Noname manuscript No.
(will be inserted by the editor)

New Approximation Results for Resource Replication Problems*

Samir Khuller · Barna Saha · Kanthi K. Sarpatwar

the date of receipt and acceptance should be inserted later

Abstract In the *Basic Resource Replication* problem, we are given a graph, embedded into a distance metric, and a set of data items. The goal is to assign one data item to each vertex so as to minimize the maximum distance any vertex has to travel to access all the data items. We consider several variants of this problem in this paper, and propose new approximation results for them. These problems are of fundamental interest in the areas of P2P networks, sensor networks and ad hoc networks, where placement of replicas is the main bottleneck on performance. We observe that the threshold graph technique, which has been applied to several k-center type problems, yields simple and efficient approximation algorithms for resource replication problems. Our results range from positive (efficient, small constant factor, approximation algorithms) to extremely negative (impossibility of existence of any algorithm with non-trivial approximation guarantee, i.e., with positive approximation ratio) for different versions of the problem.

* An extended abstract of this paper appeared in the 15th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2012

S. Khuller: Supported by NSF Awards CCF-0728839 and CCF-0937865, and a Google Research Award.

K.K. Sarpatwar: Supported by NSF Grant CCF-0728839.

S. Khuller
Department of Computer Science
University of Maryland (College Park)
E-mail: samir@cs.umd.edu

B. Saha
AT&T Shannon Research Laboratory
E-mail: barna@research.att.com

K. K. Sarpatwar
Department of Computer Science
University of Maryland (College Park)
E-mail: kasarpa@cs.umd.edu

1 Introduction

Problems related to data placement and replication are of fundamental interest both in the area of large scale distributed networking systems as well as centralized storage systems. The performance of distributed systems such as P2P file sharing systems, wireless ad hoc networks, sensor networks etc., where resources are shared among clients, can be significantly impacted by placement of the replicated resources [17, 18, 2]. On the other hand, centralized storage systems, such as in Netflix, might have data distributed across different data centers so that it is necessary to keep data closer to the demand to prevent over loading the network. Demand patterns for data can also vary widely, especially in the context of video on demand distribution.

There is a lot of research on centralized storage systems [10] that addresses the problem of data layout when all the storage units are centrally located in a single location and thus the "distance" of each client from any storage unit is the same. However, in modern storage management systems, this assumption is not valid. Internet content providers rent storage space all over the world from different data centers in different locations. Since most interesting objective functions are NP-hard, it is of interest to consider efficient approximation algorithms.

The basic framework is the following: given a collection of k data items, we wish to distribute the k data items to a collection of n nodes modeled by a graph, where the vertices are embedded in a metric space. In the basic model, each node wishes to access each of the k data items and the goal is to minimize the *maximum* distance any node has to travel to access all k items. For this problem, Ko and Rubenstein [17] give a distributed algorithm based on a local search idea and also show that this algorithm delivers a solution with a worst case approximation guarantee of 3. We note that the algorithm is not guaranteed to run in polynomial time, however, in practice its convergence is reasonably quick. In a followup piece of work [18], Ko and Rubenstein introduced a generalization of the basic problem in which each node only required a *subset* of the items. For this problem, they develop a heuristic; however, for this heuristic, unlike the other case, there is no approximation guarantee any more.

In this paper we consider both the questions described above, along with several other generalizations and provide polynomial time approximation algorithms for them. In particular we develop a simple algorithm with a 3-approximation for the basic model, and this can be implemented in a distributed setting. We also develop a more involved centralized 3-approximation scheme for the general problem. However, we do not know how to implement this algorithm in a distributed setting as yet. In addition, we consider further generalizations where we need to provide excellent service to a given fraction of the clients and not all the clients. This is motivated from the fact that there may be a few outliers, and it may be extremely costly to provide all data items to the outliers. Here, the two problems deviate in difficulty immediately. For the basic problem we can still provide a constant approximation, but for the

general problem, somewhat surprisingly, it turns out that, assuming $P \neq NP$, there is no polynomial time algorithm with any non-trivial approximation guarantee. We give a polynomial time reduction of the *maximum k-clique* problem to the feasibility version of the general problem.

Following the works of Ko and Rubenstein [17,18], in this paper, we consider the "min-max" objective function for data placement problems. A different objective function of minimizing average data-access cost was studied by Baev et al. [1,2] under the assumption that each client only requires a particular data item. A generalization of this problem with load and capacity constraints on servers was considered by Guha et al. [12] and Meyerson et al. [22] (called the *page-placement* problem). They developed bicriteria approximation algorithms for this problem where load and/or capacity are violated by a small factor.

Our Contributions. The following is a summary of our results.

- In Section 2, we consider the basic replication problem where each client needs all k data items (*basic resource replication*) and its generalization where each client might need a subset of data items (*subset resource replication*). For the first problem, we give a distributed polynomial time 3-approximation algorithm and show that there does not exist any polynomial time algorithm achieving a $2-\epsilon$ (for any $\epsilon > 0$) approximation (Theorem 1 and Theorem 6). For the later, we give the first polynomial time 3-approximation algorithm (in a centralized setting) along with matching hardness (Theorem 5 and Theorem 6).
- In Section 3, we consider the outlier version of the basic as well as subset resource replication problem. For the former, we give a polynomial time 3-approximation algorithm while for the latter, somewhat surprisingly, we show that there does not exist any non-trivial approximation guarantee (in polynomial time). We also consider the case where each resource can be replicated at most K times and give polynomial time 5-approximation algorithm for it.
- In Section 4, we consider another natural generalization of the basic resource replication problem where each node has an upper bound (load) on the number of clients it can serve. We give a polynomial time 4-approximation algorithm for this version when load $L \geq 2k-1$ (k is the number of resources). A simple counting argument shows that this problem is infeasible if $L < k$. This implies our 4-approximation algorithm is a bicriteria approximation algorithm and the load capacity is not violated by more than a factor of 2.

2 Resource Replication Problem

2.1 Basic Resource Replication Problem

The following problem, which we call the *Basic Resource Replication* (BRR) problem, was first studied by Ko and Rubenstein [17]. The input consists of:

– set of nodes or vertices, $V = \{v_1, v_2 \dots v_n\}$
– a metric space defined by the function $d : V \times V \to \mathbb{R}^+ \cup \{0\}$
– set of resources or colors $\mathcal{C} = \{C_1, C_2, C_3, \dots, C_k\}, k \leq n$.

We seek to find an *optimal* mapping $\phi : V \to \mathcal{C}$ of colors to vertices. The objective function for optimality is defined in the following way. Define $d_r(v)$ to be the shortest distance between a vertex assigned the color C_r[1] and the vertex v. The goal of the *Basic Resource Replication* (BRR) problem is the following -

$$\min_{\phi} \quad \max_{\substack{v \in V \\ C_r \in \mathcal{C}}} \quad d_r(v).$$

This is the central problem of the work of Ko and Rubenstein [17] who give a distributed algorithm with a 3-approximation guarantee. Unfortunately, their algorithm has no proven polynomial running time bound. We give a simple distributed polynomial time 3-approximation algorithm for this problem. All the algorithms in this work use a technique called *threshold graph construction* introduced by Edmonds and Fulkerson [7] and used extensively for k-center type problems [11,16,15,14]. We observe that the use of this approach enables the design of very simple and efficient algorithms for several resource replication problems. Given $\delta \in \mathbb{R}^+ \cup \{0\}$, the *threshold graph*, denoted by G_δ, is constructed by adding edges between every pair of vertices u, v which are at distance at most δ. Our distributed algorithm (Algorithm 1) for BRR works in the following way. In the first step, each vertex v, determines the distance of the $(k-1)^{th}$ closest neighbor - call this $l_{k-1}(v)$. Now in a distributed fashion each vertex obtains the maximum value $\delta_L = \max_v l_{k-1}(v)$. We observe that the threshold graph G_{δ_L} has minimum degree at least $k-1$. Let δ_{OPT} be the minimum value of δ for which a feasible solution exists. δ_L must be a lower bound on this optimal δ value (δ_{OPT}) - because δ_L is the least value such that the threshold graph has degree at least $k-1$ and $G_{\delta_{OPT}}$ has minimum degree at least $k-1$. We set $\delta = \delta_L$, and construct the graph G_δ^2 which is the graph formed by squaring G_δ. In other words, each vertex v maintains a list of all vertices within two hops in G_δ as its neighbors. Using standard distributed algorithms (see for e.g., [21]), we compute a maximal independent set $\mathcal{I}$ in G_δ^2. Finally, each vertex in $\mathcal{I}$, colors itself with C_1 and picks $k-1$ vertices from its list of neighbors in G_δ ($N_{G_\delta}(v)$) and assigns them a distinct color from the remaining $k-1$ colors.

