New Approximation Results for Resource Replication Problems

Samir Khuller*[1], Barna Saha[2], and Kanthi K. Sarpatwar**[1]

[1] Department of Computer Science
University of Maryland (College Park)
samir,kasarpa@cs.umd.edu
[2] AT&T Shannon Research Laboratory
barna@research.att.com

Abstract. We consider several variants of a basic resource replication 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 optimal 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.

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 [16, 17, 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 [9] 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

* Supported by NSF Awards CCF-0728839 and CCF-0937865, and a Google Research Award.
** Supported by NSF Grant CCF-0728839.

not valid. Companies rent storage space all over the world from different data centers in different locations. Since the 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 [16] 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 was reasonably quick. In a followup piece of work [17], 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 as well. 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 *densest k-subgraph* [8] to the feasibility version of the general problem.

Following the works of Ko and Rubenstein [16, 17], 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. [11] and Meyerson et al. [20] (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 3). For the later, we give the first polynomial time 3-approximation algorithm (in a centralized setting) along with matching hardness (Theorem 2 and Theorem 3).
- 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 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 [16]. 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 $\mathcal{C} = \{C_1, C_2, C_3, \dots, C_k\}$.

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[3] and the vertex v. The goal of the *Basic Resource Replication* (BRR) problem is the following -

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

This is the central problem of the work of Ko and Rubenstein [16] 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*

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

introduced by Edmonds and Fulkerson [7] and used extensively for k-center type problems [10, 15, 14, 13]. 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. The algorithm for BRR works in the following way. For each vertex v, we determine the distance of the $(k-1)^{th}$ closest neighbor - and denote by δ_L the maximum of these distances. We construct the threshold graph G_{δ_L} which has minimum degree at least $k-1$. Also δ_L must be a lower bound on the optimal δ (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 $k-1$. Now in the graph $G^2_{\delta_L}$ which is the graph formed by squaring G_{δ_L}, we compute a maximal independent set $\mathcal{I}$. Finally, for each vertex in $\mathcal{I}$, we color the vertex with C_1 and pick $k-1$ vertices from its list of neighbors in G_δ and assign them a distinct color from the remaining $k-1$ colors. Due to space restrictions, we defer the details of the algorithm and discussion of this problem (along with a few other generalizations) to full version[4].

Theorem 1. *There is a distributed, polynomial time, 3-approximation algorithm for the problem of BRR.*

2.2 Subset Resource Replication Problem

In 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 [17]. 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 $|\phi(v)| \le s_v$, 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 [17] 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 1)

[4] http://www.cs.umd.edu/∼samir/grant/approx12-full.pdf

for the problem. Later, in Theorem 3, 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.

Algorithm 1 A 3-approximation algorithm for SRR

```
1: Guess the optimal value $\delta$. Construct the graph $G_\delta^2$
2: for all colors $c$ do
3:     Let $H_c$ be the subgraph of $G_\delta^2$ induced by the set of vertices that require color $c$.
4:     Compute $\mathcal{I}_c$, any maximal independent set of $H_c$.
5: end for
6: 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.
7: 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 $\delta$.
8: Compute a maximum $b$-matching in $B$ with bounds : 1 on vertices of $X$ and re-
   spective storage capacities on the nodes of $Y$.
9: 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 $v$.
```

Theorem 2. *Algorithm 1 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 7 of Algorithm 1, 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 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 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 1 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 1 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. We state the theorem here; for lack of space, the proof is given in full version.

Theorem 3. *Assuming $P \neq NP$, for any given constant $\epsilon > 0$, there is no polynomial time algorithm which guarantees an approximation ratio better than*

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

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 version 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). 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, 12, 21], outlier versions of facility location type problems [5, 19]. 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 $V = \{v_1, v_2, \dots v_n\}$, a metric $d : V \times V \rightarrow \mathbb{R}^+ \cup \{0\}$ and a set of colors $\mathcal{C} = \{C_1, C_2 \dots C_k\}$.
- A lower bound $M \in \mathbb{N}$.

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

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

A simple extension to the BRR algorithm results in a 3-approximation algorithm for this problem. Due to space restrictions, we defer our discussion to full version. Instead, we focus on 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 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 [5] gives 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 not satisfied return NO, or else return the set of centers.

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.

We can now describe our Algorithm 2 to solve the K-RBRR problem. We make the following claims about Algorithm 2 but defer the proofs to full version.

Claim. If δ is optimal distance value for an instance of K-RBRR, it is a feasible distance for the K-supplier instance in the step 2 of Algorithm 2.

Claim. The set $\mathcal{I}$ formed in the step 3 of Algorithm 2 is an independent set in G_δ^2.

