NARA: Network-Aware Resource Allocation Mechanism for Minimizing Quality-of-Service Impact while Dealing with Energy Consumption in Volunteer Networks

Sergio Gonzalo San José[a], Joan Manuel Marquès[a], Javier Panadero[b], Laura Calvet[c]

[a] *IN3 - Computer Science Dept., Open University of Catalonia, Barcelona, Spain*
{sgonzalos, jmarquesp}@uoc.edu
[b] *Computer Architecture and Operating Systems Dept., Universitat Autònoma de Barcelona, Barcelona, Spain*
javier.panadero@uab.cat
[c] *Telecommunications and Systems Engineering Dept., Universitat Autònoma de Barcelona, Barcelona, Spain*
laura.calvet.linan@uab.cat

Abstract

A large-scale volunteer computing system is a type of distributed system in which contributors volunteer their computing resources, such as personal computers or mobile devices, to contribute to a larger computing effort. Volunteer resources are connected over the Internet and together form a powerful computing system capable of providing a service without depending on a service provider. Volunteer network resource allocation is the process of assigning computing tasks or services to a network of volunteer resources. The allocation process includes identifying the needed resources, selecting appropriate volunteers, and assigning tasks or services based on their capabilities. Volunteer computing systems consist of a large number of heterogeneous resources - in terms of processing power, storage, and availability - belonging to different authorities - users or organizations - and exhibiting uncertain behavior in terms of connection, disconnection, capacity, and failure. All of this makes resource allocation a challenging task in terms of ensuring a minimum quality of service, requiring complex algorithms and optimization techniques to ensure that services are efficiently allocated while respecting the constraints of the available resource. This paper introduces the Network-Aware Resource Allocation mechanism, which leverages the location, connectivity, and network latency of volunteer nodes to minimize the time a service runs with degraded quality of service and aims to deal with the energy consumption resulting from data replication requirements. This resource allocation mechanism applies to both the initial deployment of the service in the network and to the reallocation of nodes in the event that one of the allocated nodes fails or becomes unavailable. Our method has been validated in a simulation environment of a realistic volunteer system. The analysis of the results shows how our mechanism meets the quality requirements of users while minimizing the synchronization and replication times of service data replicas, as well as the time that services run with degraded quality of service, reducing the times by more than 70% in the service deployment phase and by more than 60% in the service execution phase. It also helps to reduce overall energy consumption.

Keywords: Resource Allocation Methods, Volunteer Computing System, Network-aware Service Deployment, QoS Impact Minimization, Energy Consumption Reduction

1. Introduction

A large-scale Volunteer Computing System (VCS) [1] refers to a distributed computing infrastructure that relies on the collective computational power of volunteers who donate their spare computing resources to host services, process data, or perform computational tasks. These resources typically have high heterogeneity and uncertain behavior regarding availability, connectivity, and failure. Because networks provide access to a large pool of computing resources without investing in expensive hardware or commercial computing resources, they can be used by individuals, organizations, research groups, or academic institutions to provide services or perform complex tasks that require significant computing resources.

In this regard, Resource Allocation (RA) mechanisms play a crucial role, as they must balance the availability, scarcity, and reliability of the underlying resources. In [2], we introduced the CLARA mechanism, which leverages most of the low-available nodes (traditionally discarded or used at high cost) as part of the resource pool for service provisioning, thus increasing the overall capacity of the network. The main goal of this mechanism is to ensure the minimum Quality of Service (QoS) required by a User Service (US) deployed on volunteer nodes while minimizing the number of replica reallocations and maximizing the utilization of volunteer resources. The mechanism uses node availability and occupancy as resource selection criteria. However, other relevant network-related criteria that may affect the QoS degradation time in the US are not considered.

In this context, our motivation is to explore how network properties can be utilized to allocate resources more efficiently in large-scale VCSs. We hypothesized that 1) Network awareness and node capabilities may be considered in RA algorithms to minimize the US QoS degradation time, and 2) the network latency between nodes can be estimated based on the Euclidean distance between network coordinates, considering a Network Coordinate System (NCS) that allows to group nodes by latency levels. Hence, this paper introduces the Network-Aware Resource Allocation (NARA) mechanism, an advanced and powerful mechanism that extends the CLARA mechanism, integrating additional criteria into the RA process. The NARA mechanism uses network and node parameters to minimize the US QoS degradation time, i.e., when the US runs with a QoS below a threshold due to the unavailability or failure of the nodes currently assigned to it. Incorporating these network characteristics into the RA mechanism can reduce overall energy consumption and support deploying USs sensitive to latency between US replicas distributed in the network.

To the best of our knowledge, this paper is the first to address the combination of network awareness and node capabilities for RA with consideration of objectives and constraints specific to voluntarily contributed resources. The main contributions of this study are:

- A novel clustering-based RA mechanism that uses network and node capabilities to select resources smarter and more energy-efficiently, reducing QoS degradation time and network latency. This enables the management of various QoS levels and the deployment of USs sensitive to latency between replicas.

- A process for estimating the network latency between any pair of network nodes and incorporating it into the sorting and selection criteria of the RA mechanism. This process relies on an NCS that groups nodes by latency levels, estimating the latency between nodes based on the Euclidean distance between their network coordinates. In addition, a cache map and a lazy estimation process for network latency are defined to prevent performance degradation as the number of network nodes increases, without compromising the scalability of the RA mechanism.

- A discussion on the relationship between network latency and energy consumption in replication and periodic synchronization between data replicas. It presents an analysis of the rationale that supports the idea that energy consumption is reduced by minimizing aggregated network latency in data replication between geographically distributed replicas and, consequently, how NARA leads to a more responsible use of energy.

- A comparison with the CLARA mechanism regarding replication and QoS degradation times in the initial deployment and reassignment phases. It was conducted through simulation using a Twitter-like decentralized implementation of a microblogging social network. The simulation scenarios aim to evaluate the addition of various network characteristics into the nodes selection criteria and their impact on the QoS degradation time.

The remainder of this paper is organized as follows. Section 2 presents a literature review on how the network's properties and energy consumption are related to different types of distributed networks. The context of the CLARA mechanism is described in Section 3. The formal problem description is described in Section 4. Section 5 describes the approach of the NARA mechanism to estimate the latency between any pair of nodes in the network. Section 6 describes the NARA mechanism and the RA phases. The simulation environment and the complete set of experiments and analyses are described in Sections 7 and 8, respectively. Finally, the main conclusions are drawn, and future research lines are identified in Section 9.

2. Related Work

Quality of Service (QoS) is crucial in large-scale VCSs to ensure a good user experience, efficient communication, and resource utilization. However, QoS may be affected by network distance, congestion, and device processing capabilities. On the other hand, energy consumption in transit refers to the energy consumed by network devices to transmit and process data packets. This energy consumption varies depending on network efficiency, workload, topology, routing, and environmental factors that affect network latency. Therefore, a strong relationship exists between network device capabilities, node capabilities, network latency, and energy consumption in transit that can impact QoS. Thus, efforts must balance low latency and high throughput to reduce energy consumption during transit. Network devices are optimized to reduce power consumption using energy-efficient processors or dynamic power management techniques. At the same time, routing and congestion control algorithms are also used to improve network energy efficiency. Several studies have proposed different approaches for QoS management in RA processes and the balance between power reduction and low latency requirements.

2.1. Approaches for QoS Management in Resource Allocation Processes

Many authors propose innovative approaches in various network environments and collectively emphasize the importance of improving system performance and user experience.[3] focuses on efficient RA in dynamic mobile communication networks, introducing stochastic learning automata and continuous-time optimization techniques. In [4], the authors introduce a QoS-based successive interference cancellation (SIC) order in uplink non-orthogonal multiple access systems and RA to maximize the rate. Energy efficiency is achieved in multi-tier Heterogeneous Networks by employing a systematic methodology in [5] using a model that considers the spatial distribution of network elements. In this regard, [6] discusses the importance of increasing energy efficiency and reducing the packet loss rate in wireless communications. The authors survey scheduling and RA algorithms, proposing a new framework and user priority metrics. The RA in energy harvesting systems for Internet of Things (IoT) systems, focused on real-time periodic task allocation with QoS, is deeply analyzed in [7]. [8] introduces a novel QoS ant colony optimization metaheuristic algorithm for task scheduling in cloud computing environments.

A RA algorithm is proposed in [9] to address the challenge of channel access and completing offloading computation within specified timeframes for Mobile Edge Computing (MEC). [10] work introduces a distributed power control algorithm for the Industrial IoT. By formulating a long-term time-averaged optimization problem and presenting a real-time decomposition-based approximation algorithm, the solution ensures adaptive RA, queue stability, and meeting QoS constraints with high energy efficiency. A novel machine learning-based algorithm for traffic priority prediction, resource partitioning and user allocation to meet QoS specifications in network slicing in 5G networks is proposed in [11]. Finally, [12] analyzes a resource segmentation mechanism for 5G networks that relies on the accuracy of position coordinates for input parameters such as signal interference and noise ratio.

2.2. *Approaches to Reduce Energy Consumption and Low Latency*

[13] explores the integration of intelligent reflecting surface (IRS)-assisted unmanned aerial vehicle communications to minimize power consumption while maximizing network throughput and reliability through joint optimization of RA. Meanwhile, [14] introduces an energy-aware approach to reduce network latency in network slicing-enabled edge computing environments by dynamically allocating network resources. [15] proposes cooperative activation and caching strategies to improve the latency and energy efficiency of small cell networks. Other approaches focus on packet processing to minimize network energy consumption and latency, as proposed in [16]. In [17], the authors investigate optimizing energy-efficient computing environments through dynamic RA in EC, aiming to enhance system efficiency and energy economic performance. [18] investigates the impact of network latency on user performance, focusing on using navigation aids under different network latency conditions. On the other hand, [19] proposes a geographically distributed placement model that considers location, energy, carbon, and cost awareness to minimize energy consumption, carbon emissions, and costs while ensuring performance requirements. [20] focused on designing a nationwide edge infrastructure and its behavior under different electrical configurations, including techniques for allocating resources in edge data centers and reducing environmental impact. [21] shows how caching solutions can reduce network traffic, improve energy efficiency, and enhance user experience in MEC networks. Finally, [22] uses neural networks to optimize transmission parameters, ensuring energy-efficient transmission and minimizing energy consumption while maintaining QoS.

2.3. *Insights and Strategies to Minimize Network Latency and Energy Consumption*

[23] presents joint QoS-aware RA and MEC in multi-access heterogeneous networks to maximize total system energy efficiency and ensure users' QoS requirements, while [24] evaluates effective energy efficiency and proposes delay-outage aware RA strategies for energy-limited IoT devices under the finite blocklength regime. [25] addresses the need for efficient communication in the IoT by proposing the optimal energy-efficient RA algorithm aiming to improve task offloading and network performance. [26] proposes a RA algorithm to maximize EE in optical intelligent reflecting surfaces (IRS) and radio frequency IRS-assisted systems. [27] focuses on energy-efficient and dependable RA and task offloading in satellite-assisted IoT networks, proposing an algorithm to optimize resource distribution among IoT devices, addressing the challenge of concave optimization through a transformative approach. [28] proposes a task allocation scheme for energy optimization in FC networks that minimizes energy consumption while maintaining a latency constraint. In addition, [29] presents a compelling argument for new network architectures that offer lower latency and deterministic performance in data centres, while [30] presents a thorough taxonomy of latency compensation techniques for networked computer games. Finally, [31] focuses on energy-efficient heterogeneous cellular networks, formulating the optimization problem as a hybrid joint RA of integer programming and continuous optimization to improve total system energy efficiency while ensuring QoS, while [32] focuses on MEC tradeoffs between energy and latency and proposes a reinforcement learning-based approach for edge server selection and RA.

