Theoretical Computer Science 595 (2015) 120–129

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Improved approximation algorithms for a bilevel knapsack problem

Xian Qiu [a,*,1], Walter Kern [b]

[a] College of Computer Science, Zhejiang University, China
[b] Department of Applied Mathematics, University of Twente, Netherlands

ARTICLE INFO

Article history:
Received 20 September 2014
Received in revised form 13 May 2015
Accepted 13 June 2015
Available online 19 June 2015
Communicated by X. Deng

Keywords:
Bilevel
Knapsack
Approximation algorithm
Stackelberg

ABSTRACT

We study the Stackelberg/bilevel knapsack problem as proposed by Chen and Zhang [1]: Consider two agents, a leader and a follower. Each has his own knapsack. (Knapsack capacities are possibly different.) As usual, there is a set of items $i = 1, \ldots, n$ of given weights w_i and profits p_i. It is allowed to pack item i into both knapsacks, but in this case the corresponding profit for each player becomes $p_i + a_i$, where a_i is a given (positive or negative) number. The objective is to find a packing for the leader such that the total profit of the two knapsacks is maximized, assuming that the follower acts selfishly. We present tight approximation algorithms for all settings considered in [1].

1. Introduction

The standard knapsack problem is one of the most fundamental and well-studied problems in combinatorial optimization: There is a knapsack of prescribed capacity W and n items with given size w_i and profit p_i. The task is to select a set of items of total size at most W and maximum total profit. A first bilevel variant (in the form of a *Stackelberg game*) was introduced by Dempe and Richter [2]: There are two decision makers (players) – a *leader* and a *follower* – as well as a (universal) knapsack with flexible capacity and a set of items with given sizes as above, yet item profits may vary w.r.t. the leader and the follower, respectively. The leader first determines the capacity of the knapsack, and afterwards the follower, assumed to be selfish, packs items to the knapsack, maximizing his own profit. The (leader's bilevel) problem is to compute the knapsack capacity such that the leader's profit – defined by a linear function of the knapsack capacity plus his total profit of packed items – is maximized.

Several other bilevel variants of knapsack have been proposed as well. For example, Mansi et al. [12] study a setting in which both the leader and the follower pack items into a knapsack (of fixed capacity). DeNegre [17] investigates a bilevel version where both players own a private knapsack each and pack items from a common item set. Again, the leader acts first, selecting a set of items for his own knapsack, then the follower packs items from the remaining item set into his own knapsack, seeking to maximize his total profit. The objective of the (hostile) leader is to choose his set of items such that the follower's profit is minimized.

* Corresponding author.
E-mail addresses: xianqiu@zju.edu.cn (X. Qiu), w.kern@utwente.nl (W. Kern).

[1] Supported by the Natural Science Foundation of Zhejiang Province, No. LQ15A010001 and the Fundamental Research Funds for the Central Universities of China.

http://dx.doi.org/10.1016/j.tcs.2015.06.027
0304-3975/

Cases		Approx. ratios [1]	Lower bounds
$a_i \le 0$		$2+\epsilon$	1.5
$a_i \ge 0$	$W_1 > W_2$	$1+\sqrt{2}+\epsilon$	1.5
	$W_1 < W_2$	$2+\epsilon$	2

Fig. 1. Known lower bounds.

In this paper we consider yet another variant of the bilevel knapsack problem, due to Chen and Zhang [1]. In this setting, again, each player has his own knapsack of fixed capacities W_1 and W_2, respectively. Items $1,\dots,n$ have fixed weights w_i and profits p_i. The characteristic feature of the model in [1] is that items may be *double-packed*, *i.e.* packed by both players. In case item i is packed only by one player, it accounts for a profit of p_i, as usual, however, if i is packed by both players, its profit (for both players) is modified to $p_i + a_i$ for given *profit modifier* $a_i \in \mathbb{R}$. Again, the setting is that of a Stackelberg game, and the objective is to exhibit an optimal packing for the leader, *i.e.*, one that maximizes the total profit assuming that the second player (the follower) acts selfishly (disregarding the impact any double packing may have on the items packed by the leader). As a motivating example, Chen and Zhang mention the case of two investors, say, the government and a company with budgets W_1 and W_2, respectively. Items correspond to potential projects of cost w_i and reward p_i, resp. $p_i + a_i$ with $a_i > 0$ if both players invest in project i. Depending on the application, the numbers a_i may be positive or negative ("double booking"). In case all a_i are positive, Chen and Zhang [1] call it the *beneficial model* and if all a_i are negative, it is referred to as the *competitive model*.

Bilevel optimization is often computationally difficult and likely to extend beyond NP. In the last decades, bilevel and multilevel optimization have received much attention in the literature (*cf.* books by Migdalas, Pardalos and Värbrand [4] and Dempe [3], a survey by Colson et al. [5]). Dempe and Richter [2] introduced a mixed integer bilevel program for their problem variant and proposed an algorithm based on branch and bound. Afterwards, a dynamic programming algorithm for this problem was given by Brotcorne et al. [6]. Recently, Caprara et al. [7] proved that the first three problem variants mentioned above are Σ_2^P-hard (probably the fourth one is as well), *i.e.*, there is no way of formulating them as single-level integer programs of polynomial size unless *the polynomial hierarchy collapses* (*cf.* [7] for more details). In particular, they showed that the first two variants (*cf.* Dempe and Richter [2], Mansi et al. [12]) do not possess a polynomial approximation algorithm with finite worst case guarantee unless $P = NP$ and proposed a polynomial time approximation scheme for the third variant (*cf.* DeNegre [17]), which is known as the first approximation scheme for a Σ_2^P-hard problem. For other variants and related problems, *cf.* [8–12].

Regarding the problem to be considered in this paper, Chen and Zhang [1] proposed a $(2+\epsilon)$-approximation algorithm for the competitive model ($a_i \le 0$), and, for the beneficial model ($a_i \ge 0$), a $(1+\sqrt{2}+\epsilon)$-approximation for the case $W_1 > W_2$ and a $(2+\epsilon)$-approximation for the case $W_1 \le W_2$.

