Received July 12, 2021, accepted August 22, 2021, date of publication September 10, 2021, date of current version October 11, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3111699

Information Resilience in a Network of Caches With Perturbations

DEHAO WU AND WEI KOONG CHAI, (Senior Member, IEEE)
Department of Computing and Informatics, Bournemouth University, Poole BH12 5BB, U.K.

Corresponding author: Dehao Wu (dwu@bournemouth.ac.uk)

ABSTRACT Caching in a network of caches has been widely investigated for improving information/content delivery efficiency (e.g., for reducing content delivery latency, server load and bandwidth utilization). In this work, we look into another dimension of network of caches – enhancing resilience in information dissemination rather than improving delivery efficiency. The underlying premise is that when information is cached at more locations, its availability is increased and thus, in turn, improve information delivery resiliency. This is especially important for networks with perturbations (e.g., node failures). Considering a general network of caches, we present a collaborative caching framework for maximizing the availability of the information. Specifically, we formulate an optimization problem for maximizing the joint utility of caching nodes in serving content requests in perturbed networks. We first solve the centralized version of the problem and then propose a distributed caching algorithm that approximates the centralized solution. We compare our proposal against different caching schemes under a range of parameters, using both real-world and synthetic network topologies. The results show that our algorithm can significantly improve the joint utility of caching nodes. With our distributed caching algorithm, the achieved caching utility is up to five times higher than greedy caching scheme. Furthermore, our scheme is found to be robust against increasing node failure rate, even for networks with a high number of vulnerable nodes.

INDEX TERMS Information resilience, caching, network of caches, network with perturbations.

I. INTRODUCTION

With the dramatic increase in information/content access demand, content caching has now been an integral part of modern communication networks. Content delivery networks (CDNs) (e.g., Akamai) deploy high-capacity proxy servers for accelerated content distribution [1] while the impending 5G network exploits caching for enabling new emerging applications [2]. Cloud and, the more recent, mobile edge cloud (MEC) [3] exploit network edge caching capability for improving user experience. With these developments, cache management has received renewed attention especially due to the advent of the Information-centric Networking (ICN) paradigm [4], [5] which advocates the use of in-network caching where network elements have content caching capability [6]. This recent exploding body of work (e.g., [6], [7]) has shown the benefits of distributed content caching. However, most of these work mainly focus on improving content delivery performance in terms of latency, server load and bandwidth utilization.

The associate editor coordinating the review of this manuscript and approving it for publication was Mingjun Dai.

Since the cache locations are now geographically distributed, the availability of content and dependability of the network of caches as a whole system are also of interest. In view of this, in-network caching has been advocated for a different purpose, namely for *information resilience* [8]–[10] whereby the aim is to improve content availability and reachability in disruptive network environment rather than improving content delivery efficiency. The rationale is that the network management and control under such volatile operating conditions could benefit from a caching scheme that considers possible network perturbations which results in topology changes and even network fragmentation. It has been highlighted in [11] that there is a clear lack of understanding of how much in-network caching improves information availability under such a scenario. Network perturbations could be due to human errors and random equipment failures. They could also simply be a natural part of the network (e.g., topology changes due to node movements in mobile networks). Severe scenarios involve perturbations resulted from natural disasters or malicious attacks intending to cause maximum damage.

Information resilience focuses on protecting information/content directly. This is in contrast with traditional resilience concept in telecommunication networks which seeks to ensure continuity of service via protection and recovery of the physical infrastructure (e.g., maintenance of connectivity, fault detection, resource redundancy) [12], [13]. Traditionally, network resilience usually involves physical/link layer protection mechanisms and redundancy provisioning [12]. When a path is disrupted, resilience is conventionally achieved via traffic re-routing [13]. The idea of information resilience in our context is derived from the ICN paradigm where content is named and can be explicitly identified independent of network locations. ICN has already been identified as particularly suitable in emergency/disruptive scenarios [14], [15]. The inherent anycast capability of ICN, whereby an information object can be delivered by multiple sources or caches, allows continuous reachability of content even when the source is unavailable as the content is not strictly bounded to one host.

