J Comb Optim (2008) 15: 342–356
DOI 10.1007/s10878-007-9086-0

Partitioning a weighted partial order

Linda S. Moonen · Frits C.R. Spieksma

Published online: 15 June 2007

Abstract The problem of partitioning a partially ordered set into a minimum number of chains is a well-known problem. In this paper we study a generalization of this problem, where we not only assume that the chains have bounded size, but also that a weight w_i is given for each element i in the partial order such that $w_i \leq w_j$ if $i \prec j$. The problem is then to partition the partial order into a minimum-weight set of chains of bounded size, where the weight of a chain equals the weight of the heaviest element in the chain. We prove that this problem is $\mathcal{APX}$-hard, and we propose and analyze lower bounds for this problem. Based on these lower bounds, we exhibit a 2-approximation algorithm, and show that it is tight. We report computational results for a number of real-world and randomly generated problem instances.

Keywords Partially ordered sets · Chain decomposition · Approximation algorithms

1 Introduction

Consider a partially ordered set $(X, \prec)$. We say that two elements $i, j \in X$ are *comparable* if either $i \prec j$ or $j \prec i$. A *chain* C is defined as a subset of X such that all elements $i, j \in C$ are pairwise comparable. An *antichain* A is a subset of X such that no two elements $i, j \in A$ are comparable. The *size* of a chain (or antichain) is equal to the number of elements contained in it (Trotter 1992).

Now, the problem of partitioning a partially ordered set $(X, \prec)$ into a minimal number of chains such that each element of X belongs to at least one chain, is a

L.S. Moonen (✉) · F.C.R. Spieksma
Research Center for Operations Research and Business Statistics (ORSTAT), Katholieke Universiteit Leuven, Naamsestraat 69, 3000 Leuven, Belgium
e-mail: linda.moonen@econ.kuleuven.be

F.C.R. Spieksma
e-mail: frits.spieksma@econ.kuleuven.be

well-known, fundamental problem in operations research. This problem is solvable in polynomial time, and the size of a maximum antichain is equal to the minimum number of chains needed to cover all elements of X (Dilworth 1950).

Shum and Trotter (1996) generalize this problem by assuming that an integer B is given that bounds the size of a chain. Thus, in this setting no more than B elements can be in a chain. They show that the corresponding decision problem is $\mathcal{NP}$-complete, even for a fixed $B = 3$.

In this work we further generalize this problem by assuming that a weight w_i for each $i \in X$ is given such that $w_i \leq w_j$ if $i \prec j$. Moreover, we define the weight of a chain C as $\max_{i \in C} w_i$, thus, the weight of a chain is equal to the weight of the heaviest element in the chain. This special weight structure is motivated by a practical application of the problem encountered at a manufacturing company in the Netherlands (see Sect. 1.2). Further, we denote a chain containing at most B elements as a B-chain. The problem is now to partition X into a minimum-weight set of B-chains. We refer to this problem as Minimum Weight Partition into B-chains, or MWPB. Observe that when $w_i = 1$ for all $i \in X$ the problem dealt with by Shum and Trotter (1996) arises.

1.1 Relation to graph-coloring problems

The problem of partitioning a partially ordered set into chains of bounded size is closely related to vertex coloring problems. In the standard vertex coloring problem we want to color the vertices of a graph such that for all edges, both endpoints have different colors. The objective is to minimize the number of colors needed (Golumbic 1980). Now, given a positive integer k, a *bounded vertex coloring* of a graph $G = (V, E)$ is then defined as a usual vertex coloring in which each color is used at most k times. The bounded chromatic number of a graph G ($\gamma_k(G)$) is the smallest number of colors such that G admits a bounded coloring with color classes of size at most k (Hansen et al. 1993).

The problem of partitioning a partial order into chains of bounded size can be formulated as a bounded vertex coloring problem. Indeed, we can represent the partial order as a graph, creating a vertex for each element in the partial order, and connecting two vertices if the corresponding elements are incomparable (the resulting graph is a cocomparability graph, see Golumbic 1980). The solution to the bounded vertex coloring problem on such problem instances can be transformed to solutions to the original problem of partitioning a partial order: vertices having the same color make up a chain. Since any color can be used at most a limited number of times, we know that the chains are bounded in size as well.

Hansen et al. (1993) show that the bounded vertex coloring problem for arbitrary graphs is $\mathcal{NP}$-complete for $k \geq 3$ using a reduction from Partition into Isomorphic Subgraphs. They also conjecture that, if k is part of the input, it is $\mathcal{NP}$-complete to find γ_k for bipartite graphs. This conjecture is later proved by Bodlaender and Jansen (1993). Jarvis and Zhou (2001) show that the bounded vertex coloring problem is solvable in polynomial time for trees.

1.2 Applications

A practical application of MWPB was encountered at Bruynzeel Storage Systems, a manufacturing company in the Netherlands (see also Moonen and Spieksma 2006).

