Bin packing with colocations

Jean-Claude Bermond, Nathann Cohen, David Coudert, Dimitrios Letsios, Ioannis Milis, Stéphane Pérennes, Vassilis Zissimopoulos

Bin packing with colocations*

Jean-Claude Bermond[1], Nathann Cohen[2], David Coudert[1], Dimitrios Letsios[1], Ioannis Milis[3], Stéphane Pérennes[1], and Vassilis Zissimopoulos[4]

[1]Université Côte d'Azur, Inria, CNRS, I3S, France
[2]CNRS and Univ. Paris Sud, France
[3]Department of Informatics, Athens University of Economics and Business, Greece
[4]Department of Informatics & Telecommunications, National and Kapodistrian University of Athens

Abstract

Motivated by an assignment problem arising in MapReduce computations, we investigate a generalization of the Bin Packing problem which we call *Bin Packing with Colocations Problem*. Given a set V of items with positive integer weights, an underlying graph $G = (V, E)$, and an integer q, the goal is to pack the items into a minimum number of bins so that (i) the total weight of the items packed in every bin is at most q, and (ii) for each edge $(i, j) \in E$ there is at least one bin containing both items i and j.

We first show that when the underlying graph is unweighted (i.e., all the items have equal weights), the problem is equivalent to the q-clique problem, and when furthermore the underlying graph is a clique, optimal solutions are obtained from covering designs. We prove that the problem becomes NP-hard even for weighted paths and unweighted trees and we propose approximation algorithms for particular families of graphs, including: a $(3+\sqrt{5})$-approximate algorithm for weighted complete graphs (improving a previous 8-approximation), a 2-approximate algorithm for weighted paths, a 5-approximate algorithm for weighted trees, and an $(1+\epsilon)$-approximate algorithm for unweighted trees. For general weighted graphs, we propose a $3 + 2\lceil mad(G)/2 \rceil$-approximate algorithm, where $mad(G)$ is the maximum average degree of G. Finally, we show how to convert any ρ-approximation algorithm for the Bin Packing (resp. the Densest q-Subgraph problem) into an approximation algorithm for the problem on weighted (resp. unweighted) general graphs.

*This work is partially supported by ANR project Stint under reference ANR-13-BS02-0007, ANR program "Investments for the Future" under reference ANR-11-LABX-0031-01, the Research Center of Athens University of Economics and Business (RC-AUEB), and the Special Account for Research Grants of National and Kapodistrian University of Athens.

1 Introduction

We study the following generalization of the classical *Bin Packing* problem, which we call *Bin Packing with Colocations Problem* (BPCP). We are given a set of items $V = \{1, 2, \ldots, n\}$ with positive integer weights $w_1, w_2, \ldots, w_n$, an underlying graph $G = (V, E)$, and bins of integer capacity q. The goal is to pack the items into a minimum number of bins so that (i) the total weight of the items packed in every bin is at most q, and (ii) for each edge $(i, j) \in E$ there is at least one bin containing both items i and j. Due to the last constraint of colocating pairwise related items, we consider item weights such that $w_i + w_j \leq q$, for each edge $(i, j) \in E$, for otherwise our problem has no feasible solution. Note also that in a feasible solution (copies of) a vertex might be packed into more than one bin.

Our initial motivation for studying BPCP was the work of Afrati et al. [1, 2] on an assignment problem in MapReduce computations. In such a computation, the outputs of the mappers, of the form $\langle key - value \rangle$, are assigned to the reducers and each reducer applies a reduce function to a single *key* and its associated list of *value*'s to produce its output. However, a reducer (in fact, the machine executing it) is subject to capacity constraints (e.g. memory size), which limits the total size of data assigned to it. Moreover, for each required output, there must be a reducer receiving all inputs necessary to compute its output. For a family of problems arising in this context, an output depends on pairwise related inputs, i.e., a situation captured by the colocation constraint in BPCP.

More generally, the BPCP models any practical situation where context-related entities of given sizes are assigned to physical resources of limited capacity and there are pairwise colocation constraints. For instance, in the case where computer files are placed into memory blocks of fixed size, it is natural to ask for the colocation of pairwise related files (for example, sharing a common attribute) in the same memory block. Moreover, in large data centers, file colocation is essential for data chunks which are highly likely to be accessed together.