Here, we consider the problem of providing information resilience in the general case of a *network of caches with perturbations*. We abstract the problem focusing on the network of caches without considering the technical realization of the networking paradigm/architecture. Specifically, we consider a network consists of cache-enabled nodes (i.e., nodes are equipped with cache capacity, able to cache content and satisfy content requests). We investigate a collaborative caching framework for maximizing the joint utility of the caching nodes in a perturbed network in which the network topology changes over time. We consider a wide range of failure scenario (cf. Section III-D). We formulate an integer programming optimization problem and solve the centralized version of the problem to obtain the optimal caching solution. Since the problem is NP-hard, we then proceed to design and evaluate a distributed collaborative caching algorithm that closely approximates the performance of the optimal centralized solution.

The main contributions of this paper are summarized below:

1) We study the problem of collaborative caching in a network of caches considering possible network perturbations. Specifically, we seek to maximize the caching utility and considers the effect of network perturbations (e.g., node failures). We show that the optimization problem is convex and present a centralized solution.
2) We further study a distributed caching algorithm that approximates the centralized solution by adopting a sub-gradient Lagrangian relaxation approximation. In our distributed solution, each network node makes its own caching decision in collaboration with its neighbors. We show the utilities given by both the centralized and distributed solutions are sufficiently close (i.e., within 2%).
3) We comprehensively evaluated our proposal across different scenarios. We compare our solution against both random and greedy caching algorithms across both real and synthetic networks and across a wide range of failure probability distributions. Our results show significant improvements even under severe network perturbations.

The rest of the paper is organized as follows. In Section II, we review the background and related literature. System overview and problem formulation are given in Section III. Section IV is devoted to solving the centralized problem. In Section V, we derive a distributed solution by adopting a sub-gradient Lagrange relaxation approximation. For the distributed solution, we present a distributed caching algorithm to approximate the optimal centralized solution. In Section VI, we evaluate the performance of the proposed collaborative caching scheme. We conclude our paper in Section VII.

II. BACKGROUND AND LITERATURE REVIEW

Most work on content caching and management aim to improve content delivery efficiency (e.g., reducing latency) [7], [16], [17]. Here, we focus on approaches that can be leveraged for providing information resilience and latest related works that directly addressed information resilience.

A. ONLINE REACTIONARY APPROACH

A caching scheme that responds to network dynamics could be employed for providing information resilience. The default ICN caching scheme uses an indiscriminate caching approach [18]. Caching gain[1] is obtained opportunistically along the shortest path. In [11], it is found that when the fraction of broken links grows beyond 50%, such indiscriminate caching has diminishing returns. Arguing that such approach unnecessarily limits the potential gain, [19] investigated a joint forwarding and caching approach that allows content requests to be forwarded off the default shortest path to find the requested content. Along the same line, [20] proposed to improve caching gain by flooding content requests to discover content cached nearby the shortest path in tandem with leave copy down (LCD) caching policy [21]. However, flooding incurs high overhead. To improve scalability, [22] proposed to limit the flooding scope to three hops and argued that there is limited improvement when compared against unconstrained flooding. Also to improve scalability, [23] exploited hash-routing to route content requests, thus avoiding the complex problem on efficiently routing requests to cache nodes.

For information resilience, [24] defined a recovery process via alternative content sources when the original one fails by exploiting ICN feature whereby information is individually identifiable. In [9], an information resilience scheme for the NDN architecture [25] is studied in which the scheme proposed to keep a record of satisfied interest so that a failed request can be forwarded on an alternative path that has been successful in the past. Such reactionary event-triggered approaches do not "prepare" the network in advance to maximize content availability but rather attempt to find the requested content replica *after* perturbations.

[1]We use the generic term "caching gain" to describe the benefit obtained via caching (e.g., reduction of server load, content delivery latency).

