A 2-Approximation Algorithm for QoS-Aware and Fault-Tolerant Replica Placement

Zhihui Du
Department of Computer Science
New Jersey Institute of Technology, Nework, New Jersey, US
zhihui.du@njit.edu

Sen Zhang
Department of Mathematics, Computer Sciences and Statistics
State University of New York, College at Oneonta
zhangs@oneonta.edu

David A. Bader
Department of Computer Science
New Jersey Institute of Technology, Nework, New Jersey, US
bader@njit.edu

Jingkun Hu
Worldmoney Blockchain Management Limited
Hong Kong
kun.hu@worldmoney.org

***Abstract*—As emerging applications become more and more distributed and decentralized, how to design and build a fault-tolerant network system with high Quality of Service (QoS) guarantee has become a challenging problem. In this paper, we formulate an unique optimal replica placement problem in terms of minimizing the replica placement cost subject to both QoS and fault-tolerant constraints at the same time. Based on the generalized graph model, we prove that the optimal replica placement problem is NP-hard. To solve the problem, our proposed a novel heuristic based greedy solution that comprises of two interesting rounding algorithms that are applied in tandem on the results preprocessed by a relaxed linear programming component. We further prove our the proposed greedy algorithm actually is a 2-approximate algorithm.**

***Index Terms*—Replica Placement, Quality of Service, Fault Tolerance, Heuristic Algorithm, Approximation Ratio**

I. Introduction

Replica placement [7] is a critical technology that can be used for many different purposes, such as reducing latency, enhancing fault tolerance, optimizing bandwidth use, or improving scalability of network. So it has been employed in extensive applications such as Content Delivery/Distribution Network (CDN) [12], data grid [7], cloud [6] and edge computing environment [1]. The theoretical model of replica placement is named as facility location [3], K-median [2], minimum K-center [10] and so on under different application scenarios. The great variety and changes in application requirements are the essential reason why so many different replica placement policies have been developed and why replica placement problems remain to be an active research field. Emerging applications, such as smart city [8] and autonomous unmanned cars [15], entail critical requirements in both Quality of Service (QoS) and fault tolerance for supporting networks behind these applications. How to provide guaranteed service in real time with optimal cost yet simultaneously meeting constraints with both fault tolerance and QoS is a challenging problem.

In this paper, we investigate a novel optimal replica placement problem where both QoS and fault-tolerant constraints must be met simultaneously, and propose heuristic algorithms to tackle the problem. The major contributions of our work are as follows.

1) Formulate a novel optimal replica placement problem with both QoS and fault-tolerant constraints.
2) Prove the problem is NP-hard.
3) Develop an interesting greedy replica placement algorithm.
4) Prove the proposed greedy algorithm actually is 2-approximate.

II. Related Work

Sahoo et.al [12] provides a comprehensive survey of data replica in CDN system. Aral et.al [1] summarizes the replica placement in even broader environments and applications, which could be centralized or decentralized, with complete or partial information, and static or dynamic. These surveys show that replica placement problems vary from each other mainly to meet the varying underneath constraints. Most replica placement algorithms take QoS as their constraint or optimization object due to the importance of QoS under different scenarios. Tang et al. [13] 's work is an early and typical replica placement research that presents their algorithm solution based on general problem formulation with QoS constraint. At the same time, there are many QoS related replica placement researches focus on algorithm efficiency under practical and specific application scenarios [4] instead of analyzing the problem's hardness under general cases.

There are also replica placement problems subject to constraints related to fault tolerance. Khan et al. [9] developed the replica placement techniques to guarantee a fault-tolerant performance under the uncertainties of arbitrary server failures. However, this work did not consider the QoS requirement. There are some further research [14] [11] on fault-tolerant requirement.

In principal, the QoS requirement will reshape each node's local topology and fault-tolerant requirement will entail redundant resources. So the QoS and fault-tolerant constrains together will make the problem very different from all existing

otherwise ones. Up to date, however, little research has been found on optimal replica problem subject to both QoS and fault-tolerant constraints at the same time. In this paper, we will formulate the novel problem, analyze its hardness and propose our solution.