In this paper, we present better approximation algorithms for the beneficial model as well as the competitive model and show that the approximation ratios are tight in each case, *i.e.*, the approximation ratios can be made arbitrarily close to the known lower bounds (*cf.* Fig. 1). The main ingredients of our approach are: An ϵ-approximation of the maximum profit problem in case both players cooperate – which may be of independent interest, *cf.* (P3) in Section 2 – and a factor revealing LP for estimating the quality of our approximation algorithms (*cf.* Jain et al. [18]).

The rest of the paper is organized as follows: In the section below, we formally introduce the bilevel problem (*cf.* (P1) in Section 2) and its "cooperative" counterpart (*cf.* (P3)). In Section 3, we describe a polynomial time approximation scheme (PTAS) for the cooperative problem version (P3). In Section 4, we present new approximation algorithms and analyze their approximation ratios. Finally, in Section 5, we mention some open problems.

2. Bilevel knapsack with independent knapsacks

Let W_1, W_2 be capacities of the knapsacks owned by player 1 (leader) and player 2 (follower), respectively. Let $A = \{1,2,\dots,n\}$ be a set of items of weight w_i, profit p_i and "double packing modifier" a_i for all $i \in A$. Let $x_i, y_i \in \{0,1\}$ indicate whether item i is packed by player 1 and player 2, respectively.

Recall that the profit of item i is modified to $p_i + a_i$ if i is packed by both players. Thus the leader's problem can be formulated as a bilevel integer program as follows:

$$\max_{x} \sum_{i=1}^{n} p_i(x_i + y_i) + 2\sum_{i=1}^{n} a_i x_i y_i \tag{P1}$$

$$\text{s.t.} \sum_{i=1}^{n} w_i x_i \le W_1,$$

$$x_i \in \{0,1\}, \quad i = 1,2,\dots,n,$$

y is an optimal solution of (P2) *below*.

$$\begin{aligned} \max_{y} \ & \sum_{i=1}^{n} p_i y_i + \sum_{i=1}^{n} a_i x_i y_i \\ \text{s.t.} \ & \sum_{i=1}^{n} w_i y_i \le W_2, \\ & y_i \in \{0,1\}, \quad i = 1,2,\ldots,n. \end{aligned} \tag{P2}$$

Note that for fixed x there may exist multiple corresponding optimal solutions y for player 2. In our analysis, we always assume a worst case scenario, *i.e.* we focus on a "pessimistic" version of the above bilevel problem, where player 2 chooses an optimal solution y of (P2) minimizing the objective of (P1). We call the outcome of this pessimistic version the *competitive optimum* and – as in the paper by Chen and Zhang [1] – compare it to the so-called *cooperative* optimum, *i.e.*, the maximum total profit the two players could achieve. The latter can be expressed by a (single level) integer program

$$\begin{aligned} \max \ & \sum_{i=1}^{n} p_i (x_i + y_i) + 2 \sum_{i=1}^{n} a_i x_i y_i \\ \text{s.t.} \ & \sum_{i=1}^{n} w_i x_i \le W_1, \\ & \sum_{i=1}^{n} w_i y_i \le W_2, \\ & x_i, y_i \in \{0,1\}. \end{aligned} \tag{P3}$$

Example. Consider two knapsacks of capacity $W_1 = 1$ (for the leader) and capacity $W_2 = 2$ (for the follower). The item set contains two items of weights $w_1 = 1$, $w_2 = 2$, profits $p_1 = 2$, $p_2 = 1$ and modifiers $a_1 = -1$ (a_2 is arbitrary).

Observe that player 1 only has two options: Packing item 1 or packing nothing. If player 1 packs item 1, then player 2 gets a maximal profit 1 by either packing item 1 or item 2, resulting in a total profit 2 or 3 respectively. If player 1 packs nothing, then player 2 packs item 1, resulting in a total profit 2. Thus, the cooperative optimum may be as large as 1.5 times the competitive optimum. We aim at showing that this example is a worst case example in the sense that if all modifiers a_i are negative, then the ratio between the cooperative and competitive optimum is bounded by 1.5 and that solutions x for the leader's problem ensuring a ratio of $1.5 + \epsilon$ can be found in polynomial time. Similarly, we present tight bounds for the case of non-negative a_i and, eventually, the mixed case with both positive and negative double packing modifiers.

2.1. Dynamic programming for (P3)

We derive a straightforward pseudo-polynomial algorithm for (P3) based on dynamic programming. Let $f_i(a,b)$ be the maximum total profit with knapsack capacities a, b w.r.t. item set $\{1,\ldots,i\}$, for $a, b, i \in \mathbb{Z}^+$, $1 \le i \le n$, $a \le W_1$, $b \le W_2$. The recursive formula is defined by

$$\begin{aligned} f_{i+1}(a,b) = \max\{ & f_i(a,b), \\ & f_i(a - w_{i+1}, b) + p_{i+1}, \\ & f_i(a, b - w_{i+1}) + p_{i+1}, \\ & f_i(a - w_{i+1}, b - w_{i+1}) + 2p_{i+1} + 2a_{i+1}\}. \end{aligned}$$

The initial condition is

$$f_1(a,b) = \begin{cases} \max\{p_1, 2p_1 + 2a_1\}, & a = w_1, b = w_1, \\ p_1, & a = w_1, b < w_1 \text{ or } a < w_1, b = w_1, \\ 0, & a < w_1, b < w_1. \end{cases}$$

It is straightforward to check that $f_n(W_1, W_2)$ returns the maximum profit for (P3) and the algorithm has a running time $O(nW_1W_2)$.

3. Polynomial time approximation scheme for (P3)

As a first step, we compute approximately optimal cooperative solutions. It is well known that knapsack can be solved by a fully polynomial time approximation scheme (FPTAS) (*cf.* [13,14]). As (P3) with $a_i \to -\infty$ becomes a multiple knapsack problem with two knapsacks, we cannot expect an FPTAS for (P3) unless $P = NP$ (*cf.* [15]). In the following, we seek for a polynomial time approximation scheme (PTAS) for (P3).