This application is described as follows: given are a number of rectangular shaped boxes, each with a given length ℓ_i and width d_i. These boxes need to be loaded on top of each other onto pallets. Due to the height of the trucks that are used for transporting the pallets, each pallet can hold at most B boxes. Moreover, for any pair of consecutive boxes on a pallet it must hold that $\ell_i \le \ell_j$ and $d_i \le d_j$. This means that one can not put a larger box on top of a smaller one. This restriction ensures that the pallets are stable, and arrive in good shape at the customers. The goal is to load all boxes onto pallets so as to minimize total area, where the area of a pallet is determined by the area of its largest box. Observe that this corresponds to a special case of MWPB, where the special case arises since the partial order induced by the lengths and widths (i.e., $i \prec j$ if and only if $\ell_i \le \ell_j \wedge d_i \le d_j$) can be embedded in two dimensions; in other words, the dimension of the partial order equals 2 (see Ore 1962).

Other applications of MWPB can be found in the field of mutual exclusion scheduling (Baker and Coffman 1996; Jansen 2003), also known as batch scheduling with job compatibilities (Boudhar 2003; Finke et al. 2004). In such a problem jobs are given, each with a given processing time p_i, and for each pair of jobs it is known whether they can be processed on a same machine. A machine can process at most B jobs simultaneously, and the time a machine needs to process its jobs equals the maximum processing time of the jobs assigned to that machine. The problem is then to assign the jobs to the machines, respecting the compatibilities, while minimizing the total time needed by the machines to process all jobs. To represent the job compatibilities, often a graph is used; different types of graphs lead to different complexity results. In our setting, the graph corresponding to the job compatibilities is a comparability graph (see e.g. Golumbic 1980).

1.3 Our results

In this paper we show the following:

- Strengthening a result from Shum and Trotter (1996), we show that MWPB is $\mathcal{APX}$-hard, rendering the existence of a PTAS unlikely (we refer to Papadimitriou 1994 and Ausiello et al. 1999 for an overview of complexity and approximability issues).
- We propose two lower bounds, each of which can be arbitrarily bad when compared to the value of the optimum. The maximum of these lower bounds, however, is shown never to be less than half the optimum value.
- We describe a simple algorithm that yields a solution with a value guaranteed not to exceed twice the optimum value. The analysis is shown to be tight.
- We consider an extension of MWPB to a setting where rotation of elements is allowed.

The outline of the paper is as follows. Section 2 deals with the complexity of MWPB. In Sect. 3 we propose a number of lower bounds on the value of the optimum, and in Sect. 4 we present a 2-approximation algorithm for solving MWPB. In Sect. 5 we look at the integrality gap between the LP-relaxation corresponding to a straightforward set-partitioning model and the integer optimum. The approximation algorithm has been tested on a number of real-world problem instances, as well as on randomly generated instances, and the results from these experiments are described in Sect. 6. In Sect. 7 we discuss an interesting variant of MWPB, where the orientation of an element is taken into account. Finally, in Sect. 8, we conclude.

2 Complexity of MWPB

The decision problem corresponding to MWPB can be formulated as follows:

Given an integer B, a partial order $(X, \prec)$, weights w_i with $w_i \leq w_j$ if $i \prec j$ ($\forall i, j \in X$), and an integer K, does there exist a partition of X into B-chains such that the sum of the weights of the B-chains does not exceed K?

As stated in Sect. 1, Shum and Trotter (1996) prove that this decision problem is $\mathcal{NP}$-complete, even if $w_i = 1$, for all $i \in X$. We can strengthen their result:

Theorem 1 *MWPB is $\mathcal{APX}$-hard, even if*

- $B = 3$, *and*
- $w_i = 1$ $\forall i$, *and*
- *each element occurs in no more than* 3 *chains.*

Proof We use a reduction from the Maximum 3-bounded 3-dimensional matching problem (3DM-3), and follow the reduction from Shum and Trotter (1996). Furthermore, we apply arguments used in Chekuri and Khanna (2005).

Problem 3DM-3 is defined as follows: given are three sets U, Y, Z with $|U| = |Y| = |Z| = n$, and a set of triples $T \subseteq U \times Y \times Z$ with $|T| = m$. Each element occurs at most three times in a triple in T (therefore we can assume that $m = O(n)$). The goal is to find a matching of largest cardinality. Kann (1991) showed that this problem is $\mathcal{APX}$-hard, and Petrank (1994) showed that it is $\mathcal{NP}$-hard to decide whether there exists a matching of size n, or whether every matching has size at most $(1-\delta)n$ for some fixed $\delta > 0$.

