A Complexity and Approximability Study of the Bilevel Knapsack Problem

Alberto Caprara[1], Margarida Carvalho[2], Andrea Lodi[1], and Gerhard J. Woeginger[3]

[1] DEI, University of Bologna, Italy
[2] Departamento de Ciências de Computadores, Universidade do Porto, Portugal
[3] Department of Mathematics, TU Eindhoven, Netherlands

Abstract. We analyze three fundamental variants of the bilevel knapsack problem, which all are complete for the second level of the polynomial hierarchy. If the weight and profit coefficients in the knapsack problem are encoded in unary, then two of the bilevel variants are solvable in polynomial time, whereas the third is NP-complete. Furthermore we design a polynomial time approximation scheme for this third variant, whereas the other two variants cannot be approximated in polynomial time within any constant factor (assuming P≠NP).

Bilevel and Multilevel Optimization. In bilevel optimization the decision variables are split into two groups that are controlled by two decision makers called *leader* (on the upper level) and *follower* (on the lower level). Both decision makers have an objective function of their own and a set of constraints on their variables. Furthermore there are coupling constraints that connect the decision variables of leader and follower. The decision making process is as follows. First the leader makes his decision and fixes the values of his variables, and afterwards the follower reacts by setting his variables. The leader has perfect knowledge of the follower's scenario (objective function and constraints) and also of the follower's behavior. The follower observes the leader's action, and then optimizes his own objective function subject to the decisions made by the leader (and subject to the imposed constraints). As the leader's objective function does depend on the follower's decision, the leader must take the follower's reaction into account.

Bilevel and multilevel optimization have received much interest in the literature over the last decades; see for instance the books by Migdalas, Pardalos & Värbrand [15] and Dempe [3]. Multilevel optimization problems are extremely difficult from the computational point of view and cannot be expressed in terms of classical integer programs (which can only handle a single level of optimization). A ground-breaking paper by Jeroslow [11] established that various multilevel problems are complete for various levels of the polynomial hierarchy in computational complexity theory; see Papadimitriou [16] for more information. Further hardness results for broad families of multilevel optimization problems are due to Deng [6] and Dudás, Klinz & Woeginger [7].

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 98–109, 2013.

Standard Knapsack Problems and Bilevel Knapsack Problems. An instance of the knapsack problem consists of a set of items with given weights and profits together with a knapsack with a given weight capacity. The objective is to select a subset of the items with maximum total profit, subject to the constraint that the overall selected item weight must fit into the knapsack. The knapsack problem is well-known to be NP-complete [10].

Over the last few years, a variety of authors has studied certain bilevel variants of the knapsack problem. Dempe & Richter [4] considered the variant where the leader controls the weight capacity of the knapsack, and where the follower decides which items are packed into the knapsack. Mansi, Alves, de Carvalho & Hanafi [14] consider a bilevel knapsack variant where the item set is split into two parts, one of which is controlled by the leader and one controlled by the follower. DeNegre [5] suggests yet another variant, where both players have a knapsack on their own; the follower can only choose from those items that the leader did not pack. Section 1 gives precise definitions of these three variants and provides further information on them.

Our Contributions. We pinpoint the computational complexity of the three bilevel knapsack variants mentioned above: they are complete for the complexity class Σ_2^p and hence located at the second level of the polynomial hierarchy. If a problem is Σ_2^p-complete, there is no way of formulating it as a single-level integer program of polynomial size *unless the polynomial hierarchy collapses* (a highly unlikely event which would cause a revolution in complexity theory). The complexity class Σ_2^p is the natural hotbed for bilevel problems that are built on top of NP-complete single-level problems; as a rule of thumb, the bilevel version of an NP-complete problem should always be expected to be Σ_2^p-complete.

In a second line of investigation, we study these bilevel problems under *unary* encodings. The classical knapsack problem becomes polynomially solvable if the input is encoded in unary, and it is only natural to expect a similar behavior from our bilevel knapsack problems. Indeed, two of our three bilevel variants become polynomially solvable if the input is encoded in unary, and thus show exactly the type of behavior that one would expect from a knapsack variant. The third variant behaves differently and stubbornly becomes NP-complete.

