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 Université 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

March 17, 2018

Abstract

Motivated by an assignment problem arising in MapReduce computations, we study a generalization of the Bin Packing problem which we call *Bin Packing with Colocations Problem*. We are given a weighted graph $G = (V, E)$, where V represents a set of items with positive integer weights and E the set of related (to be colocated) items, and an integer bin capacity q. The goal is to pack the items (with repetitions) into a minimum number of bins so that (i) the total weight of the items packed in each 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 point out that, when the graph is unweighted (i.e., all the items have equal weights), the problem is equivalent to the q-clique problem, and when furthermore the graph is complete, optimal solutions, for specific values of n and q, are obtained from Covering Designs. We prove that the problem is strongly $\mathcal{NP}$-hard even for paths and unweighted trees. Then, we propose approximation algorithms for particular families of graphs, including: a 5-approximation algorithm for complete graphs (improving a previous ratio of 8), a 2-approximation algorithm for paths, and an $(1 + O(\log q/q))$-approximation algorithm for unweighted trees. For general graphs, we propose a $3+2\lceil mad(G)/2\rceil$-approximation algorithm, where $mad(G)$ is the maximum average degree of G. Finally, we show how to convert any approximation algorithm for Bin Packing (resp. Densest q-Subgraph) problem into an approximation algorithm for the problem on weighted (resp. unweighted) general graphs.

1 Introduction

In this paper, we study the following generalization of the classical *Bin Packing* problem, which we call *Bin Packing with Colocations Problem* (BPCP). We are given a weighted graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ represents the set of items with positive integer weights $w_1, w_2, \ldots, w_n$ and E the set of related (to be colocated) items, and an integer capacity q for bins. The goal is to pack the items into a minimum number of bins so that (i) the total weight of the items packed in each bin is at most q, and (ii) for each edge $(i, j) \in E$ there is at least one bin containing both items

*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.

i and j. Due to the last constraint of colocating pairwise related items, we assume that, for each edge $(i, j) \in E$, $w_i + w_j \leq q$, for otherwise our problem has no feasible solution. Note also that in a feasible solution (copies of) a vertex (item) might be packed into more than one bin.

Our initial motivation for studying BPCP was the work of [2, 1] on an assignment problem in MapReduce computations. In such computations, 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 must be assigned to physical resources of limited capacity while fulfilling pairwise colocation constraints. For instance, when 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, which is a particular case when $E = \emptyset$. As an example of this relation, consider BPCP on a star graph with $n+1$ vertices, where the central vertex has weight w_0 and the bin capacity is $q + w_0$. Obviously, BPCP is equivalent to the Bin Packing problem with input the n leaves (with their weights) and bin capacity q. In contrast to the Bin Packing problem, BPCP remains interesting even when 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 prove that it becomes $\mathcal{NP}$-hard even for trees.

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 V of n elements, a 2-$(n, q, 1)$-covering design (see [10, 13]) abbreviated here as (n, q)-covering is a collection of subsets, called *blocks*, such that each block has q elements and every pair of distinct elements of V has to appear together in *at least* one block. An (n, q)-covering is nothing else than a solution to U-BPCP for complete graphs. In the case of perfect coverings, where each pair appears in exactly one block, the (n, q)-covering is called a BIBD$(n, q, 1)$, i.e. a Balanced Incomplete Block Design, or a 2-$(n, q, 1)$-design and a lot of work has been done on necessary and sufficient conditions for the existence of such designs (see [6]). The main observation here is that, if a 2-$(n, q, 1)$-design exists, then it is an optimal solution to U-BPCP for complete graphs.

Furthermore, BPCP generalizes the so called q-Clique Covering Problem studied in [7]. In their context, a q-clique of a graph G is an induced subgraph with at most q vertices. The objective is to find the minimum number of such q-cliques such that every edge and every vertex of G is included in at least one q-clique. This corresponds exactly to U-BPCP. Our results generalize the well-known results on q-Clique covering.

Related Work. In [2, 1] the authors 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 with ratio 8. For the U-BPCP, they also proposed a $(2 + \epsilon)$-approximation algorithm in the case of complete graphs.

In [7] the authors 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 graphs is equivalent to finding an (n, q)-covering 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.

Contributions. Following the work of [2, 1], we begin with the study of U-BPCP and BPCP on complete graphs. Table 1 lists the approximation ratios for BPCP and U-BPCP obtained in the current paper. In Section 2 we study U-BPCP where we exploit existing results on covering designs. We first present an algorithm similar to the one presented in [2, 1], 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 $(n, 3)$-coverings (resp. $(n, 4)$-coverings) we get an approximation ratio less than $3/2$ (resp. $5/4$) when q is multiple of 3 (resp. multiple of 4) and $n \geq q^2$. In Section 3, we deal with BPCP on complete graphs for which an 8-approximation algorithm was given in [2, 1]; we propose a new approximation algorithm of ratio 6 and a refined one of ratio 5.

Thereafter, we move our attention to other interesting types of graphs. In Section 4, we show that BPCP is strongly $\mathcal{NP}$-hard even on paths and we propose a 2-approximation algorithm for this case. In Section 5, 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 $\epsilon = O(\log q/q)$. In Section 6, we study U-BPCP and BPCP on general graphs, and present an algorithm achieving an approximation ratio of 5 for trees and 9 for planar graphs. This algorithm actually applies to most sparse graphs, as it yields a constant approximation ratio for any graph with bounded Maximum Average Degree. Then, based on a simple greedy approach, and given any ρ-approximation algorithm for the Bin Packing problem, we obtain a $\rho \cdot (\Delta + 1)$-approximation algorithm for BPCP on general graphs, where Δ is the maximum degree of the graph. Finally, we show that any ρ-approximation for the Densest q-Subgraph problem can be converted to a $\rho \cdot O(\log n)$-approximation algorithm for the U-BPCP on general graphs.

Graph Type	**Approximation Ratio**	**Theorem**	**Section**
Unweighted Complete	$3/2 + O(1/q) + O(q/n)$	5	2.2
Complete	5	7	3.2
Paths	2	9	4
Unweighted Trees	$1 + O(\log q/q)$	11	5
Trees	5	12	6.1
Planar	9	12	6.1
Unweighted General	$n^{1/4} \log n$	14	6.2
General	$3 + 2\lceil mad(G)/2 \rceil$, $(1 + \epsilon)(\Delta + 1)$	12, 13	6.1

Table 1: Approximation ratios for BPCP and U-BPCP obtained in this paper

Notation and Preliminaries. In what follows we denote by $W = \sum_{i=1}^{n} w_i$ the total sum of the weights of all the items. We suppose that $W > q$, for otherwise the solution consists of a single bin. By b^* we denote the number of bins of an optimal solution to BPCP or U-BPCP problems.

In the next theorem we summarize some of the known results for the classical Bin Packing problem which we use in the rest of the paper. For an overview of these results we refer the reader to [5] and the references there in.

Theorem 1 *Concerning the Bin Packing problem, with n items of integer weights $w_1, \ldots, w_n$ and an integer bin capacity q:*

(i) The Next Fit (NF), First Fit (FF) and First Fit Decreasing (FFD) algorithms:

(a) Achieve approximation ratios of 2, 1.7 and 1.5 respectively.

(b) Return a solution of at most $2\left\lceil \frac{W}{q} \right\rceil$ bins.

(ii) The problem is APX-hard and it admits an Asymptotic Polynomial Time Approximation Scheme.

2 Complete Graphs: Unweighted case

In this section we deal with U-BPCP on complete graphs. Due to its close relation with the theory of combinatorial designs (see [6]), we first briefly survey some fundamental results known in this area. Then, we present approximation algorithms for the problem.

2.1 Combinatorial Designs

Given a set V of n elements, a 2-$(n, q, 1)$-*design* is a collection of subsets of V of size q, called *blocks*, such that every pair of distinct elements appears together in exactly one block. In other words it corresponds to a partition of the edges of the complete graph K_n into K_q. 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 a 2-$(n, q, 1)$-design are $(n-1) \equiv 0 \pmod{q-1}$ and $n(n-1) \equiv 0 \pmod{q(q-1)}$. These necessary 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 $q = 4, 5$ or when q is a power of a prime and $n = q^2$ (affine planes) or $n = q^2 + q + 1$ (projective planes). Furthermore, Wilson [15] has proved that these necessary conditions are also sufficient when k is fixed and n is large enough. Still, in many cases these conditions do not guarantee the existence of a 2-$(n, q, 1)$-design; for example, as guessed by Euler both a 2-$(36, 6, 1)$-design or a 2-$(43, 7, 1)$-design do not exist [6].

Clearly, a 2-$(n, q, 1)$-design is an optimal solution for U-BPCP on a complete graph with n vertices and bin capacity q. Note that this relation was not observed by Afrati et al. [2, 1] 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 a 2-$(n, q, 1)$-design has been also extended to covering designs (see the survey [10] or chapter IV.8 in Handbook of Designs [13]). Given a set V of n elements, a 2-$(n, q, 1)$-covering design abbreviated here as (n, q)-covering is a collection of subsets, which are called *blocks*, such that each block has q elements and every pair of distinct elements of V appears together in *at least* one block. A (n, q)-covering is nothing else than a solution to U-BPCP for complete graphs.

