Theory of Computing Systems (2020) 64:17–34
https://doi.org/10.1007/s00224-019-09917-z

The Clever Shopper Problem

Laurent Bulteau[1] · Danny Hermelin[2] · Dušan Knop[3] · Anthony Labarre[1] · Stéphane Vialette[1]

Published online: 22 April 2019

Abstract

We investigate a variant of the so-called Internet Shopping problem introduced by Blazewicz et al. (Appl. Math. Comput. Sci. **20**, 385–390, 2010), where a customer wants to buy a list of products at the lowest possible total cost from shops which offer discounts when purchases exceed a certain threshold. Although the problem is NP-hard, we provide exact algorithms for several cases, e.g. when each shop sells only two items, and an FPT algorithm for the number of items, or for the number of shops when all prices are equal. We complement each result with hardness proofs in order to draw a tight boundary between tractable and intractable cases. Finally, we give an approximation algorithm and hardness results for the problem of maximising the sum of discounts

Keywords Clever shopper · Internet shopping · Fixed parameter tractability

This article is part of the Topical Collection on *Computer Science Symposium in Russia (2018)*

✉ Anthony Labarre
anthony.labarre@u-pem.fr

Laurent Bulteau
laurent.bulteau@u-pem.fr

Danny Hermelin
hermelin@bgu.ac.il

Dušan Knop
dusan.knop@fit.cvut.cz

Stéphane Vialette
stephane.vialette@u-pem.fr

[1] Université Paris-Est, LIGM (UMR 8049), CNRS, ENPC, ESIEE Paris, UPEM, F-77454, Marne-la-Vallée, France

[2] Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Beersheba, Israel

[3] Department of Theoretical Computer Science, Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic

1 Introduction

Blazewicz et al. [4] introduced and described the INTERNET SHOPPING problem as follows: given a set of shops offering products at various prices and the delivery costs for each set of items bought from each shop, find where to buy each product from a shopping list at a minimum total cost. The problem is known to be NP-hard in the strong sense even when all products are free and all delivery costs are equal to one, and admits no polynomial $(c \ln n)$-approximation algorithm (for any $0 < c < 1$) unless P = NP (here n is the number of products).

A more realistic variant takes into account discounts offered by shops in some cases. These could be offered, for instance, when the shopper's purchases exceed a certain amount, or in the case of special promotions where buying several items together costs less than buying them separately. Blazewicz et al. [5] investigated such a variant, which features a concave increasing discount function on the products' prices. They showed that the problem is NP-complete in the strong sense even if each product appears in at most three shops and each shop sells exactly three products, as well as in the case where each product is available at three different prices and each shop has all products but sells exactly three of them at the same price. A variant where two separate discount functions are taken into account (one for the deliveries, the other for the prices) was also recently introduced and studied by [6].

In this work, we investigate the case where a shopper aims to buy n books from m shops with free shipping; additionally, each shop offers a discount when purchases exceed a certain threshold (discounts and thresholds are specific to each shop). We show that the associated decision problem, which we call the CLEVER SHOPPER problem, is already NP-complete when only two shops are available, or when all books are available from two shops and each shop sells exactly three books. We also obtain parameterised hardness results: namely, that CLEVER SHOPPER is W[1]-hard when the parameter is m or the number of shops in a solution, and that it admits no polynomial-size kernel. On the positive side, we give a polynomial-time algorithm for the case where every shop sells at most two books, an XP algorithm for the case where few shops sell books at small prices, an FPT algorithm with parameter n, and another FPT algorithm with parameter m. We note that CLEVER SHOPPER generalises well-studied problems such as BIN COVERING [2] and H-INDEX MANIPULATION [20].

Most of the results in this paper previously appeared in the proceedings of CSR'18 [8]. The main changes consist in the addition of pseudocode for Algorithms 1 and 2; the proof of Proposition 4; and Proposition 9, which solves an open problem from [8].

2 Notation and Definitions

Let us now formally define CLEVER SHOPPER. For $n \in \mathbb{N}$, let $[n] = \{1, 2, \ldots, n\}$. Let B be a set of books to buy, S be a set of shops, $E \subseteq B \times S$ encodes the availability of the books in the shops, and $w : E \to \mathbb{N}$ encodes the prices. Choosing a shop in which to buy each book is encoded as a subset $E' \subseteq E$, such that each book is *covered exactly once* (i.e., any $b \in B$ has degree 1 in E'). A *discount* $d_s \in \mathbb{R}^+$ is associated to each shop s and offered when a *threshold* $t_s \in \mathbb{R}^+$ is reached, which is formally defined using the following *threshold function*:

$$\delta(s, E', d_s, t_s) = \begin{cases} d_s & \text{if } \sum_{(b,s)\in E'} w(e) \geq t_s, \\ 0 & \text{otherwise.} \end{cases}$$

We refer to the function $\mathscr{D}$ that maps each shop s to the pair (d_s, t_s) as the *discount function*. To lighten the notation, we will sometimes denote the cost of a book b at a shop s using $w(b, s)$ instead of $w(\{b, s\})$ when no confusion can arise. The problem we study is formally stated below.

Clever Shopper

Input: an edge-weighted bipartite graph $G = (B \cup S, E, w)$; a discount function $\mathscr{D}$; a bound $K \in \mathbb{N}$.

Question: is there a subset $E' \subseteq E$ that covers each element of B exactly once and such that $\sum_{e\in E'} w(e) - \sum_{s\in S} \delta(s, E', d_s, t_s) \leq K$?

3 Hardness Results

We prove in this section several hardness results under various restrictions, both with regards to classical complexity theory and parameterised complexity theory. We first show that Clever Shopper is NP-complete even if there are only two shops to choose from. For this first hardness result, we need book prices to be encoded in binary (i.e. they can be exponentially high compared to input size).

Proposition 1 Clever Shopper *is NP-complete in the weak sense (i.e., prices are encoded in binary), even when* $|S| = 2$.

Proof (reduction from Partition*)* Recall the well-known NP-complete Partition problem [17]: given a finite set A and a size $\omega(a) \in \mathbb{N}$ for each element in A, decide whether there exists a subset $A' \subseteq A$ such that $\sum_{a\in A'} \omega(a) = \sum_{a\in A\setminus A'} \omega(a)$.

Let $\mathcal{I} = (A, \omega)$ be an instance of Partition, and $T = \sum_{a\in A} \omega(a)$. We obtain an instance $\mathcal{I}'$ of Clever Shopper as follows: introduce two shops s_1 and s_2 with $(d_{s_1}, t_{s_1}) = (d_{s_2}, t_{s_2}) = (1, T/2)$. Each item $a \in A$ is a book that shops s_1 and s_2 sell for the same price — namely, $\omega(a)$. It is now clear that there exists a subset $A' \subseteq A$ such that $\sum_{a\in A'} \omega(a) = \sum_{a\in A\setminus A'} \omega(a)$ if and only if all books can be purchased for a total cost of $T - 2$. □

This NP-hardness result allows arbitrarily high prices (the reduction from Partition requires prices of the order of $2^{|B|}$). In a more realistic setting, we might assume a polynomial bound on prices, i.e., they can be encoded in unary. As we show below, the problem remains hard for a few shops in the sense of W[1]-hardness. We complement this result with an XP algorithm in Proposition 7.