Now, consider an instance $\mathcal{I}$ of problem 3DM-3. We build the corresponding instance $\mathcal{I}'$ of problem MWPB with $B = 3$ as follows. For each element $u_i \in U$, $y_j \in Y$, and $z_k \in Z$ we have an element in the partial order X. In addition, for each triple $\{u_i, y_j, z_k\} \in T$, we add nine additional elements to the partial order X as is shown in Fig. 1, where an arc from element p to element q implies that $p \prec q$ for each pair $p, q \in X$. We have now specified all elements in X, and all relations between the elements, thus we have created an instance of MWPB with $B = 3$. Shum and Trotter (1996) use the same gadget depicted in Fig. 1 for their reduction from Exact Cover by Three-Sets. Observe that

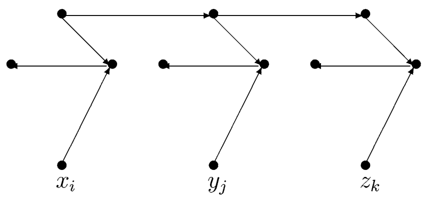


Fig. 1 Subgraph for triple $\{x_i, y_j, z_k\}$, see Shum and Trotter (1996)

(i) each chain in the instance $\mathcal{I}'$ can only consist of elements from the same configuration,
(ii) only if u_i and y_j and z_k are covered by other chains, it is possible to use 3 chains (recall that $B = 3$) for the elements in the configuration; otherwise at least 4 chains are needed.

If instance $\mathcal{I}$ has a matching of size n, then there exists a solution to instance $\mathcal{I}'$ with $4n + 3(m - n) = n + 3m$ chains: we need 4 chains to cover the elements in a configuration that corresponds to a chosen triple in the matching. For the elements in the remaining configurations, we need 3 chains to cover them, since all u_i, y_j, and z_k are covered by other chains.

Now let us consider the case that the maximum matching has size $(1 - \delta)n$ for some fixed $\delta > 0$. Consider first the elements contained in configurations corresponding to triples in the matching. We need $4(1 - \delta)n$ chains to cover these elements. Then we have covered $3(1 - \delta)n = 3n - 3\delta n$ u, y, z-elements. So there are $3\delta n$ u, y, z-elements remaining. Observe that there can be no configuration that contains more than 2 of these $3\delta n$ elements, since otherwise a better solution (i.e., a matching exceeding size $(1 - \delta)n$) exists. That means that the minimum number of configurations needed to cover these elements is $\frac{3\delta n}{2}$, and we need at least $4(\frac{3\delta n}{2})$ chains to cover all elements in these configurations. Finally, for the remaining elements we need at least 3 times the number of remaining configurations $= 3(m - (1 - \delta)n - \frac{3\delta n}{2})$ chains. So in total, we need at least $4(1 - \delta)n + 4(\frac{3\delta n}{2}) + 3(m - (1 - \delta)n - \frac{3\delta n}{2}) = n + 3m + \frac{1}{2}\delta n$ chains. Since $m = O(n)$, the $\mathcal{APX}$-hardness follows. □

3 Lower bounds for MWPB

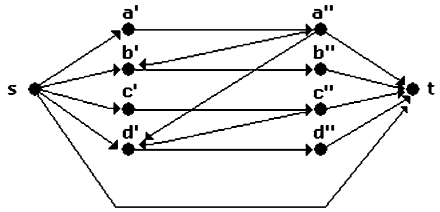
Consider an instance of MWPB, containing an integer B and n elements, each with a weight w_i, $1 \le i \le n$. We assume that the elements are ordered such that $w_1 \ge w_2 \ge \cdots \ge w_n$. Let OPT denote the value of an optimal solution to the instance. We define three lower bounds lb_i, $i = 1, 2, 3$, as follows:

1. $lb_1 = w_1 + w_{B+1} + \cdots + w_{(\lceil \frac{n}{B} \rceil - 1)B+1}$. Since the size of a chain cannot exceed B, lb_1 is obviously a lower bound for OPT.
2. When we omit the size constraint (i.e., if there is no restriction on the size of a chain), a relaxation of MWPB appears. Solving this relaxation gives a minimum-weight set of chains with value MWC. We set $lb_2 = MWC$.
3. $lb_3 = \max(lb_1, lb_2)$.

Theorem 2 *We can calculate the value of lb_2 by solving a min-cost flow problem.*

Proof In order to compute the value of lb_2, we create a directed graph $D = (V, A)$. V contains $2n + 2$ nodes: 2 nodes i' and i'' for every $i \in X$, a source s and a sink t. We draw an arc from s to each node i', with cost 0. Then we add an arc from each node i'' to t with cost w_i. Next, we add an arc from a node i' to its copy i'' with cost 0, and we add arcs from nodes i'' to j' if $i \prec j$, also with cost 0. Finally we add an arc from s to t with cost 0. All nodes have supply zero, except for s which has supply n, and t, which has supply $-n$ (a demand of n). All arc capacities are equal to 1, except for

Fig. 2 Min-cost flow network



the arc between s and t which has unbounded capacity. For the arcs from a node i' to its copy i'' we have a lower bound on the flow of 1. Now, consider a min-cost flow in D. Apart from the flow that goes directly via the arc from s to t, this flow can be seen as a set of paths such that each path carries one unit of flow. Notice that the set of paths is such that each node i'' is visited due to the lower bound on the arc between i' and i''. Moreover, the i'' nodes that are on a same path correspond to elements in X that are comparable. Since the cost of a path is determined by the last node i'' on the path, one observes that a min-cost flow in D corresponds to a solution to MWPB without the size constraint. Finally, one easily sees that a solution to MWPB without the size constraint corresponds to a min-cost flow in D. □

Notice that this algorithm solves a weighted generalization of the classical result of Dilworth (1950).