In the literature, there exists a significant amount of work on computing the minimum number of blocks in a (n, q)-covering, called the covering number and denoted $C(n, q)$. Therefore, for U-BPCP on complete graphs, the number of bins of an optimal solution is equal to $C(n, q)$.

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 (See [10], [13]) *It holds that* $C(n,q) \geq L(n,q)$. *Furthermore, if* $(n-1) \equiv 0 \pmod{q-1}$ *and* $n(n-1) \equiv 1 \pmod{q}$, *then* $C(n,q) \geq L(n,q)+1$.

The exact values of $C(n,q)$ have been determined only in some cases (see [10, 13]). For example, the exact value of $C(n,q)$ is known for $n \leq 3q$ 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 $n = 7, 9, 10$, $\epsilon = 2$ when $n = 19$, and $\epsilon = 0$ otherwise.

Finally, the following theorem, proved by Rödl [11] via probabilistic methods, bounds $C(n,q)$ asymptotically. Interestingly, it answered a conjecture of Erdős and Hanani (see [3, Chapter 4] for a proof).

Theorem 2 (Rödl [11]) *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.

2.2 Approximation algorithms

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 minimum $(g,k,1)$-covering is provided. 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 associate a bin containing 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; to get rid of them we will use intensively the bounds $\lfloor a/b \rfloor \geq a/b-(b-1)/b$ and $\lceil a/b \rceil \leq a/b+(b-1)/b$. We have also to use existing minimum $(g,k,1)$-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 minimum $(g,k,1)$-coverings are known for large k.

For $k = 2$, a case for which a trivial $(g,2,1)$-covering exists, we get Algorithm 1. It is similar with the one of [2, 1] 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 $\lfloor q/2 \rfloor$, except possibly one group of smaller size.
2: Pack every pair of groups into a bin.

Theorem 3 *Algorithm 1 achieves approximation ratios of* $\frac{2(q-1)}{q} + \frac{(q-1)(q-2)}{qn}$, *if* q *is even, and* $\frac{2q}{q-1} + \frac{q(q-3)}{(q-1)n}$, *if* q *is odd, for U-BPCP on complete graphs.*

Proof. First, as explained in the general construction the algorithm gives a valid placement. The number of groups is $g = \left\lceil \frac{n}{\lfloor q/2 \rfloor} \right\rceil$ and the number of bins is $b = \frac{g(g-1)}{2}$. We split the analysis in two cases depending on whether q is even or odd.

If q is even, then $g = \left\lceil \frac{n}{q/2} \right\rceil \leq \frac{n+q/2-1}{q/2} = \frac{2n+q-2}{q}$. Hence, the number of used bins is

$$\begin{aligned} b = \frac{g(g-1)}{2} &\leq \frac{(2n+q-2)(2n-2)}{2q^2} \\ &= \frac{2n(n-1)+(q-2)(n-1)}{q^2} \\ &= \left(2\frac{q-1}{q} + \frac{(q-1)(q-2)}{qn}\right) \cdot \frac{n(n-1)}{q(q-1)} \\ &\leq \left(2\frac{q-1}{q} + \frac{(q-1)(q-2)}{qn}\right) \cdot L(n,q) \end{aligned}$$

If q is odd, then $g = \left\lceil \frac{n}{(q-1)/2} \right\rceil \leq \frac{2n+q-3}{q-1}$, and the number of used bins is

$$\begin{aligned} b = \frac{g(g-1)}{2} &\leq \frac{(2n+q-3)(2n-2)}{2(q-1)^2} \\ &= \frac{2n(n-1)+(q-3)(n-1)}{(q-1)^2} \\ &\leq \left(2\frac{q}{q-1} + \frac{q(q-3)}{(q-1)n}\right) \cdot L(n,q) \end{aligned}$$

■

Note that, by Theorem 3, we have an approximation ratio less than 2, when q is even and $n \geq q^2/2$. When $q = 3$ or q is odd and $n \leq 3q$, we know 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 $17/6$. For any case, when q is large and n tends to infinity the ratio is close to 2. In all cases, we have a ratio of $2 + 1/q + q/n$.

For general k we can extend the algorithm in the following way:

Algorithm 2 (U-BPCP, complete graphs)

1: Choose an integer $k \geq 2$.
2: Divide the items into $g = \left\lceil \frac{n}{\lfloor q/k \rfloor} \right\rceil$ groups each of size $\lfloor q/k \rfloor$, except perhaps one group.
3: Use a $(g, k, 1)$-covering to cover the pairs of groups with blocks of size k.
4: For each block, associate a bin containing all items of the groups included in the block.

The bins obtained by Algorithm 2 form a valid placement as each bin has size at most q and each pair of items appears in at least one group or pair of groups and so in at least one bin. We can still do a complete analysis of the construction for $k = 3$ as a $(g, 3, 1)$-covering exists with $L(g,3) = \left\lceil \frac{g}{3} \left\lceil \frac{g-1}{2} \right\rceil \right\rceil$. In our analysis we use the following inequality:

Fact 4 $L(g,3) = \left\lceil \frac{g}{3} \left\lceil \frac{g-1}{2} \right\rceil \right\rceil \leq \frac{g^2+2}{6}$

Proof. It follows by a case analysis of all the 6 possible congruences. If $g \equiv 1,3 \pmod 6$, then $\left\lceil \frac{g}{3} \left\lceil \frac{g-1}{2} \right\rceil \right\rceil = \frac{g(g-1)}{6} < \frac{g^2+2}{6}$. If $g \equiv 5 \pmod 6$, then $\left\lceil \frac{g}{3} \left\lceil \frac{g-1}{2} \right\rceil \right\rceil = \left\lceil \frac{g(g-1)}{6} \right\rceil = \frac{g^2-g+4}{6} < \frac{g^2+2}{6}$. If g is even, $\left\lceil \frac{g}{3} \left\lceil \frac{g-1}{2} \right\rceil \right\rceil = \left\lceil \frac{g^2}{6} \right\rceil$. If $g \equiv 0 \pmod 6$, then $\left\lceil \frac{g^2}{6} \right\rceil = \frac{g^2}{6} < \frac{g^2+2}{6}$ and if $g \equiv 2,4 \pmod 6$, $\left\lceil \frac{g^2}{6} \right\rceil = \frac{g^2+2}{6}$ and in all cases the inequality is true with equality only in the cases $g \equiv 2,4 \pmod 6$. ∎

Theorem 5 *For k=3, Algorithm 2 achieves approximation ratios of* $\frac{3}{2}\frac{q-1}{q} + \frac{q-1}{n} + \frac{q(q-1)}{2n(n-1)}$ *when q is a multiple of* 3*, and* $\frac{3}{2}\frac{q(q-1)}{(q-2)^2} + \frac{q}{n-1} + \frac{q(q-1)}{2n(n-1)}$ *otherwise, for U-BPCP on complete graphs.*

Proof. If q is a multiple of 3, then $g = \left\lceil \frac{n}{q/3} \right\rceil \leq \frac{n+q/3-1}{q/3} \leq \frac{3n+q-3}{q}$. The number b of bins (blocks) obtained by the algorithm admits using Fact 4 the following upper bound:

$$\begin{aligned} b = L(g,3) &\leq \frac{g^2+2}{6} \leq \frac{9n^2+6n(q-3)+(q-3)^2+2q^2}{6q^2} \\ &\leq \frac{9n(n-1)+3n(2q-3)+3q^2}{6q^2} \\ &\leq \left(\frac{3}{2}\frac{q-1}{q} + \frac{(2q-3)(q-1)}{2q(n-1)} + \frac{q(q-1)}{2n(n-1)}\right) \cdot \frac{n(n-1)}{q(q-1)} \\ &\leq \left(\frac{3}{2}\frac{q-1}{q} + \frac{(2q-3)(q-1)}{2q(n-1)} + \frac{q(q-1)}{2n(n-1)}\right) \cdot L(n,q) \end{aligned}$$

If q is not a multiple of 3, the worst case appears when q is congruent to 2 (mod 3) and we have $g = \left\lceil \frac{n}{\lfloor q/3 \rfloor} \right\rceil \leq \left\lceil \frac{n}{(q-2)/3} \right\rceil \leq \frac{3n+q-5}{(q-2)}$. The number b of bins obtained by the algorithm is using Fact 4 at most

$$\begin{aligned} b &\leq \frac{g^2+2}{6} \leq \frac{9n^2+6n(q-5)+3q^2-18q+33}{6(q-2)^2} \\ &\leq \left(\frac{3}{2}\frac{q(q-1)}{(q-2)^2} + \frac{q}{n-1}\frac{(6q-21)(q-1)}{6(q-2)^2} + \frac{q(q-1)}{2n(n-1)}\right) \cdot \frac{n(n-1)}{q(q-1)} \\ &\leq \left(\frac{3}{2}\frac{q(q-1)}{(q-2)^2} + \frac{q}{n-1} + \frac{q(q-1)}{2n(n-1)}\right) \cdot L(n,q) \end{aligned}$$

∎

The general algorithm described above, for $k = 3$ (resp. $k = 4$) and q is a multiple of 3 (resp. 4), achieves an approximation ratio of at most 3/2 (resp. 4/3). More generally, for any k, if q is a multiple of k and there exists a $(g,k,1)$-covering with $L(g,k)$ blocks, for $g = \left\lceil \frac{kn}{q} \right\rceil$, we get a $\frac{k}{k-1}$-approximation ratio.

