Available online at www.sciencedirect.com

SCIENCE DIRECT®

Operations Research Letters 33 (2005) 242–248

Operations Research Letters

www.elsevier.com/locate/dsw

The maximum saving partition problem

Refael Hassin[a], Jérôme Monnot[b,*]

[a]Department of Statistics and Operations Research, Tel-Aviv University, Tel Aviv 69978, Israel
[b]LAMSADE CNRS, Université Paris-Dauphine, France

Received 3 March 2004; accepted 22 July 2004
Available online 6 October 2004

Abstract

The input to the MAXIMUM SAVING PARTITION PROBLEM consists of a set $V=\{1,\dots,n\}$, weights w_i, a function f, and a family $\mathscr{S}$ of feasible subsets of V. The output is a partition $(S_1,\dots,S_l)$ such that $S_i \in \mathscr{S}$, and $\sum_{j\in V} w_j - \sum_{i=1}^{l} f(S_i)$ is maximized. We present a general $\frac{1}{2}$-approximation algorithm, and improved algorithms for special cases of the function f.

Keywords: Partitioning; Color saving; Approximation algorithms; NP-complete

1. Introduction

Consider the following scheduling problem. Jobs (or *items*) from a given set V have to be assigned (or *packed*) to a set of identical machines for processing. Owing to various constraints, the feasible assignments to a single machine, are constrained to a family $\mathscr{S}$ of *feasible* subsets. $\mathscr{S}$ is an *independence* or *hereditary* system. This means that if a subset S of jobs can be assigned to a machine (i.e., $S \in \mathscr{S}$), then all the subsets of S are in $\mathscr{S}$. The cost of assigning a subset $S \in \mathscr{S}$ to a machine is $f(S)$, and it is a function of individual job parameters w_i $i \in S$. For example, w_i may denote the skill level required to process job i, and $f(S)=\max\{w_i | i \in S\}$ is the skill level required to process the subset S on a single machine. The problem is to partition V into feasible subsets so that the total cost is minimized.

In an interesting special case, the feasible subsets are defined solely by *pairwise* compatibility relations. These relations can be defined by a graph, where an edge indicates that its two ends are not compatible. The feasible sets are then the independent sets of the graph. In this case, our problem is a node coloring problem, in which the cost of a legal coloring is the sum of costs of its color classes as prescribed by f. Even this special case is very hard to solve or even to approximate since it generalizes the NODE COLORING PROBLEM where $f(S)=1$ for every $S \in \mathscr{S}$.

The coloring case with $f(S)=\max\{w_i | i \in S\}$ has been studied in [5], where it is proved that this problem is NP-hard even in bipartite and other restricted families of graphs. Other types of weighted coloring

* Corresponding author.
E-mail addresses: hassin@post.tau.ac.il (R. Hassin), monnot@lamsade.dauphine.fr (J. Monnot).

0167-6377/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.orl.2004.07.007

and partitioning problems are studied for instance in [1,3,4,12,13].

It is natural to assume that f is sub-additive, i.e., for any two disjoint subsets S and T, $f(S \cup T) \leqslant f(S) + f(T)$. Moreover, we assume that $f(\{i\}) = w_i$ and thus, the worst possible solution, that assigns each job to a distinct machine, costs $wor = \sum_{i \in V} w_i$.