Our third line of results studies the approximability of the three bilevel knapsack variants. As a rule of thumb Σ_2^p-hard problems do not allow good approximation algorithms. Indeed, the literature only contains negative results in this direction that establish the inapproximability of various Σ_2^p-hard optimization problems; see Ko & Lin [12] and Umans [18]. Two of our bilevel knapsack variants (actually the same ones that are easy under unary encodings) behave exactly as expected and do not allow polynomial time approximation algorithms with finite worst case guarantee, assuming P$\neq$NP. For the third variant, however, we derive a polynomial time approximation scheme. This is the first approximation scheme for a Σ_2^p-hard optimization problem in the history of approximation algorithms, and from the technical point of view it is the most sophisticated result in this paper.

Our investigations provide a complete and clean picture of the complexity landscape of the considered bilevel knapsack problems. We expect that our results will also be useful in classifying and understanding other bilevel problems, and that our hardness proofs will serve as stepping stones for future results.

Organization of the Paper. Section 1 defines the three bilevel knapsack variants and summarizes the literature on them. Section 2 presents the Σ_2^p-completeness results for these problems (under the standard binary encoding) and also discusses their behavior under unary encodings. Section 3 discusses the approximability and inapproximability behavior of the considered bilevel problems.

1 Definitions and Preliminaries

In bilevel optimization the follower observes the leader's action, and then optimizes his own objective function value subject to the decisions made by the leader and subject to the imposed constraints. This statement does not fully determine the follower's behavior: there might be many feasible solutions that all are optimal for the follower but yield different objective values for the leader. Which one will the follower choose? In the *optimistic* scenario the follower always picks the optimal solution that yields the *best* objective value for the leader, and in the *pessimistic* scenario he picks the solution that yields the *worst* objective value for the leader. All our negative (hardness) results and all our positive (polynomial time) results hold for the optimistic scenario as well as for the pessimistic scenario.

In the following subsections, we use x and $x_1, \ldots, x_m$ to denote the variables controlled by the leader, and $y_1, \ldots, y_n$ to denote the variables controlled by the follower. Furthermore we use a_i, b_i, c_i and A, B, C, C' to denote item profits, item weights, cost coefficients, upper bounds, and lower bounds; all these numbers are non-negative integers (or rationals). As usual, we use the notation $a(I) = \sum_{i \in I} a_i$ for an index set I, and $a(x) = \sum_i a_i x_i$ for a 0-1 vector x.

1.1 The Dempe-Richter (DR) Variant

The first occurrence of a bilevel knapsack problem in the optimization literature seems to be due to Dempe & Richter [4]. In their problem variant DR as depicted in Figure 1, the leader controls the capacity x of the knapsack while the follower controls all items and decides which of them are packed into the knapsack. The objective function of the leader depends on the knapsack capacity x as well as on the packed items, whereas the objective function of the follower solely depends on the packed items.

All decision variables in this bilevel program are integers; the knapsack capacity satisfies $x \in \mathbb{Z}$ and the variables $y_1, \ldots, y_n \in \{0, 1\}$ encode whether item i is packed into the knapsack ($y_i = 1$) or not ($y_i = 0$). We note that in the original model in [4] the knapsack capacity x is continuous; one nasty consequence

$$\text{Maximize} \quad f_1(x,y) \;=\; Tx + \sum_{i=1}^{n} a_i y_i \tag{1a}$$

$$\text{subject to} \quad C \;\le\; x \;\le\; C' \tag{1b}$$

$$\text{where } y_1,\ldots,y_n \text{ solves the follower's problem}$$

$$\max \sum_{i=1}^{n} b_i y_i \quad \text{s.t.} \quad \sum_{i=1}^{n} b_i y_i \le x \tag{1c}$$

Fig. 1. The bilevel knapsack problem DR

of this continuous knapsack capacity is that the problem (1a)–(1c) may fail to have an optimal solution. The computational complexity of the problem remains the same, no matter whether x is integral or continuous.

Dempe & Richter [4] discuss approximation algorithms for DR, and furthermore design a dynamic programming algorithm that solves variant DR in pseudo-polynomial time. Brotcorne, Hanafi & Mansi [1] derive another (simpler) dynamic program with a much better running time.

