/coloring is thus at least $\Omega(p^2 \log^2 p)$. Hence, a $\Omega(\log p)$ separation between these models. As $n = p \cdot \log p$, this result also implies a $\Omega(\log n)-$separation.

References

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

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

[FK96] U. Feige and J. Kilian. Zero Knowledge and the Chromatic number. *Proc. of the 11th IEEE conference of Computational Theory,* pp. 278–287, 1996.

[GJ79] M. R. Garey and D. S. Johnson. Computers and intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, 1979.

[J97] K. Jansen. The Optimum Cost Chromatic Partition Problem. *Proc. of the Third Italian Conference on Algorithms and Complexity (CIAC '97).* LNCS 1203, 1997.

[K89] E. Kubicka. The Chromatic Sum of a Graph. PhD thesis, Western Michigan University, 1989.

[K91] H. A. Kierstead. A polynomial time approximation algorithm for Dynamic Storage Allocation. *Discrete Mathematics.*, 88:231–237, 1991.

[KKK89] E. Kubicka and G. Kubicki, and D. Kountanis. Approximation Algorithms for the Chromatic Sum. *Proceedings of the First Great Lakes Computer Science Conf.*, Springer LNCS 507, pp. 15–21, 1989.

[KS89] E. Kubicka and A. J Schwenk. An Introduction to Chromatic Sums. *Proceedings of the ACM Computer Science Conf.*, pp. 39-45, 1989.

[L81] N. Lynch. Upper Bounds for Static Resource Allocation in a Distributed System. *J. of Computer and System Sciences*, 23:254–278, 1981.

[MPT94] R. Motwani, S. Phillips, and E. Torng. Non-Clairvoyant Scheduling. *Theoretical Computer Science*, 130:17–47, 1994.

[NS93] M. Naor and L. Stockmeyer. What Can be Computed Locally? *Proceedings of the Twenty Fifth Annual Symposium on the Theory of Computing*, pp. 184–193, 1993.

[NSS92] S. Nicoloso, M. Sarrafzadeh, and S. Song. On the Wire-Length Minimization Problem. Manuscript, 1992.

[NSS94] S. Nicoloso, M. Sarrafzadeh, and X. Song. On the Sum Coloring Problem on Interval Graphs. Istituto di Analisi dei Sistemi ed Informatica (IASI-CNR), R. 390, 1994.

[SG98] A. Silberschatz and P. Galvin. Operating System Concepts. Addison-Wesley, 5th Edition, 1998.

[SP88] E. Steyer and G. Peterson. Improved Algorithms for Distributed Resource Allocation. *Proceedings of the Seventh Annual Symposium on Principles of Distributed Computing*, pp. 105–116, 1988.

[TEA+89] C. Thomassen, P. Erdös, Y. Alavi, J. Malde, and A. J. Schwenk. Tight Bounds on the Chromatic Sum of a Connected Graph. *J. of Graph Theory,* 13:353–357, 1989.

[T95] A. S. Tannenbaum, *Distributed Operating Systems.* Prentice-Hall Int., Editing, 1995.

A Some Proofs

A.1 Proof of Theorem 3.3

Let Z, W be two subsets of the vertices, such that $Z \subseteq W \subseteq V$. Let $G(Z)$ and $G(W)$ be the graphs induced by Z and W. Suppose we define two (separate) instances of p-SMC on $G(W)$ and $G(Z)$ by defining $x_Z(v)$ values for each $v \in Z$, and $x_W(v)$ value for each $v \in W$. Next assume that for each $v \in Z$, $x_W(v) \geq x_Z(v)$. (Note that in this scenario we may assume $Z = W$, and assign $x_Z(v) = 0$ values to vertices that were previously in $W \setminus Z$.) Let $\Delta = \mathcal{S}(G(W)) - \mathcal{S}(G(Z))$.

Observation A.1 $\text{pSMC}(G(W)) \geq \text{pSMC}(G(Z)) + \Delta$

Proof: First suppose that $G(Z)$ and $G(W)$ differ only in one vertex v and in one unit. Namely, $x_W(v) = x_Z(v) + 1$, and all $u \neq v$, $x_W(u) = x_Z(u)$. Let OPT_W be an optimum multi-coloring for $G(W)$. Construct a multi-coloring Ψ for $G(Z)$ as follows. Use OPT_W, except that remove v from the last independent set v has appeared in, in OPT_W. Clearly, in this way $f(v)$ is reduced by *at least* 1 unit. The coloring Ψ is feasible for $G(Z)$. Hence, we get that $\text{pSMC}(G(W)) \geq \text{pSMC}(G(Z)) + 1$. Now, the observation is deduced by repeatedly applying the above. $\square$