BPCP is clearly a generalization of the Bin Packing problem, where the input graph is an independent set, i.e., $E = \emptyset$. As an example of this relation consider of BPCP on a star graph of $n + 1$ vertices (items), where the central vertex has weight w_0, the leaves have weights $w_1, w_2, \ldots, w_n$ and the bin capacity is $q + w_0$. Obviously, BPCP is equivalent to the Bin Packing problem with input the n leaves items and bin capacity q. In contrast to the Bin Packing problem, BPCP remains interesting even in the case where all the items have the same weight and we refer to this case as *Unweighted BPCP* (U-BPCP). It is easy to see that U-BPCP is trivial on a star graph or on a path but we will see that it becomes $\mathcal{NP}$-hard even for trees.

More interestingly, U-BPCP for complete graphs falls in the well known area of combinatorial design theory (the interested reader is referred to [6]

for a survey of this area). In this context, given a set S of n elements, an (n, q, λ)-design is a collection of subsets, which are called *blocks*, such that each block has q elements and every pair of distinct elements appears together in *exactly* λ blocks. A lot of work has been done on necessary and sufficient conditions for the existence of such an (n, q, λ)-design and the main observation here is that, if an $(n, q, 1)$-design exists, then it is an optimal solution to U-BPCP for complete input graphs. The notion of (n, q, λ)-designs has been extended to (n, q, λ)-covering designs (see [9, 11]) where every pair of distinct elements has to appear together in *at least* λ blocks. Again, an $(n, q, 1)$-covering is nothing else than a solution to U-BPCP for complete input graphs.

Furthermore, BPCP generalizes the so called *q-Clique Covering Problem* studied by Goldschmidt et al. [7]. In their context, a q-clique of a graph $G = (V, E)$ is a subset $C \subseteq V$ of q vertices and it is not compulsory that every edge of the q-clique exists in G. The objective is to find the minimum number of such cliques of size at most q so that every edge and every vertex of G is included in at least one clique. Clearly, this corresponds to U-BPCP.

Related Work. Afrati et al. [1, 2] studied BPCP for complete and complete bipartite graphs. For both cases, they proved that BPCP is $\mathcal{NP}$-hard, via a reduction from the Partition problem, and they proposed greedy approximation algorithms of ratio 8. For the U-BPCP, they also proposed a $(2 + \epsilon)$-approximation algorithm in the case of complete graphs.

Goldschmidt et al. [7] have proposed approximation algorithms for the q-Clique Covering Problem which corresponds to U-BPCP on general graphs. In fact, for the special cases where $q = 3$ and $q = 4$ (q is the bin capacity), they obtained approximation ratios $7/5$ and $7/3$, respectively. When the bin capacity is arbitrary, they showed that the problem admits an $O(q)$-approximation algorithm.

As described above, U-BPCP on complete input graphs is equivalent to finding an $(n, q, 1)$-covering design with the minimum number of blocks (bins). Therefore, the results obtained in combinatorial design theory apply to U-BPCP on complete graphs too and we elaborate on them in Section 2.

Finally, as BPCP is a generalization of the Bin Packing problem, we refer the reader to [5] for a recent review of the latter problem. Bin Packing is known to be APX-hard as it cannot be approximated within a factor smaller than $3/2$ (in terms of the absolute approximation ratio) and simple greedy algorithms as Next-Fit, First-Fit and First-Fit Decreasing achieve approximation ratios of 2, 1.7 and 1.5, respectively. Moreover, it admits asymptotic polynomial-time approximation schemes (APTAS).

Contributions. Following the work of Afrati et al. [1, 2], in Section 2, we begin with the study of U-BPCP and BPCP on complete graphs. We start

with U-BPCP where we can use the results obtained on covering designs. We first present an algorithm similar to the one presented in [1, 2] for the case q even, but our analysis is tighter. Our algorithm achieves an approximation ratio less than 2 when q is even and $n \geq q^2/2$. This algorithm can be generalized and, by using $(n, 3, 1)$-coverings (resp. $(n, 4, 1)$-coverings) we get an approximation ratio less than $3/2$ (resp. $5/4$) when q is multiple of 3 (resp. multiple of 4) and $n \geq q^2$. For BPCP an 8-approximation algorithm was given in [1, 2]; we propose a new improved 6-approximation and a refined one of approximation ratio $(3 + \sqrt{5})$.