1.2 The Mansi-Alves-de-Carvalho-Hanafi (MACH) Variant

Mansi, Alves, de Carvalho & Hanafi [14] consider a bilevel knapsack variant where both players pack items into the knapsack. There is a single common knapsack for both players with a prespecified capacity of C. The item set is split into two parts, which are respectively controlled by the leader and the follower. The leader starts the game by packing some of his items into the knapsack, and then the follower adds some further items from his set. Figure 2 specifies the bilevel problem MACH. The 0-1 variables $x_1,\ldots,x_m$ (for the leader) and $y_1,\ldots,y_n$ (for the follower) encode whether item i is packed into the knapsack.

Mansi, Alves, de Carvalho & Hanafi [14] describe several applications of their problem in revenue management, telecommunication, capacity allocation, and

$$\text{Maximize} \quad f_2(x,y) \;=\; \sum_{j=1}^{m} a_j x_j + \sum_{i=1}^{n} a'_i y_i \tag{2a}$$

$$\text{subject to} \quad y_1,\ldots,y_n \text{ solves the follower's problem}$$

$$\max \sum_{i=1}^{n} b'_i y_i \quad \text{s.t.} \quad \sum_{i=1}^{n} c'_i y_i \le C - \sum_{j=1}^{m} c_j x_j \tag{2b}$$

Fig. 2. The bilevel knapsack problem MACH

transportation. Variant MACH has also been studied in a more general form by Brotcorne, Hanafi & Mansi [2], who reduced the model to one-level in pseudo-polynomial time.

1.3 The DeNegre (DN) Variant

DeNegre [5] proposes another bilevel knapsack variant where both players hold their own private knapsacks and choose items from a common item set. First the leader packs some of the items into his private knapsack, and then the follower picks some of the remaining items and packs them into his private knapsack. The objective of the follower is to maximize the profit of the items in his knapsack, and the objective of the hostile leader is to minimize this profit.

$$\text{Minimize} \quad f_3(x,y) = \sum_{i=1}^{n} b_i y_i \tag{3a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} a_i x_i \le A \tag{3b}$$

$$\text{where } y_1,\dots,y_n \text{ solves the follower's problem}$$

$$\max \sum_{i=1}^{n} b_i y_i \quad \text{s.t.} \sum_{i=1}^{n} b_i y_i \le B \quad \text{and} \tag{3c}$$

$$y_i \le 1 - x_i \quad \text{for } 1 \le i \le n \tag{3d}$$

Fig. 3. The bilevel knapsack problem DN

Figure 3 depicts the bilevel problem DN. The 0-1 variables $x_1,\dots,x_n$ (for the leader) and $y_1,\dots,y_n$ (for the follower) encode whether the corresponding item is packed into the knapsack. The interdiction constraint $y_i \le 1 - x_i$ in (3d) enforces that the follower cannot take item i once the leader has picked it. Note that leader and follower have exactly opposing objectives.

2 Hardness Results

As usual, we consider the decision versions corresponding of our optimization problems: "Does there exist an action of the leader that makes his objective value at least as good as some given bound?" Theorem 1 summarizes the results under the standard binary encoding; its proof follows from the fact that all decision problems are in the class Σ_2^p (see Chapter 17 in Papadimitriou's book [16]) and by reductions from the decision problem Subset-Sum-Interval, which has been proved to be Σ_2^p-complete by Eggermont & Woeginger [8].

Theorem 1. *The decision versions of (a) DR, (b) MACH, and (c) DN in binary encoding are Σ_2^p-complete.*

If the input data is encoded in unary, the corresponding problem variants unary-DR and unary-MACH are solvable in polynomial time by dynamic programming. These results are routine and perfectly expected, and their proofs use as main tool the polynomial time algorithm for the standard knapsack problem under unary encodings (see Garey & Johnson [10]). The third variant unary-DN is much more interesting, as it turns out to be NP-complete. Our reduction is from the VERTEX-COVER problem in undirected graphs; see [10].

Problem: VERTEX-COVER

Instance: An undirected graph $G = (V, E)$; an integer bound t.

Question: Does G possess a vertex cover of size t, that is, a subset $T \subseteq V$ such that every edge in E has at least one of its vertices in T?

A *Sidon sequence* is a sequence $s_1 < s_2 < \cdots < s_n$ of positive numbers in which all pairwise sums $s_i + s_j$ with $i < j$ are different. Erdős & Turán [9] showed that for any odd prime p, there exists a Sidon sequence of p integers that all are below $2p^2$. The argument in [9] is constructive and yields a simple polynomial time algorithm for finding Sidon sequences of length n whose elements are bounded by $O(n^2)$.