[1] We may abuse the notation and use same expression, $d_r(v)$, when r represents a color.

Algorithm 1 Distributed 3-approximation algorithm for BRR

1: Choosing $\delta = \delta_L$, where δ_L is the smallest value such that G_{δ_L} has minimum degree $\geq k-1$.
2: Each node maintains a list of its neighbors in G^2_δ.
3: Compute a maximal independent set $\mathcal{I}$ in a distributed fashion [21].
4: **for** $v \in \mathcal{I}$ **do**
5: Color v with C_1, and arbitrarily pick $(k-1)$ vertices from the set $N_{G_\delta}(v)$ say $\{v'_1, v'_2, \ldots v'_{k-1}\}$ and colors them with $C_2 \ldots C_k$ respectively.
6: **end for**
7: Assign arbitrary colors to vertices which have not received any color so far.

We illustrate Algorithm 1 by a simple example, as shown in Figure 1.

Theorem 1 *Algorithm 1 gives a 3-approximation for the BRR problem.*

Proof We prove that for every vertex v and every color r, $d_r(v) \leq 3 \times \delta_L$. Since $\delta_L \leq \delta_{OPT}$, the result follows. If $v \in \mathcal{I}$, by construction $d_r(v) \leq \delta_L$. For vertices v, which are adjacent to some vertex (i.e., one hop distance) of $\mathcal{I}$ in G_δ, $d_r(v) \leq 2 \times \delta_L$ and for vertices at two hop distance from $\mathcal{I}$, $d_r(v) \leq 3 \times \delta_L$. There are no vertices at ≥ 3 hop distance from $\mathcal{I}$, since the latter is a maximal independent set in $G^2_{\delta_L}$. □

2.1.1 Generalizations of Basic Resource Replication Problem

We first consider the following generalization of BRR- each color C_i has a bound $K \in \mathbb{N}$, which is the number of copies of C_i that can be used. This problem is also a natural generalization of the k-center problem (where there is a single resource with bound k). We note that a simple modification to Algorithm 1 solves this generalized version of BRR with capacity bound on colors. In fact the only difference between the algorithms for BRR and this version is how we choose δ (and we do not follow step 7). For this case, we must try out all the values of optimal δ (there are at most $O(n^2)$ such values) and choose the smallest δ which satisfies the following two properties -

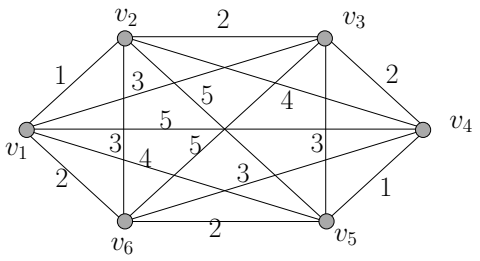
1. Each vertex of G_δ has degree $\geq k-1$.
2. the computed maximal independent set in G^2_δ has size at most K.

Clearly, this gives a feasible solution for the problem, as we follow steps 4-6 of Algorithm 1 to assign color.

Lemma 1 *The minimum value of δ, δ_{OPT}, for the generalized BRR with capacity bound of C for each color must satisfy*

1. *Each vertex of $G_{\delta_{OPT}}$ has degree $\geq k-1$.*
2. *Any maximal independent set in $G^2_{\delta_{OPT}}$ has size at most K.*

Proof The first condition is obvious. To see the second condition, suppose there exists a maximal independent set $\{u_1, u_2, ..u_L\}$ in $G^2_{\delta_{OPT}}$ whose size $L > K$. Then, in $G_{\delta_{OPT}}$, it is not possible to satisfy all $u_1, u_2, ..u_L$ by at most K copies of a single color, because then there exists at least a pair of vertices u_i and u_j, $i, j \in [1, L]$, within at most two hops. □

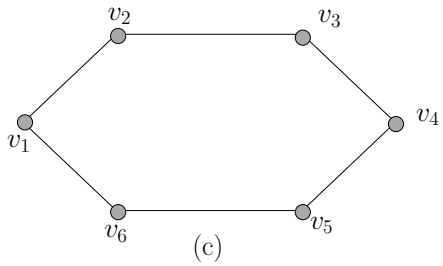


(a)

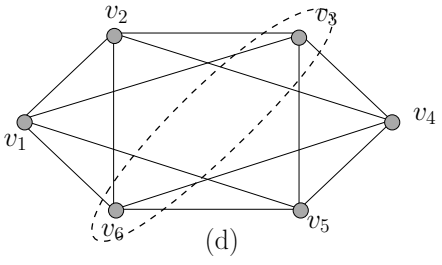
Vertex	$\delta_{\mathbf{k-1}}$
v_1	2
v_2	2
v_3	2
v_4	2
v_5	2
v_6	2

$\delta_L = 2$

(b)



(c)



(d)

Vertex	**Color**
v_1	C_3
v_2	C_2
v_3	C_1
v_4	C_3
v_5	C_2
v_6	C_1

(e)

Fig. 1 An illustrative example for the BRR algorithm: (a) Given a BRR instance with $k = 3$ and the edge weights representing pairwise distances. (b) Computation of δ_{k-1} values for various vertices. The threshold δ_L is set to the maximum of these values. (c) Threshold graph, G_{δ_L}: Delete all edges with weights greater than δ_L and keep the rest. (d) Compute $G^2_{\delta_L}$ - if u, v are within two hops of each other in G_{δ_L}, then we add an edge between them in $G^2_{\delta_L}$. Find a maximal independent set $\mathcal{I}$, for example, $\mathcal{I} = \{v_3, v_6\}$ in this case. (e) Each member v of $\mathcal{I}$, assigns itself the color C_1 and colors its neighbors (in G_{δ_L}) arbitrarily using the remaining $k - 1$ colors, using every color at least once.

The above lemma guarantees that the computed $\delta \leq \delta_{OPT}$. Now a 3-approximation guarantee follows immediately by an analogous argument to Theorem 1. Formally, we have the following theorem.

Theorem 2 *There is a polynomial time 3-approximation algorithm for the generalized BRR problem.*

Consider a further generalization of the BRR problem. Apart from the input given to the BRR instance, we are provided with placement cost of each resource j on a vertex i, c_{ij}. There can be two possible definitions of the weighted version of the problem (abbreviated as WBRR1 and WBRR2 respectively) -

1. WBRR1: Given a budget B, solve the BRR problem such that the total (sum of) cost of placement of various resources on the vertices must not exceed B.
2. WBRR2. Given budgets for each resource B_r, solve the BRR problem such that total cost of placement associated with each resource C_r does not exceed B_r.

We show that for the first version of the problem, we can easily extend Algorithm 1 to obtain a 3-approximation. This result generalizes the 3-approximation algorithm for weighted k-center problem [24].

Algorithm 2 3-approximation algorithm for weighted BRR

Let $\mathcal{D}$ be the list of possible δ values, i.e., the list of pairwise distances between the vertices of G, arranged in the non-decreasing order.
for all $\delta \in \mathcal{D}$ such that the minimum degree in G_δ is $k-1$ **do**
    Construct G_δ and G_δ^2.
    Compute a maximal independent set $\mathcal{I}$ in G_δ^2.
    for $v \in \mathcal{I}$ **do**
        Construct the complete bipartite graph, with neighbors of v (including itself) in G_δ on left hand side and k vertices (representing k slots for colors) on the other side. The edge weight on the edge incident on a vertex i and a color j is c_{ij}.
        Compute a minimum weight matching saturating all the k nodes on the right on this graph and assign colors to their matched vertices on the left.
    end for
    if total weight to all matchings (corresponding to all $v \in \mathcal{I}$) is at most B **then**
        return Current assignment of colors to vertices and exit.
    end if
end for
if no solution has been returned so far **then**
    return ϕ indicating no feasible solution exists.
end if

Theorem 3 *Algorithm 2 gives a 3-approximation for the WBRR1 problem.*

Proof If we guess the optimal δ, then for each $v \in \mathcal{I}$ has at least $k-1$ neighbors, so a minimum weighted matching saturating all the k nodes for k colors on the right can be obtained. This ensures that v must have all k colors in its neighborhood and we are opening up the cheapest possible ones. Thus for optimal δ, this should not exceed the budget B. We are returning a solution with minimum value of δ for which the total cost is at most B; hence this value serves as a lower bound to the optimum δ. Also, clearly the distance each vertex goes to get required colors is at most 3δ. □

We now observe that there is no constant approximation for the second version of Weighted Basic Resource Replication (WBRR2) problem. We reduce the

classic NP-hard problem of Subset Sum [9] to an instance of this problem. In an instance of Subset Sum problem, we are given a set of elements S, with each element $e \in S$ having a weight w_e and an real number bound B. The goal is to compute a subset S' of S such that the total sum of weights of elements in S' is exactly B.