Thereafter, we move our attention to other interesting types of graphs. In Section 3, we show that BPCP is strongly $\mathcal{NP}$-hard even on paths and we propose a 2-approximation algorithm for this case. Then, in Section 4, we show that U-BPCP is $\mathcal{NP}$-hard on trees and we propose an algorithm which asymptotically achieves an approximation ratio of $(1+\epsilon)$, where $\epsilon = O(1/q)$. Moreover, we propose a greedy 5-approximation algorithm for BPCP on trees. In Section 5, we study U-BPCP and BPCP on general graphs. Extending our ideas for BPCP on trees to BPCP on general graphs we derive an algorithm of approximation ratio $3+2\lceil mad(G)/2 \rceil$, where $mad(G)$ is the maximum average degree of the graph. This algorithm is efficient for sparse graphs and, for example, it achieves a 9-approximation ratio for BPCP on planar graphs. Then, based on a simple greedy approach, and given any ρ-approximation algorithm for the Bin Packing problem, we obtain a $\rho \cdot \Delta$-approximation algorithm for BPCP on general graphs, where Δ is the maximum degree of the graph. Finally, we show that any ρ-approximation for Densest q-Subgraph problem can be converted to a $\rho \cdot \log n$-approximation algorithm for the U-BPCP on general graphs.

2 Complete graphs

In the following we observe that U-BPCP on complete graphs is closely related to the theory of combinatorial designs (see [6]). For this reason, we briefly survey some fundamental results known in this area.

Given a set S of n elements, an $(n, q, 1)$-*design* is a collection of subsets of S, called *blocks*, such that every pair of distinct elements appears together in exactly one block. In such a design, every element appears in $(n-1)/(q-1)$ blocks and the number of blocks must be equal to $n(n-1)/q(q-1)$. Since these numbers must be integers, two necessary conditions for the existence of an $(n, q, 1)$-design are $(n-1) \equiv 0 \mod q-1$ and $n(n-1) \equiv 0 \mod q(q-1)$. These last conditions have been proved to be sufficient for certain values of n and q (see [6]), for instance when $q = 3$ (known as Steiner triple systems) and $q = 4, 5$. Moreover, when q is a power of a prime, a $(q^2+q+1, q+1, 1)$-design exists (finite projective planes of order q) and a $(q^2, q, 1)$-design exists (finite affine planes of order q). Furthermore, Wilson [12] has proved that

sufficiency is ensured also in the general case where n is large enough. On the other hand, the conditions are not always sufficient for the existence of $(n, q, 1)$-design; for example, Euler has shown that neither a $(36, 6, 1)$-design nor a $(43, 7, 1)$-design exist [6].

Clearly, when an $(n, q, 1)$-design exists, then it is an optimal solution for U-BPCP on a complete graph of n vertices and bin capacity q. Note that this relation was not observed by Afrati et al. [1, 2] who rediscovered basic results of design theory such as the existence of an $(n, 3, 1)$-design and the existence of projective planes.

The notion of an $(n, q, 1)$-design has been also extended to packing and covering designs (see the survey [9] or chapter IV.8 in Handbook of Designs [11]). Given a set S of n elements, an $(n, q, 1)$-covering (design) is a collection of subsets of S, called *blocks*, such that each block has exactly q elements and every pair of distinct elements appear together in at least one block. Clearly, an $(n, q, 1)$-covering is nothing else than a feasible solution to U-BPCP on a complete graph of n vertices where the blocks correspond to bins of capacity q. In the literature, there exists a significant amount of work on computing the covering number $C_1(n, q)$ or simply $C(n, q)$, i.e. the minimum number of blocks in an $(n, q, 1)$-covering. Therefore, for U-BPCP on complete graphs, the number of bins of an optimal solution is equal to $b^* = C(n, q)$.

In what follows, let $L(n, q) = \left\lceil \frac{n}{q} \left\lceil \frac{n-1}{q-1} \right\rceil \right\rceil$; this quantity will serve for lower bounding the number of used bins in an optimal solution.