In this paper, we are mainly interested in a maximization version of the problem, the MAXIMUM SAVING PARTITION PROBLEM, where the goal is to maximize the *saving* obtained by a solution relative to the worst case of performing each job individually on a separate machine. Formally, an instance of the MAXIMUM SAVING PARTITION PROBLEM is given by an independence system $(V, \mathscr{F})$ and a non-negative sub-additive function f from 2^V. An independence system is a pair $(V, \mathscr{F})$ where V is a ground set and $\mathscr{F}$ is a collection of subsets of V which are said to be independent, satisfying the condition i.e. if $S \in \mathscr{F}$ and $S' \subseteq S$, then $S' \in \mathscr{F}$. Trivially, any singleton of V is independent. The goal is to find a partition $(S_1, \ldots, S_k)$ of V with $S_i \in \mathscr{F}$ such that $\sum_{v \in V} w(\{v\}) - \sum_{i=1}^{k} w(S_i)$ is maximum. An interesting case of this problem, called COLOR SAVING PROBLEM, is when the solutions are node colorings. In particular when $f(S) = 1, \forall S \neq \emptyset$, an optimal cost is $|V| - \chi(G)$, i.e., the number of colors saved by an optimal coloring. For this special case of the COLOR SAVING PROBLEM with a single weight there are approximation algorithms that guarantee at least $\frac{1}{2}$ [7], $\frac{2}{3}$ [11], $\frac{3}{4}$ [9,14] and $\frac{289}{360}$ [8] of the maximum possible saving. It has been observed in [10] that the same bounds also apply to the more general problem of packingsets in an independent system. (Only [14] specifically uses the structure of the node coloring problem in a graph.) A notable example of such a problem is BIN PACKING, however, it is shown in [6], that this problem has an approximation scheme with respect to the saving criterion.

Some approximation results are given in [5] for the coloring version of the problem with $f(S) = \max\{w_i | i \in S\}$, and in particular a $\frac{1}{2}$-approximation in general graphs. We will generalize and strengthen this result by obtaining the same bound for general independence systems and a variety of optimization criteria, and by improving the bound when there are only two different weights in the input.

We now present some notation. The input to the MAXIMUM SAVING PARTITION PROBLEM consists of the family $\mathscr{S}$ of feasible subsets of $V = \{1, \ldots, n\}$, element weights w_i, $i \in V$, and the sub-additive set function f. The output is a partition $(S_1, \ldots, S_l)$ of V such that $S_i \in \mathscr{S}$, $i = 1, \ldots, l$, and $\sum_{j \in V} w_j - \sum_{i=1}^{l} f(S_i)$ is maximized. We note by *OPT* an optimal solution and by *opt* its cost. Similarly, we note by *APX* the approximate solution and by *apx* its cost. We use $opt(V')$ to note the maximum savings in the problem induced by the subsets $V' \subseteq V$, and thus $opt = opt(V)$. Similarly, we use $apx(V')$ and *apx* to denote the savings obtained by our algorithm. A w_i*-item* is an item of weight w_i, and a k-set is a set of size k. When an algorithm adds a subset S to the solution, this also means that the problem is reduced to the one induced by $V \backslash S$, i.e., the items of S are removed from V and subsets intersecting S are removed from $\mathscr{S}$.

We first describe a matching based algorithm and prove that it guarantees a $\frac{1}{2}$ approximation for several interesting function f. We then describe improved approximations. Finally, we prove some hardness results. An open question is how to use the improved results for COLOR SAVING with a single weight obtained by Halldórsson [9], Duh and Fürer [8], or Tzeng and King [14] to improve the performance of our algorithms.

2. $\frac{1}{2}$-approximation

Algorithm **optimal 2-packing**:

1. Consider the collection of feasible 2-sets $\mathscr{S}_2$. For each set $S = \{i, j\} \in \mathscr{S}_2$, define $d(S) = w_i + w_j - f(S)$.
2. Compute a maximum weight matching M in $\mathscr{S}_2$ using the weights d, and add these sets to *APX*.
3. For every element add to *APX* a singleton set.

Theorem 1.

$opt \leqslant 2apx$ for each of the following set functions:
Min: $f(S) = \min\{w_i : i \in S\}$.
Max: $f(S) = \max\{w_i : i \in S\}$.
Mean: $f(S) = \frac{1}{|S|} \sum_{i \in S} w_i$.
Suppose that $S = \{1, \ldots, l\}$, $w_1 \geqslant \cdots \geqslant w_l$, *and* $0 \leqslant \alpha < 1$ *is a given constant.*
Max-Convex: $f(S) = \alpha w_1 + (1 - \alpha) w_2$.
Ext-Convex: $f(S) = \alpha w_1 + (1 - \alpha) w_l$.