Example (Min-cost flow) Suppose we are given a partial order S containing four elements, $S = \{a, b, c, d\}$, and suppose $a \prec b$, $a \prec d$ and $c \prec d$. The min-cost flow network corresponding to this example is shown in Fig. 2. (The lower and upper bounds on the arcs, the arc costs and the demands are omitted from this figure.)

If we solve this min-cost flow problem, we find a solution containing a number of paths that have positive flow value. In the solution to our original problem, we put two elements in the same chain if, in the solution to the min-cost flow problem, these elements appear on the same path with positive flow value. This corresponds to a minimum-weight set of chains, keeping in mind that we have disregarded the size requirement on the chains.

We now show that lb_1 and lb_2 can be arbitrarily bad, even in the unweighted case. Consider lb_1, and suppose we are given a problem instance with $B = n$, such that no two elements are comparable, and suppose that each element has weight 1. One easily verifies that OPT equals n, while lb_1 equals 1.

Next, consider lb_2, and suppose we have a problem instance with $B = 1$, such that all elements are comparable, and that each element has weight 1. Again, OPT equals n, while lb_2 gives a value of 1. So, we cannot give a constant performance guarantee for either of these lower bounds. However, no instance exists where both lower bounds are arbitrarily bad. Indeed, let us now consider the maximum of these two lower bounds, lb_3.

Claim 1 For each instance of MWPB: $lb_3 \geq \frac{1}{2} OPT$. Moreover, this bound is tight.

We postpone the proof of this inequality to Theorem 3; we first give an instance for which this bound is tight.

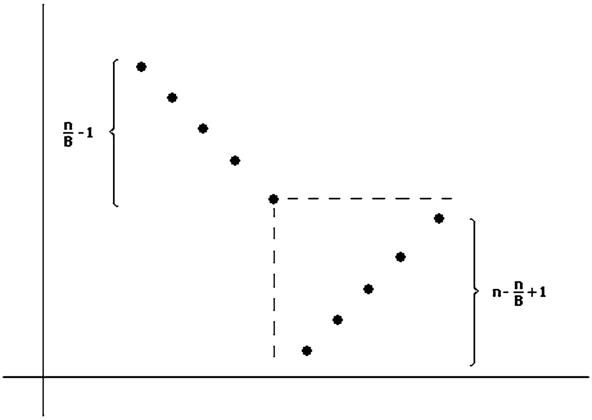


Fig. 3 A tight example

Consider the problem instance given in Fig. 3, with $n = B^2$. This problem instance represents a partial order of dimension 2, which is embedded in the plane. So, we have B^2 elements, all with weight 1, such that $n - \frac{n}{B} + 1$ of these elements are pairwise comparable, and the remaining $\frac{n}{B} - 1$ elements are pairwise incomparable. Observe that, for such instances, $lb_3 = \max(lb_1, lb_2) = B$, while $OPT = (\frac{n}{B} - 1) + \lceil(\frac{n-\frac{n}{B}+1}{B})\rceil = (B-1) + \lceil\frac{B^2-B+1}{B}\rceil = (2B-1)$.

4 A 2-approximation algorithm for MWPB

In this section we propose a 2-approximation algorithm for MWPB. It is based on first solving the relaxation of MWPB that omits the size constraint (see Theorem 2). Next, each chain that violates the size constraint (i.e. contains more than B elements) is partitioned into B-chains. We show that this method gives solutions with a value that cannot be worse than two times the value of an optimum solution, and that this bound is tight.

Consider the following heuristic H:

Step 1. Omit the size constraint, and find a minimum-weight set of chains as described in Theorem 2.

Step 2. For each chain consisting of say K elements $i_1, \dots, i_K$, with $i_1 \succ i_2 \succ \dots \succ i_K$, partition it into $\lceil\frac{K}{B}\rceil$ B-chains such that elements $i_{(j-1)B+1}$, $i_{(j-1)B+2}, \dots, i_{\min(jB,K)}$ form B-chain j, $j = 1, \dots, \lceil\frac{K}{B}\rceil$.