We start our polynomial time reduction from an arbitrary instance $G = (V, E)$ and k of VERTEX-COVER. Let $n = |V| \geq 10$, and let $v_1, \ldots, v_n$ be an enumeration of the vertices in V. We construct a Sidon sequence $s_1 < s_2 < \cdots < s_n$ whose elements are polynomially bounded in n. We define $S = \sum_{i=1}^{n} s_i$ as the sum of all numbers in the Sidon sequence, and we construct the following instance of DN as specified in (3a)–(3d).

- For every vertex v_i, we create a corresponding vertex-item with leader's weight $a(v_i) = 1$ and follower's weight $b(v_i) = S + s_i$.
- For every edge $e = [v_i, v_j]$, we create a corresponding edge-item with leader's weight $a(e) = t + 1$ and follower's weight $b(e) = 5S - s_i - s_j$.
- The capacity of the leader's knapsack is $A = t$, and the capacity of the follower's knapsack is $B = 7S$.

We claim that in the DN instance the leader can make his objective value $\leq 7S-1$ if and only if the VERTEX-COVER instance has answer YES.

(Proof of if). Assume that there exists a vertex cover T of size $|T| = t$. Then a good strategy for the leader is to put the t vertex-items that correspond to vertices in T into his knapsack, which fills his knapsack of capacity $A = t$ to the limit. Suppose for the sake of contradiction that afterwards the follower can still fill his knapsack with total weight $7S$. Then the follower must pick at least one edge-item (he can pack at most six vertex-items, and their weight would stay strictly below $7S$). Furthermore the follower cannot pick two edge-items (since every edge-item has weight greater than $4S$). Consequently the follower must pick exactly one edge-item that corresponds to some edge $e = [v_i, v_j]$.

The remaining space in the follower's knapsack is $2S + s_i + s_j$ and must be filled by two vertex-items. By the definition of a Sidon sequence, the only way of doing this would be by picking the two vertex-items corresponding to v_i and v_j. But that's impossible, as at least one of the vertices v_i and v_j is in the cover T so that the item has already been picked by the leader. This contradiction shows that the follower cannot reach an objective value of $7S$.

(Proof of only if). Now let us assume that the graph G does not possess any vertex cover of size t, and let us consider the game right after the move of the leader. Since the leader can pack at most t vertex-items, there must exist some edge $e = [v_i, v_j]$ in E for which the leader has neither picked the item corresponding to v_i nor the item corresponding to v_j. Then the follower may pick the vertex-item v_i, the vertex-item v_j, and the edge-item e, which brings him a total weight of $7S$.

Theorem 2. *The decision version of the bilevel problem DN in unary encoding is NP-complete, both for the optimistic scenario and the pessimistic scenario.*

Proof. The above construction can be performed in polynomial time. As the elements in the Sidon sequence are polynomially bounded in $|V|$, also their sum S and all the integers in our construction are polynomially bounded in $|V|$. In particular, this yields that the unary encoding length of the constructed DN instance is polynomially bounded in $|V|$. Together with the above arguments, this implies that DN in unary encoding is NP-hard.

To show containment of DN under unary encoding in class NP, we use the optimal move of the leader as NP-certificate. The certificate is short, as it just specifies a subset of the items. To verify the certificate, we have to check that the follower cannot pick any item set of high weight. Since all weights are encoded in unary, this checking amounts to solving a standard knapsack problem in unary encoding, which can be done in polynomial time. □

3 Approximability and Inapproximability

The Σ_2^p-completeness proofs for DR and MACH have devastating consequences in terms of existence of a polynomial time approximation for them: it is Σ_2^p-hard to distinguish the DR instances in which the leader can reach an objective value of 1 from those DR instances in which the leader can only reach objective value 0. An analogous statement holds for problem MACH. As a polynomial time approximation algorithm with finite worst case guarantee would be able to distinguish between these two instance types, we get the following result.

Corollary 1. *Problems DR and MACH do not possess a polynomial time approximation algorithm with finite worst case guarantee, unless P=Σ_2^p and therefore P=NP holds.* □

The statement in Corollary 1 is not surprising, as the literature on the approximability of Σ_2^p-hard optimization problems entirely consists of such negative

statements that show the inapproximability of various problems; see Ko & Lin [12] and Umans [18]. The following theorem breaks with this old tradition, and presents the first approximation scheme for a Σ_2^p-hard optimization problem.

