Management Science and Financial Engineering
Vol 18, No 2, November 2012, pp.19-22
ISSN 2287-2043 │ EISSN 2287-2361 │
http://dx.doi.org/10.7737/MSFE.2012.18.2.019

Inverse Bin-packing Number Problems: NP-Hardness and Approximation Algorithms

Yerim Chung*
School of Business, Yonsei University

(Received: July 9, 2012 / Revised: July 24, 2012 / Accepted: July 30, 2012)

ABSTRACT

In the bin-packing problem, we deal with how to pack the items by using a minimum number of bins. In the inverse bin-packing number problem, IBPN for short, we are given a list of items and a fixed number of bins. The objective is to perturb at the minimum cost the item-size vector so that all items can be packed into the prescribed number of bins. We show that IBPN is NP-hard and provide an approximation algorithm. We also consider a variant of IBPN where the prescribed solution value should be returned by a pre-selected specific approximation algorithm.

Keywords: Combinatorial Optimization, Inverse Optimization, Bin-Packing Problem, Hardness, Approximation

* Corresponding Author, E-mail: yerimchung@yonsei.ac.kr

1. INTRODUCTION

In the Bin-Packing problem, we are given a list of items and we are interested in the minimum number of bins needed to pack the items. However, in practice, it may happen that we only have a limited number of bins not enough to pack all the items. In such cases, some items should be discarded or its size should be reduced in order that all the given items can be packed using the bins in hand.

Given a prescribed solution or solution value for a combinatorial optimization problem, the problem of modifying the parameter system of the problem instance so that the optimal solution or optimal solution value in the modified instance equal the prescribed ones is called the *inverse problem* or *inverse number problem*, respectively. A variety of inverse combinatorial optimization problems have been studied during the last two decades. Main reference includes the following: (Ahuja and Orlin, 2000, 2001; Burkard *et al.*, 2008; Burton and Toint 1992; Chung and Demange, 2008, 2012; Heuberger, 2004). The inverse number problems have been studied for some NP-hard optimization problems such as the maximum independent set problem (Chung and Demange, 2008) and the minimum graph coloring problem (Chung *et al.*, 2008, 2010).

In this paper, we study the inverse bin-packing number problem. Given a set of n items, each having size $w_i \in \mathbb{Z}^+$ and K bins of capacity $B \in \mathbb{Z}^+$, the *inverse bin-packing number problem*, IBPN for short, is to determine a new item size vector $w' = (w'_1, \cdots, w'_n) \in \mathbb{Z}^{n+}$ so that all the items can be packed into K bins or less, and the perturbation $\|w - w'\|_p$ is minimum under a fixed norm L_p, for $p \in \{1, 2, \cdots, \infty\}$.

This note is organized as follows. We first introduce the computational complexity of IBPN in Section 2, followed by its approximation results in Section 3. Section 4 deals with a variant of IBPN where the prescribed solution value should be returned when applying a preselected approximation algorithm. Section 5 concludes the paper with some remarks on further research.

2. HARDNESS RESULTS OF IBPN

Let us consider the decision problem of IBPN, denoted by IBPN$_D$.

Definition 1: IBPN$_D$

Instance: A set of n items, an item-size vector $w=(w_1, \cdots, w_n)$ with positive integer entries, a positive integer bin number K, a positive integer bin capacity B, and a nonnegative value H.
Question: Is there a new item-size vector w' such that (*i*) $\|w-w'\|_p \le H$, and (*ii*) the instance defined with w' admits a feasible bin-packing, i.e., the items with modified size can be packed into K bins or less.

The bin-packing problem is polynomially reducible to IBPN*D* and this leads to the following result.

Proposition 1: *IBPN is NP-hard under the* L_p-norm, for any $p \in \{1, 2, \cdots, \infty\}$.

Proof: We first show that IBPN*D* belongs to the class **NP**. In fact, a nondeterministic algorithm needs to guess an item-size vector w' and a partition into K item subsets, and check in polynomial time the followings: whether the sum of item sizes in each subset does not exceed the bin capacity, and whether the deviation from the original item-size vector does not exceed the given threshold H. The checking step can be completed in polynomial time.

Given an instance of the bin-packing problem: n items with item-size vector w, a positive bin capacity B and a positive integer M, we construct an instance of IBPN*D* by adding a positive integer bin number K and a nonnegative value H. We set the bin number K equal to M and the threshold H to 0. Let us show that the bin-packing instance admits a positive answer if and only if the constructed IBPN*D*-instance admits an affirmative answer. First, if there is a feasible packing into M bins under the weight system w, then there is an item-size vector $w'=w$ which satisfies (*i*) $\|w-w'\|_p \le 0$, and (*ii*) the items with size w' can be packed into $K=M$ bins or less. Conversely, if it is possible to pack all the items into $K = M$ bins without modifying any item sizes, then there is a feasible packing for the original bin-packing problem with M bins or less. This result is valid regardless of the objective function measure. This implies that the optimization version IBPN of IBPN*D* is at least as hard to solve as the bin-packing problem, known to be NP-complete (Garey and Johnson, 1979). Hence, IBPN is NP-hard under the L_p-norm, for any $p \in \{1, 2, \cdots, \infty\}$.