Proof. Consider a subset $S_0 = \{v_1, \dots, v_l\} \in OPT$. We will show that S_0 contains a matching M with at least half its saving, and then the claim will follow by summation over the sets in *OPT*. W.l.o.g., suppose that $S_0 = \{1, \dots, l\}$ and $w_1 \geqslant \cdots \geqslant w_l$. If l is even take M to be $\{1, 2\}, \{3, 4\}, \dots, \{l-1, l\}$. If l is odd take M to be $\{1, 2\}, \{3, 4\}, \dots, \{l-2, l-1\}, \{l\}$. In each case,

$$\sum_{S \in M} f(S) \leqslant \frac{1}{2}\left[\sum_{i \in V} w_i + f(S_0)\right],$$

and therefore

$$apx(S_0) = \sum_{i \in V} w_i - \sum_{S \in M} f(S) \geqslant \frac{1}{2}\left[\sum_{i \in V} w_i - f(S_0)\right]. \quad \square$$

Note that for each of the functions *f* in Theorem 1, if the maximum size of a subset in the input is at most 2, then the algorithm is optimal. On the other hand, even when $w_i = 1,\ \forall i$, there exist instances yielding the ratio $\frac{1}{2}$ for each of the functions *f*.

In some cases, we have another algorithm with better time complexity $O(n \log n)$.

Algorithm **First Fit 2-packing**.

1. Sort the items in decreasing order of weight.
2. Pack the first item with the first next item that accepts it, if such an item exists. Otherwise, form a singleton set with the first item.

Theorem 2. *Algorithm* **First Fit 2-packing** *returns a* $\frac{1}{2}$*-approximation for the functions* **Min**, **Max**, **Max-Convex**, *and* **Ext-Convex**.

Proof. For **Min**: We say that a w_i-item i is *saved* by a solution if it is packed into a set that contains a w_j-item, $j > i$. We observe that if i is saved by *OPT* and not by **First Fit 2-packing**, then the latter has saved another (unique) item j_i with greater weight.

For **Max**, we will prove the result by induction on $|V|$. Let S be the first set found by the algorithm. If $S = \{1\}$, then the result is clearly true. Assume that $S = \{1, i\}$. By construction of algorithm, *OPT* cannot pack an item j for $j < i$ with 1. Thus, $opt(V\backslash S) \geqslant opt - 2w_i$. Indeed, let $S^*_{j_1}$ and $S^*_{j_2}$ be the sets of an optimal solution $\mathscr{S}^*$ containing items 1 and i respectively (maybe $j_1 = j_2$). Now, consider the solution of $V\backslash S$ given by the restriction of $\mathscr{S}^*$ to $V\backslash S$. For the set $S^*_{j_p}\backslash S$ with $p = 1, 2$, we save an item r_p (maybe this item does not exist in $S^*_{j_p}$, and in the case, we assume that we have added a fictive item with weight 0) satisfying $w_{r_p} \leqslant w_i$.

Finally, we deduce

$$apx = w_i + apx(V\backslash S) \geqslant w_i + \tfrac{1}{2}(opt - 2w_i) \geqslant \tfrac{1}{2}\, opt.$$

For **Max-Convex** and for **Ext-Convex**, the previous property of *OPT* gives $opt(V\backslash S) \geqslant opt - 2(1-\alpha)w_1 - 2\alpha w_i$. Thus, using inductive hypothesis we deduce for the two functions

$$apx \geqslant (1-\alpha)w_1 + \alpha w_i + \tfrac{1}{2}\, opt(V\backslash S) \geqslant \tfrac{1}{2}\, opt. \quad \square$$