The literature review shows that QoS, network latency, and energy consumption in RA mechanisms are mostly limited to environments highly dependent on computing and connectivity capabilities. Therefore, to the authors' knowledge, comprehensive work has yet to be devoted to studying the combination of network awareness and node capabilities in RA methods that consider specific energy consumption goals and constraints related to the use of heterogeneous and voluntary resources.

Our research aims to provide a novel RA mechanism that considers network-related properties to perform smarter node selection regarding network proximity. This approach effectively reduces periodic synchronization updates and data replication, minimizing the time services can run with degraded QoS. The network proximity of selected nodes results in lower energy consumption derived from data synchronization between nodes, enabling the deployment of services sensitive to latency between replicas while increasing data transmission efficiency.

3. Context of the CLARA Mechanism

The CLARA mechanism [2] is an RA mechanism that uses a near-optimal method of grouping low-available nodes in complementary relationships to provide combined availability equivalent to a high-available node. The new relationships enable these low-available resources for US deployment, minimizing replica reallocations, reducing the load on high-available nodes, and maximizing the utilization of all nodes in the VCS.

The mechanism categorizes nodes into High-Quality (HQ) nodes and Complementary Nodes (CN). [2] describes how CN nodes are created. Low-Quality (LQ) nodes are grouped into disjoint clusters according to the availability profile exhibited during the Availability Predictor (AP) interval. Thus, LQ nodes in a CN node have complementary availability. The AP interval is based on the availability of nodes exhibited during the last prediction weeks, that is, the number of previous weeks used to predict the nodes' availability in the following weeks.

The required quality of a US is divided between the HQ and CN nodes according to a previously defined distribution. The US QoS (Q_{US}) is the sum of the QoS of the HQ (Q_{USHQ}) and CN (Q_{USCN}) nodes over which the US is deployed. We consider a US to be operating in degraded mode if its QoS is below a specific QoS threshold (T). This threshold indicates the minimum acceptable QoS level for the US.

The mechanism is based on the Multi-Criteria Biased Randomized (MCBR) method [33] to select nodes (Fig.1). From the complete list of active nodes and a set of criteria parameters ranked in importance order, the MCBR method applies an iterative procedure based on ideas of the Lexicographic Ordering (LO) multi-criteria optimization strategy [34] to obtain a reduced set of the best nodes according to each criterion. This method potentially avoids weight factors by incorporating priorities of the individual planning criteria (objective functions) in the optimization process. The LO method assumes that the objectives can be ranked in order of importance (from best to worst). The optimal value is then obtained by minimizing/maximizing the objective functions sequentially, starting with the most important one and proceeding according to the order of importance of the objectives. This multi-objective optimization technique can be represented as an objective function $F(x) = [f_1(x), f_2(x), \ldots, f_N(x)]$ which contains a vector of N individualized functions ($f_i(x)$) ordered by importance, so that $f_1(x)$ is the most important and $f_N(x)$ the least important. The optimal value found for each objective is added as a constraint for subsequent optimizations. Thus, the optimal value of the most important objectives is preserved. Mathematically, this method can be modeled as an ordered sequence of real objective functions with a set of constraints as follows [33]:

$$Min/Max f_i(x) \tag{1}$$

subject to:

$$f_j(x) \leq f_j\left(\overset{*}{j}\right) \tag{2}$$

where $i = \{1, 2, \ldots, N\}$ and $j = \{1, 2, \ldots, i-1\}$.

Thus, as the method progresses down from level 1 to level N, the preceding objective functions are converted to new constraints with boundary values f_j^*, set by the a priori attained solutions $\min\max f_j(x)$ subject to the constraints from the upper level. Accordingly, the number of constraints increases with each level up to $N-1$, gradually reducing the feasible search space in each new level. At each iteration of the procedure, the list of nodes is sorted by one criterion. Next, a biased randomization (BR) [35] mechanism is applied to select a set of the best nodes for that criterion, discarding the remaining nodes. This subset of nodes is provided as input for the next iteration of the procedure, repeating the process for each criterion until a reduced sublist of the best nodes is obtained. Finally, a BR mechanism is applied to the final list to select the nodes to use.

The selection criteria for the HQ and CN nodes are based on node availability. The CLARA mechanism defines an AP to characterize the nodes' availability in the VCS following the insights from [36]. This study showed how their AP methods can reliably guarantee the availability of collections of VCS resources, and how this predictor is particularly useful for service placement in VCSs. The AP interval is based on the availability exhibited by the nodes during the last four weeks, i.e., the number of previous weeks to use to predict the node availability in the following weeks. The NARA mechanism leverages the node behavior and availability characterization performed in the CLARA mechanism. The AP exhibited by a node during the AP interval is converted into a normalized quality value, used to categorize nodes into HQ nodes and LQ nodes. Thus, the node selection process does not consider criteria related to the US QoS degradation. As a result, the cost of node reallocations due to node unavailability or the time the US operates in degraded mode is proportional to the availability of the selected nodes and the time required to replace the failed nodes.

4. Problem Analysis and Description

Large-scale community-owned VCSs [1] use voluntary contributions from participants to host data and services distributed over the Internet without a central authority, resulting in a diverse collection of inexpensive and off-the-shelf equipment.

In this context, a specific application type that can be deployed over volunteer resources is an application centred around distributed storage-related functions. The fluctuating nature of volunteer resources necessitates the maintenance of numerous replicas of the application data to guarantee its accessibility. As a result, the application must be able to oversee multiple data replicas across various nodes and ensure the availability of all replicas. If a data replica becomes inaccessible due to node failure or unavailability, the application data should be replicated to a newly selected node from the surviving nodes.

The balance of cost and availability is a critical goal of these systems. To achieve this, a Centralized Control System (CCS) is integrated. The CCS is a crucial component in managing information systems' deployment and replication processes. It determines which US should be managed by each node and provides relevant information upon request. The CCS also maintains a record of available nodes and decides which US needs to be replicated on which nodes to ensure accessibility. In case of a node failure causing a degraded US QoS, the missing QoS is restored in the following order: 1) the node becomes unavailable, 2) the CCS reports the unavailability during the next availability check, and 3) the CCS updates the US QoS and checks for any impact. If the US QoS is degraded, the CCS selects a new node to replace the failed node and reassigns the US data hosted on the failed node. The synchronization of the US data with the new node is initiated, and the US QoS is updated.

4.1. Problem Formulation and Objectives

Our research aims to propose a new mechanism that outperforms CLARA capabilities by performing a more intelligent RA to minimize the time in which the US operates with degraded QoS. This time can depend on node-specific capabilities and factors related to US characteristics, such as US data. Depending on the US requirements, variable user information may

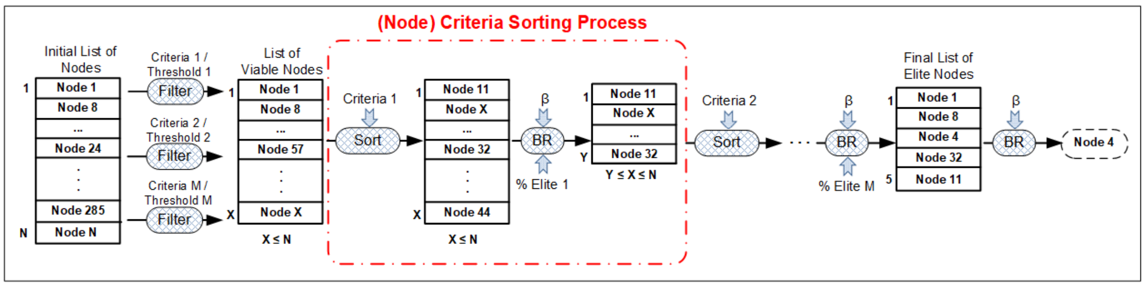


Figure 1: MCBR method overview.

need to be stored. Periodic replication between US-assigned nodes is performed to synchronize updates (data variations with respect to the last synchronization performed) between all replicas. However, if the US is reassigned to a new node, it is necessary to replicate all the data from the nodes previously assigned to the US. Therefore, it can be assumed that the volume of information varies within a specific range. Additionally, network characteristics (e.g., network location and latency) may also affect this time. Each node has a latency (Round Trip Time, RTT) to every other node in the network, which depends on the intermediate network elements, the network communication routing, and the network status, among others.

The US degradation time (T_{DEG}) is defined as the time during which the US QoS falls below the threshold T. This duration covers the time between the unavailability of one or more nodes following a failure and the restoration of US QoS in new nodes. The volatility of nodes and the typical absence of up-to-date information about the network status lead us to limit the scope of analysis to a reduced list of parameters: 1) The network latency between the source node, from which the US reallocation is started, and the destination node (TL), 2) The upload speed (VU) of the source node (O) from which the US reallocation is initiated (VU_O), 3) The download speed (VD) of the destination node (D) on which the US reallocation is performed (VD_D), and 4) the US data size that is transferred between the source and destination nodes (*USdata*). To minimize the US QoS degradation time (T_{DEG}), we consider the sum of two factors: 1) the latency between the source and destination nodes of the reassignment (TL), and 2) the transfer time of the US data, which depends on the *USdata* size and the minimum upload (VU_O) and download (VD_D) speeds of the source and destination nodes of the reassignment, respectively.

Finally, the RA has two deployment phases: 1) the initial deployment phase, which allocates resources to ensure QoS requirements for the new US, and 2) the node reassignment phase, which involves reassigning new nodes to the US to restore missing QoS due to unavailability or failures, requiring synchronization of US data from currently available nodes. In this context, the goal is to minimize the impact of node reassignments on the US Qos by intelligently selecting resources, considering not only the node availability but also network capabilities and US-specific characteristics, to improve overall US quality. Therefore, the problem can be formulated as follows.

Determine the minor set of nodes, $N \subseteq C$, that:

$$N = \arg\min_{N \subseteq C} \sum_{i \in N} \sum_{j \in M} \left(TL_{i,j} + \frac{D_{j,i}}{\min(VU_j, VD_i)} \right) \tag{3}$$

subject to:

$$Q_{US} \geq T \tag{4}$$

$$\sum_{j \in M} D_j = USdata \tag{5}$$

Symbol	Significance
	Parameters
C	Set of candidate nodes.
M	Set of nodes currently assigned to the user service.
$USdata$	Volume of user service data.
$D_{j,i}$	Slice of data to replicate from node j to node i ($D_{j,i} \in [0, USdata]$)
$TL_{i,j}$	Network latency time between nodes i and j.
VU_j	Upload speed of node j.
VD_i	Download speed of node i.
T	Minimum required quality for the user service.
Q_{US}	Quality of user service.
	Variables
N	Selected user service nodes

Table 1: Notations used in the problem description.