Theorem 4 *Assuming $P \neq NP$, there is no polynomial time constant approximation algorithm for the WBRR2 problem.*

Proof Suppose there is a constant $c > 0$ approximation algorithm, denoted by A, for the WBRR2 problem. Given an instance of the Subset Sum problem, $I = (S, B)$, we construct the following instance of WBRR2 $I' = (G, \mathcal{C} = \{C_1, C_2\}, \mathcal{B} = \{B_1, B_2\})$. G is of a collection of independent edges, one for each element of S. The distance between any vertices on two different edges is $\geq c + 1$ and that between end points of the same edge is 1. There are two colors in the instance C_1, C_2 and every vertex requires both of them. We mark one vertex on each edge as positive and other vertex as zero - the placement cost of either colors on the positive vertex is the weight of the corresponding element in the set S and the placement costs on the zero vertices is 0. We now place the bounds $B_1 = B$ and $B_2 = (\sum_{e \in S} w_e) - B$, where w_e is the weight of element e. It is easy to observe that I is a YES instance if and only if the c-approximation algorithm A returns the value 1 as the solution. □

2.2 Subset Resource Replication Problem

In the BRR model each client requires all the data items. But in general each client might be interested in a subset of resources instead of all the resources. The servers might also have capacity to hold several data items. This substantially more generalized version of resource replication problem, which we call *Subset Resource Replication* Problem (SRR) was considered by Ko and Rubenstein in a subsequent paper [18]. Formally, the problem has the following input

- a set of vertices $V = \{v_1, v_2 \ldots v_n\}$, a metric $d : V \times V \to \mathbb{R}^+ \cup \{0\}$ and a set of colors $\mathcal{C} = \{C_1, C_2 \ldots C_k\}$.
- every vertex $v \in V$ has a subset $\mathcal{C}_v \subseteq \mathcal{C}$ of "required" colors and a non-negative integer s_v as the storage capacity - that is we can assign s_v colors to vertex v.

The goal is to assign a list of colors $\phi(v) \subseteq \mathcal{C}$ to each vertex v, such that $|\phi(v)| \leq s_v$, with the following objective -

$$\delta = \min_{\phi} \max_{\substack{v \in V \\ r \in \mathcal{C}_v}} d_r(v)$$

where $d_r(v)$ is the shortest distance from v to a vertex having C_r on its list of colors. Ko and Rubenstein [18] extended their basic approach to this problem

but had no guarantee on either the approximation ratio or the running time. We give the first centralized polynomial time 3-approximation algorithm (Algorithm 3) for the problem. Later, in Theorem 6, we will prove that this is the best possible approximation one can expect, assuming $P \neq NP$.

We again use the threshold graph technique. The optimal distance δ has to be the distance between one of the $O(n^2)$ pairs of vertices. Hence, it has only polynomial number of possible values and we can assume that the value of δ is known (trying out all possible values of δ will only add a polynomial factor). Assuming δ is known, we construct the threshold graph G_δ. We now square the graph G_δ to obtain G_δ^2, i.e., add an edge between two vertices $u, v \in V$ if they are at a distance at most two in G_δ. Consider a color r and let $H_r \subseteq G_\delta^2$ be the induced subgraph on vertices that need color r (among possibly other colors). Let $\mathcal{I}_r$ be a maximal independent set in the subgraph H_r. The following is a key observation about an optimal solution.

Observation *For every vertex $v \in \mathcal{I}_r$, the optimal solution must assign a unique copy of r in the neighborhood of v in G_δ.* (†)

Indeed, in G_δ the neighborhoods corresponding to vertices in $\mathcal{I}_r$ must be mutually disjoint. If neighborhoods corresponding to vertices $u, v \in \mathcal{I}_r$ intersect, then there must exist an edge between u, v in G_δ^2 - which is impossible, as $\mathcal{I}_r$ forms an independent set in this graph. Since, every vertex must be satisfied by some copy in its neighborhood in G_δ, our observation holds. If for every vertex $v \in \mathcal{I}_r$, $d_r(v) \leq \delta$ then every vertex $u \in H_r$ has $d_r(u) \leq 3 \times \delta$. Thus to find a 3-approximation, we focus on satisfying vertices of such independent sets $\mathcal{I}_r$, for each color $r \in \mathcal{C}$. We cast this as a b-matching problem [6] on the graph $B = (X, Y)$ - where X is the union of independent sets $\mathcal{I}_r$, $\forall r \in \mathcal{C}$ (i.e., if a vertex belongs to s independent sets of the form I_r, we add s copies of the vertex to X) and Y is a copy of V with $b-$matching bounds s_v on each vertex $v \in V$. We add an edge across the partitions, if its end points are at distance at most δ from each other. From observation (†), there must exist a b-matching that saturates all the vertices of X. Figure 2 explains the construction for a simple instance.

Theorem 5 *Algorithm 3 is a 3-approximation for the Subset Resource Replication problem.*

Proof We start by proving that (assuming δ is the optimal solution), the maximum b-matching, found in step 9 of Algorithm 3, completely saturates X. It is sufficient to show that there exists of b-matching which saturates X (which implies the maximum matching also does so). In the optimal coloring, which satisfies every vertex within distance δ, let L_v^{opt} denote the list of colors placed on $v \in V$ (for feasibility, $|L_v^{opt}| \leq s_v$, where s_v is the storage capacity of v). For a color i and a vertex v, we denote the copy of v in $\mathcal{I}_i$ by v_i. We note that for every v requiring a color i, there exists a vertex $u \in Y$ which is within distance δ of v and has i in its list of colors L_u^{opt}. We now claim that the following edge set forms a b-matching which saturates X. The edge set, denoted by bM, consists one edge for each $v_i \in X$, namely $\overline{v_i u}$, where u is some vertex