III. PROBLEM DESCRIPTION

The replicated infrastructure considered in this work consists of multiple geographically distributed servers that can be classified into two categories: the servers with a replica placement and the servers serving as proxies only. A proxy server doesn't have a replica on itself, but it can delegate requests to other replica servers in its close vicinity. Each node is associated with a certain cost for a replica placement on it.

QoS requirement can be distance, response time, latency or throughput, depending on the concerned environment and application. We allow QoS to be individualized (localized) and the situation where a uniformed QoS constraint across all nodes is a naive case of our problem.

A fault-tolerant network means that it can continue to provide service in the presence of failures. Failures may happen at different levels of a network. In our work we consider failures at the replica level rather than at the node level or even lower levels. The fault-tolerant requirement is also allowed to vary from node to node.

A. System model

We model the replicated network topology using a connected undirected graph $G = (V, E)$, where $V = \{v_0, v_1, \ldots, v_{N-1}\}$ is the set of nodes whose cardinality is N, which can also be simply represented by the unique values from 0 to $N-1$. And $E = \{(u, v) | u, v \in V\}$ is a set of edges over V. Each node $v \in V$ is an abstract representation of the site that can be placed replica. An edge $e = (u, v) \in E$ represents that there is $direct$ link between the two sites identified by u and v.

For a pair of nodes v and u belonging to V, we use $l(u, v)$ or $l(v, u)$ to represent the distance of the direct link between them. If there is no direct link from v to u, the value of $l(v, u)$ would be infinite and we define $l(v, u) = \infty$. Let L denote the set of weights of all these direct links. We use $d(u, v)$ to represent the shortest distance among all paths between v and u. Since the graph is connected, $d(u, v)$ always exists.

For each node v, we introduce a parameter $s(v)$ to represent the total cost for placing a replica on the node v.

Now let us consider QoS requirement for each node. We use $q(v)$ to quantitatively describe the QoS requirement of a node v. Without loss of generality, here $q(v)$ is defined as the largest distance that is allowed for the node v to possibly get replica service from other nodes. If a node v has a replica, it can meet the request immediately; if a node v does not have a replica, it should send requests to a nearby node with a replica placed, denoted by u, that satisfies $d(v, u) \leq q(v)$. We use Q to represent all the QoS requirements of all the nodes.

Because a network is composed of N nodes, the entire replication scheme can be represented by a vector $X =< x_0, x_1, \ldots, x_{N-1} >$ where x_i gets 1 if a replica is stored at node v_i, and 0 otherwise. Let R represent the set of nodes with replicas placed. Then the total cost C of the replicated network is simply calculated by the following objective function: $C(R) = \sum_{i=0}^{N-1} x_i \times s_i$.

Furthermore, we use another parameter $m(v)$ to stand for the fault-tolerant level of node v and all nodes are represented by M. More specifically, let $mrft(v)$ represent the maximum number of replica failures the node v can tolerate, then we define $m(v) = mrft(v) + 1$. It means that in order to tolerate $m(v) - 1$ arbitrary replica failures, at least $m(v)$ replicas should be accessible for the node v. When $M =< 1, ..., 1 >$ for all the nodes, the network has zero failure tolerance, which can only happen in a replica-fault-free system.

In summary, we have established the following concepts and notations to describe the network in the rest of the discussion. Let $G = (V, E)$ be a graph representing a replicated network, where $V = \{v_0, v_1, \ldots, v_{N-1}\}$ represents the set of nodes of the graph, and E is the set of the edges of the graph built over V. Let $L =< l_0, l_1, \ldots, l_{|E|-1} >$ be the weights associated with each $e \in E$. Let $S =< s_0, s_1, \ldots, s_{N-1} >=< s(v_0), s(v_1), \ldots, s(v_{N-1}) >$ be the cost vector associated with each $v \in V$. Let $Q =< q_0, q_1, \ldots, q_{N-1} >=< q(v_0), q(v_1), \ldots, q(v_{N-1}) >$ stand for the QoS requirements on V with q_i corresponding to the QoS requirement of v_i. Let $M =< m_0, m_1, \ldots, m_{N-1} >=< m(v_0), m(v_1), \ldots, m(v_{N-1}) >$ stand for the fault-tolerant requirements on V with m_i corresponding to the fault-tolerant level of node v_i. R refers to a set of nodes which are placed replicas. M refers to the whole vector of fault-tolerant levels. The same convention also applies to S, Q, L and other notations. In the following discussion, all these notations will be used whenever they don't cause any ambiguity.