3 Complete graphs: Weighted case

The general idea of partitioning the items into appropriate groups can be also extended to BPCP on complete graphs. Note that in [2, 1], it is shown that a variant of Algorithm 1 achieves an approximation ratio of 8 for BPCP problem on complete graphs (instead of partitioning the items we greedily pack them into groups of size at most $q/2$). Here, we initially present a 6-approximation

algorithm via a better, but still simple, grouping. Then, we improve it via a more enhanced grouping to a 5-approximation one.

Our analysis uses the lower bound on the optimal number of bins b^* provided by the next lemma. Although this lemma is also implicitly shown in [2, 1], we restate and prove it here for the sake of completeness.

Lemma 2 *For BPCP on complete graphs it holds that* $b^* \geq \frac{1}{q}\sum_{i=1}^{n} w_i \left\lceil \frac{W-w_i}{q-w_i} \right\rceil > \frac{W^2}{q^2}$.

Proof. When item i belongs to some bin, the other items in the same bin have a sum of weights at most $q - w_i$. So, item i should appear in at least $\left\lceil \frac{W-w_i}{q-w_i} \right\rceil$ bins in order to be colocated with all other items. Altogether, the total capacity required by item i is at least $w_i \left\lceil \frac{W-w_i}{q-w_i} \right\rceil$. Thus, for the optimal number of bins we have $b^*q \geq \sum_{i=1}^{n} w_i \left\lceil \frac{W-w_i}{q-w_i} \right\rceil$ and the first bound follows. Moreover, $\frac{W-w_i}{q-w_i} \geq \frac{W}{q}$, since $w_i < q < W$, for each item i. Therefore, $b^*q \geq \frac{W}{q}\sum_{i=1}^{n} w_i$ and $b^* \geq \frac{W^2}{q^2}$. ∎

Consider now the relation between the weights, w_i, of the items and the capacity q of the bins. Indeed, in the BPCP on complete graphs, there can be at most one item of weight greater than $q/2$, for otherwise there is no feasible solution. We denote by $\text{BPCP}_{q/2}$ the special case of BPCP on complete graphs where all the items have weights at most $q/2$. The next lemma relates this special case with BPCP on complete graphs in terms of approximation algorithms and allows us to work in the sequel on $\text{BPCP}_{q/2}$.

Lemma 3 *If there is a* ρ*-approximation algorithm for* $BPCP_{q/2}$ *on complete graphs, which returns a solution of* $b \leq \rho\frac{W^2}{q^2} < \rho b^*$ *bins, then there is a* $\max\{\rho, 4\}$*-approximation algorithm for BPCP on complete graphs.*

Proof. Let w_1 be the single weight of BPCP greater than $q/2$. By Lemma 2 the optimal number of bins of BPCP satisfies

$$b^* \geq \frac{1}{q}\left(w_1\left\lceil\frac{W-w_1}{q-w_1}\right\rceil + \sum_{i=2}^{n} w_i\left\lceil\frac{W-w_i}{q-w_i}\right\rceil\right) \geq \frac{1}{q}w_1\left\lceil\frac{W-w_1}{q-w_1}\right\rceil + \frac{W}{q^2}(W-w_1), \tag{1}$$

since we have that $w_i < q < W$, it implies $\frac{W-w_i}{q-w_i} \geq \frac{W}{q}$, and $\sum_{i=2}^{n} w_i = W - w_1$.

Now we consider the following algorithm for BPCP. First, we use a Bin Packing algorithm (NF, FF or FFD) to pack all the items, but item 1 into bins of capacity $q - w_1$. Then, we add item 1 in all the bins and so it is colocated in at least one bin with each other item. By Theorem 1(i)(b), the number of the bins used by such a standard Bin Packing algorithm satisfies

$$b_1 \leq 2\left\lceil\frac{W-w_1}{q-w_1}\right\rceil < \frac{4}{q}w_1\left\lceil\frac{W-w_1}{q-w_1}\right\rceil,$$

since $w_1 > q/2$, that is $2 < 4w_1/q$.

Next, we pack the items different from 1 into bins of capacity q to guarantee that each one of then will be colocated with each other in at least one bin. All of these items have weights at most $q/2$ and we use for their packing the existing, by the hypothesis of the lemma, ρ-approximation algorithm for the corresponding $\text{BPCP}_{q/2}$ with $n-1$ items of total weight $W - w_1$. Thus, the number of bins used by this algorithm satisfies

$$b_2 \leq \rho\frac{(W-w_1)^2}{q^2} \leq \rho\frac{W}{q^2}(W-w_1).$$

Therefore the total number of bins used by both steps above is

$$b = b_1 + b_2 \leq \frac{4}{q} w_1 \left\lceil \frac{W - w_1}{q - w_1} \right\rceil + \rho \frac{W}{q^2}(W - w_1). \tag{2}$$

Combining the lower bound on an optimal solution to BPCP (1) and the upper bound on the solution of the algorithm above (2) the lemma follows. ∎

3.1 A 6-approximation algorithm

We present now a 6-approximation algorithm for $\text{BPCP}_{q/2}$ and, hence, by Lemma 3, for BPCP on complete graphs.

The main idea of the algorithm is to a) partition the items into groups of 3 different types and; b) pack these groups into bins. For each pair of items, their corresponding groups will appear together in a bin, ans thus their respective colocation constraint is satisfied.

For each group we define its density d, as the sum of the weights of the items in this group divided by q. So, if a group has density d, it contains items of total weight dq. Note that in order to pack groups together they should have a density of at most $1/2$. We construct three types of groups, denoted by A, B and C, according to their densities, as follows:

- Groups of type A with density $1/3 < d \leq 1/2$
- Groups of type B with density $1/4 < d \leq 1/3$
- Groups of type C with density $d \leq 1/4$.

We denote the number of groups of each type A, B and C, by n_A, n_B and n_C, respectively. The groups are constructed by packing the items greedily in an arbitrary order. The next item of the order is packed to an existing group if possible, or to a new group, otherwise. By the construction of the groups we ensure that there is at most one group of type C. Indeed, two groups of type C could be merged into a single group of type A, B or C, as the density of a group of type C is $d_C \leq 1/4$. That is, $n_C \in \{0, 1\}$. Furthermore, if $n_C = 1$ and $n_B \geq 1$, then $d_C \geq 1/6$; otherwise, the group of type C can be merged with a group of type B to obtain a new group of density at most $1/6 + 1/3 = 1/2$.

Then, our algorithm returns the following bins:

- $\binom{n_A}{2}$ bins for each pair of groups of type A.
- $n_A n_B$ bins for each pair of groups made of one group of type A and one group of type B.
- $\left\lceil \frac{n_B}{3} \left\lceil \frac{n_B - 1}{2} \right\rceil \right\rceil$ bins for each triplet of groups of type B in a $(n_B, 3, 1)$-covering design. This is possible as the sum of their densities is at most $3 \cdot 1/3 = 1$.
- $n_A n_C$ bins for each group of type A and the group of type C, if it exists (recall that $n_C \in \{0, 1\}$).
- $\left\lceil \frac{n_B}{2} \right\rceil n_C$ bins : we cover the groups of type B with $\left\lceil \frac{n_B}{2} \right\rceil$ pairs in such a way that each group of type B appears in at least one pair and put each such pair with the group of type C, if it exists. This is possible as the sum of their densities is at most $2 \cdot 1/3 + 1/4 \leq 1$.

Theorem 6 *There is a 6-approximation algorithm for BPCP on complete graphs.*

Proof. For the sum W of the weights of all items it holds by definition of the densities that

$$\frac{W}{q} \geq \frac{n_A}{3} + \frac{n_B}{4} + n_C \cdot d_C$$

So we get the following inequality for $\frac{W^2}{q^2}$ which is by Lemma 2, a lower bound on the number of bins b^* in an optimal solution

$$\frac{W^2}{q^2} \geq \frac{1}{144}\left(16n_A^2 + 9n_B^2 + 24n_An_B + 96n_An_Cd_C + 72n_Bn_Cd_C + 144n_C^2d_C^2\right) \tag{3}$$

On the other hand, the number b of bins used by the algorithm is:

$$b \leq \binom{n_A}{2} + n_An_B + \left\lceil \frac{n_B}{3}\left\lceil \frac{n_B-1}{2}\right\rceil\right\rceil + n_An_C + \left\lceil \frac{n_B}{2}\right\rceil n_C \tag{4}$$

We will show that $b \leq 6W^2/q^2$. First we note that if $n_A + n_B \leq 2$, as $n_C \leq 1$, our algorithm returns at most three bins and the theorem is true as $W/q > 1$. So we suppose in what follows that $n_A + n_B \geq 3$. In that case we will show that $F = 144W^2/q^2 - 24b \geq 0$ where, by neglecting the coefficient $144n_C^2d_C^2$ in inequality (3),

$$F = 4n_A^2 + 12n_A + 24n_An_C(4d_C - 1) + 9n_B^2 - 24\left\lceil \frac{n_B}{3}\left\lceil \frac{n_B-1}{2}\right\rceil\right\rceil + 12n_C\left(6d_Cn_B - 2\left\lceil \frac{n_B}{2}\right\rceil\right) \tag{5}$$

Note the important fact (due to the choices of the densities) that the coefficient of n_An_B is 0. We did not write the factor with d_C^2 which is at most 4. To conclude we distinguish 2 cases:

- $\mathbf{n_B = 0}$; then $F \geq 4n_A^2 - 12n_A \geq 0$ as $n_A = n_A + n_B \geq 3$.
- $\mathbf{n_B > 0}$. Recall that in that case, if $n_C = 1$, then $d_C \geq 1/6$ (otherwise, the group of type C can be merged with a group of type B). Then using by Fact 4 $\left\lceil \frac{n_B}{3}\left\lceil \frac{n_B-1}{2}\right\rceil\right\rceil \leq \frac{{n_B}^2+2}{6}$ we proceed as follows. If $n_C = 0$, then $F \geq 4n_A^2 + 12n_A + 5n_B^2 - 8$. If $n_C = 1$, then $F \geq 4n_A^2 + 5n_B^2 + 4n_A - 20$. In both cases, $F \geq 0$ as $n_A + n_B \geq 3$.