3. APPROXIMATION RESULTS OF IBPN

Consider the *maximum bin load problem*, MBLP for short, defined as follows: given a set of n items and a set of m bins of capacity B, MBLP is concerned with assigning as many items as possible to m identical bins, even if the overall size of items assigned to a bin exceeds the bin capacity. For each bin, *bin load* is measured by the minimum value of the bin capacity and the overall size of the items packed in a bin. The objective is to maximize the total bin loads over m bins.

Recall that IBPN under the L_1-norm is concerned with minimizing the total amounts of excess-the parts of items that cannot fit into each bin-when assigning all the items to m bins. Hence, we assume $K = m$ for IBPN. Clearly, IBPN under the L_1-norm has a min-max relation with the problem MBLP. This min-max relation implies that if MBLP admits a PTAS, then there exists an input-dependent, asymptotic approximation algorithm for IBPN under the L_1-norm.

To show that MBLP admits a PTAS, we consider the *multiple subset sum problem*, MSSP for short. Given a set N of n items and a set M of m bins, MSSP is concerned with finding an item subset of maximum weight sum that can be legally packed into the given m bins. It is known that MSSP admits a PTAS (Caprara *et al.*, 2008). MBLP is similar to MSSP, and a PTAS for MBLP can be obtained by using the similar but simpler arguments given in (Caprara *et al.*, 2008). Here, we provide a brief sketch of the scheme that we propose for solving MBLP.

1: Partition the set of items into two subsets L and S according to item-sizes.
2: Partition the large item set L into a constant number of subsets so that the number of large items is bounded by a constant. Then, there is a constant number of feasible bin-packing for L and the instance associated with L can be solved in constant time by complete enumeration.
3: If the number of large items is not bounded by a constant, then use the item grouping technique, introduced in (Caprara *et al.*, 2008).
  3-1: Sort the items of L in the non-decreasing order of weight, and partition the items of L into $p+1$ subsets so that p is bounded by a constant.
  3-2: Define a new instance with $p+1$ subsets of items that has a constant number $p+1$ of distinct item weights. In such an instance, the number of feasible bin types is bounded by a constant.
  3-3: Formulate a mixed integer linear program with a constant number of variables, and solve it using the algorithm given by (Lenstra *et al.*, 1977).
  3-4: Convert the solution obtained into a solution for the instance before grouping.

The above algorithm solves MBLP instances restricted to large items, and returns a solution of value λ_L, which satisfies the following: $\lambda_L \ge (1-2\varepsilon)\beta_L + \varepsilon W$ for any fixed $\varepsilon > 0$, where β_L denotes the optimal solution value of the MBLP instance restricted to L. In addition, this algorithm can be extended into a polynomial time approximation scheme for solving general instances of MBLP. For the conciseness of the paper, we omit the proof. The existence of a PTAS for MBLP leads to the following result.

Proposition 2: *IBPN under the* L_1*-norm admits an input-dependent, asymptotic approximation algorithm.*

Let us present a negative approximation result for IBPN.

Proposition 3: *There is no approximation algorithm for IBPN satisfying* $\lambda_{IBPN} \le \beta_{IBPN} + (\mu - 2)$ *for a constant* $\mu \ge 2.$

Proof: Assume that there is an approximation algorithm A that satisfies $\lambda_{IBPN} \le \beta_{IBPN} + (\mu - 2)$. Consider an instance of the partition problem; let $X = \{x_1, \cdots, x_n\}$ be n numbers such that $\sum_{i=1}^{n} x_i = 2B$ with a positive integer B. From this, we construct an instance of IBPN as follows. For a positive integer $\mu \ge 2,$ let $N = \{1, \cdots, n\}$ be the set of n items, each having size $w_i = \mu x_i$ for $i = 1, \cdots, n-1,$ and $w_n = \mu x_n + 1,$ and $\sum_{i=1}^{n} w_i = 2\mu B + 1.$ Two bins of capacity $\mu B + 1$ are given. If there exists a partition, then $\beta_{IBPN} = 0$. Hence, due to the hypothesis, the algorithm A solves this instance and returns a solution of value λ_{IBPN} such that $\lambda_{IBPN} \le \beta_{IBPN} + (\mu - 2) \le \mu - 2.$