B. Optimal QoS-aware and fault-tolerant replica placement

Given a graph $G = (V, E)$ with each node annotated by s, q and m values, and each edge annotated by l, if any replica placement solutions exist for the network, then the objective of the problem is to find a replica placement solution R whose total cost of replicas is minimized subject to the constraints of QoS and fault tolerance of every node at the same time. Here we name it as OQFRP problem. Alternatively, the proposed problem can also be modeled as an optimization problem whose objective is to minimize the total cost of the network subject to the Q and M constraints:

$$Min : C(R) = \sum_{i=0}^{N-1} x_i \times s_i, \text{subject to} \tag{1}$$

$$\sum_{d(i,j) \leq q(i), \forall j \in \{0,\ldots,N-1\}} x_j \geq m_i, \forall i \in \{0, ..., N-1\}, \text{and} \tag{2}$$

$$x_i \in \{0, 1\} \tag{3}$$

The problem turns out to be an integer linear programming problem, which, generally being NP-hard, suggests that our

problem is likely also NP-hard. In next subsection, we will provide a proof the NP-completeness of the decision version of the OQFRP problem.

C. Intractability of the problem

The decision form of the OQFRP, denoted by $DOQFRP$, asks whether or not a network has a replica solution that has the cost less or equal to a specified value k:

$$\{< G' = (V', E'), L, Q, M, k > | \exists R(V') \wedge \sum_{v \in R(V')} s(v) \leq k\}$$

It basically asks that given a k and a graph G that satisfies the preconditions L, Q, and M [1], whether there exists a replica placement solution set $R(V')$ which has a total cost less than or equal to k.

We now show that DOQFRP is NP-complete. First, it is easy to see that DOQFRP is in NP, since given a solution R and k, we know these input can always be encoded in polynomial length and whether or not the total cost of R is less than k can always be verified in polynomial time. Then, we show that DOQFRP can be reduced from the dominating set problem (DS), a well-known NP-complete problem, i.e., $DS \leq_p DOMQRP$.

Dominating set(DS): For a graph $G = (V, E)$, a Dominating Set $DS(V)$ of G is defined as a subset of V such that $\forall v \in (V - DS(V))$ is adjacent to $\exists v' \in DS(V)$. It can be formally described using the following language.

$$\{< G = (V, E), k > | \exists DS(V) \wedge |DS(V)| \leq k\}$$

First, we show that a polynomial time reduction [5] can be constructed from any instance G of DS into an instance G' of DOMQRP problem using the following straightforward method. For $\forall v \in V$, we create a node $v' \in V'$, and assign $q(v') = 1$, $m(v') = 1$ and $s(v') = 1$ to the node. For every edge in E, we create an edge for E' and associate the edge with $l(u, v) = 1$, Obviously this reduction can be carried out in polynomial-time. We then need to show that a positive solution to one problem implies a positive solution to the other problem under the above described polynomial time transformation. More specifically, we need to show the instance $< G' = (V', E'), L, Q, M, k > \in DOMQRP$ has a replica set of whose total cost is no more than k if and only if the instance $< G = (V, E), k > \in DS$ has a dominating set of which the cardinality is no greater than k.

Sufficient Condition $\rightarrow$ If there exists a dominating set $DS(V)$ for G with cardinality less than k for G, for each $v \in DS(V)$, following the above construction mapping scheme, we can always mark the mapped node $v' \in V'$. Since the mapping is a one-to-one relationship and the costs, links, QoS, and fault-tolerant level in G's are all unit values, we can claim that all the marked v's constitute the $R(V')$ that is a replica solution set whose total cost is less than k for G'.

[1]For those graphs that inherently cannot meet the preconditions of Q, L M, it is trivial to conclude that no feasible solutions can be found.