Lemma 1 *It holds that* $C(n, q) \geq L(n, q)$. *Furthermore, if* $(n-1) \equiv 0 \mod (q-1)$ *and* $n(n-1) \equiv 1 \mod q$, *then* $C(n, q) \geq L(n, q) + 1$.

The exact values of $C(n, q)$ have been determined only in some cases (see [9, 11]). For example, the exact value of $C(n, q)$ is known for $n <= 3q$ and for $q = 2, 3, 4$ where we have:

- $C(n, 2) = L(n, 2) = \frac{n(n-1)}{2}$ (trivial as a block contains one pair),
- $C(n, 3) = L(n, 3) = \left\lceil \frac{n}{3} \left\lceil \frac{n-1}{2} \right\rceil \right\rceil$, and
- $C(n, 4) = L(n, 4) + \epsilon$, where $\epsilon = 1$ when $n = 7, 9, 10$, $\epsilon = 2$ in the case where $n = 19$, and $\epsilon = 0$, otherwise.

Finally, the following theorem, which has been proved by Rödl [10] via probabilistic methods, bounds $C(n, q)$ asymptotically. Interestingly, it answered a conjecture of Erdös and Hanani (see Chapter 4 of [3] for a proof).

Theorem 1 (Rödl [10]) *For any fixed* q, *it holds that* $C(n, q) \leq (1 + o(1))L(n, q)$, *where the term* $o(1)$ *approaches zero as* n *tends to infinity.*

Unfortunately, this theoretical result does not give answers for practical values of n and q and, for such cases, we propose some simple greedy algorithms.

2.1 Unweighted case

The main idea for designing an approximation algorithm consists in partitioning the items into $g = \lceil n/\lfloor q/k \rfloor \rceil$ groups of equal size $\lfloor q/k \rfloor$ (except possibly one), where k is a chosen positive integer for which a $(g, k, 1)$-covering exists. All the items of such a group are then considered as one element and we cover the pairs of groups with blocks of size k. For each block, we use a bin consisting of all items of the groups in the block. As a block contains k groups, a bin will contain at most $k\lfloor q/k \rfloor \leq q$ items. Furthermore, each pair of items belongs to some bin. Indeed, consider a pair $\{i, j\}$; i belongs to some group A and j to some group B. Then the pair $\{i, j\}$ belongs to the bin associated to the block containing the pair of groups A and B if A and B are distinct, or to every bin containing A if $A = B$.

The analysis of this general algorithm might be difficult as we have various floors and ceils and also it assumes the existence of a $(g, k, 1)$-covering. Moreover, the approximation ratio obtained will depend of the size of the groups; indeed the pairs of items belonging to the same group will be repeated many times. So we have interest to choose a large k, but very few $(g, k, 1)$-coverings are known for large k.

For $k = 2$, a case for which a trivial $(g, 2, 1)$-covering exists, we get Algorithm 1. This is similar with the one of [1, 2] for even values of q and simpler than their algorithm for odd values of q. However, here we present a tighter analysis resulting in slightly better approximation ratios.

Algorithm 1 (U-BPCP, complete graphs)

1: Partition the items into g groups each of size $\lfloor q/2 \rfloor$, but at most one group.
2: Pack every pair of groups into a bin.

Theorem 2 *Algorithm 1 achieves approximation ratios of* $2\frac{q-1}{q} + \frac{q-2}{n}$*, if* q *is even, and* $2\frac{q}{q-1} + \frac{q-1}{n}$*, if* q *is odd, for the U-BPCP on complete graphs.*

Note that, by Theorem 2, we have an approximation ratio less than 2, when q is even and $n \geq q^2/2$. When q is odd, the algorithm has no interest for $q = 3$ and $n \leq 3q$ as we know in that case the exact value of $C(n, q)$. So, we will use the algorithm only for $q \geq 5$ and $n > 3q$, in which case the approximation ratio is less than 17/6. Note also that when q is large and n tends to infinity the ratio is near to 2.

We can also analyze the general algorithm described above for $k = 3$ (resp. $k = 4$) and q is a multiple of 3 (resp. 4), to get an approximation ratio at most 3/2 (resp. 5/4). More generally, for any k, if n is a multiple of q and there exists a $(g, k, 1)$-covering, for $g = \lceil \frac{kn}{q} \rceil$, we get a $\frac{k}{k-1}$-approximation ratio.