Remark 3. For the function **Mean**, Algorithm **First Fit 2-packing** does not guarantee the ratio $\frac{1}{2}$. For instance, consider COLOR SAVING with **Mean**, and G consists of the bipartite graph where the left set is $L = \{v_1, v_4, v_5\}$ and the right set is $R = \{v_2, v_3\}$, and only edges (v_1, v_3) is missing. The weights are given by $w_1 = w_2 = K$ and $w_3 = w_4 = w_5 = 1$. We have $apx = \frac{K+3}{2}$ given by $\{v_1, v_3\}, \{v_2\}$ and $\{v_4, v_5\}$ whereas $opt = \frac{7K+11}{6}$ given by the bipartition $L = \{v_1, v_4, v_5\}$ and $R = \{v_2, v_3\}$. Thus, we obtain that $\frac{apx}{opt}$ approaches $\frac{3}{7}$ as K goes to infinity.

Remark 4. If we modify Algorithm **First Fit 2-packing** to accept more than two elements in a set when possible, then we may not achieve the ratio $\frac{1}{2}$. For instance, if we consider COLOR SAVING with **Min** and G consists of 2 triangles $\{v_1, v_2, v_3\}, \{v_2, v_3, v_6\}$ and 2 edges (v_4, v_6) (v_5, v_6) with $w_1 = w_2 = w_3 = 3$ and $w_4 = w_5 = w_6 = 1$, then $apx = 4$ given by $\{v_1, v_4, v_5\}$, $\{v_2\}$, $\{v_3\}$ and $\{v_6\}$, whereas $opt = 9$ given by $\{v_1, v_6\}, \{v_2, v_4\}$ and $\{v_3, v_5\}$.

On the other hand, if we consider function **Max**, then the modified Algorithm **First Fit 2-packing** which accepts a maximal number of items for each subset is also a $\frac{1}{2}$-approximation, even in the unweighted version. This bound is attainable, as it can be seen from the following example. Consider COLOR SAVING and G consists of a chain of four nodes (v_1, v_4, v_3, v_2) and $w_1 = w_2 = w_3 = w_4 = 1$;

then $apx = 1$ given by $\{v_1, v_2\}$, $\{v_3\}$ and $\{v_4\}$, whereas $opt = 2$ given by $\{v_1, v_3\}$ and $\{v_2, v_4\}$.

3. Generic algorithm

This algorithm is a generalization of Hassin and Lahav's algorithm [11] and it will be used in Sections 4–6.

Algorithm 1.

1. While there exists a 3-set add it to the solution;
2. Apply Algorithm **optimal 2-packing**.

4. Min criterion: approximation results

We now describe an improved algorithm that guarantees a better than $\frac{1}{2}$ approximation factor for the **Min** criterion.

Suppose that there are r different values of weights and they are sorted in decreasing order $w_1 > \cdots > w_r$, and assume $w_{i+1} = \alpha_i w_i$ with $0 < \alpha_i < 1$.

Algorithm 2.

1. For $i = 1$ to r do
1.1. While there exists a feasible 3-set S, $|S| = 3$ and S contains at least two w_i-items, add it to the solution;
1.2. Consider the family $\mathscr{S}_2^i$ of feasible 2-sets containing at least one w_i-item. Solve a maximum matching M in $\mathscr{S}_2^i$. Add to the solution the sets resulting from the matching and add to the solution a singleton for every w_i-item.