We start with some notations. For any set S of items, let $w(S)$ and $p(S)$ denote the total weight and the total profit of items in S, respectively, *i.e.*, $w(S) = \sum_{i\in S} w_i$ and $p(S) = \sum_{i\in S} p_i$. Since the profits of items may be modified due to double-packing, $p(S)$ may also denote the modified total profit of items in S if no misunderstanding is possible.

As it turns out, *negative items*, *i.e.*, those with $a_i < 0$, and *non-negative items* (those with $a_i \geq 0$) can be treated independently. Therefore, we simplify matters by first assuming that all items are negative (non-negative items will be dealt afterwards). For technical reasons (to be explained in the proof) we slightly generalize our problem, assuming that certain items, say, items in the set $N_1 \subseteq N$, are not allowed to be double-packed. We let $n_1 = |N_1|$ denote the number of items that are prescribed to be single, and n_2 is the number of items in $N_2 = N\backslash N_1$ that may be double-packed. Thus $n_1 + n_2 = n$. Our proof for the approximation ratio will be by induction on $n_1 + 2n_2$.

We describe an $O(1 - 1/k)$-approximation algorithm – again denoted by ALG – proceeding in a similar way to that of Sahni [16]. One difference is that for a double-packable item i we distinguish between its *primary* copy i with an associated profit p_i and its *secondary* copy i with profit $p_i + 2a_i < p_i$. Let $S^* = S_1^* \cup S_2^* = \text{supp } x^* \cup \text{supp } y^*$ be an optimal solution of (P3). Observe that – as we distinguish primary and secondary copies – the set S^* may be understood as a *set* rather than a multiset. In phase I, ALG seeks to "guess" the k most profitable items from the optimum solution $S_1^* \cup S_2^*$ to include them in the initial packing. In case an item i is double-packed in S^*, and (the primary copy of) i belongs to the k most profitable items, we want ALG to include also the secondary copy into the initial packing. For this reason, we let ALG start from all initial packings with up to k primary items plus some of their secondary copies and let $S = S_1 \cup S_2$ be this set of items, with S_1, S_2 packed on knapsacks 1 and 2, respectively.

The second phase, again, considers the remaining items in order of non-increasing profit rates. As in the single knapsack case, it proceeds in a true online manner as explained below. In particular, whenever ALG checks an item i, it immediately decides upon packing or not packing i, but does not yet decide whether i should be double-packed. Note that, as we stick to the case of negative items here, we have $p_i + 2a_i < p_i$, so that primary items come with higher profit rates and are checked for inclusion before their corresponding secondary copies arrive.

Summarizing, in phase II, ALG considers the (copies of) items in $\{1, \ldots, n\}\setminus S$ in order of non-increasing profit rates

$$r_{i_1} \geq r_{i_2} \geq \ldots \geq r_{i_m} \quad (m = n_1 + 2n_2 - |S|)$$

defined as explained above.

Whenever ALG checks a primary item $i = i_t$, the item is packed wherever it fits. In case it does not fit anywhere, the item is skipped. Whenever ALG checks a secondary item $i = i_t$, it is perfectly clear on which knapsack i should be packed and ALG seeks to accommodate item i there, say, on knapsack 1, by "switching" single items from knapsack 1 to knapsack 2 if necessary. More precisely, ALG considers all single (primary) items from $\{1, \ldots, n\}\setminus S$ currently packed on knapsack 1 in some order and switches them onto knapsack 2 whenever they fit there, until either item i can eventually be accommodated on knapsack 1 or no further single item can be switched to knapsack 2 while item i still cannot be added to knapsack 1. In the latter case, item $i = i_t$ is skipped. The order in which items are considered for switching from knapsack 1 to knapsack 2 is not relevant, but it is convenient to use a "last in first out" order as switching rule.

Note that any packed item remains packed, only the assignment to a particular knapsack may be revised (possibly even several times), due to switching.

Lemma 1. *ALG as described above yields a* $(1 - 2/k)$*-approximation for (P3) (assuming that all items are negative).*

Proof. The proof is by induction on $n_1 + 2n_2$. The claim is obviously true for $n_1 + 2n_2 \leq k$. Indeed, whenever the optimal solution $S^* = S_1^* \cup S_2^*$ contains $2k$ or less items, then ALG will exhibit S^* in phase I.

Hence, let us assume that $n_1 + 2n_2 > k$ and follow the computation of ALG, assuming that in phase I, $S = S_1 \cup S_2$ containing the k largest profit items from S^* are packed in the same way and with the same multiplicities as in the optimal solution. Let $i = i_t$ denote the first item that is skipped by ALG. If $i \in N_1$, the proof is almost identical to the single knapsack case: As w_i exceeds the remaining capacity, say, c_1, c_2 respectively, for both knapsacks, the total remaining capacity $c_1 + c_2$ satisfies $c_1 + c_2 < 2w_i$. If we compare the current total profit P_{t-1} of ALG, just after item i_{t-1} has been placed, the total used capacity equals $W_1 + W_2 - (c_1 + c_2)$ and – since ALG packs in order of non-increasing profit rates – no other packing can achieve a higher profit on this part. Consequently, no other packing algorithm can achieve a total profit larger than or equal to $P_{t-1} + 2(c_1 + c_2)r_i$ (the amount that would be achieved by packing an item with profit rate equal to r_i, but smaller size (c_1, c_2 respectively), thereby exhausting exactly both knapsack capacities). Hence, the final total profit P achieved by ALG compares to the optimum profit P^* as follows:

$$P^* \leq P + (c_1 + c_2)r_i < P + 2w_i r_i = P + 2p_i.$$

Thus, if $i \in S_1^* \cup S_2^*$, then $p_i \leq P^*/k$ and we get $P \geq P^*(1 - 2/k)$ as required. Else, if $i \notin S_1^* \cup S_2^*$, then P and P^* remain unchanged if we remove item i, thereby decreasing $n_1 + 2n_2$ by one, so the claim follows by induction.