B. COOPERATIVE CACHING

An alternative approach to improve information resilience is via node cooperation. Cooperative caching has been adopted in several past works. For instance, [26] proposed an online distributed cache management solution for information-centric networks in which cache "managers" periodically exchange messages to identify changes needed in their cache store to improve caching gain. Citing prohibitive signaling overhead in cooperative caching, [27] attempted to strike a balance between performance and scalability via a lightweight local cooperation caching design. In [28], an agent reinforcement learning-based cooperative content caching policy is proposed for the mobile edge networks (MENs) when the users' preference is unknown. Some other caching policies and algorithms are studied and compared in MENs in [29]. Delay due to the backhaul and mobility effect are been considered. In [30], network bottleneck and interruption problems are raised for the MENs. However, the issue has not been addressed by any available solution. These work, however, neglected possible network failures and as such, could be complementary to our proposal here when operating under normal network conditions (i.e., no failures). Since various works pointed to the benefits of having nodes to cooperate [31]–[34] attempted to answer the question on how much coordination is required to optimize in-network caching in ICN. The authors concluded that the level of coordination between nodes is highly dependent on the content request distribution.

For information resilience, [10] considered implicit cooperation amongst caching nodes by exploiting features of network topology and proposed a modularity-based caching scheme that allows nodes within a community structure to cache content with origins outside of that community. This solution does not require explicit communication between caching nodes but nonetheless, still increases the diversity of content cached within the community. However, such a scheme may not offer significant gain when the network topology does not exhibit strong community structure.

C. CONTENT PLACEMENT

Finally, another approach that can be potentially leveraged is through adding redundant content replicas through strategic content placement. Such approach has been adopted in the past such as in CDNs [1], telco-CDNs [35] and web caching [36]. Server duplication mechanism is often used for load balancing when demand is high. Such mechanism involves the full replication of all content in the server which often requires high-capacity servers. With ICN, new solutions based directly on information are developed. For instance, [37] considered how popular content could be distributed across information-centric networks comprises of access and transit domains considering the cost and utility of caching content that is deemed popular. The authors in [38] investigated content placement for achieving *fairness* amongst the caching nodes. It considered each node is owned by a different stakeholder and are selfish and would individually maximize its own utility. The authors employed game theory to find stable caching solutions such that all nodes converge to an optimal caching solution. Our problem formulation takes this work as the starting point but we focus on information resilience instead. Content placement problem over a network of caches is also studied in [39] where a probabilistic content placement algorithm is proposed offering a bound of $1 - 1/e$ factor from the optimal solution.

For information resilience, [8] extended [39] and proposed an optimal distributed content placement algorithm that maximizes the caching gain in the presence of failures. In this work, the network failure probability is explicitly considered when making a caching decision. However, *a priori* knowledge of the failure probability of each node is required to implement the caching decisions. This presents a challenge as it is difficult to estimate and quantify failure probability in reality. As shown in [40], [41], node failure probability is affected by a combination of factors (e.g., variety and varying operating environment). As such, deriving an exact model to predict the failure probability is a challenging research topic on its own. In this work, we instead exploit metrics from network science [42] as an indicator of failure probability. Furthermore, our framework also employs cooperative caching for maximizing the joint utility of caching nodes in network with perturbations.

D. SUMMARY

We take elements from approaches discussed in Section II-B and II-C. In our approach, nodes collaborate with their neighbors within a limited radius to maximize cache hit considering different possible permutations of network perturbations. We do not "blindly" reduce cache redundancy by simply caching different content but instead seek to place content optimally within limited distance away from requesting nodes.

III. SYSTEM OVERVIEW AND PROBLEM FORMULATION

A. SYSTEM MODEL

We consider an arbitrary undirected network, $G(V, E)$ with $V = \{v_1, v_2, \ldots, v_N\}$ cache-enabled nodes and $E = \{e_1, e_2, \ldots, e_M\}$ links where their cardinalities are $N = |V|$ and $M = |E|$ respectively. Each v_i is equipped with a cache store with capacity C_i. Further, we define $l^*_{i,j}$ as the number of nodes involved in a content delivery path between v_i and v_j inclusive of both these nodes and the set of nodes involved in this path as $\zeta_{i,j}$. Thus, $l^*_{i,j} = |\zeta_{i,j}|$. By this definition, the path length, $l_{i,j} = l^*_{i,j} - 1$. Considering network perturbations, we also define p_i as the likelihood of failure for node v_i. We consider p_i as a function of different factors and evaluate their implications to information resilience (cf. Section III-D).