Theorem 5. *Algorithm* 2 *is a β-approximation, where $\beta = \min\{\frac{2}{3}, \frac{1}{1+\alpha}\}$ and $\alpha = \max_i \alpha_i$.*

Proof. The proof is by induction on $|V|$. Consider $i \in \{1, \ldots, r\}$ and let S be a 3-set chosen in Step 1.1. Observe that by construction, there is no w_j-item with $j < i$. By the induction hypothesis,

$$apx \geqslant 2w_i + \beta opt(V \setminus S) \geqslant 2w_i + \beta(opt - 3w_i) \geqslant \beta opt.$$

Algorithm 2 obtains a maximum saving from w_i-items, but an optimal solution may do better with respect to the x-items with $x \in \{w_{i+1}, \ldots, w_r\}$. Let S be the item set of the matching M and the free w_i-items found in Step 1.2. Denote by l the number of sets in the matching. We have by induction:

$$apx = lw_i + apx(V \setminus S) \geqslant lw_i + \beta opt(V \setminus S).$$

On the other hand, we observe that

$$opt \leqslant l(w_i + w_{i+1}) + opt(V \setminus S).$$

We now explain this inequality. When adding items to a given set, each added item may add to *opt* at most the weight of one item (either itself—if its weight is not the minimum in its set, or the weight of another item that this item replaces as the minimum weight item in a set). In our case, the saving per added item can be w_j with $j > i$ or w_i. However, the maximum saving due to weights w_i is gained by our algorithm in Step 1.2, and it is exactly lw_i. Hence, no higher saving of w_i weights is possible, and the maximum saving for the optimum is due to weights w_{i+1} and therefore, $opt - opt(V \setminus S) \leqslant lw_i + lw_{i+1}$.

Combining the two inequalities with $\alpha_i \leqslant \alpha$, we obtain

$$\frac{apx}{opt} \geqslant \frac{lw_i + \beta opt(V \setminus S)}{l(w_i + w_{i+1}) + opt(V \setminus S)} \geqslant \min\left\{\frac{w_i}{w_i + w_{i+1}}, \beta\right\} \geqslant \min\left\{\frac{1}{1+\alpha}, \beta\right\}. \quad \square$$

If the weights are in $\{1, \ldots, B\}$ with $B \geqslant 2$ then $\alpha \leqslant \frac{B-1}{B}$ and Algorithm 2 returns at least a $\frac{B}{2B-1}$-approximation.

Now, we analyze Algorithm 1 from Section 3 for the **Min** criterion.

Theorem 6. *Algorithm* 1 *is a γ-approximation, where $\gamma = \frac{2}{3}\alpha_1\alpha_2 \cdots \alpha_r$.*

Proof. In its first step, Algorithm 1 inserts into the solution 3-sets. Let S be such a set, then $opt(V \setminus S) \geqslant opt - 3w_1$ and thus

$$\begin{aligned} apx &\geqslant 2w_r + apx(V \setminus S) \geqslant 2w_r + \gamma(opt - 3w_1) \\ &= (2\alpha_1\alpha_2 \cdots \alpha_r w_1 - 3\gamma w_1) + \gamma opt. \end{aligned}$$

In the second stage, since there are no 3-sets, Algorithm 1 applies **optimal 2-packing** and produces an optimal solution in the remaining instance. $\square$

Let δ be the value of α that solves $\frac{1}{1+\alpha} = \frac{2}{3}(\alpha_1\alpha_2 \cdots \alpha_r)$. If $\alpha = \max_i \alpha_i \leqslant \delta$ then we apply

Algorithm 2. Otherwise, we apply Algorithm 1. The resulting bound is $\frac{2}{3}$ for $\alpha \leqslant \frac{1}{2}$, $\frac{1}{1+\alpha}$ for $\frac{1}{2} \leqslant \alpha \leqslant \delta$, and $\frac{2}{3}(\alpha_1\alpha_2 \cdots \alpha_r)$ for $\alpha \geqslant \delta$. It obtains the lowest value when $\alpha = \delta$.

Consider now the bi-valued case; we have $\delta = \frac{\sqrt{7}-1}{2} \approx 0.823$, and when $\alpha = \delta$ the resulting bound is $\gamma = \frac{\sqrt{7}-1}{3} \approx 0.548$.

Corollary 7. *There is a* $\frac{\sqrt{7}-1}{3}$*-approximation for* **Min** *criterion when* $w_i \in \{s, b\}$.