Proposition 2 Clever Shopper *is W[1]-hard for* $m = |S|$ *in the strong sense (i.e., even when prices are encoded in unary).*

Proof (reduction from Bin Packing*)* Recall the well-known Bin Packing problem: given n items with weights $w_1, w_2, \dots, w_n$ and m bins with the same given capacity W, decide whether each item can be assigned to a bin in such a way that the total weight of the items in any bin does not exceed W. Bin Packing is NP-complete in the strong sense and W[1]-hard for parameter m, even when $\sum_{i=1}^{n} w_i = mW$ and all weights are encoded in unary [16].

We build an instance $\mathcal{I}$ of Clever Shopper from an instance of Bin Packing with the aforementioned restrictions as follows:

- create m identical shops, each with $t_s = W$ and $d_s = 1$;
- create n books, where book i is available in every shop at price w_i;
- set the budget to $m(W - 1)$.

In other words, any solution requires to obtain the discount from every shop, which is only possible if purchases amount to a total of exactly W per shop before discount. Therefore, the solutions to $\mathcal{I}$ correspond exactly to the solutions of the original instance of Bin Packing. □

We can obtain another hardness result under the assumption that all books are sold at a unit price. Here we cannot bound the total number of shops (we give an FPT algorithm for parameter m in Proposition 8 in this setting), but only the number of *chosen* shops (i.e., shops where at least one book is purchased).

Proposition 3 Clever Shopper *with unit prices is W[1]-hard for the parameter "number of chosen shops".*

Proof (reduction from Perfect Code*)* Given a graph $G = (V, E)$ and a positive integer k, Perfect Code asks whether G contains a *perfect code of size* k, i.e., a size-k subset $V' \subseteq V$ such that for each vertex $u \in V$ there is precisely one vertex in $N[v] \cap V'$ (where $N[v]$ is the *closed* neighbourhood of v, i.e., v and its adjacent vertices, as opposed to the *open* neighbourhood $N(v) = N[v] \setminus \{v\}$). This problem is known to be W[1]-hard for parameter k [9].

Let $\mathcal{I} = (G = (V, E), k)$ be an instance of Perfect Code. Write $V = \{u_1, u_2, \dots, u_n\}$. We obtain an instance $\mathcal{I}'$ of Clever Shopper as follows:

- build a bipartite graph $G' = (B \cup S, E')$ where $B = \{b_i : u_i \in V\}$, $S = \{s_i : u_i \in V\}$ and $E' = \{\{b_j, s_i\} : u_j \in N_G[u_i]\}$;
- set all prices to 1;
- for each shop $s_i \in S$, set $\mathscr{D}(s_i) = (d_G(u_i) + 1, 1)$ (*i.e.,* a unit discount will be applied from $d_G(u_i) + 1$ of purchase).

Figure 1 illustrates the construction.

We claim that there exists a size-k perfect code for G if and only if all books can be bought for a total cost of $n - k$.

$\boxed{\Rightarrow}$ Let $V' \subseteq V$ be a size-k perfect code in G. For every $u_i \in V$, let $u_{\mathrm{pc}(i)}$ be the unique vertex in $N[v] \cap V'$ (pc is well-defined since V' is a perfect code). Then buying each book $b_i \in B$ at shop $b_{\mathrm{pc}(i)}$ yields a solution for $\mathcal{I}'$, and it is simple to check that its cost is $n - k$.

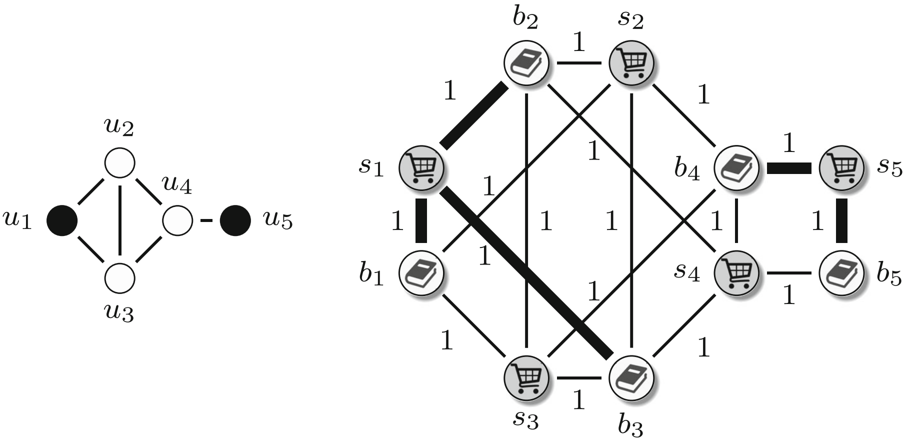

Fig. 1 Reducing Perfect Code to Clever Shopper. Left: The input graph with a size-2 perfect code (bold). Right: The corresponding bipartite graph and a solution with total cost $5 - 2 = 3$ (bold)

$\boxed{\Leftarrow}$ Suppose that all books can be bought for a total cost of $n - k$. Since n books must be bought at unit price and shops only offer a unit discount, k shops must be chosen in the solution. Let $S' \subseteq S$ denote these k shops. Since $\mathscr{D}(s_i) = (1, d_G(u_i) + 1)$ for each shop $s_i \in S$, we conclude that for each book $b_i \in B$ there is precisely one shop in $N[b_i] \cap S'$. Then $\{u_i : s_i \in S'\}$ is a size-k perfect code in G.

Note that the number of selected shops corresponds exactly to the total discount received (i.e. to parameter k in the reduction). □

We now prove the non-existence of polynomial kernels (under standard complexity assumptions) for Clever Shopper parameterised by the number of books. To this end, we use the or-composition technique [7]: given a problem $\mathscr{P}$ and a parameterised problem $\mathscr{Q}$, an or-composition is a reduction taking t instances $(I_1, I_2, \ldots, I_t)$ of $\mathscr{P}$, and building an instance (J, k) of $\mathscr{Q}$, with k bounded by a polynomial on $\max_{t' \leq t} |I_{t'}| + \log t$, such that (J, k) is a yes-instance if and only if there exists $t' \leq t$ such that $I_{t'}$ is a yes-instance. If $\mathscr{P}$ is NP-hard, then $\mathscr{Q}$ does not admit a polynomial kernel unless NP $\subseteq$ coNP/poly [7].

Proposition 4 Clever Shopper *admits no polynomial kernel unless NP $\subseteq$ coNP/poly.*

The following classical NP-complete problem [17] will be useful in that regard. It is known to remain NP-complete when each x_i appears in exactly 3 sets of $\mathscr{C}$ [14], an additional constraint which we will use in our proof.
Exact Cover By 3-Sets (X3C)

Input: a set $X = \{x_1, x_2, \ldots, x_{3m}\}$ of items, a collection $\mathscr{C}$ of 3-sets of X.
Question: is there a subset $\mathscr{C}'$ of $\mathscr{C}$ that covers each item of X exactly once?

Proof We build an OR-COMPOSITION using EXACT COVER BY 3-SETS. Consider t instances of X3C over the same number n of items. They are represented as bipartite graphs $(S_h \cup [n], E_h)$ for $h \in [t]$, where the 3-sets of $[n]$ are represented as degree-3 vertices $u \in S_h$. An example of the reduction is depicted in Fig. 2.