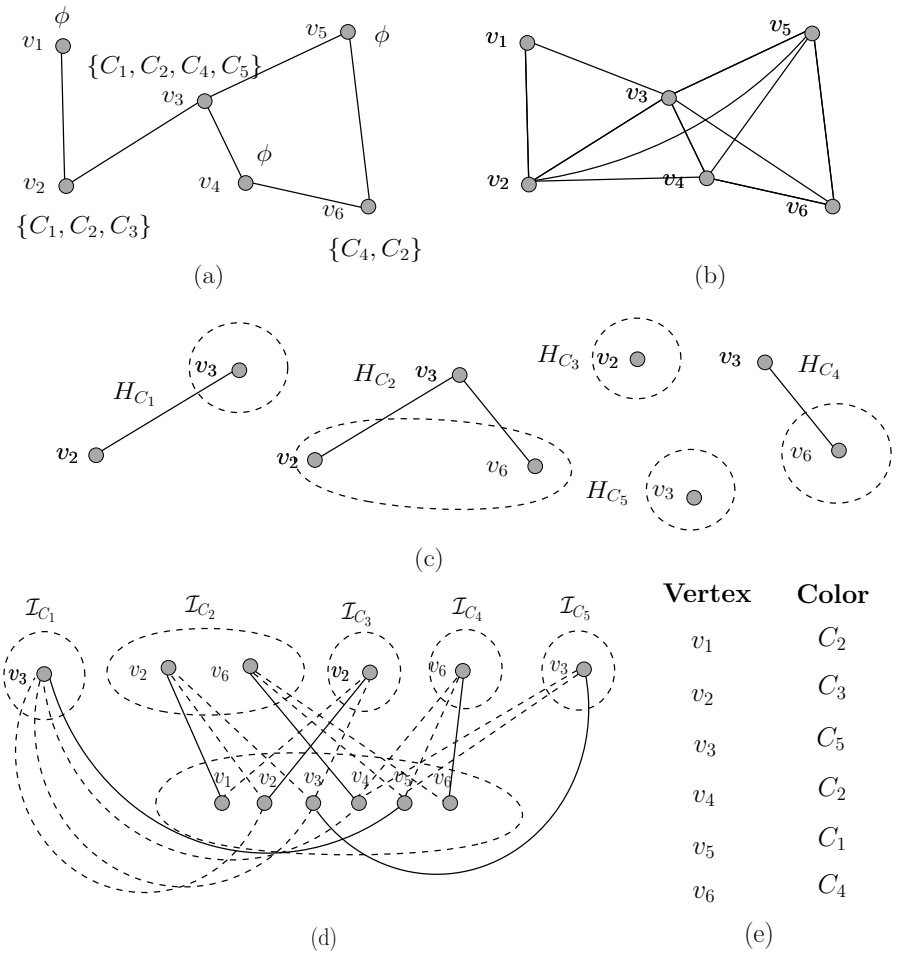


Fig. 2 Illustrative example for the SRR algorithm: (a) Guess δ and construct the threshold graph, G_δ. The required color set for each vertex is shown next to it. (b) Construction of G_δ^2. (c) For each color C_i, H_{C_i} is the subgraph of G_δ^2 over vertices that require C_i. The "dashed" circles represent maximal independent set $\mathcal{I}_{C_i}$ in each H_{C_i}. (d) Bipartite graph construction: On one side we have copies of vertices in $\mathcal{I}_{C_i}$ and on the other side we have all the vertices in G_δ. We add edges ("dashed lines") between copies of two vertices, if they had an edge in the threshold graph G_δ. We then compute a B-matching on this graph. The solid lines represents a valid B-matching (where each vertex has a capacity of 1). (e) Using the B-matching, we compute the final color assignments for each vertex.

within distance δ of v_i such that $i \in L_u^{opt}$. We only have to show that bM is a feasible b-matching, because it saturates X by its definition.

In order to prove that bM is a feasible $b-$matching, we show that the number of edges incident on each vertex is within the allocated bounds - s_v for $v \in Y$ and 1 for $v_i \in X$. The latter bound is trivially verified. To prove that the bounds s_v are not violated, we observe that no two vertices of X with same color index i, say v_i and w_i, are matched to the same vertex $u \in Y$ with respect

Algorithm 3 A 3-approximation algorithm for SRR

Let $\mathcal{D}$ be the list of possible δ values, i.e., the list of pairwise distances between the vertices of G, arranged in the non-decreasing order.
for all $\delta \in \mathcal{D}$ **do**
  for all colors c **do**
    Let H_c be the subgraph of G_δ^2 induced by the set of vertices that require color c.
    Compute $\mathcal{I}_c$, any maximal independent set of H_c.
  end for
  Let X denote the union of copies of each $\mathcal{I}_c$ (i.e., if a vertex is contained in s independent sets of form $\mathcal{I}_c$, we add s copies of that vertex to X). Let Y be a copy of set of vertices in V with non-zero storage capacities.
  Construct the bipartite graph $B = (X, Y)$: add an edge between $x \in X$ and $y \in Y$ if the nodes they represent are at distance at most δ.
  Compute a maximum b-matching in B with bounds : 1 on vertices of X and respective storage capacities on the nodes of Y.
  For every node $v \in Y$, let $S_v \subseteq X$ be matched subset of nodes, assign the list of colors L_v of nodes of S_v to v. Call this coloring ϕ.
  if the requirement of every vertex of G is satisfied within a distance of 3δ **then**
    return ϕ (and exit the function)
  end if
end for

to bM. Indeed, this would imply that v and w are adjacent in G_δ^2, which is a contradiction to the fact that they belong to a maximal independent set (in some induced subgraph of G_δ^2). Thus the number of edges of bM incident on u, is at most $|L_u^{opt}| \leq s_u$. Hence, bM is a valid b-matching which saturates all the vertices of X.
To finish the proof, we now show that every node v requiring a color i finds a node hosting i at distance at most 3δ. Indeed, there exists some $u_i \in X$, such that u is a neighbor of v in H_i (note that the distance between such u and v is at most 2δ). Now, if $\overline{u_i w} \in bM$, w is the vertex hosting i at distance at most 3δ. Hence, Algorithm 3 is a 3-approximation algorithm for the subset resource replication problem. □

2.3 Hardness of BRR and SRR

We now prove some lower bounds on the above problems. The following theorem shows that Algorithm 3 provides the best possible guarantee for the SRR problem, while there is a small gap between the algorithm and the lower bound proven for the BRR problem.

Theorem 6 *Assuming $P \neq NP$, there is no polynomial time algorithm which guarantees an approximation ratio better than*

1) $2 - \epsilon$ *for basic resource replication problem.*
2) $3 - \epsilon$ *for subset resource replication problem.*

Proof (1) The following problem is called *domatic partition problem* and was shown to be NP-complete in [9] - Given a simple undirected graph $G = (V, E)$ and an integer k, does there exist a partition of V into k subsets $V_i : i \in$

$[1, k]$ such that each V_i is a dominating set of G. We reduce any instance $(G = (V, E), k)$ of the domatic partition problem into an instance of BRR $(V, \mathcal{C} = \{C_1, C_2 \dots C_k\}, d)$ in the following way. The function $d(.)$ is defined as follows -

$$d(\overline{uv}) = \begin{cases} 1 & \text{if } \overline{uv} \in E \\ 2 & \text{otherwise} \end{cases}$$

It is easy to check that $d : V \times V \to \mathbb{R}$ is a metric. We note that, for the above instance of BRR, there are only two possible values for optimal distance - namely 1 or 2 (assuming a non-trivial instance with non-zero optimal distance). We claim that (G, k) is a YES instance of domatic partition problem if and only if the optimal distance value for the BRR instance is 1. Indeed, if we can partition V into k dominating sets, we can give each such set one of the k colors (uniquely). Every vertex, by the definition of dominating sets, must be adjacent to some vertex in each of the other dominating sets. Hence, each vertex will find all the k colors within distance 1. On the other hand, if every vertex finds a color within distance 1, each color group forms a dominating set, hence (G, k) will be a YES instance for the domatic partition problem. Thus, if (G, k) is a YES instance the optimal distance value is 1 and if it is a NO instance the optimal distance is 2. Thus, if there exists an algorithm $\mathcal{A}$ with polynomial running time such that it approximates BRR within a factor $2 - \epsilon$, it will be able to differentiate YES and NO instances of domatic partition problem and hence is a polynomial time algorithm for domatic partition problem. But this would imply $P = NP$.

(2) The following problem is NP-complete [8] - Given a bipartite graph $B = (X, Y)$ partition Y into k subsets such that each subset dominates X. This problem is called, the *one-sided domatic partition* (ODP) problem. Now, given an instance of ODP, (B, k), we reduce it to an instance of the SRR, $\mathcal{I}' = (V, \mathcal{C} = \{C_1, C_2 \dots C_k\}, \{\mathcal{C}_v \subseteq \mathcal{C} : v \in V\}, \{s_v : v \in V\}, d)$ in the following way. Set $V = X \cup Y$. All vertices of X have 0 capacity $(s_v = 0 : v \in X)$ and all vertices of Y have 1 capacity $(s_v = 1 : v \in Y)$. All vertices of X require every color in $[1, k]$ $(\mathcal{C}_v = \mathcal{C} : v \in X)$ and all vertices of Y require no color $(\mathcal{C}_v = \phi : v \in X)$. The distance metric is

$$d(\overline{uv}) = \begin{cases} 1 & \text{if } \overline{uv} \in E(B) \\ 2 & \text{if } u, v \in X \text{ or } u, v \in Y \\ 3 & \text{otherwise} \end{cases}$$

Following the same argument as in part (1), it is easy to show (B, k) is a YES instance of ODP $\iff$ $\mathcal{I}'$ has optimal distance value 1. We observe that the above instance only takes values 1 or 3. Hence, if one could approximate it within $3 - \epsilon$, ODP becomes polynomial time solvable implying $P = NP$. $\square$

3 Robust Resource Replication Problem

The objective of minimizing the maximum distance over all vertices may result in a much larger distance if there are few distant "outliers". Even a good approximation algorithm, in this case, will raise δ to a very high value and many nodes could get a bad solution. It is therefore natural to study outlier versions of such problems. In such a model, the objective remains the same but we are allowed to ignore a few far away vertices (the outliers). Figure 3 shows a simple example where "ignoring" a few far away vertices improves the quality of solution significantly.