Theorem 3. *Problem DN has a polynomial time approximation scheme.*

The rest of this section is dedicated to the proof of Theorem 3. We apply and extend a number of rounding tricks from the seminal paper [13] by Lawler, we use approximation schemes from the literature as a black box, and we also add a number of new ingredients and rounding tricks.

Throughout the proof we will consider a fixed instance of problem DN. Without loss of generality we assume that no item i in the instance satisfies $b_i > B$: such items could never be used by the follower, and hence are irrelevant and may as well be ignored. Let ε with $0 < \varepsilon < 1/3$ be a small positive real number; for the sake of simplicity we will assume that the reciprocal value $1/\varepsilon$ is integer.

Our global goal is to determine in polynomial time a feasible solution for the leader that yields an objective value of at most $(1+\varepsilon)^4$ times the optimum. This will be done by a binary search over the range $0, 1, \ldots, B$ that (approximately) sandwiches the optimal objective value between a lower and an upper bound. Whenever we bisect the search interval between these bounds at some value U, we have to decide whether the optimal objective value lies below or above U. If the optimal objective value lies below U, then Lemma 5 (derived in Section 3.1) and Lemma 6 (derived in Section 3.2) show how to find and how to verify in polynomial time an approximate solution for the leader whose objective value is bounded by $(1+\varepsilon)^3\, U$. If these lemmas succeed then we make U the new upper bound. If the lemmas fail to produce an approximate objective value of at most $(1+\varepsilon)^3\, U$, then we make U the new lower bound. The binary search process terminates as soon as the upper bound comes within a factor of $1+\varepsilon$ of the lower bound. Note that we then lose a factor of $1+\varepsilon$ between upper and lower bound, and that we lose a factor of at most $(1+\varepsilon)^3$ by applying the lemmas. All in all, this yields the desired approximation guarantee of $(1+\varepsilon)^4$ and completes the proof of Theorem 3.

3.1 How to Handle the Central Cases

Throughout this section, we assume that U is an upper bound on the optimal objective value of the considered instance with

$$B/2 \;\le\; U \;\le\; B/(1+\varepsilon). \tag{4}$$

The items $i = 1, \ldots, n$ are partitioned according to their b-values into so-called *large* items that satisfy $U < b_i$, into *medium* items that satisfy $\varepsilon U < b_i \le U$, and into *small* items that satisfy $b_i \le \varepsilon U$. We denote by L, M, S respectively the set of large, medium, small items. Furthermore a medium item i belongs to class $\mathcal{C}_k$, if it satisfies

$$k\varepsilon^2 U \;\le\; b_i \;<\; (k+1)\varepsilon^2 U.$$

Note that only classes $\mathcal{C}_k$ with $1/\varepsilon \leq k \leq 1/\varepsilon^2$ play a role in this classification. By (4) the overall size of $2/\varepsilon$ medium items exceeds the capacity of the follower's knapsack, so that the follower uses at most $2/\varepsilon$ medium items in his solution.

In the following we analyze two scenarios. In the first scenario, the solution x^* for the leader and the solution y^* for the follower both will carry a superscript*. The sets of large, medium, small items packed by x^* into the leader's knapsack will be denoted respectively by L_x^*, M_x^*, S_x^*, and the corresponding sets for y^* and the follower are denoted L_y^*, M_y^*, S_y^*. In the second scenario we use analogous notations with the superscript$^\#$. The first scenario is centered around an optimal solution x^* for the leader. The second scenario considers another feasible solution $x^\#$ for the leader that we call the *aligned* version of x^*.

- Solution $x^\#$ packs all large items into the knapsack; hence $L_x^\# = L$.
- Solution $x^\#$ selects the following items from class $\mathcal{C}_k$: it picks an item $i \in M_x^* \cap \mathcal{C}_k$ if and only if $\mathcal{C}_k - M_x^*$ contains at most $2/\varepsilon$ items j with $b_j \leq b_i$. (By this choice, the $2/\varepsilon$ items with smallest b-value in $\mathcal{C}_k - M_x^*$ coincide with the $2/\varepsilon$ items with smallest b-value in $\mathcal{C}_k - M_x^\#$.) Note that $M_x^\# \subseteq M_x^*$.
- For the small items we first determine a $(1+\varepsilon)$-approximate solution to the following auxiliary problem (Aux): find a subset $Z \subseteq S$ of the small items that minimizes $b(Z)$, subject to the covering constraint $a(Z) \geq a(L_x^\# \cup M_x^\#) + a(S) - A$. Solution $x^\#$ then packs the complementary set $S_x^\# = S - Z$.