The same argument works without any change in case $i = i_t \in N_2$ is a primary item, except that by removing i, we decrease $n_1 + 2n_2$ by two. Thus assume now that $i = i_t \in N_2$ is already packed, say, on knapsack 1, but the secondary i does not fit on knapsack 2, even after switching a maximal set of items from knapsack 2 to knapsack 1. The situation can be described as follows: Besides the initial packing S_1, S_2, there is a certain set D of items that are double-packed and

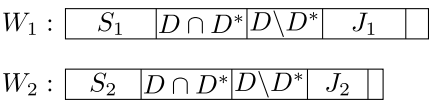

Fig. 2. Packing negative items.

sets J_1, J_2 of single/primary items on knapsack 1 and 2, respectively. By assumption, the remaining capacities c_1 and c_2 on knapsack 1 and 2 satisfy: $c_2 < w_i$ (item i does not fit) and $c_1 \le w_j$ for every $j \in J_2$ (no $j \in J_2$ can be switched to knapsack 1). Let D^* denote the set of items that are double-packed in the optimum solution. Fig. 2 below illustrates the current situation.

First note that, again, we may assume that $i \in J_1 \cap D^*$. Otherwise, if $i \in J_1 \backslash D^*$, prescribing i as single would reduce $n_1 + 2n_2$ but otherwise affect neither ALG nor the optimum solution, and the claim would follow by induction. Hence $i = i_t \in J_1 \cap D^*$ indeed. Hence $p_i \le \frac{1}{k} p(S_1 \cup S_2) \le \frac{1}{k} P^*$ follows.

A similar argument shows that we may assume $j \notin S_1^* \cup S_2^*$ for any $j \in J_2$: If $j \in S_1^* \cup S_2^*$, then $p_j \le \frac{1}{k} p(S_1^* \cup S_2^*) \le \frac{1}{k} P^*$. Thus if c_1, c_2 are the remaining capacities on knapsack 1 and 2 respectively, then (due to the fact that ALG proceeds according to non-increasing profit rates) we have $r_j \ge r_i$ and hence

$$\begin{aligned} P^* &\le P + (c_1 + c_2) r_i \le P + c_1 r_j + c_2 r_i \\ &< P + w_j r_j + w_i r_i = P + p_j + p_i \\ &\le P + \frac{2}{k} P^* \end{aligned}$$

and the claim would follow. Hence we may assume indeed that $J_2 \cap (S_1^* \cup S_2^*) = \emptyset$.

As we have seen above, we may assume that $p_i \le P^*/k$, so we can afford skipping the secondary item $i = i_t$. The problem is that i_{t+1}, i_{t+2} etc. might be similar secondary items that we have to skip. Thus how much skipping can we afford? The answer is given by the size of J_2 and $D \backslash D^*$: As the items in J_2 are not packed in the optimum solution and the items in $D \backslash D^*$ are at least not double-packed in the optimum solution. ALG spends a total of $w = w(J_2) + w(D \backslash D^*)$ packing items at rate $r \ge r_i$ which are not packed by an optimal algorithm, say, OPT. Instead, OPT packs the secondary items $i = i_t$ and subsequent secondary items at rate $r \le r_i$. Thus we can afford skipping secondary items of total size w without falling back behind OPT (the one that builds $S^* = S_1^* \cup S_2^*$). The good news is that, roughly, no more than this is about to come: If OPT builds the optimum solution S_1^*, S_2^*, then it packs S_2 and $D \cap D^*$ on knapsack 2, so the remaining space for secondary items $i \in S_1^* \cup S_2^*$ is bounded by

$$W_2 - w(S_2) - w(D \cap D^*) \le w(D \backslash D^*) + w(J_2) + w_i = w + w_i.$$

Thus the total size of secondary items that we skip can be compensated by the profit we gain while packing J_2 and $D \backslash D^*$, except for an amount of at most $w_i r_i = p_i$.

Thus at least after skipping that many secondary items something else must happen: Either a single item i that we skip or a secondary item $j \in J_2$ that we cannot accommodate on knapsack 1. But then, again, even without any effort of reassigning items, we conclude that $j \in S_1^* \cup S_2^*$ (otherwise prescribe j as single and apply induction), implying that $p_j \le P^*/k$ and, hence – disregarding a total of w for the skipped secondary items compensated by J_2 and $D \backslash D^*$ – all we loose is bounded by p_i (on knapsack 2) and p_j (on knapsack 1). Summarizing, we get $P \ge (1 - 2/k) P^*$ also in this case.

We like to stress that the above argument holds also when the secondary items that ALG skips do not come in a row. For example, it might well happen that ALG skips $i = i_t$, but i_{t+1} is a (primary or secondary) item that fits well on knapsack 2, thereby enlarging D or J_2. It may also happen that – possibly after reassigning some items in J_1 to knapsack 2 – an item $j = i_{t+1}$ can be accommodated on knapsack 1. In any case, whenever the next secondary item j to be put on knapsack 2 is skipped by ALG, then this is because ALG has tried in vain to accommodate j by switching single items from knapsack 2 to knapsack 1. According to the switching rule, ALG reassigns items in a "last in first out" order, ensuring that the set J_2 that was blocking the secondary item $i = i_t$ will stay packed on knapsack 2. (Thus, at a later stage, we might even have a larger set $J_2' \supset J_2$, $J_2' \cap (S_1^* \cup S_2^*) = \emptyset$, to compensate for skipping.) □

It is an easy exercise to bound the running time: There are $O(n^{k+1})$ sets S of size $|S| \le k$ to choose for the primary items. Given S, we are left to fix for each $i \in S$ whether to double-pack i or not, and, in the latter case, where to pack it (knapsack 1 or 2). Thus in total the number of "guesses" is bounded by $O(n^{k+1} 3^k)$.