For the next observation, let Ψ be some multi-coloring. Let $V' \subseteq V$. Suppose that we distribute the first i colors of Ψ to each vertex. Further suppose that by this assignment only the vertices of V' are (partially) colored. Further assume that for all $v \in V'$, $x(v) \geq i$. Let G' be the resulting reduced graph. Finally, let $\text{SMC}(G', \Psi)$ be the multi-sum resulting from applying Ψ to G'. Namely, using the same independent sets as Ψ (without the first i independent sets) starting with color 1 instead of $i + 1$. Then

Observation A.2 $\text{SMC}(G, \Psi) = n \cdot i + \text{SMC}(G', \Psi)$.

Proof: In $\text{SMC}(\Psi, G')$ we have the sum of maximum colors, under the assumption that the Ψ starts coloring the vertices of G' with color 1. Compared with $\text{SMC}(\Psi, G)$, one must add to *every* vertex i units since when the Ψ operates on G, it starts multi-coloring G' only with colors $i + 1$ or more.. (*Each* vertex is in a sense "delayed" i units, because $x(v) \geq i$ for every $v \in V'$.) $\square$

We prove the theorem by induction on $\mathcal{S}(G)$. The basis of the induction is with $\mathcal{S}(G) = 1$. Since we assume color requirements $x(v)$ to be non-negative integers, $\mathcal{S} = 1$ implies that the graph contains only a single vertex whose color requirement is 1. BC is optimal in such a case.

Now, for large values of S, let us first assume that:

Case 1: $\alpha(G) \leq \gamma(n)$.

In this case, without loss of generality, the algorithm gives the first m colors to the vertices of V_1 and the next m colors to the vertices of V_2. Let $V_1(m)$ be the subset of vertices of V_1 whose x value is m. Let $n_1(m) = |V_1(m)|$. Let G' be the reduced graph resulting after the first $2m$ colors are distributed. Clearly, G' is a graph where all the x values are reduced by m, and all the vertices with $x(v) = m$ were deleted. That is, for all v the residual color-requirement $x'(v)$ of v is $x'(v) = x(v) - m$. Similarly as in Observation A.2 it can be shown that:

$$\text{SMC}(G, BC) = (2n - n_1(m))m + \text{SMC}(G', BC) \tag{3}$$

Equation (3) follows since when comparing $\text{SMC}(G, BC)$ and $\text{SMC}(G', BC)$, one most add the m units of "delay" of all the vertices of G in the first m colors, and in the next m colors all the vertices in $V \setminus V_1(m)$ were delayed for an extra quantity m.

Now, let G^*_m be the reduced graph resulting after the first m colors of an optimum multi-coloring. Since $x(v) \geq m$ for all v, we have by Observation A.2 that

$$\text{pSMC}(G) = mn + \text{pSMC}(G^*_m). \tag{4}$$

Let $x_O(v)$ be the color requirement remaining for v after the first m colors of the optimum multi-coloring. In G^*_m, each $x(v)$ was reduced by at most m, that is, $x_O(v) \geq x(v) - m$. Since in G' *every* $x(v)$ was reduced by m, we have that: $x'(v) \geq x_O(v)$. Therefore, we are in the situation of Observation A.1. Clearly, the sum of color requirement in G^*_m, $S(G^*_m)$, is at least $S - m\alpha$, since S is reduced by at most α in each of the first m colors. On the other hand, $S(G') = S - mn$, since in G' we have reduced the color requirement of *each* vertex in G by m. Thus,

$$\Delta = S(G^*_m) - S(G') \geq (n - \alpha)m \tag{5}$$

Combining Equations (4) and (5), and using Observation A.1 we get that

$$\text{pSMC}(G) \geq (2n - \alpha)m + \text{pSMC}(G') \tag{6}$$

We now may apply the induction hypothesis and get that

$$\frac{\text{SMC}(G', BC)}{\text{pSMC}(G')} \leq 1.5 - 1/(2n) \tag{7}$$

Now, by the assumption that $\alpha(G) \leq \gamma(n)$, it easily follows that:

$$\frac{(2n - n_1(m)) \cdot m}{(2n - \alpha) \cdot m} \leq \frac{(2n-1)m}{(2n - \alpha)m} \leq 1.5 - 1/(2n) \tag{8}$$

Combining Equations (3), (4), (7) and (8) we get that $\text{SMC}(G, BC)/\text{pSMC}(G) \leq 1.5 - 1/(2n)$, as required.