This completes the description of $x^\#$, which is easily seen to be a feasible action for the leader. Note that also the optimal solution x^* packs all the large items, as otherwise the follower could pack a large item and thereby push the objective value above the bound U. Then $L_x^\# = L_x^*$ and $M_x^\# \subseteq M_x^*$ imply $a(L_x^* \cup M_x^*) \geq a(L_x^\# \cup M_x^\#)$, which yields

$$A \;\geq\; a(L_x^* \cup M_x^* \cup S_x^*) \;\geq\; a(L_x^\# \cup M_x^\#) + a(S_x^*). \tag{5}$$

As $a(S_x^*) = a(S) - a(S - S_x^*)$, we conclude from (5) that the set $S - S_x^*$ satisfies the covering constraint in the auxiliary problem (Aux). Hence the optimal objective value of (Aux) is upper bounded by $b(S - S_x^*)$, and any $(1+\varepsilon)$-approximate solution Z to (Aux) must satisfy $b(Z) \leq (1+\varepsilon)\, b(S - S_x^*)$, which is equivalent to

$$b(S - S_x^\#) \;\leq\; (1+\varepsilon)\, b(S - S_x^*). \tag{6}$$

The following lemma demonstrates that the aligned solution $x^\#$ is almost as good for the leader as the underlying optimal solution x^*.

Lemma 4. *If the leader uses the aligned solution $x^\#$, then every feasible reaction $y^\#$ for the follower yields an objective value $f_3(x^\#, y^\#) \leq (1+2\varepsilon)\, U$.*

Proof. Suppose for the sake of contradiction that there exists a reaction $y^\#$ for the follower that yields an objective value of $f_3(x^\#, y^\#) > (1+2\varepsilon)\, U$. Based on $y^\#$ we will construct another solution y^* for the follower in the first scenario:

- Solution y^* does not use any large item; hence $L_y^* = \emptyset$.
- Solution y^* picks the same number of items from every class $\mathcal{C}_k$ as $y^\#$ does. It avoids items in x^* and selects the $|\mathcal{C}_k \cap M_y^\#|$ items in $\mathcal{C}_k - M_x^*$ that have the smallest b-values.

– Finally we add small items from $S - S_x^*$ to the follower's knapsack, until no further item fits or until we run out of items.

Solution $y^\#$ packs at most $2/\varepsilon$ medium items, and hence uses at most $2/\varepsilon$ items from $\mathcal{C}_k$. By our choice of medium items for $x^\#$ we derive $b(\mathcal{C}_k \cap M_y^*) \leq b(\mathcal{C}_k \cap M_y^\#)$ for every k, which implies

$$b(M_y^*) \;\leq\; b(M_y^\#) \;\leq\; B. \tag{7}$$

Solution y^* only selects items that are not used by x^*, and inequality (7) implies that all the selected items indeed fit into the follower's knapsack. Hence y^* constitutes a feasible reaction of the follower if the leader chooses x^*.

Next, let us quickly go through the item types. First of all neither solution y^* nor solution $y^\#$ can use any large item, so that we have

$$b(L_y^*) \;=\; b(L_y^\#) \;=\; 0. \tag{8}$$

For the medium items, the ratio between the smallest b-value and the largest b-value in class $\mathcal{C}_k$ is at least $k/(k+1) \geq 1-\varepsilon$. Hence we certainly have $b(\mathcal{C}_k \cap M_y^*) \geq (1-\varepsilon)\, b(\mathcal{C}_k \cap M_y^\#)$, which implies

$$b(M_y^*) \;\geq\; (1-\varepsilon)\, b(M_y^\#). \tag{9}$$

Let us turn to the small items. Suppose that y^* cannot accommodate all small items from $S - S_x^*$ in the follower's knapsack. Then some small item i with $b_i < \varepsilon U$ does not fit, which with (4) leads to $b(y^*) > B - \varepsilon\, U \geq U$. As this violates our upper bound U on the optimal objective value, we conclude that y^* accommodates all such items and satisfies $S_y^* = S - S_x^*$. This relation together with (6) and the disjointness of the sets $S_x^\#$ and $S_y^\#$ yields

$$b(S_y^*) \;=\; b(S - S_x^*) \;\geq\; \frac{b(S - S_x^\#)}{1+\varepsilon} \;\geq\; \frac{b(S_y^\#)}{1+\varepsilon} \;>\; (1-\varepsilon)\, b(S_y^\#). \tag{10}$$