2.2 Weighted case

In this section, we extend the previous ideas to the BPCP on complete graphs by using an appropriate grouping of jobs. Initially, we present a 6-approximation algorithm via a simple grouping which we then improve via a more enhanced grouping. Our analysis uses the lower bound on the optimal number of bins, b^*, provided by the next lemma.

Lemma 2 *For the BPCP on a complete graph it holds that* $b^* \geq \frac{1}{q}\sum_{i=1}^{n} w_i \lceil \frac{s-w_i}{q-w_i} \rceil > \frac{s^2}{q^2}$, *where* $s = \sum_{i=1}^{n} w_i$.

In [1, 2], the authors showed that Algorithm 1 can also be used for weighted graph and gives an approximation ratio of 8. In Algorithm 2, we use a better grouping which achieves a feasible solution and improves the approximation ratio to 6. Note that, in Algorithm 2, we suppose w.l.o.g. that all the weights are at most $q/2$. Indeed, there can be at most one item of weight greater than $q/2$ as the input graph is complete. In such a case, the large item can be packed independently with all the other items and the remaining pairs of items can be packed with Algorithm 2.

Algorithm 2 (BPCP, complete graphs)

1: Partition the items into three types of groups A, B, C;
the size of a group $s_{A_i}, s_{B_i}, s_{C_i}$ is the sum of the weights of the items in the group:
- α groups of type A; $A_i \in A$ has size $\frac{q}{3} < s_{A_i} \leq \frac{q}{2}$
- β groups of type B; $B_i \in B$ has size $\frac{q}{4} < s_{B_i} \leq \frac{q}{3}$
- γ groups of type C; $C_i \in C$ has size $s_{C_i} \leq \frac{q}{4}$

2: Form bins containing either a pair of groups or when possible three groups.

Theorem 3 *Algorithm 2 achieves an approximation ratio of* 6 *for the BPCP on complete graphs.*

We can refine the above idea, by partitioning the items into four types of groups, to get an even better approximation ratio.

Theorem 4 *There is a* $(3+\sqrt{5})$*-approximation algorithm for the BPCP on complete graphs.*

3 Paths

In this section we consider the BPCP on paths; recall that U-BPCP is trivial on paths. We first show that the BPCP on paths is strongly $\mathcal{NP}$-hard via a reduction from the Bin Packing problem.

Theorem 5 *The BPCP on paths is strongly $\mathcal{NP}$-hard.*

We also present a 2-approximation algorithm by a reduction to the shortest path problem on an appropriate directed graph and the use of the Next-Fit algorithm for the Bin Packing problem. Starting from a path $G = (V, E)$, with $V = \{1, 2, \ldots, n\}$ and $E = \{(i, i+1) | 1 \leq i \leq n-1\}$, we construct an auxiliary weighted directed graph $\overrightarrow{G}$ which contains a node for each vertex $i \in V$. Then, for each pair (i, j) such that $1 \leq i < j \leq n$, we denote by $W(i, j) = \sum_{k=i}^{j} w_k$ the total weight of the vertices $i, i+1, \ldots, j$ and, if $W(i, j) \leq q$, then $\overrightarrow{G}$ contains an arc (i, j) of weight $W(i, j)$. Clearly, any $(1, n)$-path (i.e. a path from node 1 to node n) P of $\overrightarrow{G}$ corresponds to a feasible solution of our problem; for each arc $(i, j) \in P$ we use a bin to pack vertices $i, i+1, \ldots, j$. For a path P of $\overrightarrow{G}$, we denote by $W(P) = \sum_{(i,j)\in P} W(i, j)$ its total weight. The following lemma provides a lower bound on the optimal number of bins, b^*, for the BPCP on paths which we use in our analysis.

Lemma 3 *For the BPCP on paths it holds that $b^* \geq \frac{1}{q} \cdot W(P^*)$, where P^* is a minimum weight $(1, n)$-path in the auxiliary graph $\overrightarrow{G}$.*

Our Algorithm 3 considers each arc in a minimum weight $(1, n)$-path in $\overrightarrow{G}$ as an item for the Bin Packing problem and packs them using the Next-Fit algorithm.

Algorithm 3 (BPCP, paths)

1: Find a minimum weight $(1, n)$-path P^* in $\overrightarrow{G}$.
2: For each arc $(i, j) \in P^*$, create an item of weight $W(i, j)$.
3: Pack the new items using the Next-Fit algotithm.