Summarizing, we have shown that $b \leq 6\frac{W^2}{q^2}$, that is a 6-approximation ratio for $\text{BPCP}_{q/2}$ and, by Lemma 3, also for BPCP on complete graphs. ∎

3.2 A 5-approximation algorithm

In this section we use a more enhanced grouping to improve the approximation ratio of our previous algorithm from 6 to 5. We again present our algorithm for $\text{BPCP}_{q/2}$ and use Lemma 3 for BPCP on complete graphs.

The reason for which the algorithm in the previous subsection achieves a better approximation ratio compared to the algorithm by Afrati et al. [2, 1] is that a triplet of groups of type B can be packed in a bin. We refine this idea by also packing one group of type A with 2 groups of type B in a bin if it is possible, i.e., if the sum of their densities is at most 1. So, we partition the groups of type A (resp. B) into 2 types A_1 and A_2 (resp. B_1 and B_2). The new types of groups are denoted by A_1, A_2, B_1, B_2, C, and the number of groups of each type by $n_{A_1}, n_{A_2}, n_{B_1}, n_{B_2}, n_C$, respectively. Let $A = A_1 \cup A_2$, $B = B_1 \cup B_2$, $n_A = n_{A_1} + n_{A_2}$ and $n_B = n_{B_1} + n_{B_2}$. We define the following types of groups:

- Groups of type A_1 with density $4/10 < d \leq 1/2$
- Groups of type A_2 with density $1/3 < d \leq 4/10$
- Groups of type B_1 with density $3/10 < d \leq 1/3$
- Groups of type B_2 with density $1/4 < d \leq 3/10$
- Groups of type C with density $d \leq 1/4$

The groups are constructed by packing the items greedily in an arbitrary order similarly to the algorithm of Section 3.1. Consequently, we know that there is at most one group of type C, i.e. $n_C \in \{0, 1\}$, and if there is also a group of type B, i.e. $n_B > 0$, then $d_C \geq 1/6$. Additionally, we ensure that, if $n_{A_2} > 0$, then $d_C \geq 1/10$; otherwise, the group of type C can be merged with a group of type A_2, as $4/10 + 1/10 \leq 1/2$. The algorithm produces the following bins:

- $\binom{n_A}{2}$ bins for each pair of groups of type A.
- $n_{A_1} n_B$ bins for each group of type A_1 and each group of type B.
- $n_{A_2} n_{B_1}$ bins for each group of type A_2 and each group of type B_1.
- $n_{A_2} \left\lceil \frac{n_{B_2}}{2} \right\rceil$ bins for each group of type A_2 and $\left\lceil \frac{n_{B_2}}{2} \right\rceil$ pairs of groups of type B_2 in such a way each group of type B_2 appears in at least one such pair. This is possible as the sum of their densities is at most $4/10 + 2 \cdot 3/10 = 1$.
- $\left\lceil \frac{n_B}{3} \left\lceil \frac{n_B - 1}{2} \right\rceil \right\rceil$ bins for each triplet of groups of type B in a $(n_B, 3, 1)$-covering design. This is possible as the sum of their densities is at most $3 \cdot 1/3 = 1$.
- $n_A n_C$ bins for each a group of type A and group of type C
- $\left\lceil \frac{n_B}{2} \right\rceil n_C$ bins : we cover the groups of type B with $\left\lceil \frac{n_B}{2} \right\rceil$ pairs in such a way each group of type B appears in at least one pair and put each such pair with the group of type C, if it exists. This is possible as the sum of their densities is at most $2 \cdot 1/3 + 1/4 \leq 1$.

Theorem 7 *There is a 5-approximation algorithm for BPCP on complete graphs.*

Proof. The proof is similar to the proof of Theorem 6. We will show that $b \leq 5W^2/q^2$. Again if $n_A + n_B \leq 2$, as $n_C \leq 1$, 3 bins are sufficient and so the result is true as $W > q$. So we suppose in what follows that $n_A + n_B \geq 3$.

For the sum W of the weights of all items it holds that

$$\frac{W}{q} \geq \frac{4}{10} \cdot n_{A_1} + \frac{1}{3} \cdot n_{A_2} + \frac{3}{10} \cdot n_{B_1} + \frac{1}{4} \cdot n_{B_2} + n_C \cdot d_C$$

We obtain the following lower bound where we do not distinguish between A_1 and A_2 (resp. B_1 and B_2) in the expression of the squares n_A^2 (resp n_B^2) and of the product $n_A n_C$ (resp. $n_B n_C$) and we use only the fact that $d_A \geq 1/3$ (resp. $d_B \geq 1/4$).

$$\begin{aligned} 5\frac{W^2}{q^2} \geq\ & (5/9)n_A^2 + (12/10)n_{A_1}n_{B_1} + n_{A_1}n_{B_2} + n_{A_2}n_{B_1} + (10/12)n_{A_2}n_{B_2} \\ & + (5/16)n_B^2 + (10/3)d_C n_A n_C + (10/4)d_C n_B n_C + 5d_C^2 n_C^2 \end{aligned} \tag{6}$$

On the other hand, the number b of bins used by the algorithm is

$$b \leq \binom{n_A}{2} + n_{A_1} n_B + n_{A_2} n_{B_1} + n_{A_2} \left\lceil \frac{n_{B_2}}{2} \right\rceil + \left\lceil \frac{n_B}{3} \left\lceil \frac{n_B - 1}{2} \right\rceil \right\rceil + n_A n_C + \left\lceil \frac{n_B}{2} \right\rceil n_C \tag{7}$$

We use inequalities (6) and (7) to compute $5W^2/q^2 - b = F = F_A + F_B$. In the expression of F, we observe that the coefficients of $n_{A_1} n_{B_2}$ and $n_{A_2} n_{B_1}$ are 0. We do not write the positive terms $(2/10) n_{A_1} n_{B_1}$ and $5 d_C^2 n_C^2$. Additionally, we define the term ϵ_B (resp. ϵ_{B_2}) equal to 1 if n_B is odd (resp. n_{B_2} is odd) and zero, otherwise; that is: $\left\lceil \frac{n_B}{2} \right\rceil = \frac{n_B + \epsilon_B}{2}$ (resp. $\left\lceil \frac{n_{B_2}}{2} \right\rceil = \frac{n_{B_2} + \epsilon_{B_2}}{2}$).

$$\begin{aligned} F_A \geq \;& (1/18) n_A^2 + (1/2) n_A + (10/3 \cdot d_C - 1) n_A n_C + n_{A_2} \cdot (n_{B_2}/3 - \epsilon_{B_2}/2) \\ F_B \geq \;& (5/16) n_B^2 - \left\lceil \frac{n_B}{3} \left\lceil \frac{n_B - 1}{2} \right\rceil \right\rceil + (10/4 \cdot d_C - 1/2) n_B n_C - (1/2) \epsilon_B n_C \end{aligned} \tag{8}$$

In what follows, we distinguish three cases:

- $\mathbf{n_B = 0}$; then, $F_B = 0$ and, $F_A \geq (1/18) n_A^2 + (1/2) n_A + (10/3 \cdot d_C - 1) n_A n_C$.

  If $n_{A_2} > 0$, recall that $d_C \geq 1/10$, and so $F_A \geq (1/18) n_A^2 + (1/2) n_A - (2/3) n_A$ and so, as $n_A + n_B = n_A \geq 3$, $F_A \geq 0$.

  If $n_{A_2} = 0$, then the groups of type A are all of type A_1 and so $W/q \geq (4/10) n_A$; on the other side $b \leq \binom{n_A}{2} + n_A n_C$ and $5W^2/q^2 - b \geq (3/10) n_A^2 - n_A/2 > 0$ as $n_A \geq 3$.

- $\mathbf{n_B \geq 2}$. To lower bound F_B, we use Fact 4: $\left\lceil \frac{n_B}{3} \left\lceil \frac{n_B - 1}{2} \right\rceil \right\rceil \leq \frac{{n_B}^2 + 2}{6}$. We also note that either $n_C = 0$ or $n_C = 1$, but in that case as $n_B > 0$, $d_C \geq 1/6$ and so we get:

  $F_B \geq (5/16 - 1/6) n_B^2 - 2/6 + (5/12 - 1/2) n_B - \epsilon_B/2$.