Necessary Condition $\leftarrow$ If $R(V')$ exists for G', following the above one-to-one mapping construction, for each node $v' \in R(V')$, we can mark a node v in V on the graph G. Then we claim all the marked nodes form the $DS(V)$ for G. Since every node $v' \in V' - R(V')$ can be directly linked to at least a node in $R(V')$ due to unit M, QoS, links and costs, a node not in $DS(V)$ must have a neighbor in $DS(V)$. Clearly, $|DS(V)| \leq k$ due to the one-to-one mapping of the transformation. So we can find a solution $DS(V)$ mapped from $R(V')$ for the reduced dominating set instance.

Hence we have proved that $DOQFRP$ is NP-complete, which immediately follows that $OQFRP$ is NP-hard.

D. Influencing Set and Influenced Set

Two critical data structures, which will be used in the subsequent sections to facilitate the analysis of the problem, are defined here.

Influencing set: $\forall v \in V$, the influencing set of v is the set of nodes u whose distances to v are within the QoS requirement of v.

$$IG(v) = \{u | d(v, u) \leq q(v)\}.$$

Influenced set: $\forall v \in V$, the influenced set of v is the set of nodes u whose distances to v are within the QoS requirement of u.

$$ID(v) = \{u | d(v, u) \leq q(u)\}.$$

From the perspective of a given node v, $IG(v)$ denotes a set of nodes that can meet the QoS requirement of v, which means if any member in $IG(v)$ is placed replica, it can serve the request of node v. $ID(v)$ denotes a set of nodes whose QoS requirements can be met by v, which means if v is placed a replica, those nodes in $ID(v)$ will be served. From fault tolerance perspective, $IG(v)$ denotes a set of nodes whose failures may influence node v; while $ID(v)$ denotes a set of nodes whose service may be influenced by failure of replica on node v. Please see Fig.1 (a) and (b) for the highlighted illustrations of the two sets for node v_1. Since $d(v_1, v_1) = 0$, $d(v_2, v_1) = 6$, $d(v_5, v_1) = 13$ and $d(v_6, v_1) = 14$ all can meet the QoS requirement of $q(v_1) = 15$, $IG(v_1) = \{v_1, v_2, v_5, v_6\}$. Similarly, $d(v_1, v_1) = 0 < q(v_1) = 15$, $d(v_1, v_2) = 6 < q(v_2) = 10$, $d(v_1, v_3) = 16 < q(v_3) = 17$, $d(v_1, v_4) = 22 < q(v_4) = 24$ so the influenced set $ID(v_1) = \{v_1, v_2, v_3, v_4\}$.

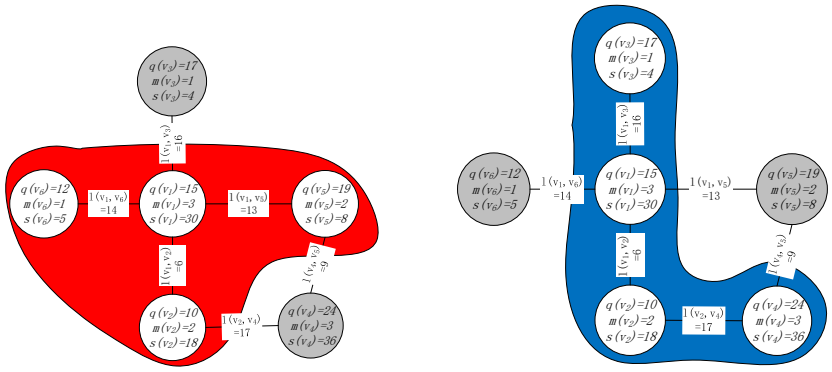
E. The essential properties of the problem

The following properties of the problem and the interplay between parameters are very important for us to design effective heuristics.

1) *Existence of a solution.* Without considering cost constraints, QoS can always be guaranteed. Given fault-tolerant levels $M =< m_0, m_1, \ldots, m_{N-1} >$, the necessary and sufficient conditions for the existence of a replica placement solution R is as follows:

$$m(v) \leq |IG(v)|, \forall v \in V.$$

This property reveals the essential relationship between M and Q.



(a) The Influencing Set of v_1

(b) The Influenced Set of v_1

Fig. 1. (a) An example of the influencing set $IG(v_1) = \{v_1, v_2, v_5, v_6\}$ of node v_1 highlighted in red; (b) An example of the influenced set $ID(v_1) = \{v_1, v_2, v_3, v_4\}$ of node v_1 highlighted in blue.