We now deal with the second case.

Case 2: $\alpha(G) > \gamma(n)$.

In this case, let m be the minimum x value in the maximum independent set I that was chosen. BC gives the first m colors to the vertices of I. Let G' be the reduced graph resulting after these m first colors of BC are distributed. According to Observation A.2 (with $V' = I$) we have that

$$\text{SMC}(G, BC) = nm + \text{SMC}(G', BC) \tag{9}$$

Now, clearly for every v, $x(v) \geq x'(v)$. Also, $S(G') = S - \alpha m$. Since $S(G) - S(G') \geq \alpha m$, by Observation A.1 we have that

$$\text{pSMC}(G) \geq \alpha m + \text{pSMC}(G') \tag{10}$$

Now, by the induction hypothesis, we may assume that Equation (7) holds with our new definition of G'. Also, since we are in case 2,

$$\frac{nm}{\alpha m} \leq 1.5 - 1/(2n) \tag{11}$$

Therefore, the required ratio follows from Equations (7),(9), (10) and (11). This completes the proof. $\square$

A.2 Proof of Theorem 3.5

First observe that each edge can delay one vertex; the delay is the amount of colors that is the smaller between the color requirements of its endpoints. Thus, the multi-chromatic sum of Sorted Greedy is bounded by

$$\text{SMC}(G, \mathsf{SG}) \leq \mathcal{S}(G) + \sum_{(u,v)\in E} \min(x(u), x(v)). \tag{12}$$

Now, assume that the maximal degree in G is $\Delta > 1$. Let $(u,v) \in E$. Denote by $D_u(v)$ the number of colors given to u, that are smaller than $f(v)$. Clearly,

$$\mathrm{pSMC}(G) \geq \mathcal{S}(G) + \sum_{v \in V} \left(\max_{u \in N(v)} \{D_u(v)\} \right) \geq \mathcal{S}(G) + \sum_{v \in V} \left(\frac{\sum_{u \in N(v)} D_u(v)}{\Delta} \right)$$

In the above, each edge $e = (u,v)$ contributes $(D_u(v) + D_v(u))/\Delta$ to the sum. Since it is easy to verify that $D_u(v) + D_v(u) \geq \min\{x(u), x(v)\}$ it follows that

$$\mathrm{pSMC}(G) \geq \mathcal{S}(G) + \sum_{(u,v) \in E} \min(x(u), x(v))/\Delta \,. \tag{13}$$

Let

$$d = \frac{2\sum_{(u,v)\in E} \min(x(u), x(v))}{\mathcal{S}(G)}. \tag{14}$$

Then,

$$\frac{\mathrm{SMC}(G, \mathsf{SG})}{\mathrm{pSMC}(G)} \leq f(d) \equiv \frac{1 + d/2}{1 + d/(2\Delta)} \tag{15}$$

Since $f(d)$ is monotone increasing, and $d \leq \Delta$, we have

$$f(d) \leq f(\Delta) = \frac{1 + \Delta/2}{3/2} = \frac{2 + \Delta}{3}.$$

A matching lower bound was shown in [BBH$^+$98] for the MCS problem; thus, it also applies to the pSMC.

For the non-preemptive case, np-SMC, a vertex v can be delayed not only by the lengths of its neighbors but also by "gaps" in the set of available colors that are too small for coloring v contiguously. There can be as many gaps as neighbors, and each gap can be of length $x(v) - 1$. Hence,

$$\mathrm{SMC}(G, \mathsf{SG}) \leq \mathcal{S}(G) + \sum_{(u,v)\in E} (x(u) + x(v)) \leq (\Delta + 1)\mathcal{S}(G) + \sum_{(u,v)\in E} \min(x(u), x(v)). \tag{16}$$

The performance ratio is then, by (16) and (13), at most $\Delta + 1$. $\square$

A.3 Proof of Theorem 3.6

Given a line graph G, form a graph H that is a disjoint collection $\{C_1, C_2, \ldots, C_{|V(H)|}\}$ of the maximal cliques in G. Add a singleton clique for each vertex that appears only once.