Table 1 shows the notation used in the problem description. Equation (1) represents the objective function of the optimization problem that minimizes the degradation time. It considers the nodes currently assigned to the US, their capabilities, the size of the data to be replicated, the network location, and the latency between the selected and currently assigned nodes. Equation (2) is a constraint that guarantees that the US QoS equals or exceeds the minimum QoS required for the US. Equation (3) limits the parallel replication of the US data from all the nodes currently assigned to the US. The US data is divided into slices and sent from the source nodes to the destination node (reassigned node) to minimize the reassignment time.

The definition of the problem would require modelling the set of interconnected networks (i.e., Internet – that itself is a network of networks – and private networks of each donator of computational resources) involved in the communication between the nodes that form the large-scale VCS to define a better optimization mechanism and do a better validation. Unfortunately, this is currently unfeasible for many diverse reasons: each network involved in the communication between the nodes of the VCS is administrated by a different authority with different policies that do not provide detailed information about the insights of the network (such as network routing, congestion, equipment efficiency, bandwidth, and performance, among many others); traffic evolves dynamically depending on the overall traffic load (not only the traffic generated by the large-scale VCS); network evolves continuously; different technologies are used (electronic, wireless and optical); many different devices are used; etc.

In this context, we have opted for a simple yet effective model that allows us to quickly select nodes that minimize the objective function (QoS degradation time) in a sub-optimal way without requiring much information about the set of underlying networks involved in the communication between replicas. More in detail, our model uses latency – the extra time introduced by the set of networks involved in the transmission of a piece of data when this data travels from the origin node to the destination node – as the estimator that summarizes all the complexities of each network that participates in the communication between two replicas. The network latency is estimated by measuring the latency between the node and a set of predefined nodes (called landmarks). The balance (trade-off) between the update frequency of the latency metrics performed from nodes to landmarks and the overhead generated over the Centralized Control System (CCS) from the bins update to make better decisions during the MCBR process for selecting the sub-optimal candidate nodes, is a system design parameter strongly dependent on the requirements of the type of application to be deployed over the network.

5. Network Nodes Latency Estimation

The NARA mechanism uses a NCS to estimate the latency between any pair of network nodes.

5.1. Network Coordinate Systems

As stated in [37], an NCS can predict the delay between nodes in a network. The nodes can be represented in a coordinate system, and the delay can be calculated by measuring the distance between them.

NCS research started in 2001 by defining several centralized alternatives, such as Global Network Positioning (GNP), virtual landmarks, and Internet coordinate systems. In these systems, some nodes are defined as reference nodes. The delays each node produces are measured with each reference node, thereby determining their coordinates. Since 2004, various distributed NCSs have emerged, such as Vivaldi, Practical Internet Coordinates, Network Positioning System, and Weight-Based Distributed Coordinate System. These systems have a distributed implementation that does not require reference nodes. NCS systems based on matrix decomposition have also been developed since 2006. These systems predict the distance between nodes by decomposing the latency matrix, avoiding the violation of the triangle inequality problem that occurs in Euclidean distance-based coordinate systems. In summary, NCS systems can be divided into two categories: those based on Euclidean distance and those based on matrix decomposition. The first category, represented by GNP and Vivaldi, maps nodes to a geometric space where each node has its own coordinates and the distance between them is calculated. The second category, represented by the Internet distance estimation service, enables the acquisition of an input vector and an output vector for each node. This is achieved by decomposing the delay matrix and calculating the distance between two nodes based on a specific calculation rule applied to the input and output vectors.

5.2. NARA Network Latency Estimation Approach

The NARA mechanism uses a GNP solution to group nodes based on their proximity using a binning scheme. This solution is based on an estimation model was already used and validated in [38] and it is supported on the classical binning model proposed by [39]. In this scheme, nodes are divided into bins so that nodes within the same bin are relatively close to each other regarding latency. As a result, server selection and overlay construction show significant performance improvements with only approximate topological information through this scheme.

This estimation scheme is based on a set of reference nodes, known as landmarks, that are distributed throughout the network. Each node measures its RTT to each of these reference nodes and selects a container (bin) based on the results of its measurements. The scheme is simple, as it requires only minimal infrastructure support and a few servers that respond to ping requests from other nodes. They are reactive nodes that do not measure or distribute any information. Therefore, the system is scalable because each node independently discovers its bin without interacting with other nodes. However, nodes would need to periodically send this information to the CCS so that it can allocate or reallocate US among nodes. This information is small in volume and is updated periodically.

The node binning is determined on the basis of the relative distances (latencies) between each node and the set of reference nodes (landmarks). Each node measures its RTT with each landmark and sorts the sequence of landmarks in ascending order by their RTT. Specifically, if $L = \{l_0, l_1, ..., l_{m-1}\}$ is the set of "m" landmark nodes, then each node A will create an order in L_A in L such that i appears before j in L_A if $rtt(A, l_i) \leq rtt(A, l_j)$. Thus, based on the delay measures obtained with each landmark, each node has an ordered sequence of landmarks that represent the bin to which the node belongs. Neighboring nodes are topologically likely to have the same ordered sequence and therefore belong to the same bin.

The nodes latency measurements (RTT) result in two types of node classification. First, nodes are classified according to their relative distance from different landmarks, sorted by proximity, identifying the bin to which they belong. Second, nodes are grouped according to their absolute distance from each landmark. Thus, the bin sublevel is determined by translating the latency measured by the nodes to each landmark metric to a predefined set of ranges. By incorporating this second-order metric (sublevel), we can create tighter groups of nodes inside the bin. Fig.2 shows a scenario with three landmarks and how the bin and sublevel for nodes A and B are calculated.

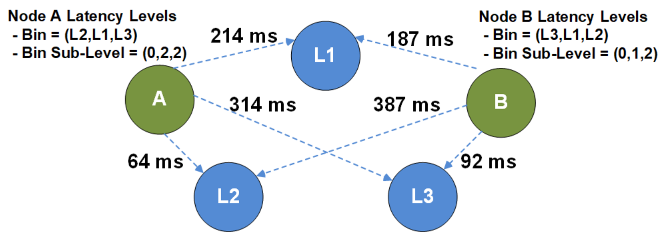


Figure 2: Latency estimation example based on three landmark nodes.

The GNP solution for the NARA mechanism reduces latency times between nodes within the same bin and bin sublevel, making them reliable indicators of node clustering. These latency times are directly proportional to the Euclidean distance between their respective bin sublevels, which is defined by the upper bounds of latency ranges. As the Euclidean distance increases, so does the latency between nodes. Therefore, nodes in bins and sublevels with smaller Euclidean distances should be given priority in node selection criteria.

Although this latency estimation model is a coarse-grained estimation model respecting other more precise or reliable models based on matrix decomposition, it is kept simple yet effective for our purpose for several reasons: 1) It requires less complexity for the CCS managing the network, delegating the measuring and binning tasks to the nodes themselves, 2) the volatility of nodes in terms of availability requires a simple and flexible model that quickly adapts to the network variability, 3) the absence of network status information enforces to infer this information from the network latency measured by each nodes respecting the landmark nodes, and 4) the application type for which the NARA mechanism is designed for does not require precise information about network latency, being a coarse-grained estimation enough for the nodes sorting and selection process.

6. Network-Aware Resource Allocation (NARA) Mechanism

This section introduces the NARA mechanism, outlining its basis, scope, and network capabilities. It addresses the phases of US lifecycle management that require RA and how it addresses them. The mechanism also discusses how network awareness, specifically minimizing aggregate latency between replicas, may reduce energy consumption.

6.1. Description of the NARA Mechanism

Fig.3 shows the components of the NARA mechanism and their relationship to each other. In the first stage, the mechanism includes a process for estimating the latency between any pair of nodes in the system. Thus, each node evaluates the RTT with respect to a set of landmark nodes and returns the metrics to the CCS. The CCS uses these metrics to group the nodes into bins. The nodes are grouped into sublevels for each bin according to the latency ranges defined for the latency metrics. A process later uses this node binning scheme to estimate network latency between any pair of nodes in the network when required. Later, in the second step, additional network-related capabilities are included in the RA process (based on the MCBR method). These capabilities can be divided into node capabilities, network characteristics, and US characteristics.

Regarding *node capabilities*, each volunteer node has its own capabilities in terms of processing, storage, and connectivity. The connectivity of the nodes depends on the access network technology and bandwidth. Thus, the NARA mechanism considers the download and upload speeds of the nodes as part of the MCBR selection criteria. Regarding *network characteristics*, nodes can be connected anywhere using different access network technologies. This means that the proximity of nodes in terms of networking can have a relevant impact on data replication and synchronization. Network latency between replicas of the same US has high relevance, so the proper selection of resources that are close in terms of connectivity is imperative. Thus, the NARA mechanism also considers network latency to be part of the MCBR selection criteria. Finally, regarding the *US characteristics*, the US typologies that can be deployed may have different requirements regarding QoS, availability, connectivity, latency sensitivity, data replication, and performance. USs require a minimum QoS, and each US must maintain updated replicas of the US data on all assigned nodes. Consequently, the NARA mechanism considers all these requirements when selecting the right resources.

In this context, the size of US data significantly impacts both QoS and data replication. The data replication time is directly related to the QoS degradation time for new nodes reassigned to the US. In contrast, the periodic data refresh to keep all nodes synced depends on the data size to replicate and the refresh frequency. In both cases, there is a direct dependency on the data transfer rates and latencies between the source and destination nodes and, thus, on the nodes network location and the node characteristics in terms of connectivity and bandwidth.