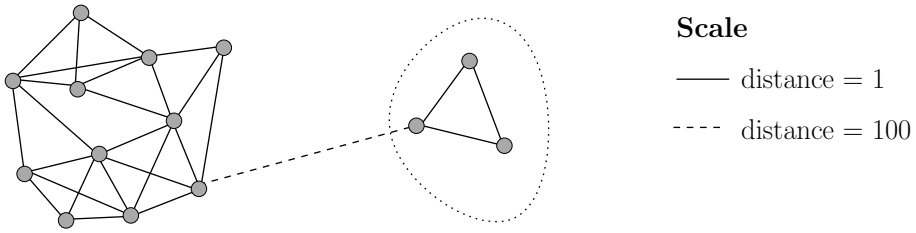


Fig. 3 A case for the outlier version: For $k = 4$, non-outlier version has a cost of 100, while if we ignore the vertices in the "dotted" circle, the solution has a cost of 1.

Several well known problems have been studied under the "outlier" model like outlier versions of k-center problem [5] (called *robust k-centers*), scheduling with outliers [4,13,23], outlier versions of facility location type problems [5, 20]. In this section, we initiate the problem of *robust basic resource replication* (RBRR) or the resource replication problem with outliers. In the RBRR problem, the input is the same as the BRR problem along with a lower bound M - which is the number of vertices that have to be satisfied. Formally, the input instance $\mathcal{I} = (V, \mathcal{C}, M, d)$ is defined as following.

- a set of vertices $V = \{v_1, v_2, \ldots v_n\}$, a metric $d : V \times V \to \mathbb{R}^+ \cup \{0\}$ and a set of colors $\mathcal{C} = \{C_1, C_2 \ldots C_k\}$.
- a lower bound $M \in \mathbb{N}$.

The objective function of the Robust Basic Resource Replication problem is defined as-

$$\min_{\substack{\phi \\ S \subseteq V \\ |S| \geq M}} \max_{\substack{v \in S \\ C_r \in \mathcal{C}}} d_r(v)$$

We show that a simple extension to Algorithm 1 gives a 3-approximation for the RBRR. Algorithm 4 describes the procedure. Again, we begin by guessing the optimal value δ and construct the threshold graph G_δ. We note that there should be at least M vertices with degree at least $k-1$ for δ to be a feasible solution distance, because at least M vertices should get $k-1$ colors from their neighborhood to be satisfied. We then construct an independent set in G_δ^2 by adding only these "high" vertices, as long as possible. Finally, for each

vertex v in the independent set, we pick $k-1$ of its neighbors in G_δ and assign k colors one to each of the k vertices (v and its $k-1$ neighbors).

Algorithm 4 A 3-approximation for RBRR

1: Let $\mathcal{D}$ be the list of possible δ values, i.e., the list of pairwise distances between the vertices of G, arranged in the non-decreasing order.
2: **for all** $\delta \in \mathcal{D}$ **do**
3: Construct G_δ and mark vertices which have degree $\geq k-1$.
4: Construct G_δ^2.
5: Construct an independent set $\mathcal{I}$ of G_δ^2 by adding marked vertices as long as possible (i.e., maximal with respect to marked vertices).
6: **for all** $v \in \mathcal{I}$ **do**
7: Choose $k-1$ vertices from neighborhood of v in G_δ. Color these k vertices with k colors arbitrarily.
8: **end for**
9: Color all uncolored vertices arbitrarily.
10: **if** the number of vertices that are satisfied with in a distance of 3δ is at least M **then**
11: **return** the current assignment and exit.
12: **end if**
13: **end for**

Theorem 7 *Algorithm 4 gives a 3-approximation for the RBRR problem.*

Proof Let δ be the optimal value, such that M vertices have all the k colors within distance δ. We prove that Algorithm 4 satisfies at least M vertices within distance 3δ. We claim that every vertex of degree at least $k-1$ (in G_δ) has all the k colors within a distance 3δ. Indeed, if a vertex u has degree $k-1$ then either it belongs to $\mathcal{I}$ or has a node in $\mathcal{I}$ at distance at most 2δ. Since each $v \in \mathcal{I}$ has all colors within δ, every such vertex u is completely satisfied within 3δ. Clearly every vertex satisfied by the optimal algorithm must have degree $k-1$ and therefore there are at least M nodes of degree $k-1$. Hence, Algorithm 4 will satisfy at least M nodes within 3δ. □

We now consider a more interesting generalization of the Robust Basic Resource Replication problem called the K-Robust Basic Resource Replication (K-RBRR) problem. In this problem we only allow K copies of each resource, while the rest of input and output structure remains the same as RBRR. This problem is a natural generalization of the robust K-center problem- the former problem has k resources and the latter has only one. The robust K-center problem is the outlier version of K-center problem and was studied, along with several other outlier variants of facility location type problems by Charikar et al. [5]. One variant of particular interest to our work is the robust K-supplier problem, for which Charikar et al. [5] give a 3 -approximation algorithm. The robust K-supplier is the outlier variant of K-supplier problem. In the K-supplier problem, we have a set of suppliers and a set of clients, embedded in a metric. The goal is to choose K suppliers which can hold a resource

(there is only one resource here) such that the maximum "client to nearest resource distance" is minimized over all clients. In the robust K-supplier problem, we have the same objective but we may satisfy only M clients. We use the 3-approximation algorithm of [5] as a sub-routine and obtain a 5-approximation algorithm for K-RBRR problem. For the sake of completeness, we briefly describe the algorithm from [5] here.

For a given value δ, the algorithm of [5] proceeds in the following way.

- For each supplier v, construct G_v as the set of clients within distance δ and E_v as the set of clients within distance 3δ of v.
- Repeat the following steps K times:
  - Greedily pick a supplier v as a center whose set G_v covers most number of yet uncovered clients. (†)
  - Mark all the clients in E_v as covered.
- If at least M vertices are satisfied return the centers, or else return NO.

For a proof on why this algorithm guarantees a 3-approximation, we refer the reader to [5]. We make a small modification to the above algorithm before using it as a sub-routine. In the step(†), if there are no more clients to be covered we can stop (this will clearly not affect the performance or feasibility of the algorithm). Otherwise, there is at least one new uncovered client which is now covered by v. We pick one such newly covered client arbitrarily and label it $U(v)$. Note that this process assigns a distinct client to each supplier. Algorithm 5 gives a 5-approximation for the K-RBRR problem. We make the following claims about Algorithm 5.

Claim 1 *If δ is the optimal distance value for an instance of K-RBRR, it is a feasible distance for the K-supplier instance in step 4 of Algorithm 5.*

Proof Consider an optimal coloring of the graph which gives distance δ for the K-RBRR instance. Now, there are at least M vertices of degree at least $k-1$, which have a particular color C_1 within distance δ (in fact, these vertices have all the colors within δ). Now, just pick those nodes colored with color C_1 as centers. This implies M vertices of V_c (the set of clients in Algorithm 5) are satisfied, as all vertices with at least $k-1$ degree in G_δ are represented in V_c. Therefore δ is a feasible distance for the K-supplier instance constructed in step 4 of Algorithm 5. □

Claim 2 *The set $\mathcal{I}$ formed in the step 11 of Algorithm 5, is an independent set in G_δ^2.*

Proof We prove that any two elements $U(v), U(w)$ are at distance strictly greater than 2δ from each other. Let us assume this is not the case. Let v be chosen as a center before w. Since the distance between v and $U(v)$ is $\leq \delta$ and distance between $U(v)$ and $U(w)$ is $\leq 2\delta$, the distance between v and $U(w)$ must be $\leq 3\delta$. But this implies $U(w)$ would be covered when v was picked, a contradiction to the fact that $U(w)$ is an uncovered vertex when w was picked. □