Now let us wrap things up. If the leader chooses x^*, the follower may react with the feasible solution y^* and get an objective value

$$\begin{aligned} f_3(x^*, y^*) &= b(L_y^*) + b(M_y^*) + b(S_y^*) \\ &> (1-\varepsilon)\, b(L_y^\#) + (1-\varepsilon)\, b(M_y^\#) + (1-\varepsilon)\, b(S_y^\#) \\ &= (1-\varepsilon)\, f_3(x^\#, y^\#) \;>\; (1-\varepsilon)(1+2\varepsilon)\, U \;>\; U. \end{aligned}$$

Here we used the estimates in (8), (9), and (10). As this objective value violates the upper bound U, we have reached the desired contradiction. □

Lemma 5. *Given an upper bound U on the objective value that satisfies (4), one can compute in polynomial time a feasible solution x for the leader, such that every reaction y of the follower has $f_3(x, y) \leq (1+\varepsilon)^3\, U$.*

Proof. If we did not only know the bound U but also an optimal solution x^*, then we could simply determine the corresponding aligned solution $x^\#$ and apply Lemma 4. We will bypass this lack of knowledge by checking many candidates for the set $M_x^\#$. Let us recall how the aligned solution $x^\#$ picks medium items from class $\mathcal{C}_k$.

- If $|\mathcal{C}_k - M_x^*| \leq 2/\varepsilon$ then $M_x^\# \cap \mathcal{C}_k = M_x^* \cap \mathcal{C}_k$. Note that there are only $O(|\mathcal{C}_k|^{2/\varepsilon})$ different candidates for $M_x^\# \cap \mathcal{C}_k$.
- If $|\mathcal{C}_k - M_x^*| > 2/\varepsilon$ then $M_x^\# \cap \mathcal{C}_k$ is a subset of M_x^*; an item i from $M_x^* \cap \mathcal{C}_k$ enters $M_x^\#$ if there are at most $2/\varepsilon$ items $j \in \mathcal{C}_k - M_x^*$ with $b_j \leq b_i$. Note that $M_x^\# \cap \mathcal{C}_k$ is fully determined by the $2/\varepsilon$ items with smallest b-value in $\mathcal{C}_k - M_x^*$. As there are only $O(|\mathcal{C}_k|^{2/\varepsilon})$ ways for choosing these $2/\varepsilon$ items, there are only $O(|\mathcal{C}_k|^{2/\varepsilon})$ different candidates for $M_x^\# \cap \mathcal{C}_k$.

Altogether there are only $O(|\mathcal{C}_k|^{2/\varepsilon})$ ways of picking the medium items from class $\mathcal{C}_k$. As every class satisfies $|\mathcal{C}_k| \leq n$ and as there are only $1/\varepsilon^2$ classes to consider, we get a polynomial number $O(n^{2/\varepsilon^3})$ of possibilities for choosing the set $M_x^\#$ in the aligned solution. Summarizing, we only need to check a polynomial number of candidates for set $M_x^\#$.

How do we check such a candidate $M_x^\#$? The aligned solution always uses $L_x^\# = L$, and the auxiliary problem (Aux) is fully determined once $M_x^\#$ and $L_x^\#$ have been fixed. We approximate the auxiliary problem by standard methods (see for instance Pruhs & Woeginger [17]), and thus also find the set $S_x^\#$ in polynomial time. This yields the full corresponding aligned solution $x^\#$. It remains to verify the quality of this aligned solution for the leader, which amounts to analyzing the resulting knapsack problem at the follower's level. We use one of the standard approximation schemes for knapsack as for instance described by Lawler [13], and thereby get a $(1+\varepsilon)$-approximate solution for the follower's problem.