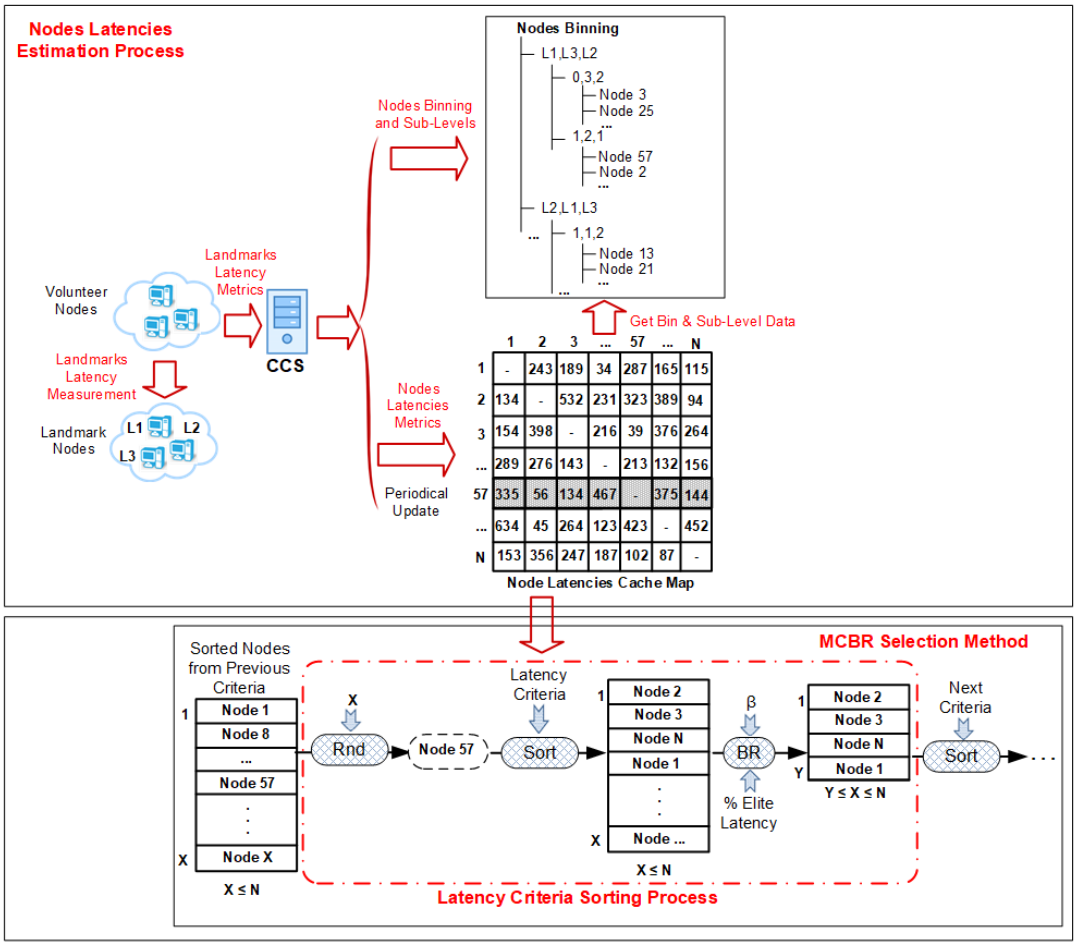
6.2. Node Latencies Cache Map

The node latencies cache map is a latencies cache that aims to reduce the computational effort in calculating latency between nodes during RA phases. It transforms the network into an L-dimensional geometric space, where L is the number of landmark nodes used as the coordinate system. Each node is identified by a set of L coordinates representing its network position. This allows for classifying nodes by location and estimating latency between any two nodes based on the Euclidean distance between their respective network coordinates.

To avoid scalability or performance impact as the network nodes increase, the node latencies cache generation and refresh uses a lazy approach, calculating latencies as long as needed for a subset of nodes. The analysis of the scalability of the network latency cache map is performed in Section 6.5.

6.3. Network Latency as MCBR Selection Criterion

The NARA mechanism uses the MCBR method to prioritize suitable nodes for US deployment. As described in Section



	1	2	3	...	57	...	N
1	-	243	189	34	287	165	115
2	134	-	532	231	323	389	94
3	154	398	-	216	39	376	264
...	289	276	143	-	213	132	156
57	335	56	134	467	-	375	144
...	634	45	264	123	423	-	452
N	153	356	247	187	102	87	-

Figure 3: NARA mechanism overview.

3, intrinsic node properties are categorized into different priority levels, and sub-optimization problems are solved according to the priority order. Finally, a reduced list of suitable nodes with the highest quality is obtained for US deployment. The NARA mechanism considers node intrinsic properties and network latency between communicating nodes. Therefore, a procedure is needed to integrate network latency into the MCBR sorting and selection method.

As shown in Fig.3, the latency criterion sorting process is a method that incorporates network latency into the MCBR method. It sorts the list "X" by latency for a randomly selected node, using a less skewed distribution to avoid overloading the best nodes. The algorithm uses the lazy latency calculation process of the node latency cache map to speed up the ordering process. A BR method [35] is applied to select the best nodes for the network latency criterion, discarding the remaining nodes. This method uses a non-uniform and biased distribution, such as the geometric distribution, which depends on a β parameter. Using a geometric distribution combined with BR has proven to be a good combination in the literature [40]. The higher the β value, the more often the first values of the sorted list are selected. This process ensures that nodes are selected at the top of the sorted list, diversifying the node selection.

6.4. *Algorithms*

This section describes the main algorithms developed as part of the NARA mechanism.

6.4.1. *Active Nodes Selection Algorithm*

The Active Nodes Selection algorithm (Algorithm 1) extends the MCBR method by incorporating new selection criteria for nodes' networking capabilities and network latency. The algorithm receives the list of active nodes (*activeNodesList*), the nodes selection criteria (*selectionCriteria*) and the number of nodes to select (*numberNodes*) as inputs, and returns the list of selected nodes (*selectedNodesList*). It selects active nodes based on specific criteria, starting with discarding non-compliant nodes. The remaining nodes are sorted and filtered, followed by a biased random selection process, resulting in a list of nodes that best fit the defined criteria. The algorithm's complexity depends on the selection criteria and the number of nodes in the network. The algorithm 1 has a linear time complexity depending on the number of candidate nodes ($|C|$) and the number of criteria ($|S|$). Concretely, $O(|C| \cdot |S|)$.

6.4.2. *Lazy Network Latency Estimation Algorithm*

As described in Section 5.2, nodes measure their latency to landmarks, sending this information to the CCS. Each node

Algorithm 1: Active Nodes Selection Algorithm

```
Input: activeNodesList, selectionCriteria, numberNodes
Output: selectedNodesList
Function activeNodesSelection():
    selectableActiveNodeList ← activeNodesList
    for criterion in selectionCriteria do
        if criterion != NetworkLatency then
            ctype, cthreshold ← criterion.type, criterion.threshold
            for activeNode in selectableActiveNodeList do
                nodecvalue ← activeNode.criterion.value
                if ((ctype == maximize AND nodecvalue < cthreshold)) OR (ctype == minimize AND nodecvalue > cthreshold)) then
                    selectableActiveNodeList.remove(activeNode)
                end
            end
        end
    end
    for criterion in selectionCriteria do
        ctype, celite ← criterion.type, criterion.elite
        if criterion == NetworkLatency then
            node ← selectableNodeList.randomNode()
            nodeLatencies ← nodeLatenciesCacheMap.getLatenciesByNode(node)
            selectableActiveNodeList.sortByLatencies(nodeLatencies, ascendent)
        else
            selectableActiveNodeList.sortByCriterion(criterion, ctype)
        end
        selectableActiveNodeList ← selectableActiveNodeList.biasedRandomSelection(celite)
    end
    selectedNodesList ← selectableActiveNodesList.biasedRandomSelection(numberNodes)
return selectedNodesList
```

is assigned a vector that combines the bin (landmarks proximity) and sub-bin (latencies to landmarks), estimating the nodes proximity degree using the Euclidean distance of their vectors.

The Lazy Network Latency Estimation algorithm (Algorithm 2) is used to sort nodes by proximity during the selection process. The algorithm receives the nodes latency cache map (*netLatencyEstimationMap*), the list of active nodes (*activeNodesList*), and the nodes for which calculate the latency (*nodeSet*) as inputs, and it returns the cache map with the updated latencies for the nodes in *nodeSet* (*netLatencyEstimationMap*). It efficiently updates network latency estimations for only the nodes required on the selection process. The algorithm categorizes nodes into bins and sub-bins based on landmark metrics, computes network latency, and creates a new tuple in the latencies cache map if a latency metric does not exist. The latency information is only updated for a limited subset of nodes following a lazy process. Thus, the algorithm's complexity depends on the list of candidate nodes ($|C|$) and the size of this subset of nodes ($|R|$). However, given that the latency sorting process in MCBR use a randomly selected node, this subset of nodes is reduced to 1, concluding a linear complexity of $O(|C|)$.

The accuracy and efficiency of the latency estimation method were assessed using cross-validation (CV) techniques, which are valid for evaluating the stability and generalization of the estimation method even when only estimated values are present. CV partitions the dataset into multiple subsets, training the estimation model on a subset of the data, getting insights as result about how well the estimation method generalizes to different subsets of the data, providing an indication of its efficiency and accuracy in absence of true values. Two types of CV strategies were evaluated: 1) k-fold CV, and 2) leave-one-out CV. Regarding the regression model applied over the latency metrics dataset, we considered a Gaussian Process Regression (GPR). The GPR model is a non-parametric, kernel-based approach that can capture complex, non-linear relationships between variables, modelling the relationship between variables as a distribution over functions, allowing for flexible and adaptive data modelling. This model is suitable for problems where the relationship between variables is non-linear or may not be well-captured by a predefined parametric model, providing uncertainty estimates for predictions and allowing for a probabilistic interpretation of the output of the model. Thus, it allows for capturing complex and non-linear patterns in the latency data and providing uncertainty estimates for the predictions.

We have created sets of one hundred randomly selected nodes. For every node in the set, we have performed a network latency estimation respecting each of the rest of the nodes of the set. Then, using the resulting latency estimation matrixes, we run both cross-validation strategies. Fig.4 exhibits the Root Mean Squared Error (RMSE) for each set, providing a measure of the error relative to the latency scale of the data set. As result, we have a latency estimation error between 15% and 20% depending on the CV strategy. Though apparently not good, this estimation requires analysis under the specific evaluation context and the nature of the data being analyzed. The inherent variability in the data (latencies), the scale and noise of the data, and the lack of actual data to be used in the predictive model lead to thinking that the performance of the predictive model should be considered as a promising model. Thus, it is expected that as the model could be fed with actual data in a real scenario, the accuracy of the estimation model could be increased.

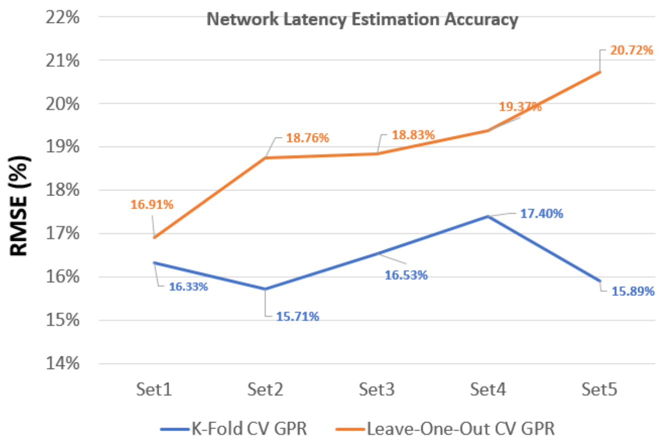


Figure 4: Network Latency Estimation Method Accuracy.

6.5. *Scalability Analysis*

This section aims to analyze the scalability of the NARA mechanism in all the critical design aspects.

Firstly, regarding the *binning scheme strategy*, the model is designed to operate with minimal support from any measurement infrastructure. This simplicity contributes to its scalability, as it does not necessitate global knowledge about the network. Each node only needs to know about a small set of well-known landmark nodes, underscoring its efficiency. The strategy is entirely distributed, as it does not require communication or cooperation between the nodes being binned, further enhancing its ease of implementation and effectiveness. The binning scheme only requires very little support from the infrastructure, i.e., a small number of relatively stable landmark machines which only need to echo "ping" messages. Thus, regarding scalability and performance, landmarks nodes do not actively initiate measurements nor gather or disseminate measurement information. These nodes independently discover their bins without communicating or coordinating with other nodes.