Let O be the content population in the system where o_k is the k^{th} item in the content set. s_k denotes the size of content o_k. We assume that there exists an origin server for each content object and each content request is routed towards this origin server via a pre-determined shortest path. The content request can be satisfied by any nodes along the path having cached the requested content (i.e., a cache hit). In this case, the requested content is returned without the involvement of the origin server. Furthermore, we define an $N \times |O|$ demand matrix, W with its element $w_{i,k}$ denoting the expected or estimated content request arrival rate for content o_k at v_i.

B. COLLABORATIVE CACHING FRAMEWORK

We consider a collaborative caching framework in which nodes collaborate with nearby nodes to satisfy content requests. In our evaluation, the request is generated one at a time and originated from a randomly chosen node that is not the node hosting the content itself. Once a request is issued, the request is forwarded over a fixed path to the origin server. At each hop, the node receiving the request searches its neighborhood for the requested content if that content is not cached in its own cache store. We define r_i as the search radius (in hop count) of v_i. It indicates that v_i will search all nodes within r_i hops away from itself for the requested content. Furthermore, let η_i be the set of nodes within r_i hops away from v_i. As r_i increases, the cost of collaboration grows. Table 1 summarizes the notations.

TABLE 1. Summary of notations.

Symbol	Description
N	The number of nodes in the network, $\|V\|$
M	The number of links in the network, $\|E\|$
C_i	The cache capacity of v_i
$l^*_{i,j}$	The number of nodes involved in a content delivery path between v_i and v_j inclusive of both nodes
$l_{i,j}$	The path length between v_i and v_j
$\zeta_{i,j}$	The set of nodes involved in the path between v_i and v_j
p_i	The likelihood of failure for node v_i
O	The content set, o_k represents the k^{th} item
s_k	The size of the k^{th} content item o_k
$w_{i,k}$	The demand of content o_k at v_i, requests per second
X	$x_{v_{i,k}}$ represents whether v_i caches o_k locally
Y	$y_{v_{i,j,k}}$ represents whether v_i retrieves o_k from v_j where the cache hit occurred
r_i	The search radius (in hop count) of node v_i
η_i	The set of nodes in v_i's neighborhood (within r_i hops)
U_i	The utility function of node v_i

In a collaborative caching framework, we would like to fetch the requested content from a node as near to the requester as possible to reduce content delivery latency and as this involves fewer nodes, it is also less likely to encounter a failed node. This then involves two decisions. The first decision relates to whether a node decides to cache a specific content in its cache store and second, whether a node retrieves a content from another neighboring node instead of following the pre-computed path to the origin server. We represent the above with two decision variables, X and Y. Specifically, $x_{(i,k)} \in \{0, 1\}$ denotes whether v_i caches content o_k while $y_{(i,j,k)} \in \{0, 1\}$ denotes whether v_i retrieves the content o_k from v_j. For both variables, we follow the convention of denoting a negative decision with "0" and the alternative with "1". For instance, $x_{i,k} = 1$ if node v_i decides to cache content o_k.

Considering the total number of network nodes and the cached objects is large, to reduce the computation complexity for the caching decision, the integer constraints on X and Y are relaxed as [0,1]. The details on this relaxation is given in Section IV.

C. PROBLEM FORMULATION

Given the above system model, we formulate a joint utility maximization problem to find the optimal caching strategy.

1) UTILITY

A node derives utility from successfully satisfying a content request it receives. The utility depends on whether the node can satisfy the request (e.g., a node may not be able to reach the origin server in some fragmented perturbed networks) and how fast it can serve this request (e.g., edge nodes can directly serve content requesters).