We first define some "shop identifiers". Write $J = \{0, 1\} \times [\lceil \log t \rceil]$. For each integer $h \in [t]$, let Key_h be the size-$\lceil \log t \rceil$ subset of J containing (b, j) if the jth digit in the binary representation of h is equal to b. Note that for $1 \leq h < h' \leq t$, we have $\mathrm{Key}_h \neq \mathrm{Key}_{h'}$. We now build a new instance $(B \cup S, E, w, \mathscr{D}, K)$ as follows:

- Create shops σ_j for all $j \in J$. Let $\Sigma = \{\sigma_j \mid j \in J\}$. The global set of shops is $S = \Sigma \cup \bigcup_{h \in [t]} S_h$. Note that $|S| = t + 2 \log t$.
- Create books x^i_j for all $i \in [n]$ and $j \in J$. The global set of books is $B = \{x^i_j \mid i \in [n], j \in J\} \cup [n]$. Let $n' = |B| = n(2 \log t + 1)$.
- For each edge $e = \{s, i\} \in E_h$, where $h \in [t]$, $s \in S_h$, $i \in [n]$, add edges $\{s, i\}$ and $\{s, x^i_j\}$ for all $j \in \mathrm{Key}_h$. Add also edges $\{\sigma_j, x^i_j\}$ for all $i \in [n]$ and $j \in J$. The overall set of edges is denoted E.
- Let all costs be equal to $T+1$ where T is any non-negative integer (say, $T = 42$).
- Let shop $s \in S$ have $t_s = k(T + 1)$ and $d_s = k$ where k is the degree of s. In other words, all shops give a discount of 1 per book only the buyer buys all books available from the shop.
- The budget is $K = Tn'$.

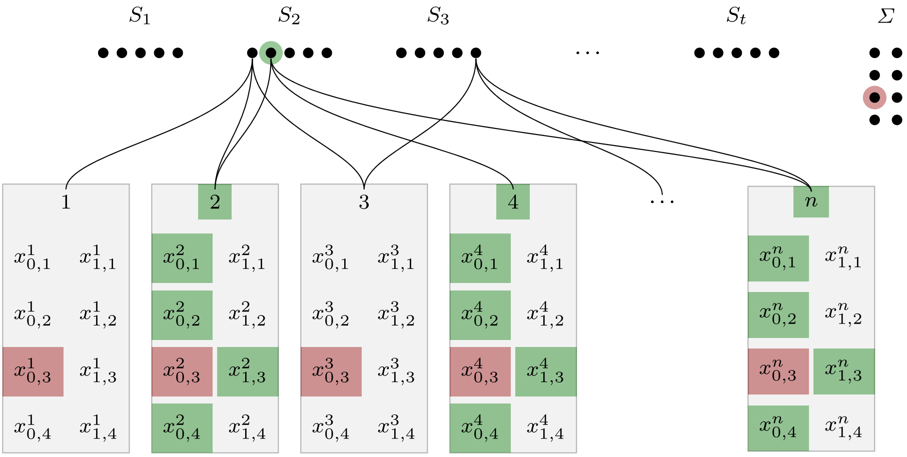

Fig. 2 Illustration of the reduction from EXACT COVER BY 3-SETS. The instances are drawn in the top part, as bipartite graphs between S_h and books $[n]$ (for better readability, most edges are ommited, but in fact all vertices in sets S_h have degree 3). The reduction introduces shops in Σ, and books x^i_j. Books sold in one of the shops from S_2 are highlighted in green. They correspond to the books i to which this element is connected in the corresponding instance of EXACT COVER, as well as books x^i_j where j visits the elements of Key_2 (the "identifier" of S_2). Since 2 is written 0010 in binary, the key contains positions (0, 1), (0, 2), (1, 3), (0, 4). Books sold by shop $\sigma_{0,3} \in \Sigma$ are highlighted in red. They are books $x^i_{0,3}$ for all i. Notice that those two shops have no books in common, since $(0, 3) \notin \mathrm{Key}_2$

Note that due to the pricing and discount functions, the average cost of a book in a shop is between T and $T+1$, and it reaches T if and only if all books in this shop have been purchased. Since the shopper needs to buy n' books with a budget of Tn', she must either buy all books from the shops she visits, or none at all. Thus she is faced with the problem of finding a set of shops whose available books correspond exactly to the set of books she needs.

The intuitive idea behind our construction is the following. We first build a set of shops behaving exactly like the union of all the sets in the instances of X3C. This is achieved directly by selling the original books in the corresponding shops: the fact that each book must be taken exactly once directly gives an exact cover. The rest of the shops (Σ) and books (x^i_j) ensure that shops are used from a single set S_h. More precisely, half of the books x^i_j are purchased together with book i, and they correspond to an identifier of the shops, while the other half must be purchased in shops from Σ, and enforce that the identifiers are the same for all books. That is, all books are purchased in shops from the same set S_h, which yields a solution to the hth instance of X3C.

We now formally prove that there is a solution to this instance of CLEVER SHOPPER if and only if *some* instance $(S_h \cup [n], E_h)$ of EXACT COVER BY 3-SETS for $h \in [t]$, is a yes-instance, which completes the OR-COMPOSITION.

$\boxed{\Leftarrow}$ Let $h \in [t]$ be such that $(S_h \cup [n], E_h)$ is a yes-instance. Let S'_h be the solution (that is, a subset of S_h such that all vertices in $[n]$ have exactly one neighbour in S'_h). Let $\Sigma' = \{\sigma_j \mid j \in J \setminus \text{Key}_h\}$. We show that buying all books in all shops of $S' = S'_h \cup \Sigma'$ gives a valid solution. Pick $i \in [n]$. Since S'_h is a solution to the X3C instance, then there exists a single $s \in S'_h$ such that $(s, i) \in E$. Books i and x^i_j for $j \in \text{Key}_h$ are thus sold by shop s, but not by any other shop in S' (in particular, not by any shop in Σ' since $j \notin \text{Key}_h$). Consider now a book x^i_j with $i \in [n]$ and $j \in J \setminus \text{Key}_h$. Then x^i_j is sold by shop $\sigma_j \in \Sigma' \subset S'$, and by no other shop in S'. Overall, each book is sold by exactly one shop in S', so the shopper buys all books from those shops, for a base price of $n'(T+1)$ and with a discount of n'.

$\boxed{\Rightarrow}$ As we have already remarked, in any solution, the set of shops $S' \subseteq S$ must contain each book exactly once. Consider first book 1: it is sold by a shop $s_1 \in S' \cap S_{h_1}$ for some $h_1 \in [t]$ (since no shop in Σ sells books in $[n]$). Consider now books x^1_j, with $j \in J$. If $j \in \text{Key}_{h_1}$, then x^1_j is sold by s_1, and thus $\sigma_j \notin S'$. If $j \in J \setminus \text{Key}_{h_1}$, then x^1_j is not sold by s_1, nor by any other shop in $S' \setminus \Sigma$ (such a shop would also sell book 1, which is already taken from shop s_1). Thus x^1_j must be sold by some shop from Σ, which can only be σ_j by construction. Hence $\{\sigma_j \mid J \setminus \text{Key}_h\} \subseteq S'$. Consider now any index $h_2 \neq h_1$. There exists some $j \in \text{Key}_{h_2} \setminus \text{Key}_{h_1}$. If there exists some shop $s \in S_{h_2} \cap S'$, then s sells book x^i_j for some $i \in [n]$. However, this book is already taken at shop $\sigma_j \in S'$ (since $j \in J \setminus \text{Key}_{h_1}$), hence there is no such shop s. Overall, $S' \setminus \Sigma \subseteq S_{h_1}$, that is, the shopper uses only shops from the same set S_{h_1} as well as some shops from Σ'. We can now prove that $S'_{h_1} = S' \cap S_{h_1} = S' \setminus \Sigma$ is a solution to the instance

$(S_{h_1} \cup [n], E_{h_1})$ of Exact Cover By 3-Sets. Indeed, consider any $i \in [n]$, then it is sold by a single shop in $s \in S'$, which cannot be in Σ, hence $s \in S'_{h_1}$. In other words, there is exactly one $s \in S'_{h_1}$ such that $(s, i) \in E_{h_1}$, so S'_{h_1} is a valid cover of $[n]$. □

4 Positive Results

We now give exact algorithms for Clever Shopper: a polynomial-time algorithm for the case where every shop sells at most two books, and four parameterised algorithms based respectively on the number of books, the number of shops, a bound on the prices, and the combination of the number of shops and the largest price of a book.