Consider now the case where a partition does not exist. In this case, we show that $\lambda_{IBPN} > \mu - 2$ for $\mu \ge 2$. In fact, since there is no partition, we can imagine two subsets, say A and A^c, such that $\sum_{x\in A} x > B$ and $\sum_{x\in A^c} x < B.$ The subset A^c gives a feasible packing for the IBPN-instance. However, it is not the case for the set A. Assume first that $x_n \in A^c$. Then, as $\sum_{x\in A} x > B,$ we have: $\sum_{x\in A} \mu x \ge \mu(B+1) > (\mu B + 1) + (\mu - 2)$ for any $\mu \ge 2.$ Assume now that $x_n \in A.$ Then, the bin load for the items associated with A is bounded as follows: $1 + \sum_{x\in A} \mu x > \sum_{x\in A} \mu x \ge \mu(B+1)$ for any $\mu \ge 2.$ So, $\sum_{x\in A} \mu x > (\mu B + 1) + (\mu - 2)$. In both cases, we need to decrease the item size by μ-1 or more. So, any feasible solution for the IBPN-instance has a value strictly greater than μ-2. Hence, $\lambda_{IBPN} > \mu$-2.

So, if there is an approximation algorithm A satisfying the following relation: $\lambda_{IBPN} \le \beta_{IBPN} + (\mu - 2),$ it can be converted into an algorithm which efficiently solves the partition problem. This is not possible because the partition problem is NP-hard.

4. IBPN AGAINST A SPECIFIC ALGORITHM

In the usual framework of inverse optimization, the objective is to find a new instance with the minimum modification so that the pre-fixed solution (or solution value) becomes optimal. However, when the original problem is NP-hard, it is often not easy to check the optimality of the fixed solution in the modified instance. In such cases, it is relevant to fix a specific algorithm and modify the instance so that the fixed solution can be returned by the algorithm. This variant can have a great interest in practice. In fact, when playing games, if we know the algorithm used by our adversary, then we can force him to choose the solution that we propose. Inverse problems against a specific algorithm have been studied for the maximum independent set problem (Chung and Demange, 2008) and the minimum traveling salesman problem (Chung and Demange, 2012) with respect to greedy algorithms and local search heuristics.

In this section, we deal with a variant of IBPN under the L_1 norm where we are given a specified approximation algorithm for the bin-packing problem as well as a target bin number *K*. The task is to compute the minimum perturbation to the item size vector so that all the items can be packed, when applying the fixed algorithm, into *K* bins of capacity *B*. Here, *K* is not required to be the minimum bin number but to be the bin number needed to pack all the items by using the given algorithm. This variant is called IBPN *against a specific approximation algorithm*.

Let us study IBPN with respect to the Next Fit algorithm. Next Fit is a simple sequential algorithm for the bin-packing problem, which guarantees an approximation ratio of 2 (Ausiello *et al*., 1999). It processes the items one by one in the same order as they are given in the input. The first item enters the first bin. Let B_j be the bin to which Next Fit assigned the *i*-th item; the bins $B_1, \cdots, B_j$ are (partially or completely) filled with the items $1, \cdots, i.$ Next Fit assigns the item $i+1$ to the bin B_j if B_j has enough room for $i+1,$ otherwise Next Fit closes B_j and opens the bin B_{j+1} to put the item $i+1$ in it.

The inverse bin-packing number problem against Next Fit, IBPNNF for short, is concerned with finding a new item size vector w' so that Next Fit returns a feasible solution with *K* bins or less in the new instance defined with w', and the the deviation $\|w - w'\|_1$ from the original vector *w* is minimum under the L_1-norm. To have a non-trivial instance, we suppose that *K* is not greater than the number *n* of items and that Next Fit returns a solution of value strictly greater than *K*, when applied to the original bin-packing instance.

The weakness of Next Fit is that when assigning an item to a bin, if the last used bin does not have enough space, Next Fit definitely closes the bin and never tries to reuse it for the other items. In the inverse version, however, for each item that cannot enter the last bin, there are always two choices: open a new bin and put the item in it, or put the item in the last used bin (ignoring the infeasibility) and get the penalty for that. Then, it is easy to see that IBPNNF is equivalent to the problem of finding a shortest path in a binary tree. However, the complexity is huge; in fact the Bellman algorithm solves this problem in $O(|V|^2)$ times, with $|V| = 2^K.$ We conclude this section by proposing a dynamic programming that solves IBPNNF in polynomial time.