The utility of node v_i, U_i, is then given by:

$$U_i = \sum_{o_k \in O} s_k w_{i,k}(1-p_i)x_{i,k} + \sum_{o_k \in O} \sum_{v_j \in \eta_i} \left[\frac{s_k w_{i,k}}{l^*_{i,j}+1} y_{i,j,k} \prod_{n \in \zeta_{i,j}} (1-p_n) \right] \quad (1)$$

where $X = [x_{i,k}]_{v_i \in V, o_k \in O} \in \{0,1\}^{|V| \times |O|}$, and $Y = [y_{i,j,k}]_{v_j \in_i, o_k \in O} \in \{0,1\}^{|V| \times |V-1| \times |O|}$.

The first term on the right hand side of Eq. 1 computes the utility derived from satisfying a content request locally by v_i while the second term sums the utility gained from collaboration with nodes in its neighborhood, η_i. It reflects that the utility decreases as the distance to retrieve a content from a remote node increases. The content request is routed along a computed path towards the origin server until a cache hit occurs or the request reaches the server. At this point, the requested content is sent back to the requester via the reverse of the path taken by the content request. The product expression $\prod_{n \in \zeta_{i,j}}(1-p_n)$ in Eq. 1 reflects that the utility gained reduces when the content delivery path length increases. It becomes critical especially for the scenarios with nodes having high failure probability (e.g., malicious attacks).

To obtain high utility, each node aims to serve its clients with both the lowest possible delay and the highest content request satisfaction rate. However, considering the aggregated demands of the entire network, we are seeking the optimum among all nodes, which points to the need of a collaborative caching scheme whereby utility is maximized through satisfaction of content requests via combination of local caching as well as redirections to nearby neighbors.

2) OPTIMIZATION OBJECTIVE

Consequently, we define our utility maximization problem as follows:

$$\max \prod_{v_i \in V} U_i \tag{2}$$

Taking the logarithm of Eq. 2, we obtain the following:

$$\ln\,(\max \prod_{v_i \in V} U_i) = \max\ \ln(\prod_{v_i \in V} U_i). \tag{3}$$

By taking the negation, Eq. 3 is equivalent to the following:

$$\max \sum_{v_i \in V} \ln(U_i) \Rightarrow \min \sum_{v_i \in V} -\ln(U_i) \tag{4}$$

Substituting Eq. 1 to Eq. 4, we have:

$$\min \sum_{v_i \in V} -\ln\Big\{ \sum_{o_k \in O} w_{i,k}(1-p_i)x_{i,k} + \sum_{o_k \in O} \sum_{v_j \in \eta_i} [\frac{w_{i,k}}{l_{i,j}} y_{i,j,k} \prod_{n \in \zeta_{i,j}} (1-p_i)] \Big\} \tag{5}$$

3) CONSTRAINTS

Our problem is subjected to the following constraints:

$$\sum_{o_k \in O} x_{i,k} \leq C_i, \quad \forall v_i \in V,\ o_k \in O \tag{6}$$

$$\sum_{v_j \in \eta_i} y_{i,j,k} \leq 1, \quad \forall v_i \in V,\ o_k \in O \tag{7}$$

$$x_{i,k} \in \{0, 1\}, \quad \forall v_i \in V,\ o_k \in O \tag{8}$$

$$y_{i,j,k} \in \{0, 1\}, \quad \forall v_i\ v_j \in V,\ o_k \in O \tag{9}$$

$$0 \leq p_i \leq 1 \tag{10}$$

$$y_{i,j,k} \leq x_{i,k}, \quad \forall v_i\ v_j \in V,\ o_k \in O \tag{11}$$

Constraint (6) ensures that content stored at node v_i does not exceed its cache capacity, C_i. Constraint (7) ensures that a node can only retrieve a maximum of one complete object per request. Constraints (8) and (9) are the domains of the decision variables. Constraint (10) defines the probability of failure for node v_i. Constraint (11) means that v_i can retrieve o_k from v_j if the cache hit occurred at v_j and v_i can retrieve the content from v_j only if v_j cached it.

D. FAILURE PROBABILITY DISTRIBUTIONS

In this paper, we study the impact of various node failure patterns. We define different node failure probability distributions to represent different network perturbation scenarios. We differentiate random failure patterns which are usually modeled with random events from malicious attacks that are intended to incur maximum disruption [43]. In the following, we detail the three different failure models considered in this paper.