Algorithm 1 solves the case where each shop sells at most two books. As we shall see in Section 5, this bound is best possible. Its running time is dominated by the time required to find a maximum matching in a graph with $|B \cup S|$ vertices. We prove its correctness below.

Algorithm 1 CleverShopperMatching(G, $\mathscr{D}$).

```
Input: an edge-weighted bipartite graph $G = (B \cup S, E, w)$ with $deg(s) \leq 2$
       for each $s \in S$, and a set $\mathscr{D}$ of pairs $(d_s, t_s)$ for each $s \in S$.
Output: an optimal solution to the given Clever Shopper instance in the
        form of an assignment of books in $B$ to shops in $S$.
/* build new graph $G'$                                              */
$G' \leftarrow$ graph on vertex set $B \cup S$ with empty edge set $E'$ and edge weight
function $w'$;
$E' \leftarrow \{\{b, s, d_s + p(b) - w(b, s)\} \mid b \in B, s \in S, w(b, s) \geq t_s\}$;
foreach $s \in S$ with neighbours $b_1$, $b_2$ and $w(b_1, s) + w(b_2, s) \geq t_s$ do
    $z \leftarrow d_s + p(b_1) - w(b_1, s) + p(b_2) - w(b_2, s)$;
    if $b_1$ and $b_2$ are adjacent in $G'$ then
        $w'(b_1, b_2) \leftarrow \max(z, w'(b_1, b_2))$;
    else
        $E' \leftarrow E' \cup \{b_1, b_2, z\}$;
/* compute maximum weight matching on $G'$ and return
   converted solution                                                */
$M \leftarrow$ maximum weight matching on $G'$;
solution $\leftarrow \emptyset$;
foreach $\{u, v\} \in M$ do
    if $u \in B$ and $v \in B$ then
        solution $\leftarrow$ solution $\cup \{\{u, s\}, \{v, s\}\}$ // $s$ is the shop that
            sells both $u$ and $v$ that we kept in $G'$
    else                                    // either $u$ or $v$ is a shop
        solution $\leftarrow$ solution $\cup \{\{u, v\}\}$;
return solution;
```

Proposition 5 CLEVER SHOPPER *is in P if every shop sells at most two books.*

Proof Let $\mathcal{I}$ be an instance of CLEVER SHOPPER given by an edge-weighted bipartite graph $G = (B \cup S, E, w)$ and a pair (d_s, t_s) for each $s \in S$, where $d_s, t_s \in \mathbb{R}^+$. Vertices in S (resp. in B) have degree at most 2 (resp. at least 1). Note that vertices in S can be made to have degree exactly 2, by adding dummy edges with arbitrarily high costs, with no impact on the solution. We will therefore assume for simplicity that those degrees are all 2. For $b \in B$, let $p(b)$ be the cheapest available price for book b (discount excluded), i.e., $p(b) = \min\{w(\{b, s\}) \mid s \in S\}$.

Algorithm 1 constructs a new, non-necessarily bipartite graph G' from G, then computes a maximum weight matching on G' which it then converts into a solution for G. Figure 3 illustrates the construction. Note that G' may contain edges with negative weights: they may be safely ignored, but we keep them to avoid case distinctions in the rest of this proof. Since a maximum weight matching for G' can be found in polynomial time [11], it is now enough to prove the following claim: G' admits a matching of weight at least W if and only if instance $\mathcal{I}$ of CLEVER SHOPPER admits a solution of total cost at most $\sum_{b \in B} p(b) - W$.

⇐ Assume that instance $\mathcal{I}$ admits a solution $E^* \subseteq E$ of total cost $\sum_{b \in B} p(b) - X$, where $X \geq 0$ (the sum of the minimum prices of the books is an upper bound of the optimal solution). We build a matching M for G' as follows. Let $s \in S$ be

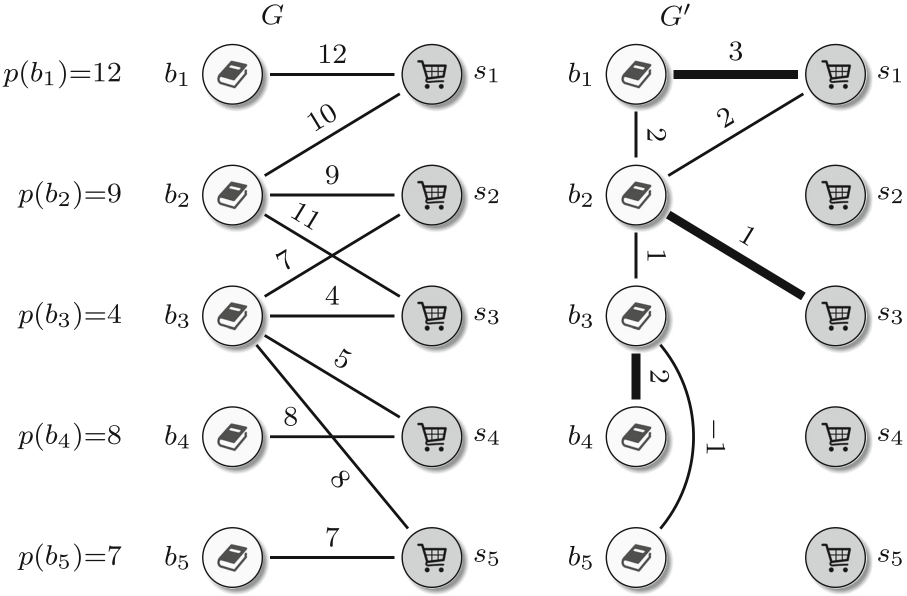

Fig. 3 Each shop offers a discount of 3 on a purchase of value ≥ 10. Bold edges indicate how to obtain optimal discounts: buy book b_1 from shop s_1, book b_2 from shop s_3, and books b_3 and b_4 from shop s_4. The remaining books are bought at their cheapest available price (so here we buy b_5 from s_5). Our clever customer used the discounts to buy all books for 6 less than if she had bought each book at its lowest price: 3 for b_1, 1 for b_2, 2 for b_3 and b_4 together

any *discount shop*, i.e., a shop whose discount is claimed, and let b_1 and b_2 be the books it sells. Then at least one of them has to be bought from s to get the discount. We distinguish between the following three cases, where the computation of the amount spent at each shop follows directly from how weights in G' were assigned:

- If $\{b_1, s\} \in E^*$ and $\{b_2, s\} \notin E^*$, add $\{b_1, s\}$ to M. The amount spent at this shop is $w(\{b_1, s\}) - d_s = p(b_1) - w'(\{b_1, s\})$.
- Similarly, if $\{b_2, s\} \in E^*$ and $\{b_1, s\} \notin E^*$, add $\{b_2, s\}$ to M. The amount spent at this shop is $w(\{b_2, s\}) - d_s = p(b_2) - w'(\{b_2, s\})$.
- Finally, if $\{b_1, s\} \in E^*$ and $\{b_2, s\} \in E^*$, then add $\{b_1, b_2\}$ to M. The amount spent at this shop is $w(\{b_1, s\}) + w(\{b_2, s\}) - d_s \geq p(b_1) + p(b_2) - w'(\{b_1, b_2\})$.