The minimum contiguous multi-coloring sum of H is given by ordering the vertices of each C_i in a non-decreasing order of color requirements, for

$$\mathrm{pSMC}(H) = \mathcal{S}(H) + \sum_{(u,v)\in E(H)} \min(x(u), x(v)).$$

Observe that since each vertex in G appears at most twice in H, $\mathcal{S}(H) = 2\mathcal{S}(G)$, and any multi-coloring of G corresponds to a multi-coloring of H of at most double the weight, $\mathrm{pSMC}(H) \leq 2 \cdot \mathrm{pSMC}(G)$. Further, there is a one-one correspondence between the edges of G and H. Thus, we have

$$\mathrm{pSMC}(G) \geq \mathcal{S}(G) + \frac{1}{2} \sum_{(u,v)\in E(G)} \min(x(u), x(v)). \tag{17}$$

Using d as defined in (14), we bound the performance ratio by

$$\frac{\mathrm{SMC}(G, \mathsf{SG})}{\mathrm{pSMC}(G)} \leq g(d) \equiv \frac{1 + d/2}{1 + d/4} \ .$$

Since $f(d)$ is monotone increasing, and $d \leq \Delta$, we have

$$f(d) \leq f(\Delta) = 2 - 4/(\Delta + 4) \ .$$

This matches the bound proved for sum coloring for regular edge graphs. □

We note that a stronger result indeed is apparent from the proof. The *total delay* of a multicoloring is the sum of the finishing times less the sum of the color requirements and the *average delay* is the total delay divided by the number of jobs. In the unit-time case, this corresponds to a coloring where the index of the first color is 0 instead of 1. We can observe from the above proof that the Sorted Greedy algorithm finds a coloring of line graphs that approximates the average delay by a factor of 2.

We can generalize the argument for line graphs to intersection of set systems where each set is of size at most k. These are $k+1$-*claw* free graphs, containing no induced star with $k+2$ nodes. They include various classes of geometric graphs, e.g. unit-disk graphs are 6-claw free. We now have that each vertex is contained in at most k maximal cliques. Thus, as similar argument shows that

$$\mathrm{pSMC}(G) \geq \mathcal{S}(G) + \frac{1}{k} \sum_{(u,v) \in E} \min(x(u), x(v)) \ ,$$

and thus

$$\mathrm{SMC}(G, \mathsf{SG}) \leq k \left(1 - \frac{2(k-1)}{\Delta + 2k}\right) \mathrm{pSMC}(G) \ .$$

A.4 Proof of Theorem 4.4

We first illustrate our approach by an example. Given a bipartite graph G, we color the vertices into sets C_1 and C_2. Let $C_i[a, b]$ denote the set of vertices in C_i whose length is between a and b, inclusive. The idea is as follows:

> Color all vertices in $C_1[1, 1]$ with 1 color, those in $C_2[1, 2]$ with the next 2 colors, then $C_1[2, 4]$ with the next 4 colors, and $C_2[3, 8]$ with the following 8 colors.
>
> In general, color $C_1[2^{2i-2} + 1, 2^{2i}]$, followed by $C_2[2^{2i-1} + 1, 2^{2i+1}]$, for $i = 0, 1, \ldots$.

For a given vertex v in say C_1, the worst case occurs when $x(v) = 2^{2i} + 1$, for some i. Then, v is finished in step $x(v) + \sum_{j=0}^{i} (2^{2j} + 2^{2j+1}) = x(v) + 2^{2i+2} - 1 = 5x(v) - 5$. The same can be argued for any $u \in C_2$. Thus, we have bounded the *worst case* completion time of *any* vertex by a factor of 5, giving not only the system a guarantee on the schedule completion but also each player.

One idea to improve this schedule is to swap the roles of C_1 and C_2. That is, start with coloring $C_2[1, 1]$, followed by $C_1[1, 2]$ etc. The worst case for the first schedule, a vertex v in C_1 of length $x(v) = 2^{2i} + 1$, now gets completed in fewer than $3x(v)$ steps. This, in fact, holds for any vertex v, that the sum of the completion times in the two schedules is at most $8x(v)$. Hence, the better of the two (which we can test), has an overall cost of at most $4 \cdot \mathcal{S}(G)$.

This idea can be taken further. We still see that bad instances have vertices with quite particular color requirements. If we could uniformly vary the locations of the intervals, e.g. instead of $C_1[9, 16]$ use $C_1[11, 18]$, we may do better on average. This is most naturally done by randomly selecting a starting point, from which the doubling steps are taken. Given the exponential growth in the steps, the distribution from which the point is drawn should be uniform in its *logarithm*. Finally, instead of doubling, we evaluate the optimal step size to use.

Let G be a graph whose k-coloring $C_1, C_2, \ldots C_k$ is given. Let a be a constant, and let $d = a^k$. Let $C_i[x, y]$ denote the set of vertices in C_i of lengths in the interval $[x, y]$.