2) *Relationship among M, Q and R.* Furthermore, if a solution R exists, then for every node v the following inequality holds.

$$m(v) \leq |IG(v) \cap R|, \forall v \in V.$$

It means that given node v, $m(v)$ must not exceed the cardinality of the replica placed nodes belonging to the influencing set of v, since the fault-tolerant level of a node v simply refers to the number of replicas being placed on a subset of $IG(v)$. The size of the replica solution for the whole network should be no smaller than $m(v)$ of any node v and the replicas that meet m requirements for node v should be the intersection of $IG(v)$ and R.

3) *Lower bound and upper bound of M.* Lower bound (or the lowest level) and upper bound (or the highest level) of fault tolerance are the limitations inherent in the graph subject to QoS constraints. The lowest level of fault tolerance occurs when there is no redundant replica. It means that the fault-tolerant value of every node is assigned with 1. Thus, the lower bound of fault-tolerant value is $M =< 1, 1, \ldots, 1 >$. The highest fault-tolerant value a network can achieve occurs at the most aggressive case where all the nodes of the influencing set of any v are placed replicas. As a result, the upper bound for the fault-tolerant value of the whole network system is $M =< |IG(v_0)|, |IG(v_1)|, \ldots, |IG(v_{N-1})| >$. The two bounds show the range of M values. This analysis also shows that in our problem the QoS and fault-tolerant requirements are not independent, they will interact with each other.

IV. Approximation Algorithm

The basic idea of the proposed 2-approximation algorithm includes two phases. The first phase is relaxing the binary integer constraints $x_i \in \{0, 1\}$ of Eq.3 as $0 \leq x_i \leq 1$ so the proposed integer programming problem can be changed into a linear programming problem. Based on this relaxation method the linear programming problem can be solved by employing the existing polynomial time complexity method. According to the optimal linear programming solution, during the second phase, we develop an efficient rounding algorithm that can meet all the integer constraints and the total cost is no more than 2 times of the optimal linear programming solution. In this section, we will give the detailed description of the rounding algorithm and then we prove that the approximation factor is no more than 2.

A. Two Steps Rounding

We develop a two steps rounding mechanism to implement our rounding algorithm. The pseudo-code description is given in Alg. 1. During the first step , the value of linear programming solution $LX =< LX_0, ..., LX_{N-1} >$ will be used to decide which fractional value will be rounding to 1. The proposed *half rounding* mechanism works in this way. If $LX_i \geq 0.5$, we will round this fractional value to 1 ($IX_i = 1$ and the initial value of $IX =< 0, .., 0 >$). The first *half rounding* mechanism will generate the decision vector IX and it can guarantee that its worst cost is no more than 2 times of the linear optimal solution LX. If the rounding result IX cannot meet all vertexes' constraints, we will enter the second step and employ the *cheapest amortized cost rounding* mechanism to select the fractional value $LX_i \in LX and LX_i < 0.5$ rounding to 1 and meet the constraints of the integration programming.

The basic idea is that for any unsatisfied vertex v, we will select the first $m(v)$ cheapest amortized cost replicas (see definition IV.3) to meet it and round the corresponding fractional values to 1. The *cheapest amortized cost rounding* mechanism will be repeated until all the vertexes can be met by IX.

B. Approximation factor analysis and proof

We will prove that the proposed two step rounding mechanisms have a 2-approximation factor. Since our rounding mechanisms includes two mechanisms, *half rounding* and

input : $G = (V, E), Q, M, S,$
$LX =< LX_0, ..., LX_{N-1} >$
output: a 2-approximation replica placement solution
$IX =< IX_0, ..., IX_{N-1} >$