Note that edges added to M are indeed present in E', since in order to obtain the discount from s, the book prices must satisfy the same condition as for creating the corresponding edges. Note also that M is a matching, since each book can be bought from at most one shop. Let B^* be the set of books bought from discount shops. Summing over all these shops, the total price paid for the books in B^* is at least $\sum_{b \in B^*} p(b) - \sum_{e \in M} w'(e)$.

The books in $B \setminus B^*$ do not yield any discount, so the total price paid for them is at least $\sum_{b \in B \setminus B^*} p(b)$. Overall, the cost of the books is at least $\sum_{b \in B} p(b) - \sum_{e \in M} w'(e)$, therefore $\sum_{e \in M} w'(e) \geq X$.

$\boxed{\Rightarrow}$ Let M be a matching of G' of weight W; the conversion that takes place from line 10 to line 11 in Algorithm 1 proceeds as follows. For each edge $e \in M$, let s_e be the shop for which e was introduced. For an edge $e = \{b, s_e\} \in M$, buy book b from shop s_e. The price is sufficient to reach the threshold for the discount, so we pay $w(\{b, s_e\}) - d_e = p(b) - w'(e)$. For an edge $e = \{b_1, b_2\} \in M$, buy books b_1 and b_2 together from shop s_e. We again get the discount, and pay $w(\{b_1, s_e\}) + w(\{b_2, s_e\}) - d_e = p(b_1) + p(b_2) - w'(e)$. Note that for $e \neq f \in M$, $s_e \neq s_f$, so we never count the same discount twice. For every other book, buy them at the cheapest possible price $p(b)$, without expecting to get any discount. The total price paid is at most $\sum_{b \in B} p(b) - \sum_{e \in M} w'(e) = \sum_{b \in B} p(b) - W$. □

We now give a dynamic programming FPT algorithm with the number of books as parameter.

Proposition 6 CLEVER SHOPPER *admits an FPT algorithm for parameter n with running time $O(m3^n)$.*

Proof Given $j \in [m]$ and $B' \subseteq B$, let $p_j(B')$ be the price for buying all books in B' together from shop s_j (discount included), and $p_{\leq j}(B')$ be the lowest price that can be obtained when purchasing all books in B' from a subset of $\{s_1, s_2, \ldots, s_j\}$. Our goal is to compute $p_{\leq m}(B)$.

For $j = 1$, clearly $p_{\leq 1}(B') = p_1(B')$ for every B'. For any other j, consider an optimal way of buying the books in B' from shops $s_1, s_2, \ldots, s_j$. This way the customer buys some (possibly empty) subset B'' of books in s_j, and the rest, i.e., $B' \setminus B''$, at the lowest price from shops $s_1, s_2, \ldots, s_{j-1}$. Therefore:

$$p_{\leq j}(B') = \begin{cases} p_j(B') & \text{if } j = 1, \\ \min_{B'' \subseteq B'}\{p_j(B'') + p_{\leq j-1}(B' \setminus B'')\} & \text{otherwise.} \end{cases}$$

The values of $p_j(B')$ for all j and B' can be computed in $O(m2^n)$ time. Then the dynamic programming table requires to enumerate, for all j, all subsets B' and B'' such that $B'' \subseteq B' \subseteq B$. Any such pair B'', B' can be interpreted as a vector $v \in \{0, 1, 2\}^n$, where $i \in B'' \Leftrightarrow v_i = 2$ and $i \in B' \Leftrightarrow v_i \geq 1$. Therefore, filling the dynamic table takes $m3^n$ steps, each requiring constant time. □

As usual with dynamic programming, this algorithm yields the optimal price that can be obtained. One gets the actual solution (i.e., where to buy each book) with classic backtracing techniques.

The NP-hardness of CLEVER SHOPPER for two shops (using large prices, encoded in binary) and its W[1]-hardness when the parameter is the number of shops leave a very small opening for positive results: we can only consider small prices (encoded in unary) for a constant number of shops. The following result proves the tractability of this case.

Proposition 7 CLEVER SHOPPER *admits an XP algorithm running in time $O(nm\mathcal{W}^m)$, where $\mathcal{W}$ is the sum of all the prices of the instance, n is the number of books, and m is the number of shops.*

Proof We propose the following dynamic programming algorithm, which generalises the classical pseudo-polynomial algorithm for PARTITION. Let $i \in [n]$ and $p_s \in [\mathcal{W}]$ for $s \in S$. Define $T[i, p_{s_1}, \ldots, p_{s_m}]$ as 1 if it is possible to buy books 1 to i by spending exactly p_s (discount excluded) at shop s; and 0 otherwise. For $i = 0$, $T[0, p_{s_1}, \ldots, p_{s_m}] = 1$ if and only if $p_s = 0$ for all $s \in S$. The following formula allows to fill the table recursively for $i \geq 1$:

$$T[i, p_{s_1}, \ldots, p_{s_m}] = \max_{e \in E, i \in e} T[i-1, p'_{s_1}, \ldots, p'_{s_m}] \text{ where } p'_s = \begin{cases} p_s - w(e) & \text{if } s \in e, \\ p_s & \text{otherwise.} \end{cases}$$

Finally, we need to check whether the table contains a valid solution. To this end, we need to take the discounts into account. Clearly, an entry $T[n, p_{s_1}, \ldots, p_{s_m}] = 1$ leads to a solution if the following holds:

$$\sum_{i=1}^{m} p_{s_i} - \sum_{i=1, p_{s_i} \geq t_{s_i}}^{m} d_{s_i} \leq K.$$

The running time corresponds exactly to the time needed to fill the table: any of the $n\mathcal{W}^m$ cells requires at most m look-ups, which yields the claimed running time. □

The CLEVER SHOPPER problem also turns out to be fixed parameter tractable in the number of shops in the case where all books are sold at the same price. It is based on the following notion. Let $G = (B \cup S, E)$ be a bipartite graph, and $f : B \cup S \to \mathbb{N}$;

an *f-star subgraph* [13] is a subgraph G' of G such that the degree of each vertex $u \in B \cup S$ is at most $f(u)$ in G', and every connected component of G' is isomorphic to $K_{1,p}$ for some integer p.

Let $S' \subseteq S$. We write $f_{S'} : B \cup S \to \mathbb{N}$ for the following function:

$$\begin{aligned} f_{S'}(b) &= 1 \text{ for } b \in B, \\ f_{S'}(s) &= t_s \text{ for } s \in S', \\ f_{S'}(s) &= 0 \text{ for } s \notin S'. \end{aligned}$$

Algorithm 2 enumerates all subsets S' of S in time $2^{|S|}$, and for each subset, computes a maximum $f_{S'}$-star subgraph in time $O(|E| \log |B \cup S|)$ [13]. We prove its correctness in Proposition 8.