Our algorithm is as follows:

```
**Steps**$(G, a)$
  Let $X$ be a random number uniformly chosen from $[0, 1]$.
  $d \leftarrow a^k$
  $Y \leftarrow d^X$.
  for $i \leftarrow 0$ to $\log_d p$ do
    for $j \leftarrow 1$ to $k$ do
      $A_{ij} \leftarrow d^{i-1} a^j Y$
      Color vertices of $C_j[A_{ij}/d, A_{ij}]$ using the next $\lfloor A_{ij} \rfloor$ colors
```

The coloring cost of a vertex v is intimately related to the length of the interval of color requirements in which v is colored. Let the size of the interval refer to the upper bound on the color requirement of vertices colored in that interval. The proof of the following lemma is omitted.

Lemma A.3 *Let ℓ_v be a random variable representing the size of the interval within which vertex v was colored. Then, its its density function is given by*

$$F_v(z) \equiv Pr[\ell_v = z \cdot x(v)] = \frac{1}{\ln d} \cdot \frac{1}{z} x(v)$$

on its domain $[1, d]$.

Theorem A.4 *The expected multi-color sum of* Steps *is at most $(1.544k + 1)\mathcal{S}(G)$ on a k-colored graph G, and at most $2.796\mathcal{S}(G)$ on a bipartite graph G.*

Proof: Observe, that if v is colored in interval of size ℓ_v, then the *delay time* d_v of v, or the time until v starts to get colored, is at most

$$d_v \leq \sum_{i=1}^{\infty} \frac{\ell_v}{a^i} = \ell_v \frac{1}{a-1}.$$

¿From Lemma A.3,

$$E[\ell_v] = \int_1^d z \cdot F_v(z)\, dz = \left[\frac{1}{\ln d}\right]_1^d x(v) = \frac{d-1}{\ln d} x(v).$$

Thus,

$$E[d_v] \leq \frac{d-1}{\ln d} \cdot \frac{1}{d^{1/k} - 1} \cdot x(v). \tag{18}$$

For bipartite graphs, this implies that

$$E[d_v] \leq \frac{a^2 - 1}{2 \ln a} \cdot \frac{1}{a-1} \cdot x(v) \leq \frac{a+1}{2 \ln a} x(v).$$

The function $f(x) = \frac{x+1}{\ln(x)}$ is minimized when $x = 3.5911$, which results in the bound $E[d_v] \leq 1.796 x(v)$. The expected cost of the coloring is thus

$$E[\mathrm{SMC}(G, \mathsf{Steps}(G, 3.5911))] = \sum_v E[d_v] + \mathcal{S}(G) \leq 2.796\, \mathcal{S}(G).$$

For k-colored graphs, we can use that $1+x \le e^x$ to bound $d^{1/k} - 1 \ge (\ln d)/k$, obtaining from (18) that

$$E[d_v] \le \frac{d-1}{\ln d} \cdot \frac{k}{\ln d} \cdot x(v).$$

The function $g(x) = \frac{x-1}{\ln^2 x}$ takes a minimum at $x = 4.9215$, resulting in $E[d_v] \le 1.544k \cdot x(v)$. Hence,

$$E[\mathrm{SMC}(G, \mathsf{Steps}(G, 4.9215^{1/k}))] \le (1.544k + 1)\,\mathcal{S}(G).$$

$\square$

Steps can be derandomized, by examining a set of evenly spaced candidates for the random number X. The additive error term will be inversely proportional to the number of schedules evaluated.

We can also bound the delay from a worst-case perspective. For each vertex v, it holds that $\ell_v \le d \cdot x(v)$, since $A_{i+1,j}/A_{ij} = d$. It follows that

$$d_v \le \frac{d}{d^{1/k} - 1} x(v) \le \frac{d \cdot k}{\ln d} x(v).$$

This is minimized when $d = e$, for a worst case bound of

$$f_\Psi(v) \le (ek+1)x(v), \qquad \text{for each vertex } v.$$

B The MAXIS heuristic

In this appendix we first discuss the exact performance ratio of MAXIS for the sum-coloring problem. We are able to show a (rather evolved) example on which the MAXIS algorithm has performance ratio of 4. This matches an upper bound of 4 proven in [BBH+98]. Our construction exploits the weaknesses of MAXIS by forcing it to choose the independent sets across the color sets of the optimal solution, using considerably more colors than necessary. We then discuss the performance ratio of MAXIS for the sum multi-coloring problems. We show that in the sum multi-coloring case MAXIS performance is not as "good" as in the sum coloring case.