Theorem 3 *H is a 2-approximation algorithm for problem MWPB.*

Proof We assume that the elements are ordered such that $w_1 \geq w_2 \geq \dots \geq w_n$. Suppose we find a solution using heuristic H with value V_H, where in the first step we find a decomposition into p chains, $C_1, \dots, C_p$. In the second step we partition each of these p chains into a number of B-chains. We refer to the indices of the maximal elements of C_ℓ as u_ℓ, $1 \leq \ell \leq p$. All other indices of maximal elements of B-chains are referred to as v_ℓ, $1 \leq \ell \leq k$. Assume, without loss of generality, that

$v_1 \le v_2 \le \cdots \le v_k$. Notice that we can associate to each element with index v_ℓ a unique set of B elements that belong to the same chain found in Step 1 as this element, and are the smallest B elements that dominate it. Let us refer to this set of elements as $S(v_\ell)$, $1 \le \ell \le k$.

Claim 2 $v_\ell \ge \ell B$ ($\ell = 1, \ldots, k$).

Argument Consider the sets $S(v_\ell)$, $\ell = 1, \ldots, k$. Since these sets are pairwise disjoint, the number of elements that must precede the element with index v_ℓ, $1 \le \ell \le k$, equals at least ℓB.

The inequality from Claim 2 implies $\sum_{\ell=1}^{k} w_{v_\ell} \le \sum_{\ell=1}^{k} w_{\ell B} \le lb_1$. And, obviously, $\sum_{\ell=1}^{p} w_{u_\ell} = lb_2$. Thus, we have that $V_H = \sum_{\ell=1}^{p} w_{u_\ell} + \sum_{\ell=1}^{k} w_{v_\ell} \le lb_1 + lb_2 \le 2OPT$. Also, notice that $lb_3 + lb_3 \ge lb_1 + lb_2 \ge V_H \ge OPT$, implying Theorem 3. □

It is not clear yet whether the bound derived is tight. Indeed, the example for which lb_3 is shown to be worst possible is solved to optimality by H. We know that a solution found by heuristic H can be no worse than twice the optimal value. Can we find problem instances for which this gap is tight?

For the problem instances shown in Fig. 4 (again, we have a partial order of dimension 2, embedded in the plane), we have $n = B^2$ elements, all with weight 1, and there are B elements that are pairwise incomparable, and the remaining $B(B-1)$ elements are pairwise comparable. For such an instance, we have $OPT = lb_1 = lb_2 = lb_3 = B$, while the heuristic H gives a solution with value $(B-1) + \lceil \frac{B^2-B+1}{B} \rceil = (2B-1)$, so these are asymptotic worst-case instances with respect to H.

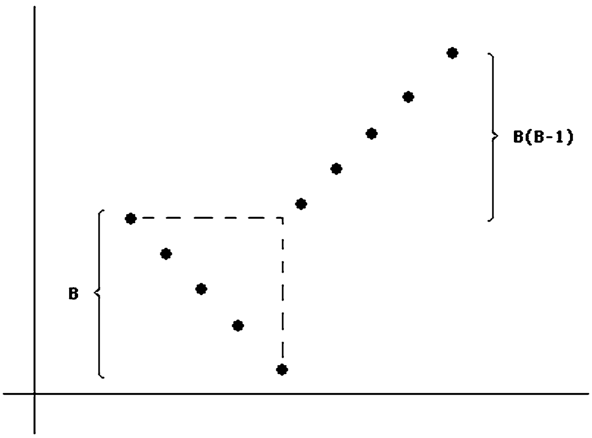


Fig. 4 Worst-case instance with respect to H

5 Integrality gap

In the last section we established an approximation algorithm with worst-case ratio equal to 2. An interesting question is whether we can do better than this. A natural approach is then to look at the integrality gap as measured by the ratio between the optimal integral solution and the solution to the LP-relaxation corresponding to some IP model. This gap serves as a lower bound for any approximation algorithm that is based on straightforward rounding of the solution of the linear program. In this section we analyze the gap between the LP-relaxation corresponding to a straightforward set-partitioning model and the integer optimum.

We define a decision variable x_k for every B-chain k:

$$x_k = \begin{cases} 1 & \text{if } B\text{-chain } k \text{ is in the solution,} \\ 0 & \text{otherwise.} \end{cases}$$

We define λ_k as the weight of a B-chain k. The set-partitioning model is then as follows:

$$\min \quad \sum_k \lambda_k x_k \tag{1}$$

$$\text{s.t.} \quad \sum_{k: i \in \mathcal{I}_k} x_k = 1 \quad \forall i; \tag{2}$$

$$x_k \in \{0, 1\} \quad \forall k. \tag{3}$$

Here, $\mathcal{I}_k$ is the set of all elements contained in B-chain k. The objective (1) is to minimize sum of the weights of the B-chains. Constraints (2) state that each element has to be in exactly one B-chain, and constraints (3) are the zero-one constraints on the x_k variables. For the LP-relaxation, we replace constraints (3) by constraints (4):

$$x_k \geq 0 \quad \forall k. \tag{4}$$

Let us start with a small example:

Example (Integrality gap) Suppose we are given a problem instance containing three elements a, b, and c, each with a weight equal to one, such that all three elements are pairwise comparable ($a \prec b \prec c$), and B is equal to 2. In the solution to the LP-relaxation we find three different B-chains: one B-chain containing elements a and b, a second B-chain containing elements a and c, and a third B-chain containing elements b and c. All three B-chains occur in the LP-solution with value $\frac{1}{2}$, so the value of the solution to the LP-relaxation V_{LP} is $\frac{3}{2}$. However, the optimal integral solution has value equal to 2 for this problem instance: an example of such a solution has a B-chain containing elements a and b and a second B-chain containing the single element c. This means that, for this simple problem instance, the gap between the integer optimum and the LP-solution is $\frac{4}{3}$.