Algorithm 2 CLEVERSHOPPERUNITPRICES(G, $\mathscr{D}$).

```
Input: an edge-weighted bipartite graph $G = (B \cup S, E, w)$ with $w(e) = 1$ for
       each $e \in E$, and a set $\mathscr{D}$ of pairs $(d_s, t_s)$ for each $s \in S$.
Output: an optimal solution to the given CLEVER SHOPPER instance in the
        form of an assignment of books in $B$ to shops in $S$.
OPT ← ∅, best ← 0;
foreach $S' \subseteq S$ do
    $H \leftarrow$ maximum $f_{S'}$-star subgraph of $G$;
    if $|E(H)| > best$ then OPT ← $H$, best ← $|E(H)|$;
return OPT;
```

Proposition 8 CLEVER SHOPPER *admits an* FPT *algorithm for parameter* m *when all prices are equal.*

Proof We assume without loss of generality that all prices are equal to 1. We write $d_{S'} = \sum_{s \in S'} d_s$ and $t_{S'} = \sum_{s \in S'} t_s$. Let $\mathcal{I} = (B \cup S, E, w, \mathscr{D}, K)$ be an instance of CLEVER SHOPPER with $w(e) = 1$ for all $e \in E$. We show that $\mathcal{I}$ is a yes-instance if and only if there exists $S' \subseteq S$ with $|B| - d_{S'} \leq K$ such that $(B \cup S, E)$ admits an $f_{S'}$-star subgraph with $t_{S'}$ edges, thereby proving the correctness of Algorithm 2.

$\boxed{\Rightarrow}$ Let $E' \subseteq E$ be a solution and S' be the set of shops whose threshold t_s is reached. Since the total price is $|B| - d_{S'}$, we have $|B| - d_{S'} \leq K$. Since every weight equals 1, all vertices of S' have degree at most t_s in E'. Let $E'' \subseteq E'$ be a subset obtained by keeping exactly t_s edges incident to each $s \in S'$ and no edge incident to $s \notin S'$. Then E'' is an $f_{S'}$-star subgraph of size $t_{S'}$.

$\boxed{\Leftarrow}$ Let $G' = (B \cup S, E')$ be a spanning $f_{S'}$-star subgraph of G of size $t_{S'}$ with $S' \subseteq S$, and $|B| - d_{S'} \leq K$. The degree and size constraints force all vertices in S' to have degree exactly t_s in G'. We build a solution as follows: for each book $b \in B$, if E' contains an edge $\{b, s\}$ incident to b, then buy b from shop s, otherwise buy b from any other shop. Overall, at least t_s books are purchased from a shop $s \in S'$, so the total price is at most $|B| - d_{S'}$. □

Finally, another fixed parameter tractability result can be obtained based on the number of shops and the largest price at which a book is sold. The algorithm we design is based on n-fold integer programming, or n-fold IP for short. An instance of n-fold integer programming is given by a matrix $E = \binom{D}{A}$ with $E \in \mathbb{Z}^{(r+s)\times t}$, a positive integer n, and integral vectors $\mathbf{w}, \mathbf{b}, \mathbf{l}, \mathbf{u}$. The solution is an nt-dimensional vector $\mathbf{x}$ – a minimiser of the following integer linear program:

$$\min \left\{ \mathbf{wx} \mid E^{(n)}\mathbf{x} = \mathbf{b}\,,\ \mathbf{l} \le \mathbf{x} \le \mathbf{u}\,,\ \mathbf{x} \in \mathbb{Z}^{nt} \right\},$$

$$\text{where}\ \ E^{(n)} := \begin{pmatrix} D & D & \cdots & D \\ A & 0 & \cdots & 0 \\ 0 & A & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A \end{pmatrix}.$$

Note that vector $\mathbf{x}$ splits naturally into n so-called *bricks*. Hemmecke et al. [15] gave the first FPT algorithm for parameters r, s, t, and $\Delta = \|E\|_\infty$. Its running time has been improved in subsequent works [1, 12, 18, 19]; the following theorem gives the currently best known running time. We use $\langle\cdot\rangle$ to express the length of binary encoding.

Theorem 1 [12, 19] *There is an algorithm for solving n-fold IP in time* $\Delta^{O(r^2s+rs^2)} \cdot (nt)^2 \cdot \log(n)\langle\|\mathbf{w}\|_\infty, \mathbf{b}, \mathbf{l}, \mathbf{u}\rangle$.

Proposition 9 CLEVER SHOPPER *admits an FPT algorithm for combined parameter m and $w_{\max}$, where $w_{\max}$ is maximal price of a book.*

Proof We may assume a set of shops $S_D \subseteq S$ where the shopper gets the discount is fixed, since m is a parameter and we may thus try all possibilities for S_D.

We present an n-fold IP formulation for CLEVER SHOPPER. We associate a brick with each book. Let D be an $m \times (m(w_{\max} + 1))$ matrix consisting of m copies of a row vector $(0, 1, \ldots, w_{\max})$; here each row is associated with a shop in S. Before we reveal the use of D (which will be responsible for passing the thresholds in the selected shops S_D), let us first consider matrix A and the intuition behind variables $\mathbf{x}$, more precisely, its bricks $\mathbf{x}^b$ for $b \in B$. The entries of $\mathbf{x}$ are $x^b_{s,\omega}$, where $b \in B$, $s \in S$, and $\omega \in \Omega = \{0, 1, \ldots, w_{\max}\}$. The matrix A has only one row corresponding to the condition

$$\sum_{s\in S}\sum_{\omega\in\Omega} x^b_{s,\omega} = 1\,, \tag{1}$$

that is, we are going to buy one copy of each book in B as requested. We set the lower bounds on $\mathbf{x}$ to 0 while the upper bounds are set as follows

$$x^b_{s,\omega} \le \begin{cases} 1 & \text{if the shop } s \text{ sells the book } b \text{ at price } \omega \\ 0 & \text{otherwise.} \end{cases}$$

Note that by this $x^b_{s,\omega} = 0$ whenever it is not possible to buy a book b in a shop s for price ω.

Returning our attention back to D, using this matrix we introduce the following m conditions (each for a shop $s \in S$)

$$\sum_{b \in B} \sum_{\omega \in \Omega} \omega \cdot x^b_{s,\omega} \geq \tau_s \,, \tag{2}$$

where $\tau_s = t_s$ if $s \in S_D$ and $\tau_s = 0$ otherwise.

Finally, we are going to minimize the following objective

$$\sum_{b \in B} \sum_{s \in S} \sum_{\omega \in \Omega} w \cdot x^b_{s,\omega} - \sum_{s \in S_D} d_s \,.$$

This finishes the description of our model and thus it remains to conclude that the solution of the above n-fold IP is indeed a solution to CLEVER SHOPPER and determine its parameters in order to use Theorem 1. To see this, recall that by (1) we buy one copy of each book $b \in B$ and by (2) we buy books for at least t_s in all shops $s \in S_D$. To this end, the largest coefficient Δ is $w_{\max}$, the matrix D has m rows, and the matrix A has only one row; thus, by Theorem 1 one can solve the above IP in $(w_{\max})^{O(m^2)} n^2 \log(n)$ time. Consequently, our algorithm solves the given instance of CLEVER SHOPPER in $2^m (w_{\max})^{O(m^2)} n^2 \log(n) = (w_{\max})^{O(m^2)} n^2 \log(n)$ time, since we may assume $w_{\max} \geq 2$. □

5 Approximations

Since variants of CLEVER SHOPPER are, by and large, hard to solve exactly, it is natural to look for approximation algorithms. However, our hardness proofs can be modified to imply the NP-hardness of deciding whether the total price (including discounts) is 0 or more. For instance, in Proposition 1, we can set the discounts to $T/2$ instead of 1, so the PARTITION instance reduces to checking whether the optimal solution has cost 0. Therefore, we start with the following bad news:

Corollary 1 CLEVER SHOPPER *admits no approximation unless* P = NP.