Secondly, regarding *the generation and refresh of the node latencies cache*, the process follows a lazy approach in which node latencies are calculated as required (i.e., only for a small subset of nodes) instead of the total number of nodes. Thus, leveraging the binning scheme, the network latency cache map is partially updated when required. Several reasons justify this decision: 1) From the service point of view, there is no need to have a filled latency cache for all the nodes in the network. Adding the network latency as part of the node selection criteria is crucial. The node selection algorithm, which selects a random node for sorting the nodes by network latency, requires only the network latency from this node to the rest of the nodes, estimated from the node partitioning in the binning scheme and stored only for this subset in the cache until their value expires, and 2) the lazy latency estimation process leverages the binning classification of nodes by network proximity, reducing the scope of nodes for sorting and selection. Even in the worst case, the node selection process will require N latency calculations, with N the total number of nodes in the network. This time is negligible compared with the required time for updating the whole latencies cache. Additionally, 3) the required memory storage using the lazy calculation approach is minimized because only the latencies required by the selection process are calculated and stored for a maximum period (validity period). Finally, 4) the lazy latency calculation prioritizes candidate nodes in the same bin as the failed node, limiting calculations to the number of nodes in the bin, only extended for the next closest bin if no nodes are available.

About the *computational overhead of the Lazy Latency Estimation Algorithm*, Fig.5 shows an analysis of the computing time of the lazy latency calculation process for different numbers of nodes. This process significantly reduces the total execution time for nodes, allowing the CCS to reduce its computing needs. The simulation shows total execution times around 400 ms for a scenario of 10,000 nodes. Given that the number of calculations performed in the lazy process is reduced to the number of nodes in the MCBR sorted list of the previous criteria or the nodes in the same bin in case of node reassignments, the total number of network nodes supported by these figures would be much higher. Additionally, as the number of criteria in the MCBR selection criteria reduces the list of candidate nodes, the latency calculations are consequently reduced. However, this may lead to additional costs from managing the binning scheme, requiring a trade-off between bin granularity and binning scheme management in the initial system design.

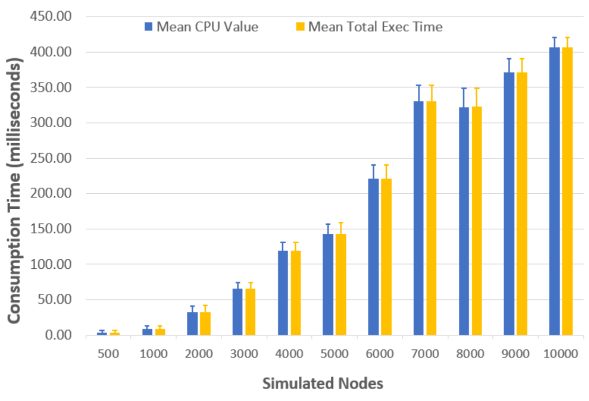


Figure 5: Cache computing time metrics.

Finally, about the *computational overhead of the Active Nodes Sorting and Selection Algorithm*, Fig.6 exhibit the computing behavior of the active nodes sorting and selection (MCBR) algorithm for different workloads and different selection criteria. The evaluation considered workloads ranging from 500 to 10,000 nodes, and the selection criteria used in all the evalu-

Algorithm 2: Lazy Network Latency Estimation Algorithm

```
Input: $netLatencyEstimationMap, activeNodesList, nodeSet$
Output: $netLatencyEstimationMap$
Function $lazyNodesNetworkLatenciesCacheMap()$:
    $binningSchema \leftarrow empty$
    for $activeNode$ in $activeNodesList$ do
        $landmarkMetrics \leftarrow activeNode.getLandmarksMetrics()$
        $binLevel, subBinLevel \leftarrow getBinLevelFromLandmarkMetrics(landmarkMetrics)$
        $bin, subBin \leftarrow binningSchema.createBinLevelIfNotExist(binLevel, subBinLevel)$
        $binningSchema.update(activeNode, bin, subBin)$
    end
    for $nodeA$ in $nodeSet$ do
        for $activeNodeB$ in $activeNodesList$ do
            $binA, subBinA \leftarrow binningSchema.getBinAndSubBin(nodeA)$
            $binB, subBinB \leftarrow binningSchema.getBinAndSubBin(activeNodeB)$
            $fromAtoB.latency \leftarrow euclideanDistance(BinA, subBinA, binB, subBinB)$
            $tupleAB \leftarrow networkLatencyEstimationMap.getTuple(nodeA, activeNodeB)$
            if $tupleAB$ not exist then
                $tupleAB \leftarrow createTuple(nodeA, activeNodeB, fromAtoB.latency)$
                $netLatencyEstimationMap.add(tuple)$
            else
                $tupleAB.addLatency(fromAtoB.latency)$
                $netLatencyEstimationMap.update(tupleAB)$
            end
        end
    end
return $netLatencyEstimationMap$
```

ated scenarios considered both node and networking capabilities. The results show how, as the number of nodes and the selection criteria increase, the average CPU consumption time also increases, with values lower than 60 ms for 10,000 network nodes. In this case, the average CPU consumption time added by the NARA mechanism respecting the CLARA mechanism is equal to 20.04 ms, which can be considered negligible and a good indicator of the scalability of the NARA mechanism.

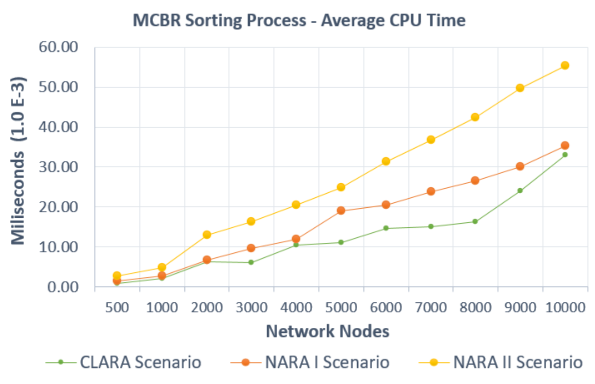


Figure 6: Analysis of CPU consumption scalability for the MCBR process.

6.6. *Resource Allocation Phases*

The NARA mechanism is crucial in two stages of the US life cycle. The setup phase ensures that deploying a new US in the network meets all US requirements. In the reallocation phase, it selects replacement resources, assigns them to the US, and replicates US data to them. This section outlines the phases, actions taken, and criteria used to select suitable resources.

6.6.1. *US Initial Deployment (Setup Phase)*

The initial deployment of a US in a network involves selecting nodes based on the MCBR node sorting process. The first node is randomly selected from the ordered list, while the remaining nodes are selected as the closest nodes in terms of network latency, distinguishing between HQ and CN nodes.

The *initial selection of HQ nodes* (characterized by their upload (VU_A) and download (VD_A) rates) is made according to the quality of the nodes and the node sorting criteria. The criteria for HQ nodes are based on node availability, upload and download speed, and network latency. The replication time to the remaining nodes depends on the data size ($US data$), the upload and download speeds, and the network latency between nodes involved in data replication. Thus, the initial deployment of a US involves an offload data process to one node, followed by replication to the rest of the nodes assigned to the US.

The *initial selection of CN nodes* is similar to the process for HQ nodes, with some differences. First, the replication process is unique from the initial random node to the CN nodes. Second, node selection is based on the same criteria as HQ nodes, except for node availability. However, CN nodes must efficiently combine candidate nodes based on their complementary availability with the minimum number of LQ nodes while minimizing latency. A two-stage selection mechanism is applied to achieve both goals. First, the complementarity degree for the remaining CN nodes is computed, and the nodes are ordered in decreasing order by their Euclidean distance. Then, the aggregated latency with respect to the rest of CN nodes is computed, selecting the node with lowest accumulated latency.

6.6.2. US Nodes Reallocation (Reassignment Phase)

The US reassignment phase involves assigning new nodes to restore the US QoS after degradation. The selection process considers network latency and upload and download speeds to minimize degradation time. Prioritizing nodes based on latency and upload and download speeds allows the selection of nodes with better performance and shorter replication times. The connection between any pair of nodes is defined by the download rate of the destination node ($V_{D_{dest}}$), the upload rate of the source node ($V_{U_{src}}$), and the network latency ($L_{src,dest}$). The link speed will be the minimum value of both rates.

On the other hand, we assume that the data are split and sent from the currently active nodes assigned to the US to the new node. The data each node sends depends on the network characteristics and the bandwidth between the source and destination nodes. Therefore, we can consider the following assumptions: 1) each active node of the US contributes at least some data to the destination node, 2) the bandwidth is not shared (multiplexed) between several source nodes and the destination node, and 3) only nodes assigned to the same US and in "active" (US-assigned and available) state are considered for user data replication. The total user data required for replication to the new destination node is D. Each currently assigned node contributes a portion of D based on the link bandwidth between the source and destination nodes. Then, we can distinguish two alternatives regarding the data slice to be replicated:

- **D1** or data slice when the link speed between the source and destination nodes is limited by the download speed of the destination node ($V_{D_{dest}}$).

$$D1 = \frac{D \cdot V_{D_{dest}}}{\sum_{i \in N} V_{i-D}} \quad (6)$$

where N is the set of nodes currently assigned to the US from which replication is performed, and V_{i-D} is the minimum value between the upload speed of the source node i and the download speed of the destination node. Then, the data slice download time (T_{D1}) will be equal to:

$$T_{D1} = L_{i-D} + \frac{D1}{V_{D_{dest}}} \quad (7)$$

where L_{i-D} represents the latency between the source node i and the destination node D.

- **D2** or data slice when the link between the source and destination nodes is limited by the upload speed of the source node ($V_{U_{src}}$).

$$D2 = \frac{D \cdot V_{U_{src}}}{\sum_{i \in N} V_{i-D}} \quad (8)$$

where N is the set of nodes currently assigned to the US from which replication is performed, and V_{i-D} is the minimum value between the upload speed of the source node i and the download speed of the destination node. Then, the data slice download time (T_{D2}) will be equal to:

$$T_{D2} = L_{i-D} + \frac{D2}{V_{U_{src}}} \quad (9)$$

where L_{i-D} represents the latency between the source node i and the destination node D.

Thus, the user data D to be replicated in the data slices from the source nodes to the destination node will be equal to:

$$D = \sum_{i \in N_1} D1_i + \sum_{j \in N_2} D2_j \quad (10)$$

Where:

- N_1 is the set of active nodes currently assigned to the US whose link speed to the destination node is limited by the download speed of the destination node ($V_{D_{dest}}$).
- N_2 is the set of active nodes currently assigned to the US whose link speed to the destination node is limited by the upload speed of the source node ($V_{U_{src}}$).
- $D1_i$ and $D2_j$ are the D1 and D2 data slices to be replicated from nodes i and j, respectively.

Therefore, the total replication time of US data from source nodes to a new destination node is calculated by adding the latency times of each pair of source and destination nodes and the download times of the data slices $D1$ (T_{D1}) and $D2$ (T_{D2}).

$$T_{replication} = \sum_{i \in N1} \left(L_{i-D} + \frac{D1_i}{V_{D_{dest}}} \right) + \sum_{j \in N2} \left(L_{j-D} + \frac{D2_j}{V_{U_{src}}} \right) \quad (11)$$

Finally, regarding the potential impact of simultaneous node failures, the node reassignment process will start sequentially replacing each failed node one by one, replicating the service data from the surviving (active) nodes. As nodes are being replaced, they will be incorporated into the scheme to restore the user service data in the following failed nodes to be replaced. In this case, the total replication time will be the sum of replication times for all the replaced nodes.