  If $n_B \geq 3$, $F_B \geq 63/48 - 2/6 - 3/12 - 1/2 \geq 11/48$. If $n_B = 2$, $F_B \geq 28/48 - 2/6 - 2/12 = 1/12$. So in all the cases $F_B \geq 1/12$

  For F_A using $d_C \geq 1/6$ when $n_C = 1$, we get: $F_A \geq (1/18) n_A^2 + (1/18) n_A + n_{A_2} \cdot (n_{B_2}/3 - \epsilon_{B_2}/2)$.

  Note that, if $n_{B_2} \neq 1$, then $n_{B_2}/3 - \epsilon_{B_2}/2 \geq 0$. Therefore when $n_{B_2} \neq 1$ or $n_{A_2} = 0$ we have $F_A \geq (1/18) n_A^2 + (1/18) n_A \geq 0$.

  If $n_{B_2} = 1$ and $n_{A_2} > 0$, we have $F_A \geq (1/18) n_A^2 + (1/18) n_A - (1/6) n_{A_2}$. So $F_A \geq 0$ if $n_A \geq 2$. If $n_A = n_{A_2} = 1$, $F_A \geq -(1/18)$. In all the cases $F = F_A + F_B \geq 1/12 - 1/18 \geq 0$

- $\mathbf{n_B = 1}$. Then, $F_B \geq 5/16 - 1/12 - \epsilon_B/2 = -13/48$.

  Like in the preceeding case, if $n_{B_2} \neq 1$ or $n_{A_2} = 0$ we have $F_A \geq (1/18) n_A^2 + (1/18) n_A$. Therefore, as $n_A + n_B = n_A + 1 \geq 3$, $F_A \geq 1/3$ and so $F = F_A + F_B \geq 1/3 - 13/48 \geq 0$.

  If $n_{B_2} = 1$ and $n_{A_2} > 0$, we have $F_A \geq (1/18) n_A^2 + (1/18) n_A - (1/6) n_{A_2}$. So, if $n_A \geq 4$, $F_A \geq 4/9$ and $F = F_A + F_B \geq 4/9 - 17/48 \geq 0$.

  To conclude it remains to deal with the very particular small cases $n_C = 1$, $n_B = n_{B_2} = 1$; $n_{A_2} \geq 1$ and $2 \leq n_A \leq 3$ (in the case $n_A = 1$, $n_A + n_B \leq 3$). If $n_A = 2$, our algorithm uses at most 4 bins: let the groups of type A (resp. B, C) be A^1, A^2 with A^1 of type A_2 (resp B^1, C^1). We use the bins (A^1, A^2), (A^2, B^1), (A^2, C^1) and (A^1, B^1, C^1). Note that the last bin content has weight at most $4/10 + 3/10 + 1/4 < 1$. On the other hand, the optimal solution uses at least one bin. Finally, if $n_A = 3$, we have $W/q \geq 1/6 + 1/4 + 3 \cdot 1/3 \geq 17/12$

and so $W^2/q^2 \geq 2$. But we can pack our groups in 8 bins; namely let the groups of type A (resp. B, C) be A^1, A^2, A^3 with A^1 of type A_2 (resp B^1, C^1). We use three bins for the pairs of groups of type A, one for (A^1, B^1, C^1); and four for $(A^2, B^1), (A^3, B^1), (A^2, C^1), (A^3, C^1)$. Therefore $b = 8 \leq 10 \leq 5W^2/q^2$.

In summary, we have shown that $b \leq 5\frac{W^2}{q^2}$, that is a 5-approximation ratio for $\text{BPCP}_{q/2}$ and, by Lemma 3, also for BPCP on complete graphs. ∎

Remark: An interesting question is whether the groups of type A and B can be partitioned in a different way in order to achieve a better approximation ratio. Let d_A (resp. d_B) be new density values that replaces 4/10 (resp. 3/10) for partitioning the groups of type A (resp. B). Our algorithm requires that a group of type A_2 can be packed with a pair of groups of type B_2 in a bin, i.e. $d_A + 2d_B \leq 1$. In the expression $\rho W^2/q^2 - b$, the coefficients of terms $n_{A_1} n_{B_2}$ and $n_{A_2} n_{B_1}$ are $2\rho \cdot d_A/4 - 1$ and $2\rho \cdot d_B/3 - 1$, respectively. In order to end up having $\rho W^2/q^2 - b \geq 0$, we must choose non-negative such coefficients. That is, $2\rho \cdot d_A/4 \geq 1$ and $2\rho \cdot d_B/3 \geq 1$. So, $\rho(d_A + 2d_B) \geq 5$; but $d_A + 2d_B \leq 1$. The best achievable ratio in this way is $\rho = 5$, by picking $d_A = 4/10$ and $d_B = 3/10$.

4 Paths

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

Theorem 8 *BPCP on paths is strongly $\mathcal{NP}$-hard.*

Proof. Let $I = \{1, 2, \cdots, n\}$ be a set of items. Starting from an instance of the classical Bin Packing problem with n items $i \in I$ of integer weights w_i and integer bin capacity q, we construct an instance of BPCP as follows. The graph G is a path with $2n - 1$ vertices; vertices $2i - 1$, $1 \leq i \leq n$, (each one corresponding to the original item $i \in I$) have weight nw_i and vertices $2i$, $1 \leq i \leq n - 1$, (called dummy vertices) have unit weights. Each dummy vertex is connected with two original item vertices. The bin capacity of BPCP problem is $nq + (n - 1)$. We claim that there is a feasible solution for Bin Packing using b bins if and only if there is a feasible solution for BPCP using also b bins.

Consider, first, a feasible solution S of Bin Packing using b bins. Let B_k be the set of items packed in the k-th bin of S, $1 \leq k \leq b$. Each item $i \in I$ is packed in at least one bin and $\sum_{i \in B_k} w_i \leq q$. We construct a feasible solution S' for BPCP using also b bins as follows. For each bin B_k of the solution S, with $1 \leq k \leq b$, we create a bin B'_k of the solution S' by packing into B'_k, for each item $i \in B_k$, the vertex $2i - 1$ and its dummy neighbors in the path, if they are not already packed. Thus, as each item $i \in I$ is packed in at least one bin B_k, every edge of the path appears in at least one bin B'_k. Moreover, as we have at most $n - 1$ dummy vertices in a bin B'_k, the total weight packed in a bin B'_k is at most $\sum_{i \in B_k} nw_i + (n - 1) \leq nq + (n - 1)$. Therefore, S' is a feasible solution of BPCP using b bins.

Consider, next, a feasible solution S' to BPCP using b bins. Let B'_k be the set of vertices packed in the k-th bin of S', with $1 \leq k \leq b$, and let $B_k = \{i \in I \mid (2i - 1) \in B'_k\}$ be the subset of non-dummy vertices packed in this bin. Since S' is feasible,

$$\sum_{i \in I | (2i-1) \in B'_k} nw_i \leq nq + (n - 1) \Rightarrow \sum_{i \in B_k} w_i \leq q + \frac{n - 1}{n} \Rightarrow \sum_{i \in B_k} w_i \leq q,$$

where the last inequality holds since the w_i's and q are integers. Therefore, there exists a feasible solution for Bin Packing problem using b bins. ∎

We now present a 2-approximation algorithm by a reduction to the shortest path problem on an appropriate directed graph and the use of Theorem 1(i)(b). Consider an instance of BPCP on a path $G = (V, E)$, with $V = \{1, 2, \ldots, n\}$ and $E = \{(i, i+1) | 1 \leq i \leq n-1\}$, and bin capacity q. Let $W(i, j) = \sum_{k=i}^{j} w_k$ denote weight of the subpath between vertices i and j of G, with $i < j$. We construct a weighted directed graph $\overrightarrow{G}$, with the same vertex set V as G and where there exists an arc from i to j with weight $W(i, j)$ if and only if $W(i, j) \leq q$.

Let $\overrightarrow{P}$ be a directed path from vertex 1 to vertex n in $\overrightarrow{G}$ and $W(\overrightarrow{P}) = \sum_{(i,j)\in\overrightarrow{P}} W(i, j)$ be its total weight. Each arc $(i, j) \in \overrightarrow{P}$ corresponds to the subpath $i, i+1, \ldots, j$ of G which has a total weight $W(i, j) \leq q$. We call such an arc of $\overrightarrow{G}$ or subpath of G, a group (of vertices). Clearly, for any path $\overrightarrow{P}$ in $\overrightarrow{G}$ there is a trivial solution to BPCP on G, packing each group of $\overrightarrow{P}$ into a different bin. Note that each edge of G appears in one of the groups and so in one bin. However, we can do better using Algorithm 3, where we consider the groups of path $\overrightarrow{P}$ as input to a classical Bin Packing problem.

Algorithm 3 (BPCP, paths)

1: Construct the directed graph $\overrightarrow{G}$.
2: Find a minimum weight path $\overrightarrow{P}$ in $\overrightarrow{G}$ from 1 to n.
3: For each arc $(i, j) \in \overrightarrow{P}$, create a group of weight $W(i, j)$.
4: Pack the groups by a Bin Packing algorithm (NF, FF or FFD).