This example shows that, even for a very small problem instance, the integrality gap is already equal to $\frac{4}{3}$. Now, we can expand the example as follows: assume we have a problem instance containing $B+1$ elements, each with a weight equal to one, such that all elements are pairwise comparable. In the solution to the LP-relaxation there will be a variable with positive value for each B-chain containing exactly B elements. There are exactly $B+1$ of these B-chains, and each of them occurs in the LP-solution with value $\frac{1}{B}$. This means that the solution to the LP-relaxation has value $\frac{B+1}{B}$, while the integer optimum has value equal to 2. For large B, this leads to an integrality gap of a factor 2, thereby showing that a straightforward rounding approach will not improve the heuristic H with respect to the performance guarantee.

6 Computational results

In order to assess the quality of our approximation algorithm on different types of instances we did some (limited) computational experiments. Our focus here is not to find high-quality solutions to MWPB, but is to see the computational behavior of the approximation algorithm. We implemented the 2-approximation algorithm in C++, using the CPLEX network solver to solve the min-cost flow problems, and we tested it on a number of real-world and randomly generated problem instances. We use 3 different data sets for the experiments. The first data set contains 50 real-world instances provided to us by Bruynzeel Storage Systems, the second data set contains 50 randomly generated instances, and the third data set contains 50 randomly generated instances that have small clique-width (see Moonen and Spieksma 2006). The problem instances for all three data sets contain between 20 and 200 elements. We solve each problem instance for 5 different values of B, so we have 250 experiments for each data set. Since the computation time for each instance is less than a second, we omit these from the tables with results.

In Table 1 we compare the lower bounds. As lb_3 is defined as the maximum of lb_1 and lb_2, we want to know how many times lb_3 equals lb_1, and how many times it equals lb_2. So in Table 1 we give, for each of the three data sets, the percentage of the number of times that lb_3 equals lb_1 and the number of times that lb_3 equals lb_2.

From these results we see that the performance of the lower bounds is very different for the different data sets. For the first data set, that contains the real-world problem instances, lb_1 is clearly better than lb_2: in 97.20% of all experiments, the value of lb_1 is larger than the value of lb_2. If we look at the second data set, we see that lb_1 still performs better compared to lb_2, but the percentage of experiments for which lb_1 is larger than lb_2 is only 57.60% for data set 2. However, for data set 3

Table 1 Results for lower bounds

B	*Data set 1*		*Data set 2*		*Data set 3*	
	$lb_3 = lb_1$	$lb_3 = lb_2$	$lb_3 = lb_1$	$lb_3 = lb_2$	$lb_3 = lb_1$	$lb_3 = lb_2$
3	100%	0%	96%	4%	96%	4%
6	100%	0%	74%	26%	74%	26%
9	98%	2%	50%	50%	28%	72%
12	94%	6%	42%	58%	14%	86%
15	94%	6%	26%	74%	8%	92%

Table 2 Results for data set 1: real-world instances

B	lb_1	lb_2	lb_3	V_H	Δ_3 (%)	
					avg_{Δ_3}	$\max_{\Delta_3}$
3	1085.06	96.02	1085.06	1096.12	1.45	5.49
6	556.12	96.02	556.12	574.74	4.18	17.39
9	379.96	96.02	380.22	396.66	6.18	27.55
12	293.76	96.02	294.82	316.08	8.12	32.05
15	240.40	96.02	242.18	259.64	8.89	25.16

Table 3 Results for data set 2: random instances

B	lb_1	lb_2	lb_3	V_H	Δ_3 (%)	
					avg_{Δ_3}	$\max_{\Delta_3}$
3	6518.64	1508.44	6589.06	6935.18	11.12	31.73
6	3317.18	1508.44	3338.78	3877.36	19.22	39.22
9	2259.08	1508.44	2314.28	2867.24	18.71	37.17
12	1726.68	1508.44	1818.90	2348.64	18.45	43.90
15	1411.90	1508.44	1585.92	2066.94	17.49	46.88

Table 4 Results for data set 3: instances with small clique-width

B	lb_1	lb_2	lb_3	V_H	Δ_3 (%)	
					avg_{Δ_3}	$\max_{\Delta_3}$
3	3215.70	1357.98	3219.66	3607.64	12.76	29.47
6	1686.28	1357.98	1765.54	2200.00	23.61	38.59
9	1177.96	1357.98	1451.42	1762.92	20.28	43.54
12	926.42	1357.98	1389.50	1574.78	13.72	37.79
15	775.38	1357.98	1364.50	1470.34	9.16	39.91

we see that lb_2 performs slightly better than lb_1: in 56.00% of all experiments the value of lb_2 is larger than the value of lb_1. The differences concerning the quality of the lower bounds with respect to the different data sets can be explained by the fact that the real-world problem instances from data set 1 have a special structure, that is not present in the problem instances of the other data sets. As a consequence of this special structure, we need very few chains to cover all elements in case B is large, which means that the quality of lb_2 is very poor for these instances.