Since this result seems resilient to most natural restrictions on the input structure (bounded prices, bounded degree, etc.), our proposed angle is to maximise the total discount rather than minimise the total cost. However, maximising the total discount is only relevant when the base price of the books is the same in all solutions (otherwise the optimal solution might not be the one with maximum discount), i.e., each book b has a fixed price w_b, and $w(\{b, s\}) = w_b$ for every $\{b, s\} \in E$. We call this variant MAX-DISCOUNT CLEVER SHOPPER. This "fixed price" constraint is not strong (all reductions from Section 3 satisfy it). In this setting, Proposition 1 shows that it is NP-hard to decide whether the optimal discount is 1 or 2. This yields the following corollary:

Corollary 2 MAX-DISCOUNT CLEVER SHOPPER *is* APX*-hard: it does not admit a* $(2-\epsilon)$*-approximation unless* P = NP.

Whether or not Max-Discount Clever Shopper admits a fixed-ratio approximation remains open. We show that it remains APX-hard even when the degrees of vertices in the input graph are bounded.

Proposition 10 Max-Discount Clever Shopper *is APX-hard even when each shop sells at most 3 books, and each book is available in at most 2 shops.*

Proof We reduce from Max 3-Sat (the problem of satisfying the maximum number of clauses in a 3-sat instance), known to be APX-hard when each literal occurs exactly twice [3]. Let $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be such a 3-CNF formula over a set $X = \{x_1, x_2, \ldots, x_n\}$ of boolean variables. For every $1 \leq i \leq m$ and $1 \leq j \leq 3$, let $\ell_{i,j}$ be the j-th literal of clause C_i. We obtain an instance $\mathcal{I}$ of Max-Discount Clever Shopper by first building a bipartite graph $G = (B \cup S, E)$ as follows (for ease of presentation, C_i, x_i and $\ell_{i,j}$ will be used both to denote respectively clauses, variables and literals in 3-CNF formula context, and the corresponding vertices in G):

$$\begin{aligned}
B &= \{\ell_{i,j} : 1 \leq i \leq m \text{ and } 1 \leq j \leq 3\} \ \cup \ \{x_i : 1 \leq i \leq n\} \\
S &= \{C_i : 1 \leq i \leq m\} \ \cup \ \{t_i, f_i : 1 \leq i \leq n\} \\
E &= E_1 \ \cup \ E_{2,p} \ \cup \ E_{2,n} \ \cup \ E_3
\end{aligned}$$

where

$$\begin{aligned}
E_1 &= \{\{\ell_{i,j}, C_i\} : 1 \leq i \leq m \text{ and } 1 \leq j \leq 3\} \\
E_{2,p} &= \{\{\ell_{i,j}, t_i\} : 1 \leq i \leq m \text{ and } \ell_{i,j} \text{ is the positive literal } x_i\} \\
E_{2,n} &= \{\{\ell_{i,j}, f_i\} : 1 \leq i \leq m \text{ and } \ell_{i,j} \text{ is the negative literal } \overline{x_i}\} \\
E_3 &= \{\{x_i, t_i\}, \{x_i, f_i\} : 1 \leq i \leq n\}.
\end{aligned}$$

Observe that each shop sells exactly 3 books and that each book is sold in exactly 2 shops. We now turn to defining the prices, the thresholds and the discounts. All shops sell books at a unit price. For the shops C_i, $1 \leq i \leq m$, a purchase of value 1 yields a discount of 1. For the shops t_i and f_i, $1 \leq i \leq n$, a purchase of value 3 yields a discount of 2. This discount policy implies that, for every $1 \leq i \leq n$, a customer cannot obtain a 2 discount both in shop t_i and in shop f_i (this follows from the fact that the book x_i is sold by both shops t_i and f_i).

First, it is easy to see that the largest discount that can be obtained is $2n + m$ (the upper bound is achieved by obtaining a discount in every shop C_i for $1 \leq i \leq m$, and in either the shop t_i or the shop f_i for $1 \leq i \leq n$). On the other side, for any truth assignment τ for φ satisfying k clauses, a $2n + k$ discount can be obtained as follows.

- For any variable x_i, $1 \leq i \leq n$, if $\tau(x_i) = \texttt{false}$, then buy 3 books from shop t_i, and if $\tau(x_i) = \texttt{true}$ then buy 3 books from shop f_i. Intuitively, if a variable is true, then all negative literals are "removed" by f_i, and all positive literals remain available for the corresponding clauses.
- For any clause $C_i = \ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3}$ satisfied by the truth assignment τ, buy book $\ell_{i,j}$ from shop C_i, where $\ell_{i,j}$ is a literal satisfying the clause C_i.

Then it follows that

$$\begin{aligned}\mathrm{opt}(I) = 2n + \mathrm{opt}(\varphi) &= 3m/2 + \mathrm{opt}(\varphi) && (\text{since } 4n = 3m)\\ &\leq 3\mathrm{opt}(\varphi) + \mathrm{opt}(\varphi) && (\text{since } 2\mathrm{opt}(\varphi) \geq m)\\ &\leq 4\mathrm{opt}(\varphi).\end{aligned}$$

Suppose now that we buy all books in B for a total discount of k'. First, we may clearly assume that $k' \geq 2n$ since a total $2n$ discount can always be achieved by buying 3 books either from shop t_i or from shop f_i, for every $1 \leq i \leq n$. Second, we may also assume that, for every $1 \leq i \leq n$, we buy either exactly 3 books from shop t_i or exactly 3 books from shop f_i. Indeed, if there exists an index $1 \leq i \leq n$ for which this is false, then buying either exactly 3 books from shop t_i or exactly 3 books from shop f_i instead results in a total k'' discount with $k'' \geq k'$ (this follows from the fact that we can get a 2 discount from t_i or f_i but only a 1 discount from any shop C_j, $1 \leq j \leq m$). We now obtain a truth assignment τ for φ as follows: for any variable x_i, $1 \leq i \leq n$, set $\tau(x_i) = \texttt{false}$ if we buy 3 books from shop t_i, and set $\tau(x_i) = \texttt{true}$ if we buy 3 books from shop f_i (the truth assignment τ is well-defined since, for $1 \leq i \leq n$, we cannot simultaneously buy 3 books from shop t_i and 3 books from shop f_i because of book x_i). Therefore, a clause C_i is satisfied by τ if and only if the corresponding shop C_i contains at least one book $l_{i,j}$ which is not bought from some other shop t_i or f_i. If we let k stand for the number of clauses satisfied by τ, then we obtain $k \geq k' - 2n$. It then follows that

$$\mathrm{opt}(\varphi) - k = \mathrm{opt}(I) - 2n - k \leq \mathrm{opt}(I) - 2n - k' + 2n = \mathrm{opt}(I) - k'.$$

Therefore, our reduction is an Ł-reduction (*i.e.*, $\mathrm{opt}(\mathcal{I}) \leq \alpha_1 \mathrm{opt}(\varphi)$ and $\mathrm{opt}(\varphi) - k \leq \alpha_2 \left(\mathrm{opt}(\mathcal{I}) - k'\right)$) with $\alpha_1 = 4$ and $\alpha_2 = 1$. □

On the other hand, we can obtain an approximability result based on the number of books sold by each shop.