Initialize $IX =< 0, ..., 0 >$;
for $(i = 0 : N - 1)$ **do**
  if $(LX_i \geq 0.5)$ **then**
    $IX_i = 1$
  end
end
if *(IX can meet all constraints)* **then**
  return IX
end
else
  Let U_R be the set of vertexes whose fractional values have not been rounded to 0.5;
  Let U_S be the set of vertexes which have not been satisfied;
  Build influencing set $IG(u)$ for any element $u \in V$;
  Build influenced set $ID(u)$ for any element $u \in V$;
  Calculate amortized cost $sa(u)$ for any vertex $u \in U_R$;
  while *($U_S! = \phi$)* **do**
    Remove any vertex u from U_S;
    Select the first $m(u)$ cheapest amortized cost vertexes in $IG(u)$ and set corresponding elements of IX to 1;
  end
  return IX
end

Algorithm 1: The two step rounding mechanisms

cheapest amortized cost rounding, we will prove that total cost of *half rounding* will not more than two times of the optimal linear programming's cost and the result of *cheapest amortized cost rounding* will not further increase the total cost of the first step. So the total cost of our rounding mechanism has a 2-approximation factor. At the same time, we will prove that by combining the two rounding mechanisms together, we will generate a feasible solution.

First, we will prove the *half rounding* theorem.

Theorem IV.1 (2 boundary of *half rounding*)**.** *The cost of* half rounding *mechanism for replica placement is no more than twice cost of the optimal linear programming solution.*

Proof. Let

$$LX =< LX_0, ..., LX_{N-1} >,$$

$0 \leq LX_i \leq 1, 0 \leq i \leq N-1$ be the solution of optimal linear programming and

$$IX =< IX_0, ..., IX_{N-1} >,$$

if $0.5 \leq LX_i$ then let $IX = 1$, otherwise $IX = 0, 0 \leq i \leq N - 1$ be the result of *half rounding*.

So the total cost of IX is

$$\sum_{i=0}^{N-1} IX_i \times s_i \leq \sum_{i=0}^{N-1} 2 \times LX_i \times s_i = 2 \times Cost(Opt).$$

Here $Cost(Opt)$ means the optimal cost of linear programming.

□

The *cheapest amortized cost rounding* mechanism's proof includes two parts. In the first part we will prove this method can generate a feasible integer programming solution. In the second part we will prove that the 2-approximation factor holds.

Lemma IV.2 (Feasibility of *cheapest amortized cost rounding*)**.** *The* cheapest amortized cost rounding *mechanism can generate a integer solution IX based on the optimal linear programming solution LX.*

Proof. For any vertex $v \in V$, if v is not satisfied by IX, it means that the number of rounded element k_1 in $IG(v)$ is less than $m(v)$. So there are additional $m(v) - k_1$ replicas will be needed. Let k_2 be the number of replicas whose fractional values are less than 0.5. Then $0.5 \times k_2 \geq m(v) - k_1$. This means that k_2 should be no less than 2 times $m(v) - k_1$. In other words, we will have at least twice times number of replicas that can be rounded to meet the requirement of $m(v)$. So the *cheapest amortized cost rounding* mechanism can continue to select the cheapest amortized cost replica until the replica fault-tolerant requirement $m(v)$ can be met. Under this condition $m(v)$ vertexes in $IG(v)$ will be put a replica and their fractional values will be rounded to 1. So vertex v will be satisfied by IX after the *cheapest amortized cost rounding* mechanism. □

We will employ a new metric named *amortized cost* to generation the rounding solution and prove this rounding procedure will also have a 2-approximation factor.

Definition IV.3. *Amortized Cost: For an optimal replica placement solution IX, if any vertex $v \in V$ has cost $s(v)$ and there is $EID(v) \subseteq ID(v)$ be the largest subset of $ID(v)$ and for any vertex in $EID(v)$ must pay the fee of the replica on node v to share the service and meet its failure tolerance requirement, then the cost $s(v)$ can be evenly amortized among $|EID(v)|$ vertexes and each vertex will share the cost as low as $sa(v) = \frac{s(v)}{|EID(v)|}$. $sa(v)$ is defined as the amortized cost of the replica on vertex v. Under this case, $EID(v) = ID(v)$.*

Based on the optimal solution LX of relaxed linear programming problem, if we reduce the cost of each vertex $v \in V$ from $s(v_i)$ to $LX_i \times s(v_i)$ and round all no zero element of LX to 1, then we will generate a solution for the cost reduced integer programming problem that has the same total cost. Our *cheapest amortized cost rounding* will be employed after the *half rounding* results. We will update the graph before the following steps. $\forall v \in V$, we will let $m(v) = m(v) -$number of rounded vertexes in $IG(v)$ and then remove the rounded

vertexes in $IG(v)$. In the following we will only consider the un-rounded vertexes.