Next we compare the values of lb_3 with the values of the approximation algorithm. Tables 2, 3, and 4 show a comparison between the values of the three lower bounds and the value of the 2-approximation algorithm. In the first column we give the value of B, and in the next four columns we show the average values of the three lower bounds (lb_1, lb_2, and lb_3) and the 2-approximation algorithm (V_H). Of course, as can be seen in the third column, the value of B does not influence lb_2. Finally, the column labeled Δ_3 shows the average (avg_{Δ_3}) and the maximum ($\max_{\Delta_3}$) difference between the values of V_H and lb_3. These differences are all given in percentages (i.e., $\frac{V_H - lb_3}{lb_3} \cdot 100\%$).

From these results we see that while the difference between the value of the approximation algorithm and the value of lb_3 could, in theory, get as large as 100%, the maximum difference among all experiments from the three data sets is equal to 32.05% for data set 1, 46.88% for data set 2, and 43.54% for data set 3. The average difference for the three data sets equal 5.76% for data set 1, 17.00% for data set 2, and 15.91% for data set 3. So again we see that the results for data set 1 are clearly better compared to the results of data sets 2 and 3, which can be explained by the special structure that is present in the problem instances of data set 1.

7 Rotation problem

In some applications, for example in the field of pallet loading problems as discussed in Moonen and Spieksma (2006), the elements are allowed to be *rotated* (i.e., the length and the width of a box are swapped) if that allows us to find a better solution. It is not difficult to exhibit instances where allowing rotation improves the solution value. So, when rotation is allowed, a relevant problem is how to choose a best orientation for each element that allows a best possible solution. Of course this question is motivated by a two-dimensional representation of the elements. However, each partial order can be embedded in d-dimensional space for some d, and, therefore the rotation variant of MWPB is interesting in higher dimensions as well. In this variant the objective is still to partition X into a minimum-weight set of B-chains. However, now we are allowed to choose an orientation for each element of X. We refer to this problem as MWPB-R.

Theorem 4 *There exists an optimal solution to MWPB-R such that, for all elements p that are contained in this optimal solution, it holds that*

$$x_p^1 \geq x_p^2 \geq \cdots \geq x_p^d \tag{5}$$

where d is the number of dimensions, and x_p^i is the i-th coordinate of element p.

Informally said, Theorem 4 states that there is no loss in rotating each element such that for each element the i-th largest coordinate becomes its size in the i-th dimension ($1 \leq i \leq d$). Before we give a proof of this theorem, we give an example for the 2-dimensional case.

Example (The 2-dimensional case) Given are n pairs of elements $\{(x_i^1, x_i^2), (x_i^2, x_i^1)\}$, $i = 1, \ldots, n$ in the 2-dimensional plane. Exactly one element from each of these n pairs must be present in a solution. Suppose we have an optimal solution that contains the chain $C = \{P, Q, R\}$ (see Fig. 5). Theorem 4 states that there exists an optimal solution such that for all the selected elements in this solution we have that $x_i^1 \geq x_i^2$. That means that we can exchange all the elements that lie above the line $x^1 = x^2$

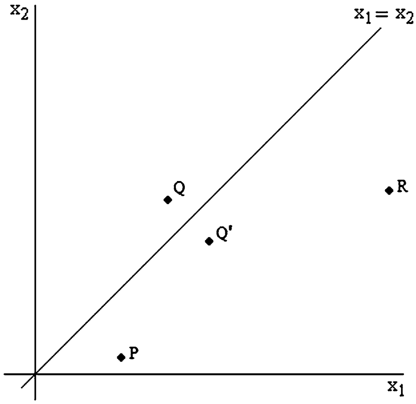


Fig. 5 Example for the 2-dimensional case

with their corresponding copies that lie below the line $x^1 = x^2$. This corresponds to the chain $C' = \{P, Q', R\}$. To see why this is true, consider the following.