Next let us turn to the beneficial model, which turns out to be easier: Assume that all items in N are non-negative, *i.e.*, $a_i \ge 0$. Our approximation algorithm – denoted by ALG – in this case, will again have a "guessing" phase I, followed by a phase II, where items are processed in order of non-increasing profit rates. Note that this time, however, double-packing yields higher profit rates than single packing, so ALG will first decide about double-packing item i (at profit rate $r_i = (p_i + a_1)/w_i$) and then – in case of non-acceptance consider single-packing i at a later stage. Correspondingly, in the beneficial model, it does not make sense to distinguish between a primary and secondary copy of item i (as both arrive

at the same time). Thus, in what follows we will interpret an optimal solution of (P3) as a multiset $S^* = S_1^* \cup S_2^*$, where $D^* = S_1^* \cap S_2^*$ is the set of double-packed items.

In phase I, ALG guesses a set $D \subseteq \{1,\ldots,n\}$ of size $|D| \le k$ (as a candidate for a set of most profitable double-packed items in an optimal solution) as well as a set $S \subseteq \{1,\ldots,n\}\backslash D$, $|S| \le k$ of most profitable single-packed items. In addition, it guesses a number $l \le n$ indicating the total number of items that should be double-packed by ALG (on top of the already chosen set D). In phase II, ALG processes the remaining items in order of non-increasing profit rates as usual. More precisely, let

$$r_{i_1} = \frac{p_{i_1} + a_{i_1}}{w_{i_1}} \ge \ldots \ge r_{i_l} = \frac{p_{i_l} + a_{i_l}}{w_{i_l}}$$

be the l highest double-packing profit rates in $N_2\backslash D$. Then ALG double-packs as much as possible of $i_1,\ldots,i_l$ (on top of D) and then continues (after packing S_1 and S_2) with single packing of the remaining items (*i.e.*, items in $(N_1 \cup N_2)\backslash(D \cup S \cup \{i_1,\ldots,i_l\})$) in order of non-increasing profit rates.

There are $O(n^{2(k+1)})$ possible choices for D and S, and $O(2^k)$ bipartitions of S. Together with the different choices for l, there are $O(n^{2k+3}2^k)$ different choices of parameters.

Lemma 2. *For suitable choice of parameters D, S_1, S_2 and l, ALG yields an approximation ratio $(1-4/k)$.*

Proof. The proof is by induction on $n_1 + 2n_2$. Let S_1^*, S_2^* be an optimal solution and let $W^* := w(S_1^* \cap S_2^*)$ be the total size of double-packed items. If $\left|S_1^* \cap S_2^*\right| \le k$, then ALG will guess $D = S_1^* \cap S_2^*$ exactly, otherwise let D denote the set of k highest profit $(= p_i + a_i)$ items in $S_1^* \cap S_2^*$ and assume l is chosen such that $w(D) + w_{i_1} + \ldots + w_{i_l} \le W^*$ but $w(D) + w_{i_1} + \ldots + w_{i_{l+1}} > W^*$.

Thus ALG, with this choice of parameters D and l will double-pack $i_1,\ldots,i_l$ (on top of D), but all further items will get at most single-packed. We claim that the total double-packing profit is at least $(p+a)(S_1^* \cap S_2^*)(1-\frac{2}{k})$, where $(p+a)(S_1^* \cap S_2^*) := \sum_{i \in S_1^* \cap S_2^*}(p_i + a_i)$. Indeed, if $i = i_{l+1} \in S_1^* \cap S_2^*$, this follows in the – by now – usual manner from $p_i + a_i \le \frac{1}{k}(p+a)(D)$. Else, if $i \notin S_1^* \cap S_2^*$, we might prescribe i as single, thus decreasing $n_1 + 2n_2$ and the claim follows by induction.

Thus ALG performs very well w.r.t. double packing, achieving almost optimal profit, by using less space in general. So we are left to analyze the single packing part. If at most k items are single-packed in the optimum solution $S_1^* \cup S_2^*$, ALG will exhibit these (along with the correct assignment to knapsack 1 and knapsack 2). Else, let $S = S_1 \cup S_2$ denote the k highest profit single-packed items in the optimum solution (with S_i packed on knapsack i). As ALG, when it starts single-packing items, has at least as much capacity left as OPT on each knapsack, S_1 and S_2 can be single-packed without any problem. The remaining items are then processed in order of non-increasing profit rates and the remaining part of the proof is by now routine: Let $i = i_t$ be the first item that cannot be accommodated, neither on knapsack 1 nor on knapsack 2. (Here we do not need to try to reassign any items.) Then the total profit, say P, including single-packed items obtained by ALG at this time, is at least a $(1-4/k)$-fraction of P^* the total profit in the optimum solution:

In case $i \notin S_1^* \cap S_2^*$, the claim follows by induction on $n_1 + 2n_2$ if we prescribe i as single item. Else, if $i \in S_1^* \cap S_2^*$, then $p_i \le p_i + a_i \le \frac{1}{k}(p+a)(S_1^* \cap S_2^*) \le \frac{1}{k}P^*$. Hence, adding a copy of item i to both knapsacks (thereby violating their capacity constraints) would yield a profit larger than $P^*(1-\frac{2}{k})$. (The factor $1-\frac{2}{k}$ takes care of the possible loss in the double-packing case.) Thus, in total, when skipping item i, we can ensure a total profit of at least $P^*(1-\frac{2}{k}) - 2p_i \ge P^*(1-\frac{4}{k})$. □

Remark 1. We like to point out that negative items cannot be treated in such an easy way: Consider, for example a list of $2n$ items with $p_i = 10$, $a_i = -5$ plus $2n$ items with $p_i = 5, a_i = -\epsilon$ and $w_i = 1$ for all $4n$ items. An optimum packing for two knapsacks of capacity $3n$ each would single-pack all items with $p_i = 10$ and double-pack all remaining items. Yet, a simple greedy heuristic like the "positive item variant" of ALG would always start double-packing the "high profit" items – at least we cannot prevent it from doing so by prescribing the number l of items to be double-packed.