B.1 A 4 lower bound for MAXIS algorithm

We prove the following theorem:

Theorem B.1 *The performance ratio of* MAXIS *on the sum coloring problem is exactly* 4*, up to low order terms.*

B.1.1 A chopping procedure

We represent a coloring of a graph G by k colors as a tuple of length k: $\langle c_1, \ldots, c_k \rangle$. The size of the set C_i of the vertices colored by i is c_i. Note that without loss of generality, in any solution to the sum coloring problem we have $c_1 \ge c_2 \ge \cdots \ge c_k$. Otherwise we could switch color classes and get a better sum coloring. By definition, for a given pattern $P = \langle c_1, \ldots, c_k \rangle$ the sum coloring is $SC(P) = \sum_{i=1}^{k} i \cdot c_i$.

Given a pattern $P = \langle c_1, \ldots, c_k \rangle$, we describe a *chopping* procedure which constructs a graph G_P with $n = \sum_{i=1}^{k} c_i$ vertices. In each step, we observe the minimum size for a maximum independent set. Then the chopping procedure forces MAXIS to pick a maximum independent set of this size such that in later steps MAXIS will find smaller sets.

In G_P, there are k independent sets $C_1, \ldots, C_k$ that cover all the vertices of the graph such that $|C_i| = c_i$. Hence, there exists a coloring whose representation is $\langle c_1, \ldots, c_k \rangle$. We now place the vertices of the graph in a matrix of size $c_1 \times k$. The ith column contains c_i ones at the bottom and $c_1 - c_i$ zeros at the top. Each vertex is now associated with a one entry in the matrix.

The chopping procedure first constructs an independent set I_1 of size c_1. It collects the vertices from the matrix line after line from the top line to the bottom line. In each line, it collects the vertices from right to left. Each one entry that is collected becomes a zero entry. Then it adds edges from I_1 to all the other vertices as long as these edges do not connect two vertices from the same column. In a same manner the procedure constructs I_2. The size of I_2 is the number of ones in the first column after the first step. At the beginning of the ith step, the procedure already constructed $I_1, \ldots, I_{i-1}$, defined all the edges incident to these vertices, and replaced all the one entries associated with the vertices of the sets $I_1, \ldots, I_{i-1}$ by zeros. During the ith step, the chopping procedure constructs in a similar manner the independent set I_i the size of which is the number of ones in the first column of the matrix at the beginning of the step. Again, each one entry that is collected becomes a zero entry. Then the procedure connects the vertices of I_i with the remaining vertices in the matrix as long as these edges do not connect two vertices from the same row. The procedure terminates after h steps when the matrix contains only zeros.

In the resulting graph, two vertices are connected unless they both belong to some C_i for $1 \leq i \leq k$ or both belong to some I_j for $1 \leq j \leq h$. Moreover, these $k + h$ sets are the only maximal independent sets in the graph. To see the last claim, observe first that due to the way I_j was constructed it follows that $C_i = I_j$ only if they contain one vertex that is connected to all the other vertices in the graph. Next observe that if $|I_j| \geq 2$, then there exist $u \in I_j \cap C_{i_1}$ and $v \in I_j \cap C_{i_2}$ for some $i_1 \neq i_2$. Therefore we cannot add another vertex to I_j because this vertex is either connected to u or to v. A similar argument could be applied for the case $|C_i| \geq 2$.

We need to show that I_j is one of the maximum independent sets in the graph induced by the vertices $\cup_{j \leq \ell \leq h} I_\ell$ denoted by G_j. To see this, observe that because $|C_1| \geq |C_2| \geq \cdots \geq |C_k|$ it follows that $I_1| \geq |I_2| \geq \cdots \geq |I_h|$. This and the maximality of the independent sets $C_1, \ldots, C_k, I_1, \ldots, I_h$ imply that the maximum independent set in G_j is either I_j or $C_1 \cap G_j$. The claim follows since I_j was chosen to be of the same size as $C_1 \cap G_j$.

After each step of the chopping procedure, the remaining ones in the matrix could be represented by $P_i = \langle c_1^i, \ldots, c_k^i \rangle$ where P_0 is the original pattern P. With this notation, in the ith step, the procedure creates the independent set I_i of size c_1^{i-1}. Note that since the first row represents at this stage an independent set of size c_1^{i-1}, MAXIS must find an independent set of at least this size. The chopping algorithm forces it to choose an "horizontal" set in order to force smaller size sets in the next steps.