1) UNIFORM MODEL

This node failure probability distribution follows uniform probability. All nodes have fixed equal failure probability, p. Each node fails independently. We use this uniform distribution to represent failure scenarios where perturbations are caused by random events such as equipment failures, human error or unpredictable natural failure events.

2) LINEAR MODEL

We further define two failure probability distributions in which node failure probability is linearly proportional to some centrality measures reflecting the node's relative importance in the network. This mimics malicious attacks on the network where the perpetrator, with the intention to cause maximum damage, logically targets nodes that are deemed to be important. Such attacks may come in physical form, via equipment tampering or electronically via computer viruses. We assume a node with higher centrality will have higher chance of being targeted. We choose two widely used centrality measures, namely degree and betweenness centrality [44]. For the case where we use degree centrality, we first limit the maximum failure probability of the node with the highest degree to $p^{\max}$. Then, we compute a base failure probability with $p_D^{base} = \frac{p^{\max}}{\max(D(v_i);\forall i)}$ where $D(v_i)$ is the degree of node, v_i. Finally, we compute v_i's failure probability by $p_i = p_D^{base} \times D(v_i)$. We use the same methodology for betweenness centrality case by replacing $D(v_i)$ with the betweenness of each node, $B(v_i)$. Figure 1 shows the cumulative distribution functions (CDF) of failure probability based on degree and betweenness of nodes in a sample ER and SF network each and three real-world networks, namely L3, Sprint and AT&T networks (cf. Section VI). From Figure 1(a), we note that the ER network has comparatively higher node failure probability with almost all nodes having failure probability higher than 0.25. In contrast, AT&T has approximately 80% of the nodes with 0.04 failure probability or lower. Meanwhile, the failure probability distribution based on betweenness showed two groups of networks. The ER, SF and AT&T networks have high number of nodes with similar betweenness (i.e., many nodes have similar failure probability). L3 and Sprint networks have "wider" betweenness spread.

3) NON-LINEAR MODEL

In real networks, failure probabilities often do not follow linear models. Correspondingly, we also define node failure probability to be proportional to degree or betweenness of node but in a non-linear fashion. In this scenario, the failure probability increases non-linearly with the degree or betweenness. For a given node v_i, its failure probability is then defined using a *Sigmoid* function as follow:

$$p_i(x) = \frac{\sigma_0}{1 + e^{-(\mu_0 + \mu_1 x)}}. \tag{12}$$

where x is either $D(v_i)$ or $B(v_i)$ of the node. We use the first half of the *Sigmoid* function. When $\sigma_0 = 1$, then the midpoint of the function equals 0.5 which, in our case, means

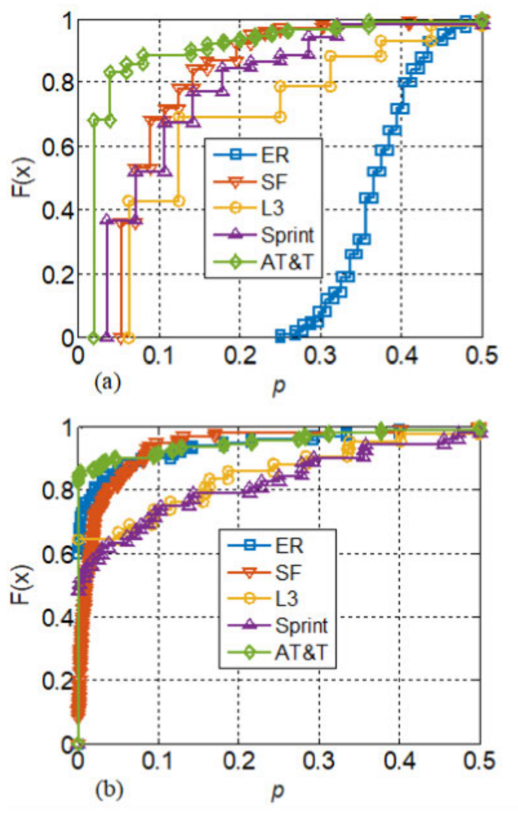


FIGURE 1. **The failure probability distribution of different networks based on node degree (a) and betweenness (b) of each network.**