We are left to combine the two algorithms for negative and non-negative items in order to deal with "mixed" sets of items. The only way we found works in a sense by "brute force": We guess the amount W_1^+ and W_2^+ of capacity an optimum solution uses on knapsack 1 and 2 for non-negative items respectively and then split the problem accordingly into one with negative and one with non-negative items. In our case, it is sufficient to approximate W_1^+ and W_2^+ up to a factor $1/k$, so we actually guess $m_i := \lceil \log_{1+1/k} W_i \rceil$ and run the algorithm with $\tilde{W}_i^+ = (1+1/k)^{m_i}$ instead of W_i^+. The total number of guesses we need is $O(k \log W_i)$ for $i = 1, 2$, thus $O(k^2(\log W_1)(\log W_2))$ in total.

This introduces another possible loss of order $1/k$ on the total profit gained with positive items, so that after all, the combined algorithm ALG will yield a total profit P with $P \ge (1-1/k)(1-4/k)P^*$, where P^* is the optimum profit. Thus we have shown

Theorem 1. *For fixed k, there exists a polynomial time algorithm ALG that approximates (P3) up to a factor $(1-5/k)$ in time $O(n^{2k+5}2^k \log W_1 \log W_2)$.*

4. Approximation algorithms for the leader

Let S_1 and S_2 be optimal solutions for the standard single knapsack instances with knapsack capacities W_1 and W_2, resp., and profits p_i for all items. In other words, S_1 and S_2 denote the support of optimal solutions of the following problem with $W = W_1$ and $W = W_2$, resp.:

$$\begin{aligned} \max\ & \sum_{i=1}^{n} p_i x_i \\ \text{s.t.}\ & \sum_{i=1}^{n} w_i x_i \le W, \\ & x_i \in \{0,1\}, \quad i = 1,2,\ldots,n. \end{aligned} \tag{P4}$$

Let (S_1^*, S_2^*) be an optimal solution of the cooperative relaxation (P3). We first note that the value OPT of (P3) satisfies

$$OPT = p(S_1^*) + p(S_2^*) + 2h^*, \text{ where } h^* := \sum_{i \in S_1^* \cap S_2^*} a_i. \tag{1}$$

Furthermore, we trivially have

$$p(S_1^*) \le p(S_1) \quad \text{and} \quad p(S_2^*) \le p(S_2). \tag{2}$$

As (S_1^*, S_2^*) can be found by dynamic programming (*cf.* Section 2.1) and can be approximated arbitrarily closely by a PTAS (*cf.* Section 3), we first present an algorithm by assuming that S_1^*, S_2^*, S_1 and S_2 can be found exactly. Afterwards, we show that we lose a factor of ϵ if these solutions are approximated correspondingly (*cf.* Section 4.3). We (first) treat the beneficial and the competitive model separately.

4.1. The beneficial model: $a_i \ge 0$

We present a very simple algorithm and show that the approximation ratios are tight in both cases ($W_1 \ge W_2$ and $W_1 < W_2$).

Algorithm 1. Pack one of S_1^* and S_1 (whichever results in a maximum total profit[1]).

Assume first that player 1 packs S_1^* and player 2 packs some set $\hat{S}$. Let $\hat{h} = \sum_{i \in S_1^* \cap \hat{S}_2} a_i$. Then $p(\hat{S}_2) + \hat{h} \ge p(S_2^*) + h^*$. Denote by ALG the value obtained by the algorithm. Thus

$$\begin{aligned} ALG &= p(S_1^*) + p(\hat{S}_2) + 2\hat{h} \\ &\ge p(S_1^*) + p(S_2^*) + h^* + \hat{h}. \end{aligned}$$

Since $p(\hat{S}_2) \le p(S_2)$, we have

$$\hat{h} \ge p(S_2^*) - p(\hat{S}_2) + h^* \ge p(S_2^*) - p(S_2) + h^*,$$

implying

$$ALG \ge p(S_1^*) + 2p(S_2^*) - p(S_2) + 2h^*. \tag{3}$$

Now assume that player 1 packs S_1. Then player 2 will get at least $p(S_2)$ by packing S_2, implying

$$ALG \ge p(S_1) + p(S_2).$$

Besides, we observe the following "knapsack constraints":

$$p(S_1) \ge p(S_2) \text{ if } W_1 \ge W_2, \tag{4}$$

$$p(S_1) \le p(S_2) \text{ if } W_1 \le W_2. \tag{5}$$

Let $\alpha = ALG/OPT$. We derive the following linear program for estimating the approximation ratio:

[1] *Cf.* Remark 2 at the end of Section 4.2 explaining how this decision can be taken.

$$\begin{aligned}
&\text{minimize } \alpha \qquad (6)\\
&\text{subject to } p(S_1^*) + p(S_2^*) + 2h^* = 1,\\
&\qquad \alpha \geq p(S_1^*) + 2p(S_2^*) - p(S_2) + 2h^*,\\
&\qquad \alpha \geq p(S_1) + p(S_2),\\
&\qquad p(S_1) \geq p(S_1^*),\\
&\qquad p(S_2) \geq p(S_2^*),\\
&\qquad p(S_1), p(S_2), p(S_1^*), p(S_2^*), h^* \geq 0\\
&\qquad \text{and the knapsack constraints hold.}
\end{aligned}$$

The minimum value equals 2/3 if $W_1 \geq W_2$ and 1/2 if $W_1 < W_2$, proving that OPT/ALG is at most 3/2 resp. 2, matching the lower bounds (*cf.* [1]).