The chopping procedure creates another partition of G into independent sets the pattern of which is $A(P) = \langle |I_1|, \ldots, |I_h| \rangle$ for some $h \geq k$. The pattern $A(P)$ is lexicographically less than the pattern P. Therefore, the sum coloring associated with $A(P)$ is greater than the one associated with P. In the above notations, $A(P) = \left\langle c_1^0, \ldots, c_1^{h-1} \right\rangle$.

We sometimes terminate the chopping procedure before the matrix contains only zeros. At some stage, we take as maximum independent sets the k columns from left to right. Assume that at the end of the ith step we already constructed the pattern $\left\langle c_1^0, \ldots, c_1^{i-1} \right\rangle$. Then the final pattern will be $A(P) = \left\langle c_1^0, \ldots, c_1^{i-1}, c_1^i, \ldots, c_k^i \right\rangle$.

Example: Suppose $P = \langle 8, 8, 1 \rangle$. Then I_1 contains 4 vertices from C_1 and 4 vertices from C_2. We are left with a pattern $\langle 4, 4, 1 \rangle$. Then I_2 contains 2 vertices from C_1 and 2 vertices from C_2 and we are left with the pattern $\langle 2, 2, 1 \rangle$. Then I_3 contains 2 vertices and I_4, I_5, and I_6 each contains one vertex. Therefore $A(P) = \langle 8, 4, 2, 1, 1, 1 \rangle$.

B.1.2 The $4 - o(1)$ Lower bound

To achieve the tight bound, we build a pattern with special properties, to be used in the chopping procedure. We let the first two entries be equal, and after two steps of chopping they should be equal to the third entry. After two additional chopping steps we want the first four entries to be equal, and so on.

Small examples of such patterns are $\langle 4, 4, 1 \rangle$ for $k = 3$ and $\langle 36, 36, 9, 4 \rangle$ for $k = 4$. Note that for $k = 3$, MAXIS produces the pattern $\langle 4, 2, 1, 1, 1 \rangle$ and already the approximation ratio is at least $20/15 > 1.333$. For $k = 4$, MAXIS produces the pattern $\langle 36, 18, 9, 6, 4, 3, 3, 2, 1, 1, 1, 1 \rangle$ and then the ratio is $240/151 > 1.589$.

More formally, for $x > 1$, consider the following pattern:

$$LB4 = \left\langle x, x, \frac{x}{4}, \frac{x}{9}, \ldots, \frac{x}{(k-1)^2}, \frac{x}{k^2} \right\rangle .$$

It follows that $n = \left(1 + \sum_{i=1}^{k} \frac{1}{i^2}\right) x$. We choose x such that $LB4$ contains only integral numbers (e.g., $x = (k!)^2$). The sum coloring of the pattern $LB4$ is:

$$SC(LB4) = x + \sum_{i=1}^{k} (i+1)\frac{x}{i^2} = x + \sum_{i=1}^{k} \frac{x}{i} + \sum_{i=1}^{k} \frac{x}{i^2} < (H_k + 2.65)x .$$

Recall that $\sum_{i=1}^{k} \frac{1}{i^2} < 1.65$, and that H_k denotes $\sum_{i=1}^{k} \frac{1}{i}$.

In the chopping procedure, once we arrive at a pattern of equal size we just take these $k+1$ columns as the next $k+1$ entries. We get that

$$A(LB4) = \left\langle x, \frac{x}{2}, \frac{x}{4}, \frac{x}{6}, \frac{x}{9}, \frac{x}{12}, \ldots, \frac{x}{(k-1)^2}, \frac{x}{(k-1)k}, \frac{x}{k^2}, \ldots, \frac{x}{k^2} \right\rangle ,$$

where $\frac{x}{k^2}$ appears $k+1$ times. The number of vertices in $A(LB4)$ is $\sum_{i=1}^{k} \frac{x}{i^2} + \sum_{i=1}^{k-1} \frac{x}{i(i+1)} + k\frac{x}{k^2}$. Since $\frac{x}{i(i+1)} = \frac{x}{i} - \frac{x}{i+1}$, it follows that $\sum_{i=1}^{k-1} \frac{x}{i(i+1)} = x - \frac{x}{k}$. Therefore, the number of vertices in $A(LB4)$ is equal to the number of vertices in $LB4$. The sum coloring of the pattern $A(LB4)$ is:

$$\begin{aligned} SC(A(LB4)) &= \sum_{i=1}^{k} \frac{2i-1}{i^2} x + \sum_{i=1}^{k-1} \frac{2i}{i(i+1)} x + \sum_{i=1}^{k} (2k-1+i)\frac{1}{k^2} x \\ &= \left(2\sum_{i=1}^{k} \frac{1}{i} - \sum_{i=1}^{k} \frac{1}{i^2} + 2\sum_{i=1}^{k-1} \frac{1}{i+1} + \frac{2k-1}{k} + \sum_{i=1}^{k} \frac{i}{k^2} \right) x . \end{aligned}$$