The next lemma gives a lower and an upper bound on the number of bins used by Algorithm 3 via the relation between a path $\overrightarrow{P}$ in $\overrightarrow{G}$ and a solution to the BPCP on G.

Lemma 4

(i) *For any path $\overrightarrow{P}$ in $\overrightarrow{G}$ from 1 to n, there is a feasible solution to the BPCP on G using $b \leq 2\left\lceil W(\overrightarrow{P})/q \right\rceil$ bins.*

(ii) *For any feasible solution to BPCP on G with b bins, there is a path $\overrightarrow{P}$ such that $W(\overrightarrow{P}) \leq bq$.*

Proof.
(i) The bound follows by Theorem 1(i)(b). Note that each edge of G appears in one of the groups and so in one bin.
(ii) Consider a feasible solution to BPCP on G with b bins. By deleting occurrences of vertices in some bins we can build a solution with the same number b of bins as the original one but satisfying the two properties:

Property 1: Each vertex appears at most once in a bin; otherwise it suffices to delete the useless other occurrences of the vertex.

Property 2: Each edge $(i, i+1)$ of G appears in at most one bin. Recall that by definition of a feasible solution each edge $(i, i+1)$ appears in at least one bin. Then, consider the vertices from 1 to n. For vertex 1 consider one of the bins, where it appears with vertex 2, keep it in this bin and delete its occurrences in all other bins where it appears with vertex 2. Then,

suppose that the property is satisfied for vertex $i-1$. Consider now vertex i. By induction it appears in exactly one bin with $i-1$. Now, if $i+1$ is also in this bin keep i in this bin and delete the occurrences of i in all other bins. Otherwise, if i appears with $i+1$ in some bins different from the one containing $i-1$, then keep i in one of these bins and delete the occurrences of i in the other bins.

Consider now a solution satisfying these two properties. By Property 1, each bin contains a set of vertices which form vertex disjoint subpaths in G and so correspond to different arcs in $\overrightarrow{G}$; recall that the weight of any subpath is at most the capacity q of the bin. By Property 2, each edge $(i, i+1)$ of G appears exactly in one bin and so the union of the of arcs appearing in all the bins form a path $\overrightarrow{P}$. Hence, the total weight $\overrightarrow{P}$ is at most bq. ∎

Theorem 9 *There is a 2-approximation algorithm for BPCP on paths.*

Proof. Consider, first, Algorithm 3 finding a minimum weight path $\overrightarrow{P}^*$ in $\overrightarrow{G}$. By Lemma 4(i) the number of bins obtained by the algorithm is $b \leq 2\left\lceil W(\overrightarrow{P}^*)/q \right\rceil$. Consider, next, an optimal solution to BPCP on a path G with b^* bins. By Lemma 4(ii) there is a path $\overrightarrow{P}$ in $\overrightarrow{G}$ such that $W(\overrightarrow{P}) \leq b^*q$. Therefore,

$$b \leq 2\left\lceil \frac{W(\overrightarrow{P}^*)}{q} \right\rceil \leq 2\left\lceil \frac{W(\overrightarrow{P})}{q} \right\rceil \leq 2\left\lceil \frac{b^*q}{q} \right\rceil = 2b^*,$$

since $\overrightarrow{P}^*$ is a minimum weight path. ∎

5 Trees

In this section we deal with U-BPCP on trees. We show that U-BPCP is $\mathcal{NP}$-hard and we also present an approximation algorithm which achieves asymptotically a ratio of $(1+\epsilon)$, where $\epsilon = O(\log q/q)$. For BPCP on trees, a 5-approximation algorithm is obtained as a corollary of an approximation result for BPCP on general graphs which we present in the next section.

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

Theorem 10 *U-BPCP on trees is strongly $\mathcal{NP}$-hard.*

Proof. The 3-Partition problem is defined as follows. We are given a set $A = \{a_1, a_2, \ldots, a_{3m}\}$ of positive integers such that $\sum_{i=1}^{3m} a_i = mB$ and $\frac{B}{4} < a_i < \frac{B}{2}$, for $1 \leq i \leq 3m$. The objective is to partition A into m pairwise disjoint subsets S_j, $1 \leq j \leq m$, where each subset S_j has exactly 3 elements $\{a_{j_1}, a_{j_2}, a_{j_3}\}$ and $a_{j_1} + a_{j_2} + a_{j_3} = B$. Note that 3-Partition is $\mathcal{NP}$-hard even in the case where each $a_i \in A$ is polynomially bounded by the size of the instance.

From an instance of 3-Partition, we construct an unweighted tree G by adding a root vertex and $3m$ pairwise disjoint paths P_i, $1 \leq i \leq 3m$, all having the root vertex as a common endpoint. Each path P_i contains the root vertex and a_i other vertices when $1 \leq i \leq 3m$. All vertices have unit weight and the bin capacity is $B+1$. Then, we claim that there exists a feasible solution to U-BPCP on the tree G using m bins if and only if A admits a 3-partition.

Assume, first, that there exists a 3-partition of A with m triples $S_j = \{a_{j_1}, a_{j_2}, a_{j_3}\}$, $1 \leq j \leq m$. Then, we obtain a feasible solution to U-BPCP on the tree G using m bins, by packing the root vertex and the vertices of the three paths $P_{j_1}, P_{j_2}, P_{j_3}$ in the j-th bin. Thus, a bin contains $1 + a_{j_1} + a_{j_2} + a_{j_3} = B + 1$ vertices and every edge of these three paths appears in this bin.

Assume, next, that there exists a feasible solution of U-BPCP on the tree G using m bins. In such a solution, each bin contains at most $B + 1$ vertices and at most B edges, as the subgraph induced by these vertices is a subgraph of a tree. Furthermore, a bin contains exactly B edges if and only if the subgraph induced by these $B + 1$ vertices is a connected subtree. We call such a bin containing $B + 1$ vertices and B edges *perfect*. The tree G has mB edges and, therefore, in a solution with m bins, each bin should contain exactly B edges and so all the bins are perfect. Now, we claim that all the edges of a path appear in a unique bin. Indeed, if the edges of a path appear in at least two bins, then consider the bin not containing the edge between the root and the next vertex of the path. Then, the vertices in this bin induce a disconnected subtree with a number of edges $< a_i < B$ and so this bin is not perfect, a contradiction. So, a perfect bin B_j contains the vertices of k paths $P_{j_1}, P_{j_2}, \ldots, P_{j_k}$ such that $1 + a_{j_1} + a_{j_2} + \ldots + a_{j_k} = B + 1$. If $k \leq 2$, we get a contradiction as $a_i < B/2$ implies $1 + a_{j_1} + a_{j_2} < 1 + B/2 + B/2 = B + 1$. If $k \geq 4$ we also get a contradiction as $a_i > B/4$ implies $1 + a_{j_1} + a_{j_2} + a_{j_3} + a_{j_4} > 1 + 4 \cdot B/4 = B + 1$. Therefore, a perfect bin B_j contains exactly the vertices of three paths $P_{j_1}, P_{j_2}, P_{j_3}$ such that $1 + a_{j_1} + a_{j_2} + a_{j_3} = B + 1$. It suffices now to associate to the three paths packed in a bin B_j the triple $S_j = \{a_{j_1}, a_{j_2}, a_{j_3}\}$ to obtain a 3-partition of A with m triples. ∎

For our approximation algorithm for U-BPCP on an input tree G, we first obtain a directed tree by orienting the edges of G away from some arbitrary node which is picked as the root. 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 obtained by deleting the edges with both endpoints in T' and then all the remaining vertices of degree 0 consists of a single tree. The following decomposition lemma is critical for designing our algorithm.

Lemma 5 *For each directed tree T, and for each $p \in [1, |V_T|]$, there exists an eligible subtree T' of T such that $p/2 \leq |V_{T'}| \leq p$.*

Proof. To each vertex $i \in V_T$ we associate the value p_i which is the number of nodes reachable from i (including i itself) in the (directed) tree T (in other words the number of vertices of the directed subtree rooted at i). Observe that, for the root r, $p_r = |V_T| \geq p$ and that, for any leaf u, $p_u = 1$. Therefore, there exists a vertex i, whose children are the vertices $i_1, i_2, \ldots, i_\ell$, such that $p_i \geq p$ and $p_{i_j} < p$, for $1 \leq j \leq \ell$. We order the children of i so that $p_{i_1} \geq p_{i_2} \geq \ldots \geq p_{i_\ell}$. If $p_{i_1} \geq (p-2)/2$, we take as eligible subtree the subtree containing i and the subtree rooted at i_1. Otherwise i has at least two children and for each child i_j, $p_{i_j} < (p-2)/2$. For each $1 \leq m \leq \ell$, we consider the subtree $T'_m = (V'_m, E'_m)$ rooted at i and containing all the children $i_1, \ldots, i_m$ and their subtrees rooted at them. Then, it holds that $|V'_m| = 1 + \sum_{j=1}^{m} p_{i_j}$. We have $|V'_1| < p/2$ and $|V'_\ell| = p_i \geq p$. Therefore, there exists some m such that $|V'_m| < p/2$ and $|V'_{m+1}| \geq p/2$; furthermore $|V'_{m+1}| = |V'_m| + p_{i_{m+1}} \leq p/2 + p/2 = p$, as $p_{i_j} < p/2$ for each i_j. So, T'_{m+1} is an eligible tree satisfying the requirements. ∎