Theorem IV.4 (The *cheapest amortized cost rounding* has a 2-approximation factor.)**.** *The cost of* cheapest amortized cost rounding *mechanism for replica placement has no more than twice cost of the optimal linear programming solution.*

Proof. Let all the elements in vertex v's influencing set $IG(v)$ have been sorted in amortized cost increasing order. So $\forall u_i, u_j \in IG(v)$, if $i \leq j$, then $sa(u_i) \leq sa(u_j)$. We will round the fractional value replica one by one until there is any vertex can be met. The *cheapest amortized cost rounding* procedure will be repeated until no unsatisfied vertex. We can prove that for any new satisfied vertex v, the following amortized cost based inequality holds.

$$\sum_{u_j \in IG(v_i), j=0}^{m(v_i)-1} (1 - LX_{u_j}) \times sa_{u_j} \leq \sum_{u_j \in IG(v_i), j=m}^{|IG(v_i)|-1} LX_{u_j} \times sa_{u_j}. \quad (4)$$

Since LX is an optimal solution, $\forall v \in V$, it must meet the constraint

$$\sum_{u_j \in IG(v), j=0}^{|IG(v)|-1} LX_{u_j} \geq m(v).$$

This inequality can be rewritten as the following form.

$$\sum_{u_j \in IG(v), j=m}^{|IG(v)|-1} LX_{u_j} \geq m(v) - \sum_{u_j \in IG(v), j=0}^{m(v)-1} LX_{u_j} = \sum_{u_j \in IG(v), j=0}^{m(v)-1} (1 - LX_{u_j}).$$

If we let $sa_{max} = max\{sa_{u_0}, ..., sa_{u_{m-1}}\} = sa_{u_{m-1}}$, then we will have the following results.

$$\sum_{u_j \in IG(v), j=0}^{m(v)-1} (1 - LX_{u_j}) \times sa_{u_j} \leq \sum_{u_j \in IG(v), j=0}^{m(v)-1} (1 - LX_{u_j}) \times sa_{max} \leq \sum_{u_j \in IG(v), j=m}^{|IG(v)|-1} LX_{u_j} \times sa_{max} \leq \sum_{u_j \in IG(v), j=m}^{|IG(v)|-1} LX_{u_j} \times sa_{u_j}.$$

Let $u_{j_{max}}$ be the rounded vertex replica which has the largest amortized cost in $IG(v)$. For any vertex v if $m(v) - 1 < j_{max}$, this means that the vertex is overprovisioned. For overprovisioned case, Eq.4 can be adjusted as follows.

$$\sum_{u_j \in IG(v), j=0}^{u_{j_{max}}} (1 - LX_{u_j}) \times sa_{u_j} \leq \sum_{u_j \in IG(v), j=j_{max}+1}^{|IG(v_i)|-1} LX_{u_j} \times sa_{u_j} + \sum_{u_j \in IG(v), j=m(v)}^{j_{max}} sa_{u_j} \quad (5)$$

We will sum all those inequalities' left hand side and right hand side together. Let FV be all the fractional value vertex and they are sorted in the amortized cost increasing order. Let $u_{g_{max}} \in FV$ be the rounded vertex replica with the largest amortized cost and OP be all the overprovisioned vertexes. Based on the definition of amortized cost, we will have the following results.

$$\sum_{v \in NS} \sum_{u_j \in IG(v), j=0}^{j_{max}} (1 - LX_{u_j}) \times sa_{u_j} = \sum_{u_j \in FV, j=0}^{g_{max}} (1 - LX_{u_j}) \times s(u_j) \leq \sum_{v \in NS} \sum_{u_j \in IG(v), j=j_{max}+1}^{|IG(v)|-1} LX_{u_j} \times sa_{u_j} + \sum_{v \in NS} \sum_{u_j \in IG(v), j=m(v)}^{j_{max}} sa_{u_j} = \sum_{u_j \in FV, j=g_{max}+1}^{|FV|-1} LX_{u_j} \times s(u_j) + \sum_{u_j \in OP} s_{u_j} \leq \sum_{u_j \in FV, j=g_{max}+1}^{|FV|-1} LX_{u_j} \times s(u_j) \times 2$$