4.2. The competitive model: $a_i < 0$

In principle, we could also apply Algorithm 1 to this case: If player 1 packs S_1^*, similar to the above argument, we have

$$ALG \geq p(S_1^*) + 2p(S_2^*) - p(S_2) + 2h^*.$$

In case of packing S_1, we want to show that $ALG \geq p(S_1) + p(S_2) + 2h$, where $h = \sum_{i \in S_1 \cap S_2} a_i$. This is clearly true if player 2 packs S_2. Otherwise, player 2 packs a set, say, $\hat{S}_2$, with $\hat{h} = \sum_{i \in S_1 \cap \hat{S}_2} a_i$, such that $p(\hat{S}_2) + \hat{h} \geq p(S_2) + h$. As $p(\hat{S}_2) \leq p(S_2)$, this implies $\hat{h} \geq h$. Thus,

$$\begin{aligned}
ALG &\geq p(S_1) + p(\hat{S}_2) + 2\hat{h}\\
&\geq p(S_1) + p(S_2) + h + \hat{h}\\
&\geq p(S_1) + p(S_2) + 2h,
\end{aligned}$$

as claimed.

Now we obtain the following linear program bounding the approximation ratio of Algorithm 1.

$$\begin{aligned}
&\min \alpha \qquad (7)\\
&\text{s.t. } p(S_1^*) + p(S_2^*) + 2h^* = 1,\\
&\qquad \alpha \geq p(S_1^*) + 2p(S_2^*) - p(S_2) + 2h^*,\\
&\qquad \alpha \geq p(S_1) + p(S_2) + 2h,\\
&\qquad p(S_1) \geq p(S_1^*),\\
&\qquad p(S_2) \geq p(S_2^*),\\
&\qquad p(S_1), p(S_2), p(S_1^*), p(S_2^*) \geq 0,\ h, h^* \leq 0\\
&\qquad \text{and the knapsack constraints hold.}
\end{aligned}$$

Letting $p(S_1^*) = p(S_1) = p(S_2) = 1$, $h = -1$ and $h^* = p(S_2^*) = 0$, we can easily see that the optimal objective value is 0. We observe that in this worst-case instance the penalty h is too large: A better choice for player 1 is to pack nothing. Then player 2 must pack S_2, implying $ALG \geq p(S_2)$. Adding this simple constraint to the above program yields an optimal objective value 0.5.

Thus, excluding items with large penalty from S_1 can improve the performance of ALG, however, it does not yield a tight approximation yet. The idea for further improvement is not only to avoid packing items with large penalties but also to pack items with *small* penalties. We define the following sets:

$$S_2^+ = \left\{ i \in S_2 \mid |a_i| > \frac{p_i}{2} \right\}, \quad S_2^- = S_2 \backslash S_2^+, \qquad (8)$$

where S_2^+, S_2^- are sets of items having large penalties and small penalties respectively. Our algorithm can now be described as follows.

Algorithm 2. Pack one of $S_1^*, S_1 \backslash S_2^+$ and $\emptyset$ (whichever results in a maximum total profit).

To analyze its performance, we distinguish three cases:

Case 1. Player 1 packs S_1^*. This is already considered in the above analysis and yields $ALG \geq p(S_1^*) + 2p(S_2^*) - p(S_2) + 2h^*$.

Case 2. Player 1 packs $S_1 \setminus S_2^+$. We prove that $ALG \geq p(S_1) + p(S_2^-) + 2h^-$, where $h^- = \sum_{i \in S_2^-} a_i$. Clearly, this is true when player 2 packs S_2. If player 2 packs some other set, say $\hat{S}_2 \neq S_2$, with $\hat{h} = \sum_{i \in (S_1 \setminus S_2^+) \cap \hat{S}_2} a_i$, then $p(\hat{S}_2) + \hat{h} \geq p(S_2) + h^-$, implying $\hat{h} \geq h^-$ (recall that $p(\hat{S}_2) \leq p(S_2)$). Hence,

$$\begin{aligned} ALG &\geq p(S_1) - p(S_2^+) + p(\hat{S}_2) + 2\hat{h} \\ &\geq p(S_1) - p(S_2^+) + p(S_2) + h^- + \hat{h} \\ &\geq p(S_1) - p(S_2^+) + p(S_2) + 2h^- \\ &= p(S_1) + p(S_2^-) + 2h^-. \end{aligned}$$

Case 3. Player 1 packs nothing. Then player 2 can guarantee a total profit $p(S_2)$ by packing S_2. Thus, $ALG \geq p(S_2)$. Furthermore, in addition to the knapsack constraints (4), (5) we have the following constraints:

$$p(S_2) = p(S_2^+) + p(S_2^-) \text{ and } h^- \geq -\frac{p(S_2^-)}{2}.$$

This finally yields the following linear program with an optimal value of 2/3 (for both $W_1 \geq W_2$ and $W_1 < W_2$).

$$\begin{aligned} &\text{minimize } \alpha \\ &\text{subject to } p(S_1^*) + p(S_2^*) + 2h^* = 1, \\ &\qquad \alpha \geq p(S_1^*) + 2p(S_2^*) - p(S_2) + 2h^*, \\ &\qquad \alpha \geq p(S_1) + p(S_2^-) + 2h^-, \\ &\qquad \alpha \geq p(S_2), \\ &\qquad p(S_2) = p(S_2^+) + p(S_2^-), \\ &\qquad h^- \geq -\frac{p(S_2^-)}{2}, \\ &\qquad p(S_1) \geq p(S_1^*), \\ &\qquad p(S_2) \geq p(S_2^*), \\ &\qquad p(S_1), p(S_2), p(S_2^+), p(S_2^-), p(S_1^*), p(S_2^*) \geq 0,\ h^*, h^- \leq 0 \\ &\qquad \text{and the knapsack constraints hold.} \end{aligned} \tag{9}$$

Hence $OPT/ALG \leq 1.5$, which is tight (*cf.* the example in Section 2).

Remark 2. Since player 2 acts after player 1, the reader may wonder how to find the maximal total profit over all cases (in Algorithms 1 and 2). This can be done by checking the inequalities "$\alpha \geq \ldots$" in (6) and (9) respectively: If any of these inequalities gets tight, the algorithm may pick the associated set (for player 1).