Since the chain containing elements P, Q, and R is in the original solution, we know that $P \prec Q \prec R$, and so we have:

$$x_P^1 \le x_Q^1 \le x_R^1,$$
$$x_P^2 \le x_Q^2 \le x_R^2.$$

We also know that elements P and R lie below the line $x^1 = x^2$, and element Q lies above this line, so we also have:

$$x_P^1 \ge x_P^2,$$
$$x_Q^1 \le x_Q^2,$$
$$x_R^1 \ge x_R^2.$$

Now, in order for C' to be feasible, we must have that $P \prec Q' \prec R$, so we must show that

$$x_P^1 \le x_Q^2 \le x_R^1, \tag{6}$$
$$x_P^2 \le x_Q^1 \le x_R^2. \tag{7}$$

Equation (6) is true since $x_P^1 \le x_Q^1 \le x_Q^2 \le x_R^2 \le x_R^1$. Equation (7) is true since $x_P^2 \le x_P^1 \le x_Q^1 \le x_Q^2 \le x_R^2$.

Now we continue with the proof of Theorem 4.

Proof Given an element $p = (x_p^1, \dots, x_p^d) \in R^d$, we refer to the element that results when sorting the x-coordinates in non-decreasing order as the *rotated copy of* p. In order to prove Theorem 4, we must show that if we have a chain C containing elements that do not satisfy condition (5), then the rotated copies of these elements that do satisfy condition (5) form a chain. So, given two arbitrary elements U and

V such that $U \prec V$, we must show that, for their rotated copies satisfying condition (5), called U' and V', it holds that $U' \prec V'$. So we have to prove that if $U \prec V$, then $U' \prec V'$. That means that we have to show, for each $i = 1, \ldots, d$, that $x^i_{U'} \leq x^i_{V'}$.

We argue by contradiction. Take the smallest i for which this does not hold, so we have $x^i_{V'} < x^i_{U'}$. This means that the i^{th}-largest coordinate of element V' is smaller than the i^{th}-largest coordinate of element U', which contradicts $U \prec V$. □

Observe that in the argument above B plays no role. Hence, using Theorem 2, we can solve instances of MWPB-R with $B \geq n$ to optimality in polynomial time.

Notice that rotating the elements as described by (5) increases the number of pairs of elements that are comparable. It would be interesting to see how this would influence the performance of lb_2 and the heuristic H on the problem instances of Sect. 6.

8 Conclusions

In this paper we discuss the problem of partitioning a weighted partially ordered set into chains of bounded size. We proposed three lower bounds for this problem, and presented a 2-approximation algorithm for solving it. The approximation algorithm is tested on a number of real-world and randomly generated problem instances. From the results of the experiments we see that, although the value of heuristic H could be up to twice the value of lb_3, the largest difference between these values over all 750 experiments is 31.92%, and the average difference equals 10.50%. We conclude from these results that the approximation algorithm performs reasonably well.

References

Ausiello G, Crescenzi P, Gambosi G, Kann V, Marchetti-Spaccamela A, Protasi M (1999) Complexity and approximation: combinatorial optimization problems and their approximability properties. Springer, Berlin

Baker BS, Coffman EG Jr (1996) Mutual exclusion scheduling. Theor Comput Sci 162:225–245

Bodlaender HL, Jansen K (1993) On the complexity of scheduling incompatible jobs with unit-times. In: Borzyszkowski AM, Sokolowski S (eds) Mathematical foundations of computer science, 18th international symposium, Gdansk, Poland, 30 August–3 September 1993. Lecture notes in computer science, vol 711, pp 291–300

Boudhar M (2003) Scheduling a batch processing machine with bipartite compatibility graphs. Math Methods Oper Res 57:513–527

Chekuri C, Khanna S (2005) A PTAS for the multiple knapsack problem. SIAM J Comput 35(3):713–728

Dilworth RP (1950) A decomposition theorem for partially ordered sets. Ann Math 51:161–166

Finke G, Jost V, Queyranne M (2004) Batch processing with interval graph compatibilities between tasks. In: Proceedings of discrete optimization methods in production and logistics

Golumbic MC (1980) Algorithmic graph theory and perfect graphs. Academic Press, New York

Hansen P, Hertz A, Kuplinsky J (1993) Bounded vertex colorings of graphs. Discret Math 111:305–312

Jansen K (2003) The mutual exclusion scheduling problem for permutation and comparability graphs. Inf Comput 180:71–81

Jarvis M, Zhou B (2001) Bounded vertex coloring of trees. Discret Math 232:145–151

Kann V (1991) Maximum bounded 3-dimensional matching is MAX SNP-complete. Inf Process Lett 37(1):27–35

Moonen LS, Spieksma FCR (2006) Exact algorithms for a loading problem with bounded clique width. INFORMS J Comput 18:455–465

Ore O (1962) Theory of graphs. American mathematical society colloquium publications, vol 38. Providence

Papadimitriou CH (1994) Computational complexity. Addison–Wesley, Reading

Petrank E (1994) The hardness of approximation: gap location. Comput Complex 4:133–157

Shum H, Trotter LE Jr (1996) Cardinality-restricted chains and antichains in partially ordered sets. Discret Appl Math 65:421–439

Trotter WT (1992) Combinatorics and partially ordered sets: dimension theory. The John Hopkins University Press, Baltimore