Algorithm 5 A 5-approximation for K-RBRR

1: Let $\mathcal{D}$ be the list of possible δ values, i.e., the list of pairwise distances between the vertices of G, arranged in the non-decreasing order.
2: **for all** $\delta \in \mathcal{D}$ **do**
3: Construct G_δ. Mark the nodes of degree $\geq k-1$. Let these "high" degree vertices form a set V_c.
4: With V_c as the set of clients, $V_s = V$ as the set of suppliers, distance between copies remaining the same as the original vertices, we solve the robust K-supplier problem [5] with δ as the input distance.
5: **if** δ is in-feasible for the above robust K-supplier instance **then**
6: By Claim 1, δ is not the correct guess for optimal solution. Hence, we can move onto to the next δ value.
7: **continue** to next δ value.
8: **else**
9: Let $S \subseteq V_s$ be the set of centers returned. By Claim 1, S is well defined.
10: **end if**
11: Let $\mathcal{I} = \{U(v) : v \in S\}$. By Claim 2, $\mathcal{I}$ is an independent set in G_δ^2. Further, each member of $\mathcal{I}$ has degree $\geq k-1$ in G_δ (because $\mathcal{I} \subseteq V_c$ and each $v \in V_c$ is of degree $\geq k-1$).
12: **for** $v \in \mathcal{I}$ **do**
13: Pick $k-1$ neighbors of v in G_δ. Assign each of these vertices along with v, one color each of the k colors.
14: **end for**
15: **if** the number of vertices satisfied within a distance 5δ is at least M **then**
16: **return** the current assignment and exit.
17: **end if**
18: **end for**

Theorem 8 *Algorithm 5 is a 5-approximation for the K-RBRR.*

Proof Claim 1 guarantees S is valid and Claim 2 guarantees there is no clash during the coloring phase (step 12) of Algorithm 5. Hence, Algorithm 5 generates a valid coloring. We now prove that at least M vertices get all k colors within 5δ distance. We note that S has the property that, at least M vertices are at distance at most 3δ from S (i.e., each of the M vertices has a vertex of S at distance at most 3δ). Since $\mathcal{I}$ was obtained by shifting the centers of S by at most δ, at least M vertices are at distance 4δ from $\mathcal{I}$. But each element of $\mathcal{I}$ has all k colors within δ, hence at least M vertices have all k colors within distance 5δ. □

Let us now consider the Robust Subset Resource Replication (RSRR) problem. In this problem, we are provided with the input for the SRR problem along with a lower bound M on the number of vertices that must be satisfied with their requirement. The objective function is -

$$\min_{\substack{\phi \\ S \subset V \\ |S| \geq M}} \max_{\substack{v \in S \\ r \in \mathcal{C}_v}} d_r(v)$$

Given that the outlier version of BRR and its extension with bound on each color has simple constant factor approximation algorithms, it is a natural question to ask whether similar bounds can be obtained for Robust SRR. But,

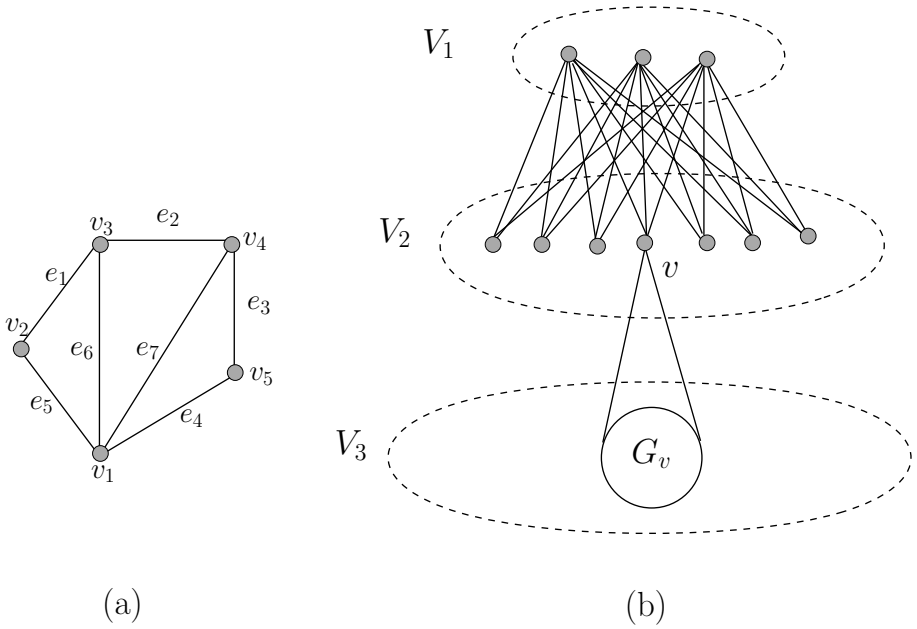


Fig. 4 Reduction of maximum k-clique subgraph instance to a Robust Subset Resource Replication Problem:(a) Given instance of the maximum k-clique problem. For this example, $k = 3$. (b) A Robust Subset Resource Replication instance. The vertex set comprises of 3 parts: V_1 with $k = 3$ vertices, V_2 with vertices corresponding to the edges of G. Each $v \in V_2$ has a m^2 clique G_v associated with it as shown. These gadgets form the set V_3. Each edge represents distance 1 and the remaining distances are computed using shortest path metric.

quite surprisingly, we show not only there does not exist any constant factor approximation algorithm for Robust SRR, but in fact, assuming $P \neq NP$, there is no polynomial time algorithm that provides any nontrivial approximation guarantee. In Theorem 9, we prove that deciding if a given instance of RSRR is feasible, is NP hard. We give a polynomial time reduction of the maximum k-clique problem to the problem of deciding the feasibility of RSRR. In the decision version of the maximum k-clique problem, we have an instance of the form $\mathcal{I} = (G, k)$ and the goal is to decide if there is a complete subgraph (clique) of G with exactly k vertices.

Theorem 9 *Assuming $P \neq NP$, there is no polynomial time algorithm which gives a positive approximation ratio for Robust Subset Resource Replication problem.*

Proof **Reduction.** Given an instance of maximum k-clique problem $\mathcal{I} = (G = (V, E), k)$, $|V| = n$, $|E| = m$ where the problem is to decide if there is a clique on k vertices - we construct an instance of RSRR, $\mathcal{I}' = (G', M, \mathcal{C}, \{\mathcal{C}_v : \forall v \in G'\})$ as follows. First, color the vertices in V with distinct colors $c_1, c_2 \ldots c_n$ arbitrarily. The vertex set of G' has 3 parts - V_1, V_2, V_3. V_1 has k vertices and V_2 has m vertices corresponding to the edges of G. The distance between any

two vertices $u \in V_1, v \in V_2$ is 1. Each vertex $v \in V_2$ has a set of m^2 vertices, G_v, associated with itself. The distance between any vertex pair of $v \cup G_v$ is 1. Rest of the distances are computed using the shortest path metric. The set $\{\mathcal{C}_v : \forall v \in G'\}$ is specified in the following way - Each vertex $u \in V_1$ requires 0 colors and hence are trivially satisfied. Each vertex $v \in V_2$ requires colors $\{a_v, c_i, c_j\}$ where a_v is a color associated uniquely with vertex v and c_i, c_j are the colors of the end points of the edge in G associated with v. Each vertex $w \in G_v$ requires colors $\{a_v, b_v^i : i \in [1 : m^2]\}$. Each one of $a_v, b_v^i : v \in V_2, i \in [1, m^2]$ is a distinct color. Let $L = \binom{k}{2}$. Set $M = m^3 + L + k$, the lower bound of the number of vertices that must be satisfied. Figure 3 shows the construction for a simple instance.

Claim: I is a YES instance of maximum k-clique problem if and only if I' is a feasible solution of Robust Subset Resource Replication problem. In other words, we prove that the feasibility question of Robust Subset Resource Replication problem is NP-hard. This would imply that there is no approximation algorithm for this problem.

Proof of the Claim. Let I be an YES instance of the maximum k-clique problem and let $H = \{v_1, v_2 \dots v_k\}$ be the k vertices that induce a clique, that is $L = \binom{k}{2}$ edges in G. We present a feasible coloring for I' as following -

- The k vertices of V_1 are colored with the k colors of H
- Each vertex $v \in V_2$ is colored with its associated color a_v.
- For each vertex $v \in V_2$, its m^2 associated vertices G_v are colored with m^2 colors of type b_v^i.