Algorithm 2 A 5-approximation for K-RBRR

```
1: Guess optimal distance value $\delta$ and construct $G_\delta$. Mark the nodes of degree $\geq k-1$.
   Let these "high" degree vertices form a set $V_c$.
2: 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 $\delta$ as the input distance. Let $S \subseteq V_s$ be the set of centers returned.
   By Claim 3, $S$ is well defined.
3: Let $\mathcal{I} = \{U(v) : v \in S\}$. By Claim 3, $\mathcal{I}$ is an independent set such that each
   member has degree $\geq k-1$ in $G_\delta$.
4: for $v \in \mathcal{I}$ do
5:     Pick $k-1$ neighbors of $v$ in $G_\delta$. Assign each of these vertices along with $v$, one
       color each of the $k$ colors.
6: end for
```

Theorem 4. *Algorithm 2 is a 5-approximation for the K-RBRR.*

Proof. We defer the proof to full version.

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 -

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

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, 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 5, we prove that deciding if a given instance of RSRR is feasible, is NP hard. We give a polynomial time reduction of the well-studied densest k subgraph [8] problem to the problem of deciding the feasibility of RSRR. In the decision version of the densest k-subgraph problem, we have an instance of the form $\mathcal{I} = (G, k, L)$ and the goal is to decide if there is a subgraph of G with exactly k vertices and at least L edges.

Theorem 5. *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 densest k-subgraph problem $\mathcal{I} = (G = (V, E), k, L)$, $|V| = n$, $|E| = m$ where the problem is to decide if there is a subgraph on k vertices with at least L edges - 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 $c_1, c_2 \ldots c_n$. 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. Set $M = m^3 + L + k$, the lower bound of the number of vertices that must be satisfied.
Claim: I is an YES instance of densest k subgraph 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 densest subgraph problem and let $H = \{v_1, v_2 \dots v_k\}$ be the k vertices that induce L 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 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 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 L edges. Hence the theorem.

4 Capacitated Basic Resource Replication Problem

Another desired quality of an assignment scheme in client-server type problems is load balancing [18, 15, 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 [15] study the load balancing version of the k-center problem which is called the *capacitated k-center problem.* Khuller and Sussman [15] 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 3) for this problem, provided $L \geq 2k - 1$. We prove in the full version that, for a feasible solution, L has to be $\geq k$. By using this fact, we observe that Algorithm 3 is in fact a bicriteria approximation algorithm - it gives an approximation guarantee of 4 while exceeding the load by a factor of 2 at most.

Algorithm 3 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.

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 $2k-1$.

Algorithm 3 A 4-approximation for CBRR

```
1:  Guess the optimal value $\delta$. Construct the graph $G_\delta$ and $G_\delta^2$.
2:  Let $\mathcal{I}$ be a maximal independent set in $G_\delta^2$.
3:  for all $v \in V$ do
4:      if $v \in \mathcal{I}$ then
5:          $Empire(v) = \{v\}$
6:      end if
7:      if $v \notin \mathcal{I}$ then
8:          if $v$ has a vertex $u \in \mathcal{I}$ at distance $\delta$. then
9:              Such a vertex is unique owing to the property that $\mathcal{I}$ is an independent set.
                Add $v$ to the empire of $u$, $Empire(u) = Empire(u) \cup \{v\}$.
10:         else if $v$ has a vertex in $\mathcal{I}$ at distance $2\delta$. then
11:             Pick one such vertex $u$ arbitrarily and add $v$ to the empire of $u$.
12:         end if
13:     end if
14: end for
15: for all $v \in \mathcal{I}$ do
16:     Each vertex $v$ has degree at least $k-1$ in $G_\delta$. 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 $k$.
17:     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.
18: end for
```

Theorem 6. *Algorithm 3 is a 4-approximation algorithm for the problem of Capacitated Basic Resource Replication problem where the allowed load $L \geq 2k - 1$.*

Proof. We defer the proof to full version.

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, David Peleg, and Guy Kortsarz. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.
9. 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.
10. Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.*, 38:293–306, 1985.
11. Sudipto Guha and Kamesh Munagala. Improved algorithms for the data placement problem. In *SODA*, pages 106–107, 2002.
12. Anupam Gupta, Ravishankar Krishnaswamy, Amit Kumar, and Danny Segev. Scheduling with outliers. In *APPROX-RANDOM*, pages 149–162, 2009.
13. 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.
14. Dorit S. Hochbaum and David B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *J. ACM*, 33(3):533–550, 1986.

15. Samir Khuller and Yoram J. Sussmann. The capacitated k-center problem. *SIAM J. Discrete Math.*, 13(3):403–418, 2000.
16. Bong-Jun Ko and Dan Rubenstein. Distributed, self-stabilizing placement of replicated resources in emerging networks. In *ICNP*, pages 6–15, 2003.
17. Bong-Jun Ko and Dan Rubenstein. Distributed server replication in large scale networks. In *NOSSDAV*, pages 127–132, 2004.
18. 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.
19. Ravishankar Krishnaswamy, Amit Kumar, Viswanath Nagarajan, Yogish Sabharwal, and Barna Saha. The matroid median problem. In *SODA*, pages 1117–1130, 2011.
20. Adam Meyerson, Kamesh Munagala, and Serge A. Plotkin. Web caching using access statistics. In *SODA*, pages 354–363, 2001.
21. Barna Saha and Aravind Srinivasan. A new approximation technique for resource-allocation problems. In *ICS*, pages 342–357, 2010.