We are, now, ready to present our approximation algorithm. Algorithm 4 starts with the directed tree T and, gradually, packs vertices of eligible subtrees into bins and removes vertices whose incident edges have been covered until a feasible solution is produced. More specifically,

Algorithm 4 (U-BPCP, trees)

1: Obtain a directed tree T by orienting the edges of G
2: $k = 1$, $f_k = q$
3: **while** $E_T \neq \emptyset$ **do**
4: **if** $|V_T| \leq q$ **then**
5: Pack V_T in B_k and set $T = \emptyset$
6: **if** $|V_T| > q$ **then**
7: **while** $f_k > 1$ **do**
8: Compute an eligible subtree T' such that $f_k/2 \leq |V_{T'}| \leq f_k$
9: Pack $V_{T'}$ in bin B_k, remove T' from T and let $f_k = f_k - |V_{T'}|$
10: $k = k + 1$, $f_k = q$
11: **return** the solution found.

it consists of phases, the bin B_k being filled up during the k-th phase. We will denote by T the current remaining tree (whose edges are not already packed) and by f_k the remaining available space of bin B_k during the execution of the k-th phase. At the beginning of phase k we consider the remaining tree T. If $|V_T| \leq q$, then we pack it entirely in the bin B_k and we get a feasible solution (lines 4-5). If $|V_T| > q$ then, thanks to Lemma 5, we can pack in B_k an eligible subtree T' such that $q/2 \leq |V_{T'}| \leq q$; while the available space is $f_k > 1$ we continue packing eligible subtrees T' such that $f_k/2 \leq |V_{T'}| \leq f_k$ (lines 6-8). At the end of each step we remove the subtree T' from T and update the available space to $f_k - |V_{T'}|$ (line 9). Then we start a new phase $k + 1$ with an available space q (line 10).

The number of steps of a phase is at most $\lfloor \log(q) \rfloor$ because the algorithm fills at least half of the empty bin capacity in each step. At the end of a phase (except perhaps the last phase) a bin contains at least $q - 1$ vertices. Therefore, the number of edges in a bin (except perhaps the last one) is at least $q - 1 - \lfloor \log q \rfloor$. This result enables us to get the approximation ratio of the next theorem.

Theorem 11 *Algorithm 4 achieves asymptotically an approximation ratio of* $(1 + \epsilon)$*, where* $\epsilon = \frac{\log q}{q-1-\log q}$*, for U-BPCP on trees.*

Proof. Since the input graph $G = (V, E)$ is a tree, there are $|E| = n - 1$ edges to be covered in total. Moreover, in any feasible solution, each bin contains at most q vertices and so at most $q - 1$ edges. Therefore, a lower bound on the number of bins used by the optimal solution is $b^* \geq \frac{n-1}{q-1}$.

On the other hand, in the solution found by Algorithm 4, each bin B_k contains at least $q-1-\log q$ edges, except perhaps the last one which contains all the edges of the remaining tree. Therefore, the number of bins used by the algorithm is

$$b \leq \left\lceil \frac{n-1}{q-1-\log q} \right\rceil \leq \frac{n-1}{q-1-\log q} + 1 = \frac{q-1}{q-1-\log q} \cdot \frac{n-1}{q-1} + 1 \leq \left(1 + \frac{\log q}{q-1-\log q}\right) b^* + 1$$

■

6 General Graphs

In this section we deal with 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 achieves a ratio of $3 +$

$2\lceil mad(G)/2\rceil$, where $mad(G) = \max_{H\subseteq G}\{2|E(H)|/|V(H)|\}$ is the maximum average degree of G, while the second one achieves a ratio of $\rho(\Delta+1)$, where Δ is the maximum degree of G, given a ρ-approximation algorithm for the Bin Packing problem. Then, we move to U-BPCP and we present an approximation algorithm of ratio $\rho \cdot O(\log n)$, given a ρ-approximation algorithm for Densest q-Subgraph problem.

6.1 Weighted case

For the first of our approximation algorithms we consider an arbitrary orientation $D = (V, \overrightarrow{E})$ of the input graph $G = (V, E)$. For each vertex $i \in V$ we consider the vertices in its in- and out-neighborhoods in D, $\Gamma^-(i)$ and $\Gamma^+(i)$, respectively. We denote by $d^-(i) = |\Gamma^-(i)|$ and $d^+(i) = |\Gamma^+(i)|$ the in- and out-degrees, respectively, of vertex i in D. Let also $\Delta^-(D) = \max_{i\in V}\{d^-(i)\}$ be the maximum indegree of D.

The first step of the algorithm consists in packing independently every node $i \in V$ together with all the vertices in its out-neighborhood $\Gamma^+(i)$. That is, for each $i \in V$, all the vertices in $\Gamma^+(i)$ are packed into bins of capacity $q - w_i$ according to the First Fit Decreasing (FFD) Bin Packing algorithm and the content of every such bin together with vertex i is considered as one group. Then, we consider these groups as an instance of the Bin Packing problem and we pack them using a standard Bin Packing algorithm (NF, FF, FFD) into bins of capacity q.

Algorithm 5 (BPCP, General graph $G = (V, E)$)

1: Construct an orientation D of the input graph G.
2: **for** each $i \in V$ with $d^+(i) > 0$ **do**
3: Pack the vertices in $\Gamma^+(i)$ into b_i bins of capacity $q - w_i$ using the FFD algorithm.
4: **for** each bin $B_k, 1 \le k \le b_i$, used by FFD algorithm in Step 3 **do**
5: Create a group of the vertices $V_k \subseteq V$ packed in B_k and vertex i, of weight $w_i + \sum_{j\in V_k} w_j$.
6: Pack all created groups into bins of capacity q by a Bin Packing algorithm (NF, FF or FFD).

Lemma 6 *Algorithm 5 achieves a $3 + 2\Delta^-(D)$ approximation ratio for BPCP on general graphs, where $\Delta^-(D)$ is the maximum in-degree of an orientation D of the input graph G.*

Proof. Initially, we observe that Algorithm 5 produces a feasible solution. Indeed, every group created has a total weight at most q, and for every edge $(i, j) \in E$ its endpoints appear together in one of the groups. In order to analyze the algorithm, we need to define a parameter that will serve for lower bounding the optimal solution. For each vertex $i \in V$ with $|\Gamma^+(i)| > 0$, we define

$$n_i^* = \left\lceil \frac{1}{q - w_i} \sum_{j\in\Gamma^+(i)} w_j \right\rceil$$

Note that n_i^* is the minimum number of bins required for packing vertex i together with each one of its outneighbours while ignoring the remaining vertices. Clearly, n_i^* is a lower bound on the number of occurrences of vertex i in an optimal solution S^*. That is, the total packed weight in S^* is at least $\sum_{i\in V} n_i^* w_i$ which implies that

$$b^* \ge \frac{1}{q} \sum_{i\in V} n_i^* w_i$$

Now, let S be the algorithm's solution using b bins, and consider the occurrences of a vertex $i \in V$. Due to its packing with all of its out-neighbours it will appear in each of the b_i bins used by the FFD algorithm in Step 2 of the algorithm. By Theorem 1(i)(a) it follows that $b_i \leq \frac{3}{2}n_i^*$. Moreover, a vertex $i \in V$ appears in $d^-(i) \leq \Delta^-(D)$ bins due to its packing with all of its in-neighbours. Therefore, a vertex $i \in V$ appears in at most $n_i \leq \frac{3}{2}n_i^* + \Delta^-(D)$ bins and a total weight of $\sum_{i \in V}(\frac{3}{2}n_i^* + \Delta^-(D))w_i$ is packed into b bins by the algorithm.

Hence, by Theorem 1(i)(b) we conclude that

$$b \leq 2\left\lceil \frac{1}{q}\sum_{i \in V}\left(\frac{3}{2}n_i^* + \Delta^-(D)\right)w_i \right\rceil \leq \left(3 + 2\Delta^-(D)\right)\left\lceil \frac{1}{q}\sum_{i \in V} n_i^* w_i \right\rceil \leq \left(3 + 2\Delta^-(D)\right) \cdot b^*$$

■

By Lemma 6, the approximation ratio of BPCP on general graphs depends on the orientation D of the input graph G. Fortunately, it is known that the orientations of a graph G are tightly related to its density as it is captured by the measure of *maximum average degree* of G which is denoted by $mad(G)$. The relation between orientations of a graph G and $mad(G)$ is best illustrated by the following result of Hakimi [8] or [12, Corollary 61.1b].

Lemma 7 *Any graph G has an orientation D with maximum outdegree $\Delta^-(D) \leq \lceil mad(G)/2 \rceil$ (which can be computed in polynomial time).*

As a consequence of this result and Lemma 6 we get the next theorem.

Theorem 12 *Algorithm 5 achieves a $3 + 2\lceil mad(G)/2 \rceil$ approximation ratio for BPCP on general graphs, where $mad(G)$ is the maximum average degree of the input graph G.*

One is easily convinced that $mad(G) < 2$ whenever G is a tree (or forest). It follows also from Euler's formula that $mad(G) < 6$ whenever G is a planar graph and, hence the following corollary of Theorem 12 follows. This corollary actually applies to graphs that can be embedded on the torus, or more generally to any class of H-minor-free graphs or minor-closed graphs as they also have a bounded maximum average degree (see [14, 9]).