Proposition 11 MAX-DISCOUNT CLEVER SHOPPER *where each shop sells at most k books admits a k-approximation.*

Proof Let B_s be the set of books sold by shop s. Our approximation algorithm proceeds as follows: start with a set of selected shops $S' = \emptyset$, a set of available books $B' = B$ and sort the shops by decreasing value of d_s. Then for each shop s, let $B'_s = B_s \cap B'$. If the books in B'_s are enough to get the discount ($\sum_{b \in B'_s} \geq t_s$), then assign all books of B'_s to shop s, add s to S' and set $B' = B' \setminus B'_s$. Finally, assign the remaining books to arbitrary shops that sell them.

We now prove the approximation ratio. For any $b \in B$, if $b \in B'_s$ for some $s \in S'$ then let $\delta(b) = d_s$, and $\delta(b) = 0$ otherwise. Thus, for any shop $s \in S'$, $d_s = \frac{1}{|B'_s|} \sum_{b \in B'_s} \delta(b) \geq \frac{1}{k} \sum_{b \in B'_s} \delta(b)$ due to the degree-k constraint. Note that for each shop of S', the amount spent at s is at least t_s, so the total discount obtained with this algorithm is $D \geq \sum_{s \in S'} d_s \geq \frac{1}{k} \sum_{b \in B} \delta(b)$.

We now compare the result of the algorithm with any optimal solution. For such a solution, let D^* be its total discount, S^* be the set of shops where purchases reach the threshold, and, for any $s \in S^*$, let B_s^* be the (non-empty) set of books purchased in shop s. Note that $D^* = \sum_{s \in S^*} d_s$.

Consider a shop $s \in S^*$. We show that there exists a book $b^*(s) \in B_s^*$ with $\delta(b^*(s)) \geq d_s$. If $s \in S^* \cap S'$, then we take $b^*(s)$ to be any book in B_s^*. Either $b^*(s) \in B'_s$, in which case $\delta(b^*(s)) = d_s$, or $b^*(s) \notin B'_s$, in which case $b^*(s)$ was assigned by the algorithm to a shop with a larger discount, i.e., $\delta(b^*(s)) \geq d_s$. If $s \in S^* \setminus S'$, since $s \notin S'$, at least one book in B_s^* is not available at the time the algorithm considers shop s; let $b^*(s)$ be such a book. Since it is not available, it has been selected as part of $B'_{s'}$ for some earlier shop s' (i.e., $d_s \leq d_{s'}$). Therefore, $b^*(s) \in B_s^* \cap B'_{s'}$ and $\delta(b^*(s)) = d_{s'} \geq d_s$. Since the sets B_s^* are pairwise disjoint for $s \in S^*$, we have $\sum_{s \in S^*} \delta(b^*(s)) \leq \sum_{b \in B} \delta(b)$. Putting it all together, we obtain:

$$D^* = \sum_{s \in S^*} d_s \leq \sum_{s \in S^*} \delta(b^*(s)) \leq \sum_{b \in B} \delta(b) \leq kD. \qquad \square$$

6 Conclusion

We introduced the Clever Shopper problem, a variant of Internet Shopping with free deliveries and shop-specific discounts based on shop-specific thresholds. We proved a number of hardness results, both in the classical complexity setting and from a parameterised complexity point of view. We also gave efficient algorithms for particular cases where restrictions apply to the number of books, the number of shops, or the nature of prices.

An interesting angle for future work is that of designing efficient exact algorithms for the general cases in which our FPT algorithms are not sufficient.

References

1. Altmanová, K., Knop, D., Koutecký, M.: Evaluating and tuning n-fold integer programming. In: D'Angelo, G. (ed.) 17th International Symposium on Experimental Algorithms, SEA 2018, June 27-29, 2018, L'Aquila, Italy, vol. 103 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, pp. 10:1–10:14 (2018)
2. Assmann, S., Johnson, D., Kleitman, D., Leung, J.-T.: On a dual version of the one-dimensional bin packing problem. J. Algorithms **5**, 502–525 (1984)
3. Berman, P., Karpinski, M., Scott, A.D.: Approximation hardness of short symmetric instances of MAX-3SAT electronic colloquium on computational complexity (ECCC) (2003)
4. Blazewicz, J., Kovalyov, M.Y., Musial, J., Urbanski, A.P., Wojciechowski, A.: Internet shopping optimization problem. Appl. Math. Comput. Sci. **20**, 385–390 (2010)
5. Blazewicz, J., Bouvry, P., Kovalyov, M.Y., Musial, J.: Internet shopping with price sensitive discounts. 4OR **12**, 35–48 (2014)
6. Blazewicz, J., Cheriere, N., Dutot, P.-F., Musial, J., Trystram, D.: Novel dual discounting functions for the internet shopping optimization problem: new algorithms. J. Sched. **19**, 245–255 (2016)
7. Bodlaender, H.L., Jansen, B.M.P., Kratsch, S.: Kernelization lower bounds by cross-composition. SIAM J. Discrete Math. **28**, 277–305 (2014)
8. Bulteau, L., Hermelin, D., Labarre, A., Vialette, S.: The clever shopper problem. In: Fomin, F.V., Podolskii, V.V. (eds.) Computer Science - Theory and Applications - 13th International Computer Science Symposium in Russia, CSR 2018, Moscow, Russia, Proceedings, vol. 10846 of Lecture Notes in Computer Science, pp. 53–64. Springer, Berlin (2018)

9. Cesati, M.: Perfect code is W[1]-complete. Inf. Process. Lett. **81**, 163–168 (2002)
10. Chatzigiannakis, I., Kaklamanis, C., Marx, D., Sannella, D. (eds.): 45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic, vol. 107 of LIPIcs Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2018)
11. Edmonds, J.: Paths, trees and flowers. Can. J. Math., 449–467 (1965)
12. Eisenbrand, F., Hunkenschröder, C., Klein, K.: Faster algorithms for integer programs with block structure. In: Chatzigiannakis et al. [10], pp. 49:1–49:13
13. Gabow, H.N.: A note on degree-constrained star subgraphs of bipartite graphs. Inf. Process. Lett. **5**, 165–167 (1976)
14. Gonzalez, T.F.: Clustering to minimize the maximum intercluster distance, Theor. Comput. Sci. **38**, 293–306 (1985)
15. Hemmecke, R., Onn, S., Romanchuk, L.: N-fold integer programming in cubic time. Math. Program. **137**, 325–341 (2013)
16. Jansen, K., Kratsch, S., Marx, D., Schlotter, I.: Bin packing with fixed number of bins revisited. J. Comput. Syst. Sci. **79**, 39–49 (2013)
17. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W. (eds.) Proceedings of a Symposium on the Complexity of Computer Computations, The IBM Research Symposia Series, pp. 85–103, Plenum Press (1972)
18. Knop, D., Koutecký, M., Mnich, M.: Combinatorial n-fold integer programming and applications. In: Pruhs, K., Sohler, C. (eds.) 25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria, vol. 87 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, pp. 54:1–54:14 (2017)
19. Koutecký, M., Levin, A., Onn, S.: A parameterized strongly polynomial algorithm for block structured integer programs, In: Chatzigiannakis et al. [10], pp. 85:1–85:14
20. van Bevern, R., Komusiewicz, C., Niedermeier, R., Sorge, M., Walsh, T.: H-index manipulation by merging articles: models, theory, and experiments. Artif. Intell. **240**, 19–35 (2016)