It is straightforward to check that the above coloring satisfies $M = m^3 + k + L$ vertices - all the vertices of V_3 are satisfied, all the vertices of V_1 are satisfied and at least L vertices of V_2 are satisfied. Now, we consider the other direction. Let there be a coloring of vertices of G' which certifies that I' is a feasible instance. We first observe that, all the m^3 colors of type b_v^i and the m colors of type a_v must be used - otherwise, there will be at least m^2 vertices out of $m^3 + m + k$ vertices which go unsatisfied and hence the bound M is not met. Since, we are only interested in the feasibility question, we can assume that m^2 vertices of G_v are colored with m^2 colors of type b_v^i and the m vertices $v \in V_2$ are colored with color a_v. Now at least $L = \binom{k}{2}$ vertices of V_2 must be satisfied and the k vertices of V_1 must be colored with k colors from $\{c_1, c_2 \dots c_n\}$ - say $\{c_1, c_2 \dots c_k\}$. We observe that the union of colors required by the $L = \binom{k}{2}$ vertices, apart from their associated colors, must be $\{c_1, c_2 \dots c_k\}$. Hence, the L edges in G corresponding to these L vertices in V_2 must be completely incident on the vertices in V corresponding to these k colors. This implies the existence of k vertices in G that induce $\binom{k}{2}$ edges, that is an existence of a k clique. Hence the theorem. □

Note. If we insist only on a lower bound on the number of satisfied node-resource pairs as opposed to the number of completely satisfied nodes, the problem becomes significantly easier. We just need to create a copy of a vertex for each color that it desires and then run the robust version of BRR. The main

hardness stems from the fact that in order for a vertex to be satisfied it requires all the desired colors.

4 Capacitated Basic Resource Replication Problem

Another desired quality of an assignment scheme in client-server type problems is load balancing [19,16,3]. In this setting, we are not allowed to "overload" a server by assigning more than a bounded number of clients. Bar-Ilan, Kortsarz and Peleg [3], Khuller and Sussman [16] study the load balancing version of the k-center problem which is called the *capacitated k-center problem*. Khuller and Sussman [16] provide the current best approximation ratio of 5 for this problem. We initiate the study of basic resource replication problem in the load balancing setting. We call it the *capacitated basic resource replication problem* (CBRR). In this problem, the input instance is defined as $\mathcal{I} = (V, \mathcal{C} = \{C_1, C_2 \dots C_k\}, d, L)$ and the goal is the same as the basic resource replication problem with an additional restriction that a vertex with a certain color is not allowed to serve more than L other vertices (including itself). We give a 4-approximation algorithm (Algorithm 6) for this problem, provided $L \geq 2k-1$. First we prove that the problem is infeasible if $L < k$.

Preposition 1 *Given an instance of CBRR, $\mathcal{I} = (V, \mathcal{C}, d, L)$, with $|\mathcal{C}| = k$, the following statements are true:*

1. *If $L \leq k-1$, then $\mathcal{I}$ is infeasible.*
2. *If $L = k$, then $\mathcal{I}$ is feasible with value $\delta \iff$ the number of vertices in each component of G_δ (threshold graph on V) is a multiple of k.*

Proof 1. Let if possible, there exist a feasible solution when $L < k$. We construct the following directed graph D, on the vertex set V. For every pair of vertices $u, v \in V \times V$ (note that u and v need not be distinct), we add a directed edge $\overline{uv}$ them if v serves u in the feasible solution. Since each vertex requires all k colors, the out-degree of each vertex is at least k. Also we note that, since each vertex can serve at most L vertices the in-degree of each vertex is at most L . Given that the problem is feasible, we have:

$$\begin{aligned} k \times |V| &\leq \sum_{v \in D} \text{out-degree of v} \\ &= \sum_{v \in D} \text{in-degree of v} \\ &\leq L \times |V| \end{aligned}$$

This implies $L \geq k$, a contradiction.
2. Fixing a component C of G_δ, we will first prove that for the instance to be feasible, $|C|$ must be a multiple of k. Firstly, we note that the vertices of C can only be satisfied by other vertices of C. Consider the feasible assignment of colors C_i $i \in [1, k]$ to C. Group all the vertices that are given a color C_i into a class B_i. Let B_s be the smallest cardinality color class. Construct

the directed graph on C as described in the part(a). Now, every vertex in C (including those in B_s) must have an edge directed into B_s (because every vertex requires the color C_s). Each vertex of B_s has an in degree $\leq L = k$. Hence, we have $|C| \leq k \times |B_s|$. This implies all the color classes have the same cardinality, which in turn implies $|C|$ is a multiple of k. □

We now give a 4-approximation algorithm for the CBRR problem where $L \geq 2k - 1$. We refer to Algorithm 6 for pseudocode. The algorithm starts by guessing the optimal δ and constructs the threshold graph G_δ. Let $\mathcal{I}$ be some maximal independent set of G_δ^2. We divide all the vertices into three levels - *level* 0, *level* 1 and *level* 2. All the elements in $\mathcal{I}$ are at *level* 0. All vertices not in $\mathcal{I}$ but adjacent (with respect to G_δ) to some element in $\mathcal{I}$ are at *level* 1. Finally all the vertices not in *level* 0 or *level* 1 are in *level* 2. For each element v at *level* 0, its empire $Empire(v)$ consists of itself along with all the adjacent(with respect to G_δ) *level* 1 vertices. Since $\mathcal{I}$ is independent in G_δ^2, all the empires defined so far are mutually disjoint. Finally, all the *level* 2 vertices are adjacent to at least one *level* 1 vertex. For each *level* 2 vertex, we pick one such *level* 1 vertex arbitrarily and assign the former to the same empire as the latter. Thus we have assigned every vertex to exactly one empire.

Algorithm 6 A 4-approximation for CBRR

Let $\mathcal{D}$ be the list of possible δ values, i.e., the list of pairwise distances between the vertices of G, arranged in the non-decreasing order.
for all $\delta \in \mathcal{D}$ **do**
  Construct the graph G_δ and G_δ^2.
  if minimum degree of $G_\delta < k - 1$ **then**
    continue onto the next δ value.
  end if
  Let $\mathcal{I}$ be a maximal independent set in G_δ^2.
  for all $v \in V$ **do**
    if $v \in \mathcal{I}$ **then**
      $Empire(v) = \{v\}$
    end if
    if $v \notin \mathcal{I}$ **then**
      if v has a vertex $u \in \mathcal{I}$ at distance δ. **then**
        Such a vertex is unique owing to the property that $\mathcal{I}$ is an independent set. Add v to the empire of u, $Empire(u) = Empire(u) \cup \{v\}$.
      else if v has a vertex in $\mathcal{I}$ at distance 2δ. **then**
        Pick one such vertex u arbitrarily and add v to the empire of u.
      end if
    end if
  end for
  for all $v \in \mathcal{I}$ **do**
    Each vertex v has degree at least $k - 1$ in G_δ. Hence, $|Empire(v)| \geq k$. Divide $Empire(v)$ into blocks, all of which have size exactly k - except possibly the last one which has size at most k.
    Color each block of size exactly k using k colors, arbitrarily. The final block, whose size is at most k, has its color requirement satisfied from one such block. Since there is at least one block of size exactly k, such an assignment is valid.
  end for
end for

In the next step, we consider one empire at a time and split it into "blocks" of vertices. Every block consists of exactly k vertices, except the last block which might have less than k vertices. A key property of vertices in a block is the following - any two vertices are at a distance of at most 4δ from each other. We now color each block of size exactly k using all k colors (since the degree of each vertex is at least $k-1$ in G_δ, every empire has at least one block of size exactly k). A vertex in a block only serves other vertices in the same block, hence the load is not more than k currently on any vertex. The vertices of the final block (which might have $\leq k$ vertices) are now served by some block of size exactly size k. Thus the load on each vertex is at most $2k-1$.

Theorem 10 *Algorithm 6 is a 4-approximation algorithm for the problem of Capacitated Basic Resource Replication problem where the allowed load* $L \geq 2k-1$.