Since $\sum_{i=1}^{k-1} \frac{1}{i+1} = \sum_{i=1}^{k} \frac{1}{i} - 1$ and since $\sum_{i=1}^{k} \frac{i}{k^2} = 1 + \frac{1}{k}$, it follows that

$$SC(A(LB4)) = 4\sum_{i=1}^{k} \frac{1}{i} + 1 - \sum_{i=1}^{k} \frac{1}{i^2} > (4H_k - 0.65)x .$$

It follows that the approximation ratio of MAXIS is

$$r > \frac{(4H_k - 0.65)x}{(H_k + 2.65)x} = 4 - o(1) .$$

Note that only for $k \geq 8290$ does the value of r exceed 3. Furthermore, $r \approx 3.9475$ for $k = 10^{100}$ and $r \approx 3.5334$ for $k = 10^{10}$.

B.2 Lower bounds for MAXIS on sum multi-coloring of graphs

In the preemptive case there are several possible definitions for MAXIS. One natural definition for MAXIS is to repeat as long as needed the following procedure: color a maximum independent set with one color, then reduce the requirements of all the vertices in the chosen set by one. The following example shows that such an algorithm has an approximation ratio of $\Omega(\sqrt{p})$.

Consider two vertices with requirement p that are connected to a clique of size $\sqrt{p}$ in which all vertices have requirement 1. MAXIS colors first the two special vertices with p colors and then colors in a sequence each vertex in the clique with one color. This yields a multi-chromatic sum of $p(2+\sqrt{p}) + \binom{\sqrt{p}}{2}$ which is $\Omega(p\sqrt{p})$. On the other hand, the optimal solution schedules the clique vertices first. This yields a multi-chromatic sum of $\binom{\sqrt{p}}{2} + 4p = O(p)$. Thus, the approximation ratio is $\Omega(\sqrt{p})$.

In the co-scheduling variant, the above procedure is the most natural way to generalize MAXIS and the counter example holds also in this model. However, it is not difficult to verify that this algorithm is a $\min\{4p, n\}$ approximation for the co-SMC problem. This turns out to be best possible even for trees. Consider the following graph on $n = 2t+1$ vertices, with $E = \{v_i v_{t+i}, v_n v_{t+i} : i = 1, \ldots, t\}$. All vertices have unit color requirement except v_n that requires p colors. MAXIS will first color v_n and the first t vertices, with the second t vertices delayed until color $p+1$. The cost of this coloring is $t(p+1) + t + p \geq np/2$. The optimal solution colors the first t vertices with color 1, the next t vertices with color 2, and the last vertex with colors $3, \ldots, p+2$. The cost of this coloring is $3t + (p+2) \leq 2(n+p)$. Hence, the approximation ratio is $\Omega(\min(n,p))$.

In the non-preemptive case, the algorithm may do still worse. Whenever the set of currently colored vertices becomes a non-maximal independent set, MAXIS in this model would find the largest set of vertices that the solution can be extended with. Consider the graph G given by three independent sets A, B, C. The edges between A and B are given by $\{(a_i, b_j) : |i-j| > 1\}$, while vertices of C are connected to all vertices of A and B except a_1 and b_1. The vertices of A and B have color requirement 2, while the vertices of C have color requirement 1.

Observe that $C \cup \{a_1, b_1\}$ forms the only maximum independent set in the graph, hence it gets colored first. In step 2, only a_1 and b_1 are scheduled, and the only vertices adjacent to neither vertex are a_2 and b_2. In step 3, similarly, a_1 and b_1 have been completed and a_2 and b_2 are still scheduled. Therefore, only a_3 and b_3 can be added. In general, vertices a_i and b_i start to get colored in step i. The cost of the schedule is then $t + 2\binom{t+2}{2} - 1 = \Omega(t^2)$. The optimal coloring colors first the vertices of C, followed by A and B, for a total cost of $t + 3t + 5t = O(t)$. Hence, the approximation ratio is $\Omega(t)$, which grows independent of p.