5. Max criterion: approximation results

We now describe an improved algorithm that guarantees a better than $\frac{1}{2}$ approximation factor for the **Max** criterion.

As previously, suppose that there are r different values of weights $w_1 > \cdots > w_r$, and assume $w_{i+1} = \alpha_i w_i$ with $0 < \alpha_i < 1$.

Algorithm 3.

1. For $i = 1$ to r do
   1.1. While there exists a feasible 3-set S which only contains w_j-items with $j \leqslant i$, add it to the solution;
   1.2. Consider the family $\mathscr{S}_2^i$ of feasible 2-sets that contain only w_j-items with $j \leqslant i$. Solve a maximum matching M in $\mathscr{S}_2^i$ and add to the solution the sets resulting from the matching;
2. For every item, add to the solution a singleton set.

Theorem 8. *Algorithm* 3 *returns a* β*-approximation, where* $\beta = \min\{\frac{2}{3}, \frac{1}{1+\alpha}\}$ *with* $\alpha = \max_i \alpha_i$.

Proof. The proof is by induction on $|V|$. Consider $i \in \{1, \ldots, r\}$, and let S be a 3-set chosen in Step 1.1. Observe that S contains at most one w_j-item with $j < i$. Thus, $opt(V \backslash S) \geqslant opt - 3w_i$ and

$$apx \geqslant 2w_i + \beta(opt - 3w_i) \geqslant \beta opt.$$

Suppose that the matching M found in Step 1.2 has l sets. Since $\alpha \geqslant \alpha_i$, we have

$$\begin{aligned} apx &\geqslant l w_i + \beta(opt - l w_i - l w_{i+1}) \\ &= \beta opt + l w_i(1 - \beta - \alpha_i \beta) \geqslant \beta opt. \end{aligned}$$

The reason again is that the matching gives the maximum possible saving of w_i-items, so *opt* may only save more w_{i+1}-values.

For Step 2, the proof is trivial since only 1-sets remain. $\square$

If the weights are in $\{1, \ldots, B\}$ with $B \geqslant 2$ we deduce that Algorithm 3 returns at least a $\frac{B}{2B-1}$-approximation.

Theorem 9. *Algorithm* 1 *is a* γ*-approximation where* $\gamma = \frac{2\alpha_1\alpha_2\cdots\alpha_r}{1+2\alpha_1\alpha_2\cdots\alpha_r}$.

Proof. The proof is by induction on $|V|$. First, assume that $S = \{i, j, k\}$ with $w_i \leqslant w_j \leqslant w_k$. By the definition of **Max**, we have $opt(V \backslash S) \geqslant opt - w_i - w_j - w_1$ since $w_k \leqslant w_1$. Thus,

$$apx \geqslant w_i + w_j + \gamma(opt - w_i - w_j - w_1) \geqslant \gamma opt.$$

since $w_i \geqslant w_r, \ w_j \geqslant w_r$.

Now, if there are no 3-sets, then it is easy to see that Algorithm **optimal 2-packing** gives an optimal solution. $\square$

Let δ be the value of α that solves $\frac{1}{1+\alpha} = \frac{2\alpha_1\alpha_2\cdots\alpha_r}{1+2\alpha_1\alpha_2\cdots\alpha_r}$. If $\alpha \leqslant \delta$ then we apply Algorithm 3. Otherwise, we apply Algorithm 1. The resulting bound is $\frac{2}{3}$ for $\alpha \leqslant \frac{1}{2}$, $\frac{1}{1+\alpha}$ for $\frac{1}{2} \leqslant \alpha \leqslant \delta$, and $\frac{2\alpha_1\alpha_2\cdots\alpha_r}{1+2\alpha_1\alpha_2\cdots\alpha_r}$ for $\alpha \geqslant \delta$.

When $r = 2$, we have the bound $\frac{\sqrt{2}}{\sqrt{2}+1} \approx 0.585$.