Proof As mentioned in the discussion above, the key observation needed is that any two vertices in the same empire (of say a vertex v) are at distance at most 4δ from each other. Indeed, all the vertices in the empire of vertex v are at distance at most 2δ from v and hence at distance 4δ from each other. The only detail that needs to be verified is that the maximum load on any vertex is at most $2k-1$. A block of size exactly k satisfies the requirement of its own members along with at most one other block (of size $< k$). Hence the maximum load is $\leq 2k-1 \leq L$. □

By using Preposition 1, we observe that Algorithm 6 is in fact a bi-criteria approximation algorithm (for arbitrary load capacity) - it gives an approximation guarantee of 4 while exceeding the load by a factor of 2 at most.

We now show a simple 8-approximation algorithm for the CBRR problem, when the capacity $L = k$. We use the construction of [16] to obtain a maximal independent set, $\mathcal{I}$ on G_δ^2, which has the following useful property -

Property *$\mathcal{I}$ can be represented as a rooted tree, $\mathcal{T}$, where any given vertex (apart from the root) has its parent (immediate ancestor) at distance $\leq 3\delta$.*

We also adopt the terminology of [16] and call each vertex in $\mathcal{I}$ a *monarch*, the rooted tree $\mathcal{T}$ a *monarch tree* and all the vertices assigned to it in a feasible solution its *empire*. Every monarch has all its neighbors in G_δ added to its empire. Every non-assigned vertex is at distance at most 2δ from some monarch (otherwise such a vertex can be added to $\mathcal{I}$) and we add the former to the latter's empire (breaking ties arbitrarily, if more than one such monarch exists).

Theorem 11 *Algorithm 7 is an 8-approximation algorithm for CBRR with* $L = k$.

Proof We prove the following two properties of Algorithm 7 which will imply the statement of the theorem:

Algorithm 7 8-approximation for load = k

Let $\mathcal{D}$ be the list of possible δ values, i.e., the list of pairwise distances between the vertices of G, arranged in the non-decreasing order.
for all $\delta \in \mathcal{D}$ **do**
  Construct G_δ.
  if minimum degree of $G_\delta < k-1$ **then**
    continue onto next δ value.
  end if
  Construct G_δ^2.
  Construct the monarch tree $\mathcal{T}$. Let r be its root.
  Call the recursive procedure, ProcessMonarch$(r, \mathcal{T})$
end for

Algorithm 8 Recursive procedure: ProcessMonarch$(r, \mathcal{T})$

Let C be the set of *children* of r in $\mathcal{T}$.
LeftOver $= \phi$
for all Child $c \in C$ **do**
  LeftOver = LeftOver $\cup$ ProcessMonarch$(c, \mathcal{T})$.
end for
Divide the set empire$(r) \cup$ LeftOver, into blocks of size exactly k, with the possible exception of the final block, which is of size $< k$. In constructing such blocks, we give least preference to the vertices in the neighborhood of r in G_δ. Since the degree of r in G_δ is at least k (otherwise δ is infeasible), the final block F is completely in the neighborhood of r in G_δ.
Color all the blocks of size exactly k with k colors.
if F is uncolored **then**
  return F as the set of "left over" vertices.
else
  return ϕ.
end if

1. Every vertex belongs to some "colored block", i.e., a block of size k which is colored using k colors. This will imply that every vertex has its requirement satisfied within the block.
2. For a given block, the maximum distance between any two vertices is at most 8δ. This will imply that Algorithm 7 is an 8-approximation.

From Preposition 1, we know that the number of vertices in a component is a multiple of k. This along with the fact that every block is of size k, implies that the LeftOver set must be empty when the Procedure 8 is called on the root r. For the second claim, lets consider a arbitrary block B which is colored when processing some monarch m. If B is completely contained in the empire of m, the maximum distance between any two vertices of B is 4δ. On the other hand, if B contains left over elements, we observe that these left over elements are from the empires of monarchs which are children of m in $\mathcal{T}$. Indeed, when we are creating blocks for a monarch, the left over vertices of its children are preferred and made into blocks first . Hence, each monarch has to deal with the left overs of its children alone. We also note that the only vertices passed on from a monarch to its parent monarch are the former's neighbors in G_δ. Hence, if u is an element in the set LeftOver of a monarch m, it must be at a distance $3\delta + \delta = 4\delta$ (since the monarch m', whose empire contains u, is a

child of m and hence is at distance 3δ from it) from m. Thus, any two elements of the block are at a distance at most $4\delta + 4\delta = 8\delta$. □

5 Conclusion

To conclude, we study several variants of the resource replication problem and prove that most of them are approximable within a small constant. A striking anomaly is the problem of RSRR, which somewhat surprisingly is hard to approximate within any non-trivial bound. Our work leaves several open problems. It would be interesting to close the gap between the approximation factor and the lower bound of the BRR problem. Extending the capacitated version to SRR, obtaining a true approximation factor for CBRR for all values of load, improving the approximation factor for K-RBRR etc. are few other future directions to consider.

References

1. Ivan D. Baev and Rajmohan Rajaraman. Approximation algorithms for data placement in arbitrary networks. In *SODA*, pages 661–670, 2001.
2. Ivan D. Baev, Rajmohan Rajaraman, and Chaitanya Swamy. Approximation algorithms for data placement problems. *SIAM J. Comput.*, 38(4):1411–1429, 2008.
3. Judit Bar-Ilan, Guy Kortsarz, and David Peleg. How to allocate network centers. *J. Algorithms*, 15(3):385–415, 1993.
4. Moses Charikar and Samir Khuller. A robust maximum completion time measure for scheduling. In *SODA*, pages 324–333, 2006.
5. Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *SODA*, pages 642–651, 2001.
6. Jack Edmonds. Paths, trees, and flowers. In Ira Gessel and Gian-Carlo Rota, editors, *Classic Papers in Combinatorics*, Modern Birkhuser Classics, pages 361–379. Birkhuser Boston, 1987.
7. Jack Edmonds and Delbert R. Fulkerson. Bottleneck extrema. *Journal of Combinatorial Theory*, 8(3):299 – 306, 1970.
8. Uriel Feige, Magnús M. Halldórsson, and Guy Kortsarz. Approximating the domatic number. In *STOC*, pages 134–143, 2000.
9. Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.
10. Leana Golubchik, Sanjeev Khanna, Samir Khuller, Ramakrishna Thurimella, and An Zhu. Approximation algorithms for data placement on parallel disks. *ACM Transactions on Algorithms*, 5(4), 2009.
11. Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.*, 38:293–306, 1985.
12. Sudipto Guha and Kamesh Munagala. Improved algorithms for the data placement problem. In *SODA*, pages 106–107, 2002.
13. Anupam Gupta, Ravishankar Krishnaswamy, Amit Kumar, and Danny Segev. Scheduling with outliers. In *APPROX-RANDOM*, pages 149–162, 2009.
14. Dorit S. Hochbaum and David B. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, 1985.
15. Dorit S. Hochbaum and David B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *J. ACM*, 33(3):533–550, 1986.
16. Samir Khuller and Yoram J. Sussmann. The capacitated k-center problem. *SIAM J. Discrete Math.*, 13(3):403–418, 2000.

17. Bong-Jun Ko and Dan Rubenstein. Distributed, self-stabilizing placement of replicated resources in emerging networks. In *ICNP*, pages 6–15, 2003.
18. Bong-Jun Ko and Dan Rubenstein. Distributed server replication in large scale networks. In *NOSSDAV*, pages 127–132, 2004.
19. Madhukar R. Korupolu, C. Greg Plaxton, and Rajmohan Rajaraman. Analysis of a local search heuristic for facility location problems. In *SODA*, pages 1–10, 1998.
20. Ravishankar Krishnaswamy, Amit Kumar, Viswanath Nagarajan, Yogish Sabharwal, and Barna Saha. The matroid median problem. In *SODA*, pages 1117–1130, 2011.
21. Michael Luby. A simple parallel algorithm for the maximal independent set problem. In *STOC*, pages 1–10, 1985.
22. Adam Meyerson, Kamesh Munagala, and Serge A. Plotkin. Web caching using access statistics. In *SODA*, pages 354–363, 2001.
23. Barna Saha and Aravind Srinivasan. A new approximation technique for resource-allocation problems. In *ICS*, pages 342–357, 2010.
24. V.V. Vazirani. *Approximation Algorithms*. Springer, 2001.