6.7. Energy Consumption Reduction by Aggregated Latency Minimization on Data Replication for Node Reassignments

To ensure availability, US data replicas must be kept on several nodes in both the setup and reallocation phases. Network latency can have a significant and non-negligible impact on energy consumption. A new node is selected when a node fails, and US data is synchronized from the current nodes. This process involves splitting US data into fragments from active nodes to the new node. The size of these fragments depends on the source and destination nodes' rates and the network latency between them. Likewise, regular updates between US data replicas are necessary to sync the data shift since the last update.

Although the size of the US data updates to periodically synchronize between replicas is much smaller than the full synchronization performed on node reassignments, the update frequency, the number of changes, and the latency can also have a relevant impact on performance and energy consumption. Therefore, we can assume the existence of a correlation between aggregated latency among replicas and energy consumption. However, the type of correlation between both concepts mostly depends on the efficiency and energy saving strategies used by the nodes and network devices involved in the data transmission. Latency can be affected by several factors, such as network bandwidth, network congestion, and network distance between devices. Energy consumption, on the other hand, depends on several factors, including the size of the data to be transmitted, network technology, network routing, and the efficiency of the network devices. Therefore, it is important to analyze the relationship between aggregate latency and energy consumption by considering different aspects of the study.

Quantitative information from network operators on energy consumption is not publicly available or only provided to estimate the entire Internet backbone. The lack of low-level information about energy consumption (e.g., network device, communication hop, network path), jointly with the lack of information about the internal routing and the network devices involved in the communication, makes it impossible to estimate the impact on the energy consumption reduction quantitatively. Thus, our research does not aim to provide a quantitative analysis of the reduction of energy consumption when the aggregated network latency is minimized. Rather, we aim to provide a comprehensive reflection on this topic to elucidate the reasons that allow us to presume that when the aggregated network latency on data replication processes is minimized, the overall energy consumption is reduced. While we cannot yet provide quantitative evidence for this reduction, our diverse analysis and compelling arguments lead us to believe that minimizing the aggregated network latency could reduce energy consumption.

In this sense, *data transmission* between geographically distributed replicas depends on network throughput, congestion, and routing factors. These factors can lead to network latency, resulting in longer transmission times and higher power consumption. Higher network throughput can reduce latency but increases power consumption for network devices involved in data transfer. Higher network latency may lower power consumption but not meet the US requirements. In our case, we cannot influence network routing selection, throughput, or congestion, so we cannot estimate total energy consumption. However, replicating data to multiple replicas over heterogeneous network links with higher latency and lower bandwidth may result in longer active connections, leading to higher power consumption as devices remain active for longer periods. Thus, selecting a node pair with lower latency could make sessions active for less time, resulting in lower power consumption.

Concerning *transmission retries*, transmission errors are more likely to occur on inter-node links where latency is higher, resulting in a higher number of transmission retries and, thus, higher energy consumption (positive correlation).

In terms of *network infrastructure*, it is necessary to maintain an active network infrastructure in all locations where there is a replica to ensure that all US replicas are up-to-date. Consequently, the higher the latency, the longer this infrastructure must remain active and consume power. Power-saving algorithms used in network devices can reduce power consumption by reducing transmission rates or putting certain network parts in standby mode. However, this can increase network latency when reactivated, impacting replication times and US QoS degradation time, especially in USs sensitive to latency.

Regarding *new replicas*, data replication to a new replica requires long-distance transfers from geographically dispersed sources to ensure availability. This data replication can result in significant aggregate latencies, increasing power consumption during the transfer phase. The energy consumption from data replication is positively correlated with the carbon footprint and energy-inefficient network infrastructure can increase energy consumption and greenhouse gas emissions (GHG). Besides, network latency can also affect energy consumption, as longer devices must be active to transmit data, resulting in higher energy consumption and carbon emissions. Therefore, given the positive correlation between the energy consumption derived from data replication and the carbon footprint, we can add some additional considerations. About the replication network distance, the distance between data replicas can increase GHG emissions due to increased time, energy, and network resources required for data transfer. Concerning hardware and software efficiency, selecting energy-efficient hardware and software can significantly reduce energy consumption during data synchronization, particularly in volunteer networks where participating nodes (most likely with low-energy efficiency) and access networks are considered. Finally, about periodic synchronization, lower latencies between US nodes and a fine-tuned synchronization strategy can minimize computational needs and network load, reducing energy consumption and carbon footprint.

In summary, higher aggregate latency in data replication or synchronization processes can lead to increased energy consumption due to longer data transmission and the need to maintain active communication infrastructure. If energy-efficient strategies are not adopted, the carbon footprint is also increased. Thus, minimizing aggregate network latency and maximizing node networking capabilities as part of the selection criteria can reduce energy consumption and carbon footprint.

6.8. Special Conditions and Considerations

The NARA mechanism is designed for specific large-scale applications requiring distributed data replicas across network nodes. The fluctuating nature of volunteer resources requires the maintenance of numerous replicas of the application data to guarantee its accessibility. The application must be able to oversee multiple data replicas across various nodes and ensure that they are synchronized. If a data replica becomes inaccessible due to node unavailability, the application data should be replicated to a newly selected node from the surviving nodes. This means the mechanism's utility is constrained to this specific domain, limiting its adaptability to other application types.

In this regard, fault tolerance, throughput, and load balancing are application characteristics and, therefore, decision parameters of the system designers during service deployment. System designers may implement different strategies for increasing the fault tolerance and redundancy of the network (e.g., use more HQ nodes, increase the used nodes for service deployment incorporating a tolerance margin for eventual failures). However, these decisions negatively impact the network due to an inefficient use of the available resources. In our context, the QoS degradation time will depend on a smart selection of the nodes that minimize this time while making fair use of the network. Thus, it is not the purpose of the model to study or discuss any of the strategies that could be implemented on top of this node selection to optimize the degradation time from the fault tolerance and load balancing perspective.

Finally, the mechanism presents challenges in the geographical distribution of landmark nodes. Ensuring an effective geographical spread of them can be a complex task, especially when considering factors such as network coverage and accessibility. Maintaining these landmark nodes adds another management layer, requiring ongoing oversight to ensure continued functionality. The mechanism depends on these landmark nodes' availability, introducing an element of uncertainty and potential disruption to the system. The distribution of landmark information as the nodes join the network requires careful orchestration to seamlessly integrate new nodes while ensuring they have access to the necessary landmark data.

7. Computational Experiments

This section details the simulation environment and experiments for evaluating the performance of the NARA mechanism.

7.1. Simulation Environment

The simulation model is based on Garlanet [41], a realistic large-scale VCS of a microblogging (Twitter-like) application that stores and replicates US data across nodes to ensure availability. This model, presented and validated in [2] and [33], follows a discrete-event model in which real-time is simulated as a series of events (e.g., nodes status changes) that occur at specific times, and how these events change the state of the whole system at that specific times. Thus, as availability events occur, nodes are connected and disconnected during the simulation. In this way, the model considers dynamic changes in the node availability and the impact of varying workloads during simulation. As result, the changes occurred in the nodes' availability and the varying service and reassignment workloads allow us to study the interactions between the system components, observing how affect the overall behavior of the mechanism. Garlanet has a CCS component that decides which US should be managed by each node. It also informs which nodes are managing each US when queried, monitors nodes status, detects available nodes and assigns the most suitable ones to clients. Additionally, it ensures all USs meet minimum replica number and QoS constraints, ensuring a smooth operation and efficient RA.

The node availability characterization is based on historical availability traces from the Failure Trace Archive (FTA) [1], a public repository of distributed system traces. This information, spanning 230,000 nodes, is used to validate fault-tolerant models and algorithms. This historical data allows simulating nodes based on real user behavior, enabling the validation and verification of the NARA mechanism and evaluating expected results in a similar real network. The simulator, developed using Java 11 Standard Edition, aims to accurately reproduce a real scenario, using an Intel quad-core 2.7 GHz processor and 16 GB RAM on a workstation running Ubuntu 22.04.

7.2. Simulation Assumptions

The simulation assumptions can be summarized as follows: 1) The CLARA mechanism simulation in [2] revealed a 50% distribution of US QoS between HQ and CN nodes, 2) The nodes upload and download rates distribution is based on monthly information published by the tool Speedtest [2]. Ookla, the company behind Speedtest, is currently the global leader in fixed broadband and mobile network testing applications and analysis. This considers symmetric (fiber) and asymmetric (adsl, cable, fiber) links with upload and download rates between 5 Mbps and 1 Gbps; 3) the distribution of the latency into bins and the number of landmarks has been selected to create a representative number of bins and nodes on each. This combination enables the evaluation of the goodness of the NARA algorithm in classifying nodes by latency, updating the cache using the lazy process for estimating the latency between nodes as required by the MCBR process; 4) The US QoS required, and the US node distribution (HQ and CN) have not been altered, respecting the analysis and simulation performed on the CLARA mechanism [2]. This enables the comparison between both mechanisms results, 5) The microblogging application uses a constant value of US data based on a number of conversations, the average number of messages per conversation, and the average size per message to ensure US QoS degradation time metrics are only dependent on network location and node capabilities, and 6) node capacities to host US range from 200 to 300.

7.3. Simulation Scenarios

The simulation aims to quantitatively compare and evaluate the NARA mechanism with the CLARA mechanism, focusing on the impact of node bandwidth parameters and internode network latency as part of the NARA mechanism's selection criteria. The goal is to minimize the time a US can operate with degraded QoS, ensuring the minimum QoS required by the US.

The selection criteria are an application-driven decision that must be aligned with the purposes and objectives of the application. As stated in Section 6.7, the NARA mechanism is designed for a type of application based on distributed storage

[1]https://fta.scem.westernsydney.edu.au/

[2]https://www.speedtest.net/global-index

among several replicas in different network nodes. It requires to prioritize nodes according to the following selection criteria and priority order: 1) By *availability (AV)* exhibited during the node AP interval, as a parameter to maximize; 2) By *node percentage of occupation (PO)* concerning the maximum number of USs it can host, as a parameter to minimize, 3) By *network latency (NLT)* of the node to a previously randomly selected node., as a parameter to minimize, and 4) By *upload speed (UPS)* and *download speed (DWS)* of the node, as parameters to maximize. The NARA mechanism is evaluated using simulation scenarios that combine different selection criteria and node selection modes. The *scenario I* (CLARA) uses node sorting criteria only, while *scenario II* and *scenario III* extend this criteria to network characteristics and latency. Table 2 summarizes the selection criteria used for each scenario. The upload and download speeds are simulated following a distribution of network access technologies randomly assigned to the nodes when created, regardless of whether they are HQ or CN nodes. In addition, the simulation considers several network workloads, ranging from 500 to 4,000, with the aim to assess its scalability and performance.

	Selection Criteria				
	AV	PO	NLT	UPS	DWS
Scenario I (CLARA)					
- HQ nodes	✓	✓			
- CN nodes		✓			
Scenario II (NARA - Network Properties)					
- HQ nodes	✓	✓		✓	✓
- CN nodes		✓		✓	✓
Scenario III (NARA - Network Properties & Latency)					
- HQ nodes	✓	✓	✓	✓	✓
- CN nodes		✓	✓	✓	✓

Table 2: Selection criteria used for simulation scenarios.

8. Analysis of the Results

This section presents the results of each simulation scenario followed by a discussion of the findings.

8.1. Experiment Results

The experimental results can be categorized into various analysis areas, including QoS, US data replication, execution and processing, and US reassignment metrics.