Corollary 10. *There is a* $\frac{\sqrt{2}}{\sqrt{2}+1}$*-approximation for* **Max** *criterion when* $w_i \in \{s, b\}$.

6. Mean criterion: approximation results

Theorem 11. *Algorithm* 1 *is a* γ*-approximation where* $\gamma = \frac{4\alpha_1\alpha_2\cdots\alpha_r}{3(1+\alpha_1\alpha_2\cdots\alpha_r)}$.

Proof. The proof is by induction on $|V|$. First, assume that $S = \{i_1, i_2, i_3\}$. By the definition of **Mean**, we have

$$opt(V \backslash S) \geqslant opt - \frac{3}{2}w_1 - \frac{w_{i_1} + w_{i_2} + w_{i_3}}{2}.$$

Indeed, let S_j^* be the set of an optimal solution containing item i_j for $j=1,2,3$ in the present instance. We have

$$opt \leqslant opt(V\setminus S) + \sum_{j=1}^{3} \times \left(\frac{|S_j^*|-1}{|S_j^*|} w_{i_j} + \frac{w(S_j^*\setminus\{i_j\})}{|S_j^*|(|S_j^*|-1)} \right)$$

where for any set S, $w(S)=\sum_{l\in S} w_l$. Finally, since $w(S_j^*\setminus\{i_j\}) \leqslant (|S_j^*|-1))w_1$ the result is deduced

Thus, we obtain

$$apx \geqslant \frac{2}{3}(w_{i_1}+w_{i_2}+w_{i_3}) + \gamma\left(opt - \frac{3}{2}w_1 - \frac{w_{i_1}+w_{i_2}+w_{i_3}}{2}\right) \geqslant \gamma opt.$$

Now, if there is no 3-set, then it is easy to see that Algorithm **optimal 2-packing** gives an optimal solution. □

7. Min criterion: hardness results

Now, we will study the version where $\mathscr{S}$ is the set of independent sets in a graph and the criterion **Min**. We call this version the WEIGHTED NODE COLORING PROBLEM with **Min**-criterion. Here, we are interested in the standard version of coloring, not in color saving. So, when $w_i=1,\ \forall i=1,\dots,n$, we exactly obtain the same coloring problem. We show that even this restricted version is hard for approximation in bipartite graphs with weights 1 and 3. On the other hand, when $w_v\in\{1,2\}$, the WEIGHTED NODE COLORING PROBLEM with **Min**-criterion is polynomial in bipartite graphs and an optimum solution is just given by a 2-coloring. Remark that these results also hold for the COLOR SAVING PROBLEM with **Min**-criterion.

Theorem 12. *The* WEIGHTED NODE COLORING PROBLEM *with* **Min**-*criterion is Strongly* NP-*hard even in bipartite graphs and the function w only takes values* 1 *and* 3.

Proof. We apply a reduction from 1-PREXT IN BIPARTITE GRAPHS. This latter problem is defined by: given a bipartite graph $G=(V,E)$ where $V=L\cup R$ and $L=\{v_1,v_2,v_3\}$, there exists a 3-coloring (S_1,S_2,S_3) of G such that $v_i\in S_i$ for $i=1,2,3$. This problem was shown to be NP-complete in [2]. Consider an instance of 1-PREXT. We build an instance $I=(G',w)$ of the WEIGHTED NODE COLORING PROBLEM. We add two nodes v_1' and v_2' in R and link v_1' to v_2, v_3 and v_2' to v_1 and v_3. Note that G' is still bipartite. Finally, we set $w_{v_1}=w_{v_2}=w_{v_3}=1$ and $w_v=3$ for the other nodes of G'.

We prove that there exists an optimum weight coloring $\mathscr{C}$ of G' with $opt \leqslant 3$ if and only if there exists a 3-coloring (S_1,S_2,S_3) of G with $v_i\in S_i,\ i=1,2,3$.