Using Lemma 3 and the fact that the Next-Fit algorithm packs a set of items of total weight W into at most $2\lceil W/q \rceil$ bins of capacity q we get the next theorem.

Theorem 6 *Algorithm 3 achieves an approximation ratio of 2 for the BPCP on paths.*

4 Trees

In this section we deal with both U-BPCP and BPCP on trees. We show that U-BPCP is $\mathcal{NP}$-hard and that it admits an $(1+\epsilon)$-approximation algorithm. We also propose a greedy 5-approximation algorithm for BPCP on trees.

4.1 Unweighted case

We show first that U-BPCP on trees is $\mathcal{NP}$-hard via a reduction from the 3-Partition problem which is known to be $\mathcal{NP}$-hard even for polynomially bounded parameters.

Theorem 7 *The U-BPCP on trees is $\mathcal{NP}$-hard.*

For our approximation algorithm, let G be the input tree of our problem and suppose that the edges of G are oriented away from some arbitrary node which is picked as the root and we obtain a directed tree T. A key ingredient for the description of our algorithm is the notion of an *eligible subtree*. Given a directed tree $T = (V(T), E(T))$, an eligible subtree T' is a subtree of T rooted at some vertex $i \in V(T)$ such that, the forest $\overline{T} = ((V(T) \setminus V(T')) \cup \{i\}, E(T) \setminus E(T'))$ consists of a single tree. That is, the removal of all the edges and all the vertices of T', but i, leaves $\overline{T}$ connected. We define the *size* of a tree T as the number of vertices that it contains and we denote it by $s(T)$. The following decomposition lemma is critical for designing our algorithm.

Lemma 4 *There exists an eligible subtree T' of a tree T of size $k/2 \leq s(T') \leq k$, for each $k \in [1, s(T)]$.*

We assume, for convenience, that the bin capacity q is a power of 2, i.e. $q = 2^a$ for some integer $a > 0$, but our analysis can be extended to arbitrary values of q. We also denote by b the number of bins that our algorithm uses. The algorithm starts with the initial tree G and, gradually, it packs vertices into bins and removes vertices whose incident edges have been covered until a feasible solution is produced. More specifically, it consists of b phases and, in each phase, a steps are performed. In the k-th phase, $1 \leq k \leq b$, the algorithm computes the content of the k-th bin, say B_k. During the algorithm's execution, we denote by f_k the free space of bin B_k and by T the current remaining tree (whose edges have not been packed before). In the beginning of the i-th step of the k-th phase, it must be the case that $f_k \leq q/2^{i-1}$. Then, if it also holds that $f_k \geq q/2^i$, based on Lemma 4, the algorithm computes an eligible subtree T' of T (the remaining part of the initial tree) with size $s(T') \in [q/2^{i+1}, q/2^i]$ and it packs T' in B_k. Moreover, the vertices of T', apart from the root, as well as the edges are removed from T. If there is sufficient space, then a second eligible subtree of the same bounded dimension is also computed, is packed in B_k and is removed from T. In this way, at the end of the i-th step, it holds that $f_k \leq q/2^i$ (Lemma 5). The algorithm proceeds until T becomes the empty graph.

Lemma 5 *At the end of the i-th step in the k-th phase, it holds that $f_k \leq q/2^i$, for $1 \leq i \leq a$ and $1 \leq k \leq b$.*

Algorithm 4 (U-BPCP, trees, $q = 2^a$)

T: directed tree obtained by orienting the edges of G
$k = 1$, $f_k = q$
while $E(T) \neq \emptyset$ **do**
  for $i = 1, 2, \ldots, a$ **do**
    Repeat twice:
    if $f_k \geq q/2^i$ **then**
      Compute an eligible subtree T' such that $s(T') \in [q/2^{i+1}, q/2^i]$.
      Pack $V(T')$ in bin B_k and remove T' from T.
  $k = k + 1$
Return the solution found.