It means that the *cheapest amortized rounding*'s added cost is no more than twice of the removed cost. □

V. CONCLUSION

Existing replica problem research only focuses on either QoS or fault tolerance. However, as more and more contemporary applications have become increasingly distributed and decentralized, the need for more sophisticated replica problems to consider both QoS and fault tolerance simultaneously is on the rise.

In this work, we propose a unique graph model to capture all attributes of replica cost, QoS and fault tolerance at the individual node and link level. Based on the proposed model, we formulate a novel optimal replica placement problem and prove that the problem is NP-hard. In order still solve the problem, we proceed to propose a novel greedy algorithm that consists of two hueristic rounding methods. Finally, the algorithm has been proved to enjoy a 2-approximate ratio.

REFERENCES

[1] Atakan Aral and Tolga Ovatman. A decentralized replica placement algorithm for edge computing. *IEEE transactions on network and service management*, 15(2):516–529, 2018.

[2] Moses Charikar, Sudipto Guha, Éva Tardos, and David B Shmoys. A constant-factor approximation algorithm for the k-median problem. *Journal of Computer and System Sciences*, 65(1):129–149, 2002.

[3] Gérard Cornuéjols, George Nemhauser, and Laurence Wolsey. The uncapicitated facility location problem. Technical report, Cornell University Operations Research and Industrial Engineering, 1983.

[4] Zhihui Du, Jingkun Hu, Yinong Chen, Zhili Cheng, and Xiaoying Wang. Optimized qos-aware replica placement heuristics and applications in astronomy data grid. *Journal of Systems and Software*, 84(7):1224–1232, 2011.

[5] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[6] Hamoun Ghanbari, Marin Litoiu, Przemyslaw Pawluk, and Cornel Barna. Replica placement in cloud through simple stochastic model predictive control. In *2014 IEEE 7th International Conference on Cloud Computing*, pages 80–87. IEEE, 2014.

[7] R Kingsy Grace and R Manimegalai. Dynamic replica placement and selection strategies in data grids—a comprehensive survey. *Journal of Parallel and Distributed Computing*, 74(2):2099–2108, 2014.

[8] Ibrahim Abaker Targio Hashem, Victor Chang, Nor Badrul Anuar, Kayode Adewole, Ibrar Yaqoob, Abdullah Gani, Ejaz Ahmed, and Haruna Chiroma. The role of big data in smart city. *International Journal of Information Management*, 36(5):748–758, 2016.

[9] S. U. Khan, A. A. Maciejewski, and H. J. Siegel. Robust cdn replica placement techniques. In *the 23rd IEEE Int. Parallel Distrib. Process. Symp. (IPDPS)*, pages 1–8, Rome, Italy, 2009.

[10] Andrew Lim, Brian Rodrigues, Fan Wang, and Zhou Xu. k-center problems with minimum coverage. *Theoretical Computer Science*, 332(1-3):1–17, 2005.

[11] Kenneth Alex Mills et al. *Algorithms for Optimal Replica Placement in Data Centers*. PhD thesis, 2017.

[12] Jagruti Sahoo, Mohammad A Salahuddin, Roch Glitho, Halima Elbiaze, and Wessam Ajib. A survey on replica server placement algorithms for content delivery networks. *IEEE Communications Surveys & Tutorials*, 19(2):1002–1026, 2016.

[13] X. Tang and J. Xu. QoS-aware replica placement for content distribution. *IEEE Transactions on Parallel and Distributed Systems*, 16(10):921–932, Oct 2005.

[14] Abdullah Yousafzai, Abdullah Gani, and Rafidah Md Noor. Availability aware continuous replica placement problem. *arXiv preprint arXiv:1605.04069*, 2016.

[15] Steven Sounyoung Yu and Sounil Yu. Autonomous unmanned road vehicle for making deliveries, 2015. US Patent App. 14/318,690.