The *QoS metrics* results show a significant reduction in the average time for initial US replication in the assigned repositories and the average time that USs are in degraded mode. Fig.7 shows how this reduction is significant for the NARA scenarios that consider node download and upload speeds as part of the sorting criteria, compared to scenario I. The time required to complete the initial deployment of a US in all replicas shows a significant reduction with the NARA node selection mode. The setup times for both NARA scenarios remain stable below 2 seconds as the number of network nodes increases, resulting in a reduction of 70.61% (for 500 nodes) and 86.18% (for 4,000 nodes) compared to scenario I (CLARA). Besides, the time that the US is in degraded mode is remarkably reduced when network parameters are part of the sorting and selection criteria, with a reduction in times between 60.90% (for 500 nodes) and 80.97% (for 3,000 nodes) compared to scenario I. Additionally, we have evaluated the latency variability in scenario III around the estimated value, aiming to assess how sensitive the Qos degradation time is to variations in the latency estimation. [42] supports the latency variability used for the simulation, which analyzes two complementary datasets to quantify the latency variation experienced by Internet end-users. These metrics remain relevant for our purpose, as supported by the insights exhibited in [43] where authors reveal that Internet delays can be characterized as constant on timescales of 15 minutes to several hours after re-examining the constancy of end-to-end latency on Internet paths. The new simulation scenario considers the latency recalculation on every node communication as an increase over the estimated value by a random value in the variability range. Thus, we evaluated the pessimistic case where the estimated value represents the minimum latency, and the random latency value symbolizes the variability on the replication link. The results showed a slight increase respecting scenario III, where only the estimated latency derived from the binning scheme was considered. This increase ranges between 100 and 350 ms, representing a percentual increase between 5% and 23%, respecting the estimated latency. Compared to scenario II, the results are slightly worse while reducing the aggregate latency between replicas. However, the results are still outstanding compared to the CLARA scenario.

The *US data replication metrics* show a significant reduction in the average time spent replicating US data to nodes during the reassignment process. Compared to scenario I, Fig.8 shows how this reduction is remarkable in both NARA scenarios II and III. The average US data replication time for node reallocations remains stable below 42 seconds as the number of network nodes increases, with a reduction of 70.06% (for 500 nodes) and 77.68% (for 4,000 nodes) compared to scenario I.

About the *execution and processing metrics*, the study focuses on: 1) the US node reassignment and 2) the MCBR node sorting processes. Regarding the US node reassignment process, Fig.9 shows that most of the execution time is processing time. The average reassignment time remains stable below 660 ms for all evaluated network nodes, reducing execution times between 67.94% and 73.60%. Processing times decrease between 68.09% and 73.69% compared to scenario I. Regarding the MCBR node sorting process, Fig.10 shows an increase in execution and processing times for NARA scenarios compared to scenario I. The inclusion of network criteria in scenario II increases the average execution and processing times by 26.84% compared to scenario I. Adding network latency as a selection criterion increases the average by 22.38% over scenario II.

Finally, regarding *US reassignment* metrics, Fig. 11 shows how the number of USs affected by node unavailability is similar across all scenarios, with equivalent QoS recovery levels achieved in almost all cases. The US recovery efficiency of scenarios I and II is almost identical, slightly higher than scenario III. This is due to the node selection performed in scenario I respecting the NARA scenarios. In scenario III, latency addition

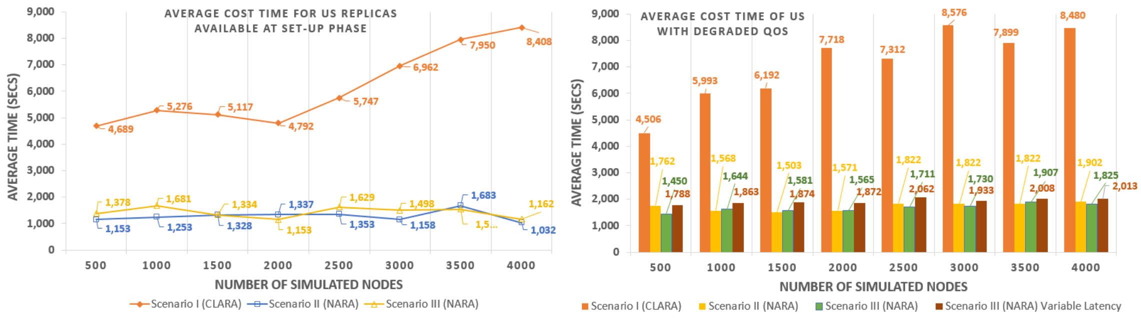


Figure 7: US QoS times.

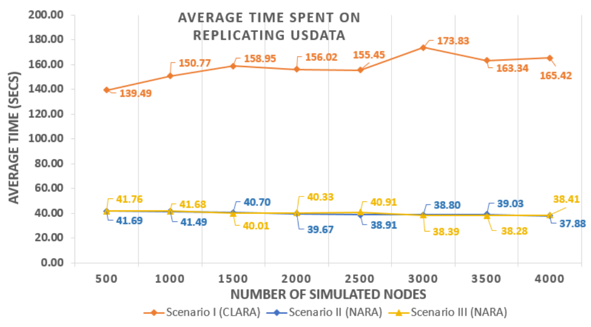


Figure 8: US data replication times.

allows for the selection of closer nodes, not prioritizing nodes that would otherwise be the best nodes. This results in a minor improvement in the efficiency of node reassignments.

8.2. *Statistical Hypothesis Testing*

Validation is based on the Wilcoxon signed rank test. This test is a non-parametric statistical hypothesis test commonly used to compare two related samples. In particular, this test aims to assess whether their population mean ranks differ. A confidence level of 95% is used. The R programming language (version 4.3.2) (R Core Team, 20234) was used to perform the hypothesis tests. The tests performed cover the main metrics used to validate the goodness of the NARA mechanism: 1) the average time for US replicas to be available at US setup, 2) the average time for US to run with degraded QoS, and 3) the average time to replicate US data from the currently assigned replicas to a new replica at reassignment.

Table 3 shows the results of the Wilcoxon signed-rank test, considering a single scenario of 1,000 nodes and all the scalability scenarios, ranging the simulated nodes from 500 to 4,000. These results exhibit that for the average time for US replicas to be available at US setup and average time to replicate US data on reassignment metrics, we cannot reject H_0, and therefore, we have no statistically significant evidence that the difference between the mean ranks is not zero. On the other hand, for the average time for the US to run with degraded QoS metric, we reject H_0. Consequently, we have statistically significant evidence that the difference between the mean ranks is not zero.

8.3. *Discussion*

This section discusses the results detailed in subsection 8.1 about considering network latency as MCBR selection criteria.

The results show that the differences between scenarios II and III are not statistically significant except for the average time a US runs with degraded QoS, with a minimal difference between both scenarios. However, scenario III improves the energy efficiency while allowing the deployment of USs sensitive to latency between replicas, reducing the time required for data replication and sync updates. This allows us to consider the choice of one scenario or the other as a trade-off between the average time that USs run with degraded QoS or an overall improvement in energy consumption while enabling the deployment of USs sensitive to latency between data replicas.

The NARA mechanism enables the use of both behaviors: 1) based on node capabilities only and 2) based on node capabilities and network latency. The CCS is responsible for determining the most appropriate mode for selecting nodes according to the US and network requirements. These requirements may include US deployment requirements, USs sensitivity to latency between replicas, power consumption constraints, replica update requirements, and the volume of US data updates.

9. Conclusions and Future Research

This paper presents the NARA mechanism as a response to two research hypotheses: 1) network awareness and node properties can be considered in RA algorithms to minimize the US QoS degradation time, and 2) the inter-node network latency can be estimated using a network coordinate system that groups nodes by latency bins. The NARA mechanism is an RA method that exploits network properties and node capabilities to minimize the QoS impact of the USs deployed in the network. As a result, the network-aware selection of nodes satisfies US QoS requirements while minimizing periodic updates and replication times of US data replicas. The NARA mechanism reduces USs QoS degradation time and energy consumption, enabling the deployment of USs sensitive to latency between replicas.

The simulation results shows how the NARA mechanism minimizes the time that USs operate with degraded QoS in the case of unavailability of the currently assigned nodes. This time is drastically reduced (between 67.82% and 81.93%) by

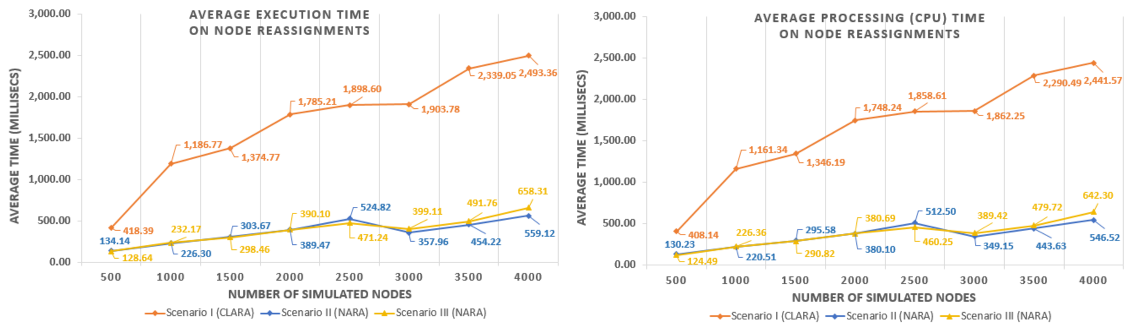


Figure 9: Execution and processing (CPU) times on US nodes reassignments.

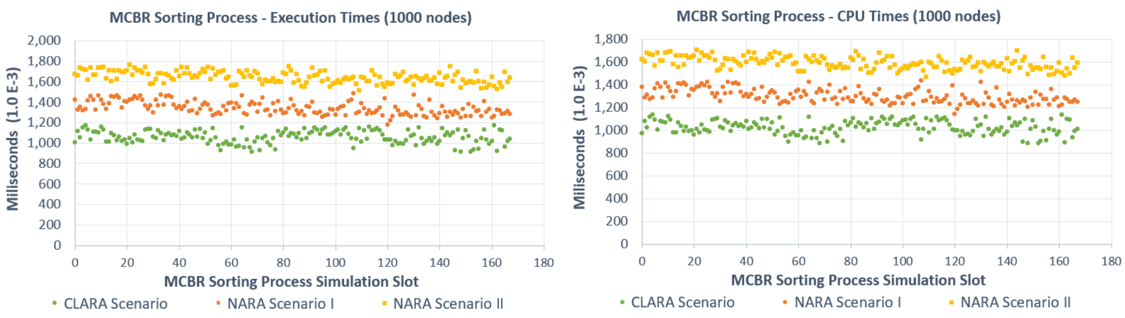


Figure 10: Comparison of CPU and execution times of MCBR sorting process on all scenarios (1,000 nodes).

	Scenario 1,000 nodes			Scalability Scenarios (500 to 4,000 nodes)		
	W	p-value	Reject H_0?	W	p-value	Reject H_0?
Average time for US replicas to be available at US setup	67	0.7197	No	7	0.1484	No
Average time for US to run with degraded QoS	22	0.03015	Yes	25	0.00828	Yes
Average time to replicate US data on reassignment	54	0.7615	No	16	0.8438	No

Table 3: Wilcoxon signed-rank analysis.