Theorem 8 *Algorithm 4 achieves asymptotically an approximation ratio of* $(1 + \epsilon)$*, where* ϵ *is* $O(1/q)$*, for U-BPCP on trees*

4.2 Weighted case

In what follows, we present a greedy 5-approximation algorithm for BPCP on trees. We consider a tree $G = (V, E)$ and we assume again that the edges are oriented away from some node $r \in V$ which is chosen arbitrarily as the root and we obtain a directed tree T. The algorithm produces a feasible solution by considering T. Initially, every node $i \in V$ is packed independently together with all its children so as to ensure feasibility of the obtained solution. More specifically, for each $i \in V$, all vertices in its out-neighborhood $\Gamma^+(i)$ are packed into bins of capacity $q - w_i$ according to the First-Fit Decreasing algorithm. Then, the content of every such bin together with vertex i is considered as one item for the Bin Packing problem and they are packed using the Next-Fit algorithm.

Algorithm 5 (BPCP, trees)

for each $i \in V$ **do**
  Pack the vertices in $\Gamma^+(i)$ into bins of capacity $q - w_i$ using the First-Fit Decreasing algorithm.
  For each bin containing a subset S of items, create an item of size $\sum_{i \in S} w_i$.
Pack the created items using the Next-Fit algorithm.

It is known that number of bins used by First-Fit Decreasing algorithm for the Bin Packing problem is at most 3/2 times the number of the optimal number of bins. Using this fact, we can bound the number of copies of each vertex when it is packed with its children and we get the next theorem.

Theorem 9 *Algorithm 5 achieves an approximation ratio of* 5 *for BPCP on trees.*

5 General Graphs

In this section we deal with the BPCP and U-BPCP on a general graph $G = (V, E)$. We first deal with BPCP and we present two approximation algorithms. The first one extents our approach for the BPCP on trees to general graphs and gives an approximation ratio which is efficient for the BPCP on sparse graphs. The second one considers each edge $(i, j) \in E$ as an item of the Bin Packing problem of weight $w_i + w_j$ and gives an approximation ratio of $O(\Delta)$, where Δ is the maximum degree of the graph. Then, we move to the U-BPCP and we present an approximation algorithm based on its relation with Densest q-Subgraph problem.

5.1 Weighted case

We first, extend our approach for the BPCP on trees to BPCP on a general graph $G = (V, E)$. More specifically, we construct an orientation D of the graph G and for each vertex $i \in V$ we consider its in- and out-neighborhood in D. Recall that in BPCP on trees each node is packed with its children and in one more bin with its parent. In the BPCP on general graphs, each node is packed with the vertices in its out-neighborhood and with each one of the vertices in its in-neighborhood in different bins. Using similar arguments as in the proof of Theorem 9 we obtain an approximation ratio of $3 + 2\Delta^-(D)$ where $\Delta^-(D)$ is the maximum in-degree of D.

The maximum average degree $mad(G)$ of the input graph G is the maximum of the average degrees $ad(H) = 2|E(H)|/|V(H)|$ taken over all subgraphs H of G, i.e., $mad(G) = \max_{H \subseteq G} \left\{ \frac{2|E(H)|}{|V(H)|} \right\}$. By applying the approach of Hakimi [8], we can construct, in polynomial time, an orientation D of a general undirected graph G, with maximum in-degree $\Delta^-(D) \leq \lceil mad(G)/2 \rceil$. Using this result we get the next theorem.

Theorem 10 *There is a $3 + 2\lceil mad(G)/2 \rceil$-approximation algorithm for the BPCP on general graphs.*

In the case of planar graphs, it holds that $mad(G) < 6$ and we obtain a 9-approximation. More generally, any class of H-minor-free graphs have bounded maximum average degree.

Next, we present an approximation algorithm for BPCP on a general graph $G = (V, E)$, which uses a ρ-approximation algorithm $\mathcal{A}$ for the Bin Packing problem. We denote by Δ the maximum degree of G.

Initially, we obtain a lower bound by packing the edges of the input graph $G = (V, E)$ instead of its vertices. Specifically, for each edge $(i, j) \in E$, we create an item $I_{i,j}$ of weight $w_i + w_j$. Let I be the set of all such items and consider the instance (I, q) of the Bin Packing problem. Clearly, any

feasible packing of (I, q) is a feasible solution for BPCP in general graphs. So, we get the following lemma.