Corollary 1 *Algorithm 5 achieves a 5-approximation ratio for BPCP on trees, and a 9-approximation ratio for BPCP on planar graphs.*

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. Specifically, for each edge $(i, j) \in E$, we create an item $e_{i,j}$ of weight $w_i + w_j$ and we run the algorithm $\mathcal{A}$ on the instance (E, q) of the Bin Packing problem.

Algorithm 6 (BPCP, general graph $G = (V, E)$)

1: For each edge $(i, j) \in E$, create an item $e_{i,j}$ of weight $w_i + w_j$.
2: Pack the set E of items into bins of capacity q using a ρ-approximation algorithm $\mathcal{A}$ for the Bin Packing problem.

Initially, we obtain a lower bound by packing the edges of the input graph G, instead of its vertices, into bins of capacity q.

Lemma 8 *Let b^* and b_e^* be the optimal numbers of bins for BPCP on a general graph G and the Bin Packing problem (E, q), respectively. Then, it holds that $b_e^* \leq (\Delta + 1) \cdot b^*$, where Δ is the maximum degree of the input graph G.*

Proof. Consider an optimal solution S^* of BPCP on G using b^* bins. Starting from S^*, we construct a feasible solution S', for the Bin Packing problem (E, q), as follows. We use a $\Delta + 1$ edge coloring that we know always exists (see any text book in graph theory). This coloring is used only for the sake of the analysis and not by Algorithm 6. For each bin B_k of S^*, $1 \leq k \leq b^*$, we create $\Delta + 1$ new bins $B_k^1, \ldots, B_k^{\Delta+1}$ in S'. Bin B_k^c contains all edges with color c whose end vertices are both included in B_k. Each pair of colocated vertices appears in at least one bin B_k and so each edge appears in bin B_k^c if it is of color c. Because all edges of the same color form a matching (i.e they consist of disjoint vertices), the weight of the edges in B_k^c is at most the sum of the vertex weights in B_k and, thus, not greater than q. Therefore, we have constructed a feasible solution of the Bin Packing problem (E, q), with at most $(\Delta+1) \cdot b^*$ bins and the lemma follows. ∎

Then, Lemma 8 implies the next theorem.

Theorem 13 *Algorithm 6 achieves an approximation ratio of $\rho \cdot (\Delta + 1)$ for BPCP on general graphs, given a ρ-approximation algorithm for the Bin Packing problem.*

Proof. Clearly, any feasible packing for the instance (E, q) of the Bin Packing problem is a feasible solution for BPCP on the input graph G. So, Algorithm 6 returns a feasible packing for the instance (E, q) using $b_e \leq \rho \cdot b_e^*$ bins. By Lemma 8 it holds that $b_e^* \leq (\Delta + 1) \cdot b^*$ and the theorem follows. ∎

6.2 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 (DqS) problem. In the DqS problem, we are given a graph $G = (V, E)$ and we ask for a subset $V' \subseteq V$ of at most q vertices, i.e., $|V'| \leq q$, such that the number of edges in the subgraph $D = (V', E')$ induced by V' is maximized.

The idea is to use repeatedly the algorithm $\mathcal{A}$ and pack at each step the vertices of a DqS in the remainder graph (obtained by removing the already covered edges). The procedure goes on until all the edges of the input graph are covered, as in Algorithm 7 below.

Algorithm 7 (U-BPCP, general graph $G = (V, E)$)

1: $k = 1$; $G_1 = G$
2: **while** $E \neq \emptyset$ **do**
3: Run a ρ-approximation algorithm $\mathcal{A}$ for the DqS problem on the graph G_k and let $D_k = (V_k, E_k)$, $|V_k| \leq q$, be the subgraph returned by $\mathcal{A}$.
4: Pack the vertices of V_k into bin B_k.
5: $G_{k+1} = (V, E \setminus E_k)$. // Vertices of degree 0 are removed
6: $k = k + 1$
7: Return $b = k - 1$

Theorem 14 *Algorithm 7 achieves an approximation ratio of $\rho \cdot O(\log n)$ for U-BPCP on general graphs, given a ρ-approximation algorithm for Densest q-Subgraph problem.*

Proof. The algorithm packs the vertices of the input graph G into b bins, numbered $B_1, B_2, \ldots, B_b$, according to the order that they are used. In fact, in each bin B_k, $1 \leq k \leq b$, are packed the vertices V_k of the DqS of the graph G_k, i.e., the remainder of G just before using bin B_k. Moreover, the edges of the subgraph $D_k = (V_k, E_k)$, induced by V_k, have their both endpoints in bin B_k, and, therefore, the algorithm produces a feasible solution. By convention, for each edge $e = (i, j) \in E_k$, which has its both endpoints in bin B_k, we write $e \in B_k$.

We denote by $m_k = |E_k|$ the number of edges of the subgraph D_k derived by the Algorithm $\mathcal{A}$ and by m_k^* be the number of edges in the DqS of G_k. As $\mathcal{A}$ is a ρ-approximation algorithm for DqS problem it follows that $m_k \geq m_k^*/\rho$ (where $\rho > 1$).

We associate to every edge $e \in B_k$, $1 \leq k \leq b$, the value $v_e = \frac{1}{m_k}$. Note that $\sum_{e \in B_k} v_e = 1$ and $\sum_{e \in E} v_e = b$, where b the number of bins used by the algorithm. We also number the edges as $e_1, e_2, \ldots, e_m$, where $m = |E|$, according to the order they are covered, breaking ties arbitrarily. That is, for every pair of edges such that $e_i \in B_k$, $e_j \in B_{k'}$ and $1 \leq k < k' \leq b$, it holds that $i < j$.

Consider now an edge $e_\ell \in B_k$, $1 \leq \ell \leq m$, $1 \leq k \leq b$, and observe that, just before e_ℓ is covered, at least $m - \ell + 1$ edges remain in G_k. In an optimal solution of BPCP on the input graph G, all the edges of G are covered by b^* bins. Obviously, all the edges of G_k can be also covered by b^* bins. Hence, among these b^* bins covering the edges of G_k there must be some bin covering at least $(m - \ell + 1)/b^*$ of them. In this bin are packed at most q vertices of G_k, which induce a subgraph of G_k. On the other hand, the DqS of G_k contains the maximum possible number of edges of any subgraph of G_k of at most q vertices and, therefore, $m_k^* \geq (m - \ell + 1)/b^*$.

Thus we have $v_{e_\ell} = \dfrac{1}{m_k} \leq \dfrac{\rho}{m_k^*} \leq \rho \cdot \dfrac{b^*}{(m - \ell + 1)}$, and

$$b = \sum_{\ell=1}^{m} v_{e_\ell} \leq \rho \cdot b^* \cdot \sum_{\ell=1}^{m} \frac{1}{m - \ell + 1} \leq \rho \cdot b^* \cdot \sum_{\ell=1}^{m} \frac{1}{\ell} \leq \rho \cdot O(\log n) \cdot b^*$$

■

The best known approximation algorithm for the DqS problem is proposed in [4] and its ratio is $O(n^{1/4})$. Therefore, Theorem 14 implies the next corollary.

Corollary 2 *There exists a $O(n^{1/4} \cdot \log n)$-approximation algorithm for U-BPCP on general unweighted graphs.*

7 Conclusions

In this paper we initiate the study of the complexity and approximability of BPCP with respect to the class of the input graph. Our findings indicate that BPCP admits efficient constant factor approximation algorithms when the input graph is sparse, i.e., it has low maximum average degree or low maximum degree, like paths, trees or planar graphs. We also propose improved constant factor approximation algorithms for BPCP and U-BPCP on complete graphs, by exploring the relation of the problem with the theory of combinatorial designs.

The major open question is the approximability of the problem on dense and general graphs. A potential direction towards answering this question is the intuitive relation of U-BPCP with the Densest q-Subgraph problem. It is known that the latter problem admits a PTAS for dense graphs and it is interesting to study if this is also the case for U-BPCP on dense graphs. It is also conjectured that the Densest q-Subgraph problem does not admit a constant factor approximation algorithm for general graphs. As a first step, we show that a ρ-appoximation algorithm for the

Densest q-Subgraph problem can be converted into a $\rho \cdot O(\log n)$-approximation algorithm for U-BPCP. It remains to verify whether one can accomplish the opposite direction as well.

References

[1] F. Afrati, S. Dolev, E. Korach, S. Sharma, and J. D. Ullman. Assignment problems of different-sized inputs in mapreduce. *ACM Transactions on Knowledge Discovery from Data*, 11(2):18:1–18:35, Dec. 2016.

[2] 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.

[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 Journal on Discrete Mathematics*, 9(3):492–509, 1996.

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

[9] A. V. Kostochka. Lower bound of the Hadwiger number of graphs by their average degree. *Combinatorica*, 4(4):307–316, 1984.

[10] 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.

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

[12] A. Schrijver. *Combinatorial optimization. Polyhedra and efficiency. Vol. B*, volume 24 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2003. Matroids, trees, stable sets, Chapters 39–69.

[13] 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.

[14] A. Thomason. An extremal function for contractions of graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 95(2):261–265, 1984.

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