If (S_1,S_2,S_3) with $v_i\in S_i,\ i=1,2,3$ is such a 3-coloring of G, then $S_1'=S_1\cup\{v_1'\}$, $S_2'=S_2\cup\{v_2'\}$ and S_3 is a coloring of G' with value 3.

Conversely, let $\mathscr{C}$ be a coloring of G' with a cost at most 3. It is easy to observe that this coloring contains at most three stable sets S_1,S_2,S_3. Assume that v_1 is in S_1; if $v_1'\notin S_1$, then the value of $\mathscr{C}$ is at least 4 since v_1' cannot be with v_2 and cannot be with v_3. Thus $\{v_1,v_1'\}\subseteq S_1$ and S_1 does not contain v_2 and v_3. We apply the same argument and deduce that $\{v_2,v_2'\}\subseteq S_2$ and thus $v_3\in S_3$, which conclude the proof. □

Corollary 13. *The* WEIGHTED NODE COLORING PROBLEM *with* **Min**-*criterion is not* $2^{p(n)}$-*approximable for any polynomial p unless* $P=NP$.

Proof. We apply the proof of Theorem 12 where we only change the value of function w by: $w_{v_1}=w_{v_2}=w_{v_3}=1$ and $w_v=3\cdot 2^{p(n)}$ for the other nodes of G'. Then, it is NP-complete to decide between $opt\leqslant 3$ and $opt\geqslant 3\cdot 2^{p(n)}$. □

References

[1] A. Bar-Noy, M. Bellare, M.M. Halldórsson, H. Shachnai, T. Tamir, On chromatic sums and distributed resource allocation, Inform. Comput. 140 (1998) 183–202.

[2] H.L. Bodlaender, K. Jansen, G.J. Woeginger, Scheduling with incompatible jobs, Discrete Appl. Math. 55 (1994) 219–232.

[3] D. de Werra, M. Demange, J. Monnot, V.Th. Paschos, A Hypocoloring model for batch scheduling, Discrete Appl. Math. (2004), to appear.

[4] D. de Werra, Y. Gay, Chromatic scheduling and frequency assignment, Discrete Appl. Math. 49 (1994) 165–174.

[5] M. Demange, D. De Werra, J. Monnot, V.Th. Paschos, Weighted node coloring: when stable sets are expensive (Extended abstract), Proceedings of the 28th International Workshop on Graph-Theoretic Concepts in Computer Science, Lecture Notes in Computer Science, vol. 2573, 2002 114–125.

[6] M. Demange, J. Monnot, V.Th. Paschos, Bridging gap between standard and differential polynomial approximation: the case of bin-packing, Appl. Math. Lett. 12 (1999) 127–133.
[7] M. Demange, P. Grisoni, V.Th. Paschos, Approximation results for the graph coloring problem, Inform. Process. Lett. 50 (1994) 19–23.
[8] R-c. Duh, M. Fürer, Approximation of *k*-set cover by semi-local optimization, in: Proceedings of the Twenty Ninth Annual ACM Symposium on Theory of Computing, 1996, pp. 256–264.
[9] M.M. Halldórsson, Approximating *k*-set cover and complementary graph coloring, Proceedings of the Fifth Conference on Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science, vol. 1084, 1996 118-131.
[10] R. Hassin, S. Khuller, *Z*-Approximations, J. Algorithms 41 (2001) 429–442.
[11] R. Hassin, S. Lahav, Maximizing the number of unused colors in the vertex coloring problem, Inform. Process. Lett. 52 (1994) 87–90.
[12] K. Jansen, Approximation results for the optimum cost chromatic partition problem, J. Algorithms 34 (2000) 54–89.
[13] Z. Tuza, Graph colorings with local constraints-a survey, Discussiones Math. Graph Theory 17 (1997) 161–228.
[14] W-G. Tzeng, G-H. King, Three-quarter approximation for the number of unused colors in graph coloring, Inform. Sci. 114 (1999) 105–126.