Lemma 6 *Let b^* and b_e^* be the optimal numbers bins for BPCP and the Bin Packing problem (I, q), respectively. Then, it holds that $b_e^* \leq \Delta \cdot b^*$.*

Algorithm 6 (BPCP, general graphs)

1: For each edge $(i, j) \in E$, create an item of weight $w_i + w_j$.
2: Pack the items with $\mathcal{A}$ into bins of capacity q.

Then, Lemma 6 implies the next theorem.

Theorem 11 *Algorithm 6 achieves an approximation ratio of $\rho \cdot \Delta$ for BPCP on general graphs, given a ρ-approximation algorithm for the Bin Packing problem.*

Recall that the Bin Packing problem admits several greedy constant-factor approximation algorithms as well as an APTAS (Asymptotic Polynomial-Time Approximation Scheme).

5.2 Unweighted case

In what follows, we present an approximation algorithm for U-BPCP on general graphs by using a ρ-approximation algorithm $\mathcal{A}$ for the Densest q-Subgraph problem (i.e. finding a set of q vertices with the maximum number of edges in the subgraph induced by them). More specifically, the algorithm packs repeatedly densest q-subgraphs of G and removes the covered edges. The procedure goes on until all edges are covered, as in Algorithm 7 below.

Algorithm 7 (U-BPCP, general graphs)

1: **while** $E \neq \emptyset$ **do**
2: Run $\mathcal{A}$ and let $D = (V', E')$, $|V'| = q$ the resulting densest q-subgraph.
3: Pack the vertices of V' into a new bin.
4: $G = (V, E \setminus E')$.

Theorem 12 *Algorithm 7 is $\rho \cdot \log n$-approximate for U-BPCP on general graphs, given a ρ-approximation algorithm for Densest q-Subgraph problem.*

The best known approximation algorithm for the Densest q-Subgraph problem was proposed by [4] and its approximation ratio is $O(n^{1/4})$. Therefore, Theorem 12 implies a $O(n^{1/4} \cdot \ln n)$-approximation algorithm for U-BPCP.

References

[1] F. N. Afrati, S. Dolev, E. Korach, S. Sharma, and J. D. Ullman. Assignment of different-sized inputs in mapreduce. In *Proceedings of the EDBT/ICDT Conference, Brussels, Belgium*, pages 28–37, Brussels, 2015.

[2] F. N. Afrati, S. Dolev, E. Korach, S. Sharma, and J. D. Ullman. Assignment problems of different-sized inputs in mapreduce. *CoRR*, abs/1507.04461, 2015.

[3] N. Alon and J. H. Spencer. *The probabilistic method*. Wiley, fourth edition, 2016.

[4] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan. Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph. In *ACM Symposium on Theory of Computing (STOC)*, pages 201–210, 2010.

[5] E. G. Coffman Jr, J. Csirik, G. Galambos, S. Martello, and D. Vigo. Bin packing approximation algorithms: survey and classification. *Handbook of Combinatorial Optimization*, pages 455–531, 2013.

[6] C. Colbourn and J. Dinitz, editors. *Handbook of Combinatorial Designs (2nd edition)*, volume 42 of *Discrete Mathematics and its Applications*. Chapman and Hall/CRC, second edition, 2006.

[7] O. Goldschmidt, D. S. Hochbaum, C. A. J. Hurkens, and G. Yu. Approximation algorithms for the k-clique covering problem. *SIAM J. Discrete Math.*, 9(3):492–509, 1996.

[8] S. L. Hakimi. On the degrees of the vertices of a directed graph. *J. Franklin Inst.*, 279:290–308, 1965.

[9] W. Mills and R. Mullin. Coverings and packings. In C. Colbourn and J. Dinitz, editors, *Contemporary design theory: A collection of surveys*, pages 371–399. John wiley and Sons, new York, 1992.

[10] V. Rödl. On a packing and covering problem. *European J. of Combinatorics*, 6:69–78, 1985.

[11] D. R. Stinson. Coverings. In C. Colbourn and J. Dinitz, editors, *The CRC Handbook of Combinatorial Designs*, volume 42 of *Discrete Mathematics and Its Applications*, chapter IV.8, pages 260–265. CRC Press, 2nd edition, 2006.

[12] R. M. Wilson. Decomposition of complete graphs into subgraphs isomorphic to a given graph. *Congressus numerantium*, 15:647–659, Nov. 1976.