While checking and scanning through the candidates, we eventually must hit a good candidate $M_x^\#$ that yields the correct aligned version x of an optimal solution. By Lemma 4 the corresponding objective value $f_3(x, y)$ is bounded by $(1+2\varepsilon)\,U$. Then the approximation scheme finds an objective value of at most $(1+\varepsilon)(1+2\varepsilon)\,U \leq (1+\varepsilon)^3 U$. This completes the proof of the lemma. □

3.2 How to Handle the Boundary Cases

Finally let us discuss the remaining cases where U does not satisfy the bounds in (4). The first case $U > B/(1+\varepsilon)$ is trivial, as the objective value never exceeds the follower's knapsack capacity B; hence in this case the objective value will always stay below $(1+\varepsilon)\,U$. The second case $U < B/2$ is settled by the following lemma; its proof is based on the framework of Pruhs & Woeginger [17] and can be found in the long version of this paper.

Lemma 6. *Given an upper bound $U < B/2$ on the objective value, one can compute in polynomial time a feasible solution x for the leader, such that every reaction y of the follower has $f_3(x, y) \leq (1+\varepsilon)\,U$.* □

References

1. Brotcorne, L., Hanafi, S., Mansi, R.: A dynamic programming algorithm for the bilevel knapsack problem. Operations Research Letters 37, 215–218 (2009)
2. Brotcorne, L., Hanafi, S., Mansi, R.: One-level reformulation of the bilevel knapsack problem using dynamic programming. Technical Report, Université de Valenciennes et du Hainaut-Cambrésis, France (2011)
3. Dempe, S.: Foundations of Bilevel Programming. Kluwer Academic Publishers, Dordrecht (2002)
4. Dempe, S., Richter, K.: Bilevel programming with Knapsack constraint. Central European Journal of Operations Research 8, 93–107 (2000)
5. DeNegre, S.: Interdiction and discrete bilevel linear programming. Ph.D. dissertation, Lehigh University (2011)
6. Deng, X.: Complexity issues in bilevel linear programming. In: Migdalas, A., Pardalos, P.M., Värbrand, P. (eds.) Multilevel Optimization: Algorithms and Applications, pp. 149–164. Kluwer Academic Publishers, Dordrecht (1998)
7. Dudás, T., Klinz, B., Woeginger, G.J.: The computational complexity of multi-level bottleneck programming problems. In: Migdalas, A., Pardalos, P.M., Värbrand, P. (eds.) Multilevel Optimization: Algorithms and Applications, pp. 165–179. Kluwer Academic Publishers, Dordrecht (1998)
8. Eggermont, C., Woeginger, G.J.: Motion planning with pulley, rope, and baskets. In: Proceedings of the 29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012. Leibniz International Proceedings in Informatics, vol. 14, pp. 374–383 (2012)
9. Erdős, P., Turán, P.: On a problem of Sidon in additive number theory, and on some related problems. Journal of the London Mathematical Society 16, 212–215 (1941)
10. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco (1979)
11. Jeroslow, R.: The polynomial hierarchy and a simple model for competitive analysis. Mathematical Programming 32, 146–164 (1985)
12. Ko, K., Lin, C.-L.: On the complexity of min-max optimization problems and their approximation. In: Du, D.-Z., Pardalos, P.M. (eds.) Minimax and Applications, pp. 219–239. Kluwer Academic Publishers, Dordrecht (1995)
13. Lawler, E.L.: Fast approximation algorithms for knapsack problems. Mathematics of Operations Research 4, 339–356 (1979)
14. Mansi, R., Alves, C., de Carvalho, J.M.V., Hanafi, S.: An exact algorithm for bilevel 0-1 knapsack problems. Mathematical Problems in Engineering, Article ID 504713 (2012)
15. Migdalas, A., Pardalos, P.M., Värbrand, P.: Multilevel Optimization: Algorithms and Applications. Kluwer Academic Publishers, Dordrecht (1998)
16. Papadimitriou, C.H.: Computational Complexity. Addison-Wesley (1994)
17. Pruhs, K., Woeginger, G.J.: Approximation schemes for a class of subset selection problems. Theoretical Computer Science 382, 151–156 (2007)
18. Umans, C.: Hardness of approximating $\sum_2^{\mathrm{p}}$ minimization problems. In: Proceedings of the 40th Annual Symposium on Foundations of Computer Science, FOCS 1999, pp. 465–474 (1999)