For any positive integers *i* and *k* such that $1 \le i \le n$ and $1 \le k \le K,$ consider the subproblem of IBPNNF, defined by the item subset $N_i = \{1, \cdots, i\}$ and the target bin number *k*. Let us denote this subproblem by $IBPN_{NF}(N_i, k).$ For any $(i, k) \in \{1, \cdots, n\} \times \{1, \cdots, K\},$ let $f_k(N_i)$ be the objective function value of $IBPN_{NF}(N_i, k).$ Then, we set $f_1(N_i) = \max\{\sum_{j=i}^{n} w_j - B, 0\}$ for any $i \in \{1, \cdots, n\}$, and $f_k(N_n) = f_k(N_{n+1}) = 0$ for any $k \in \{1, \cdots, K\}$. For any fixed $i \in \{1, \cdots, n\}$, let i_0 be the smallest item such that $i < i_0 \le n$ and $\sum_{j=i}^{i_0} w_j > B$. For any fixed $i \in \{1, \cdots, n\}$, if such an item does not exist, then $f_k(N_i) = 0$ for any $k \in \{1, \cdots, K\}$. So, we assume the existence of i_0 for each $i \in \{1, \cdots, n\}$. Then, the recursive relation between $f_{k+1}(N_i)$ and $f_k(N_i)$ can be written as follows:

$$f_{k+1}(N_i) = \min\left\{ f_k(N_{i_0}); \sum_{j=i}^{i_0} w_i - B + f_k(N_{i_0+1}) \right\}$$

for any $i \in \{1, \cdots, n\}$. The optimal solution value of IBP NNF is given by $f_K(N)$, and since K is assumed to be smaller than the total item number n, $f_K(N)$ can be computed in $O(n^2)$ time.

Proposition 4: *IBPN*NF *can be solved in* $O(n^2)$ *time.*

5. CONCLUSIONS

In this paper, we considered the inverse bin-packing number problems and its interesting variant where the prescribed solution value should be returned by a pre-selected approximation algorithm. In comparison with the inverse problem aiming for the optimality of the fixed solution or the fixed solution value, this approach seems to be a slackened version. However, it provides an alternative approach to approximate the related inverse problem. In addition, this approach is meaningful especially when the original problem under consideration is intrinsically difficult to solve. As a further work, it will be interesting to detect special cases of IBPN that are efficiently solvable and have interesting applications in the business contexts.

ACKNOWLEDGEMENT

This work was supported by the National Research foundation of Korea Grant, funded by the Korean government [NRF-2011-330-B00076].

REFERENCES

Ahuja, R. K. and J. B. Orlin, "A Faster Algorithm for the Inverse Spanning Tree Problem," *Journal of Algorithms* 34, 1 (2000), 177-193.

Ahuja, R. K. and J. B. Orlin, Inverse optimization, *Operations Research* 49, 5 (2001), 771-783.

Ausiello, G., P. Crescenzi, P. Gambosi, V. Kann, A. Mar-chetti-Spaccame-La, and M. Protasi, "Complexity and Approximation," Springer, Berlin, 1999.

Burkard, R. E., E. Gassner, and J. Hatzl, "Reverse 2-Median Problem on Trees," *Discrete Applied Mathematics* 156, 11 (2008), 1963-1976.

Burton, D. and Ph. L. Toint, "On an Instance of the Inverse Shortest Paths Problem," *Mathematical Programming* 53 (1992), 45-61.

Caprara, A., H. Kellerer, and U. Pferschy, "The Multiple Subset Sum Problem," *SIAM Journal of Optimization* 11 (1998), 308-319.

Chung, Y., J. F. Culus, and M. Demange, "Inverse Booking Problems," *Proceedings of the 2nd Workshop on Algorithms and Computation*, Lecture Notes in Computer Science, 4921 (2008), 180-187.

Chung, Y., J. F. Culus, and M. Demange, "On Inverse Chromatic Number Problems," *Proceedings of the International Symposium on Combinatorial Optimization*, Electronic Notes in Discrete Mathematics, 36 (2010), 1129-1136.

Chung, Y. and M. Demange, "The 0-1 Inverse Maximum Stable Set Problem," *Discrete Applied Mathematics* 156, 13 (2008), 2501-2516.

Chung, Y. and M. Demange, "On Inverse Traveling Salesman Problems," *4OR-A Quarterly Journal of Operations Research* 10 (2012), 193-209.

Garey, M. R. and D. S. Johnson, "Computers and Intractability-A Guide to the Theory of NP-Completeness," Freeman, San Francisco, 1979.

Heuberger, C., "Inverse Combinatorial Optimization: A Survey on Problems, Methods, and Results," *Journal of Combinatorial Optimization* 8, 3 (2004), 329-361.

Lenstra, J. K., A. H. G. Rinnooy Kan, and P. Brucker, "Complexity of machine scheduling Problems," *Annals of Discrete Mathematics* 1 (1977), 343-362.