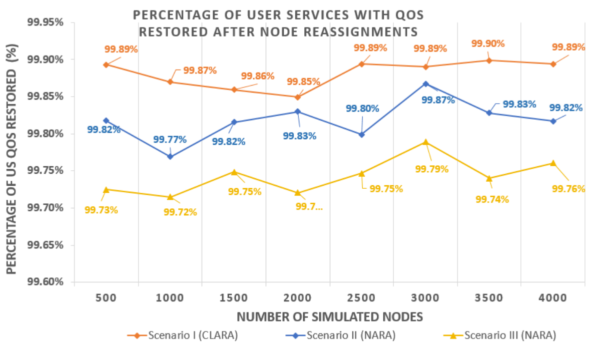


Figure 11: US QoS restoration efficiency metrics.

adding network characteristics to the sorting and selection criteria of nodes. Moreover, considering network latency as part of the selection criteria shows a remarkable improvement concerning the CLARA mechanism [2] and a negligible deviation concerning the scenario considering only the download and upload speeds of the nodes as part of the selection criteria (between 60.90% and 80.97%). This allows the selection of groups of nodes that are very close to each other in terms of connectivity, reducing the time and energy consumed by periodic replica updates and data replication in the event of node reassignment. In addition, the proximity of selected nodes allows deploying USs with high sensitivity to network latency between US replicas.

Future lines of research will focus on extending and refining the node selection criteria by incorporating customized network and node properties depending on the nature, sensitivity, and requirements of USs deployed in the network. Incorporating properties that aim to improve overall energy efficiency will be of particular importance, enabling the selection of nodes with better performance in terms of QoS, energy consumption, and environmental commitment.

Acknowledgements

This work has been partially funded by the Spanish Ministry of Science, Innovation and Universities under contract PID2023-146193OB-I00.

References

[1] M. Nouman Durrani, J. A. Shamsi, Volunteer computing: requirements, challenges, and solutions, Journal of Network and Computer Applications

39 (2014) 369–380.

[2] S. Gonzalo, J. M. Marquès, A. García-Villoria, J. Panadero, L. Calvet, CLARA: A novel clustering-based resource-allocation mechanism for exploiting low-availability complementarities of voluntarily contributed nodes, Future Generation Computer Systems 128 (2022) 248–264.

[3] Q. Wang, W. Li, A. Mohajer, Load-aware continuous-time optimization for multi-agent systems: Toward dynamic resource allocation and real-time adaptability, Computer Networks 250 (2024) 110526.

[4] Y. Wu, J. Zhu, X. Chen, Y. Zhang, Y. Shi, Y. Xie, QoS-based resource allocation for uplink NOMA networks, Computer Networks 238 (2024) 110084.

[5] A. Shabbir, S. Rizvi, M. M. Alam, F. Shirazi, M. M. Su'ud, Optimizing energy efficiency in heterogeneous networks: An integrated stochastic geometry approach with novel sleep mode strategies and QoS framework, PloS one 19 (2) (2024) e0296392.

[6] C. Jong, Y. C. Kim, J. H. So, K. C. Ri, QoS and energy-efficiency aware scheduling and resource allocation scheme in LTE-A uplink systems, Telecommunication Systems 82 (2) (2023) 175–191.

[7] Y. Yao, Y. Chen, H. Yao, Z. Ni, M. Motani, Multiple task resource allocation considering QoS in energy harvesting systems, IEEE Internet of Things Journal 10 (9) (2023) 7893–7908.

[8] A. N. Aliyu, K. Musa, A. Dutse, QoS-based resource allocation using ant colony optimizationin cloud computing, International Journal of Science Research and Technology (2024).

[9] J. Liu, Y. Wang, D. Pan, D. Yuan, QoS-aware task offloading and resource allocation optimization in vehicular edge computing networks via MADDPG, Computer Networks 242 (2024) 110282.

[10] T. Yang, J. Sun, A. Mohajer, Queue stability and dynamic throughput maximization in multi-agent heterogeneous wireless networks, Wireless Networks (2024) 1–27.

[11] S. Saibharath, S. Mishra, C. Hota, Joint QoS and energy-efficient resource allocation and scheduling in 5G network slicing, Computer Communications 202 (2023) 110–123.

[12] R. Jayaraman, B. Manickam, S. Annamalai, M. Kumar, A. Mishra, R. Shrestha, Effective resource allocation technique to improve QoS in 5G wireless network, Electronics 12 (2) (2023) 451.

[13] L. Gu, A. Mohajer, Joint throughput maximization, interference cancellation, and power efficiency for multi-IRS-empowered UAV communications, Signal, Image and Video Processing 18 (5) (2024) 4029–4043.

[14] M. A. Hossain, N. Ansari, Energy aware latency minimization for network slicing enabled edge computing, IEEE Transactions on Green Communications and Networking 5 (4) (2021) 2150–2159.

[15] J. Luo, Q. Wang, F.-C. Zheng, L. Gao, S. Gu, Cooperative activation and caching strategy for low-latency and energy-efficient small-cell networks, IEEE Wireless Communications Letters 11 (4) (2022) 756–760.

[16] K. Fujimoto, K. Natori, M. Kaneko, A. Shiraga, Energy-efficient KBP: Kernel enhancements for low-latency and energy-efficient networking, IEICE Transactions on Communications 105 (9) (2022) 1039–1052.

[17] M. H. Maturi, Optimizing energy efficiency in edge-computing environments with dynamic resource allocation, environments 13 (07) (2024) 01–08.

[18] S. Khalid, A. Alam, M. Fayaz, F. Din, S. Ullah, S. Ahmad, Investigating the effect of network latency on users' performance in collaborative virtual environments using navigation aids, Future Generation Computer Systems 145 (2023) 68–76.

[19] S. Rawas, A. Zekri, A. El-Zaart, LECC: Location, energy, carbon and cost-aware VM placement model in geo-distributed DCs, Sustainable Computing: Informatics and Systems 33 (2022) 100649.

[20] W. E. Gnibga, A. Blavette, A.-C. Orgerie, Latency, energy and carbon aware collaborative resource allocation with consolidation and QoS degradation strategies in edge computing, in: 2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS), IEEE, 2023, pp. 2630–2639.

[21] L. B. Mohammed, A. Anpalagan, M. Jaseemuddin, Energy and latency efficient caching in mobile edge networks: Survey, solutions, and challenges, Wireless Personal Communications 129 (2) (2023) 1249–1283.

[22] L.-T. Tu, A. Bradai, O. B. Ahmed, S. Garg, Y. Pousset, G. Kaddoum, Energy efficiency optimization in LoRa networks—a deep learning approach, IEEE Transactions on Intelligent Transportation Systems 24 (12) (2023) 15435–15447.

[23] M. Jalilvand Aghdam Bonab, R. Shaghaghi Kandovan, QoS-aware resource allocation in mobile edge computing networks: Using intelligent offloading and caching strategy, Peer-to-Peer Networking and Applications 15 (3) (2022) 1328–1344.

[24] F. Qasmi, M. Shehab, H. Alves, M. Latva-Aho, Effective energy efficiency and statistical QoS provisioning under markovian arrivals and finite blocklength regime, IEEE Internet of Things Journal 9 (18) (2022) 17741–17755.

[25] B. Premalatha, P. Prakasam, Optimal energy-efficient resource allocation and fault tolerance scheme for task offloading in IoT-FoG computing networks, Computer Networks 238 (2024) 110080.

[26] N. An, F. Yang, L. Cheng, J. Song, Z. Han, IRS-assisted aggregated VLC-RF system: Resource allocation for energy efficiency maximization, IEEE Transactions on Wireless Communications (2024).

[27] M. Abdullah, H. Z. Khan, U. Fakhar, A. N. Akhtar, S. Ansari, Satellite synergy: Navigating resource allocation and energy efficiency in IoT networks, Journal of Network and Computer Applications (2024) 103966.

[28] B. Kopras, B. Bossy, F. Idzikowski, P. Kryszkiewicz, H. Bogucka, Task allocation for energy optimization in fog computing networks with latency constraints, IEEE Transactions on Communications 70 (12) (2022) 8229–8243.

[29] F. Han, M. Wang, Y. Cui, Q. Li, R. Liang, Y. Liu, Y. Jiang, Future data center networking: From low latency to deterministic latency, IEEE Network 36 (1) (2022) 52–58.

[30] S. Liu, X. Xu, M. Claypool, A survey and taxonomy of latency compensation techniques for network computer games, ACM Computing Surveys (CSUR) 54 (11s) (2022) 1–34.

[31] J. Ma, H. Gao, L. Guo, H. Li, Energy-efficient joint resource allocation for heterogeneous cellular networks with wireless backhauls, AEU-International Journal of Electronics and Communications 176 (2024) 155170.

[32] C. Li, Z. Ke, Q. Liu, C. Hu, C. Lu, Y. Luo, Energy–latency tradeoffs edge server selection and DQN-based resource allocation schemes in MEC, Wireless Networks (2023) 1–27.

[33] J. Panadero, J. de Armas, J. M. Serra, Xavierand Marquès, Multi criteria biased randomized method for resource allocation in distributed systems: Application in a volunteer computing system, Future Generation Computer Systems 82 (2018) 29–40.

[34] R. T. Marler, J. S. Arora, Survey of multi-objective optimization methods for engineering, Structural and multidisciplinary optimization 26 (2004) 369–395.

[35] A. A. Juan, J. Faulín, J. Jorba, D. Riera, D. Masip, B. Barrios, On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics, Journal of the Operational Research Society 62 (6) (2011) 1085–1097.

[36] D. Lázaro, D. Kondo, J. M. Marquès, Long-term availability prediction for groups of volunteer resources, Journal of Parallel and Distributed Computing 72 (2) (2012) 281–296.

[37] R. Cheng, Y. Wang, A survey on network coordinate systems, in: MATEC Web of Conferences, Vol. 232, EDP Sciences, 2018, p. 01037.

[38] B. Tang, M. Tang, G. Fedak, H. He, Availability/network-aware mapreduce over the internet, Information Sciences 379 (2017) 94–111.

[39] S. Ratnasamy, M. Handley, R. Karp, S. Shenker, Topologically-aware overlay construction and server selection, in: Proceedings. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, Vol. 3, IEEE, 2002, pp. 1190–1199.

[40] A. Grasas, A. A. Juan, J. Faulin, J. De Armas, H. Ramalhinho, Biased randomization of heuristics using skewed probability distributions: A survey and some applications, Computers & Industrial Engineering 110 (2017) 216–228.

[41] J. M. Marquès Puig, H. Rifà-Pous, S. Oukemeni, From false-free to privacy-oriented communitarian microblogging social networks, ACM Transactions on Multimedia Computing, Communications and Applications 19 (2s) (2023) 1–23.

[42] T. Høiland-Jørgensen, B. Ahlgren, P. Hurtig, A. Brunstrom, Measuring latency variation in the Internet, in: Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies, 2016, pp. 473–480.

[43] L. Davisson, J. Jakovleski, N. Ngo, C. Pham, J. Sommers, Reassessing the constancy of end-to-end Internet latency, in: IFIP Network Traffic Measurement and Analysis Conference, 2021.