4.3. Approximating S_1^* and S_2^*

Algorithms 1 and 2 may equally well be applied, if we replace S_1, S_2, S_1^* and S_2^* by corresponding ϵ-approximations (to be computed as explained in Section 2.1, say, $\tilde{S}_1, \tilde{S}_2, \tilde{S}_1^*$ and $\tilde{S}_2^*$ with corresponding $\tilde{S}_2^+ = \{i \in \tilde{S}_2 \mid |a_i| > p_i/2\}$ and $\tilde{S}_2^- = \tilde{S}_2 \setminus \tilde{S}_2^+$). A close look at the analysis of Algorithms 1 and 2 reveals that optimality of (S_1^*, S_2^*) is only used in (1) and (3). It is easy to see that (1) becomes

$$(1-\epsilon)OPT \leq p(\tilde{S}_1^*) + p(\tilde{S}_2^*) + 2\tilde{h}^*, \text{ where } \tilde{h}^* = \sum_{i \in \tilde{S}_1^* \cap \tilde{S}_2^*} a_i.$$

Note that the "˜-version" of (2) may be assumed to be satisfied without any changes. (Replace the approximation $\tilde{S}_1$ by $\tilde{S}_1^*$ in case $p(\tilde{S}_1) < p(\tilde{S}_1^*)$.) The same is true for the knapsack constraints. Thus all that changes in the linear programs in the previous two sections is that the right hand side of the first constraint changes from 1 to $(1-\epsilon)$. Thus, a simple scaling argument shows that the resulting optimum will differ by at most an $O(\epsilon)$-fraction from the original value.

Remark 3. To be precise, the above argument is valid only if we assume that player 2 is able to solve his (lower level) problem exactly. If also player 2 applies ϵ-approximation algorithms instead, we get another factor of $(1-\epsilon)$ introduced in (3).

4.4. The mixed case

Finally, let us turn to the case where the item set contains both positive and negative items. This case is easily reduced to the two cases (beneficial and competitive model respectively) considered above: All we have to do is to split the item set A into the set A^+ and A^- of non-negative and negative items, respectively and to guess – again, up to a certain factor, say, $1/k$ – the associated parts of knapsacks 1 and 2 that are filled with non-negative and negative items, resp., in an optimal solution $S_1^* \cup S_2^*$ of (P3). Solving the "pure" (beneficial resp. competitive) cases for the approximately correct choice of corresponding knapsack capacities will then give solutions for the leader's problem that differ from the cooperative value by at most a factor of $(2+\epsilon)$ at the expense of an additional $O((\log W_1)(\log W_2))$ factor in the running time.

5. Remarks

We have presented a unified approach for computing solutions to the leader's problem with approximately optimal ratio, if compared to the outcome of the cooperative version of the problem. Without further going into details, we mention that in the lower bound examples (*cf.* the example in Section 2 and the examples in [1]), the maximum cooperative value equals the maximum value of (the optimistic version of) the bilevel problem. Thus our results also provide tight bounds for the ratio between the optimistic and pessimistic version of the bilevel problem itself.

A natural question to ask is about side payments: Player 1 can certainly enforce the optimistic value of the bilevel problem with arbitrarily small side payments. Thus, it seems natural to also investigate approximation algorithms in the optimistic setting.

References

[1] L. Chen, G. Zhang, Approximation algorithms for a bi-level knapsack problem, in: Proceedings of COCOA' 11, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 399–410.
[2] S. Dempe, K. Richter, Bilevel programming with knapsack constraints, Cent. Eur. J. Oper. Res. 8 (2000) 93–107.
[3] S. Dempe, Foundations of Bilevel Programming, Kluwer Academic Publishers, Dordrecht, 2002.
[4] A. Migdalas, P.M. Pardalos, P. Värbrand, Multilevel Optimization: Algorithms and Applications, Kluwer Academic Publishers, Dordrecht, 1998.
[5] B. Colson, P. Marcotte, G. Savard, Bilevel programming: a survey, 4OR: Quart. J. Oper. Res. 3 (2) (2005) 87–107.
[6] L. Brotcorne, S. Hanafi, R. Mansi, A dynamic programming algorithm for the bilevel knapsack problem, Oper. Res. Lett. 37 (3) (2009) 215–218.
[7] A. Caprara, M. Carvalho, A. Lodi, G. Woeginger, A complexity and approximability study of the bilevel knapsack problem, in: Proceedings of IPCO' 13, Springer, Berlin, Heidelberg, 2013, pp. 98–109.
[8] P. Loridan, J. Morgan, Weak via strong Stackelberg problem: new results, J. Global Optim. 8 (3) (1996) 263–287.
[9] Z. Wang, W. Xing, S.-C. Fang, Two-group knapsack game, Theoret. Comput. Sci. 411 (7–9) (2010) 1094–1103.
[10] Z. Wang, W. Xing, Two-person knapsack game, J. Ind. Manag. Optim. 6 (4) (2010) 847–860.
[11] X. Deng, Complexity issues in bilevel linear programming, in: Multilevel Optimization: Algorithms and Applications, Kluwer Academic Publishers, Dordrecht, 1998, pp. 149–164.
[12] R. Mansi, C. Alves, J. de Carvalho, S. Hanafi, An exact algorithm for bilevel 0–1 knapsack problems, Math. Probl. Eng. (2012), article ID 504713.
[13] O.H. Ibarra, C.E. Kim, Fast approximation algorithms for the knapsack and sum of subset problems, J. ACM 22 (4) (1975) 463–468.
[14] V.V. Vazirani, Approximation Algorithms, Springer-Verlag, New York, Inc., New York, NY, USA, 2001.
[15] C. Chekuri, S. Khanna, A PTAS for the multiple knapsack problem, in: Proceedings of SODA'00, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000, pp. 213–222.
[16] S. Sahni, Approximate algorithms for the 0/1 knapsack problem, J. ACM 22 (1) (1975) 115–124.
[17] S. DeNegre, Interdiction and discrete bilevel linear programming, Ph.D. thesis, Bethlehem, PA, USA, aAI3456385, 2011.
[18] K. Jain, M. Mahdian, A. Saberi, A new greedy approach for facility location problems, in: Proceedings of STOC' 02, 2002, pp. 731–740.