$p^{\max} = 0.5$. μ_1 controls the slope of the *Sigmoid* function whereby $\mu_1 \to 0$ leads to gradual increase in p_i with increasing x while conversely, p_i will have a steep sharp transition as x increases. The ratio $-\mu_0/\mu_1$ defines where the midpoint of the function (i.e., determines the "shift" of the *Sigmoid* function along the x-axis). Hence, we can get the x value at the midpoint of the function via this ratio. Using the maximum degree/betweenness of the network, we find the μ_0 value that satisfy $p^{\max}$.

IV. SOLVING THE CENTRALIZED PROBLEM

Equations (5) to (11) are affine functions and thus, are all log-concave. Since their composite with logarithmic functions preserves concavity, the problem is a convex optimization problem over a set of convex constraints. For such problems, a unique Pareto efficient solution exists [45].

Our optimization problem above is a typical Integer Programming (IP) problem which is NP-complete. It has been shown in [46], [47] that no pseudoplolynomial algorithm is likely to exist for the general integer programming problem as this problem is strongly NP-complete. Since the centralized problem of Eq. (5) is NP-complete, we apply the Lagrangian relaxation. The Lagrangian relaxation uses Lagrange multipliers to reduce a part of the constraints by including the constraints in the utility function and divides the original primal problem into sub problems independent of respective variables. The optimal solution is then obtained by solving its dual problem. Solving the relaxed version of the problem gives us the lower bound of the original problem. Specifically, we relax the constraints (8) and (9) as $x_{i,k} \in [0,1]$ and $y_{i,j,k} \in [0,1]$. Then the caching decision variables X and Y can be further achieved by rounding $x_{i,k}$ and $y_{i,j,k}$ for the approximated integer solution.

To solve our problem here, we first apply Lagrangian relaxation to derive the Lagrangian dual of the primal problem (5) with regard to constraint (11), as follows:

$$L(x,y,\lambda) = \sum_{v_i \in V}\Big[-\ln(U_i) + \sum_{o_k \in O}\sum_{v_j \in \eta_i} \lambda_{i,j,k}(y_{i,j,k} - x_{j,k})\Big] \tag{13}$$

where $\lambda \geq 0$ is the non-negative Lagrangian multiplier, associated with $x_{j,k}$ and $y_{i,j,k}$. However, $x_{j,k}$ is constrained by cache capacity of the node, given in (6), and $y_{i,j,k}$ is constrained by (7) which ensures that a node only retrieve one maximum of one complete object per request. Applying Lagrangian multipliers and substituting (6) and (7), we have

$$\begin{aligned} &L(x,y,\lambda,u,v,\alpha,\beta) \\ &= \sum_{v_i \in V}\Big[-\ln(U_i) + \sum_{o_k \in O}\sum_{v_j \in \eta_i} \lambda_{i,j,k}(y_{i,j,k} - x_{j,k}) \\ &\quad + \sum_{o_k \in O} u_{i,k}(x_{i,k} - C_i) + \sum_{v_j \in \eta_i} v_{j,k}(y_{i,j,k} - 1)\Big] \end{aligned} \tag{14}$$

where $u_{i,k} \geq 0$ and $v_{j,k} \geq 0$ are two Lagrangian multipliers associated with the problem (5). The constraints (8), (9) and (11) can be combined and relaxed as: $0 \leq y_{i,j,k} \leq x_{i,k} \leq 1$, $\forall v_i\ v_j \in V$, $o_k \in O$. $x_{i,k} \leq 1$ and $y_{i,j,k} \geq 0$ are further relaxed from (8) and (9) based on (11). Applying Lagrangian multipliers and substituting these two constraints, we get:

$$\begin{aligned} &L(x,y,\lambda,u,v,\alpha,\beta) \\ &= \sum_{v_i \in V}\Big[-\ln(U_i) + \sum_{o_k \in O}\sum_{v_j \in \eta_i} \lambda_{i,j,k}(y_{i,j,k} - x_{j,k}) \\ &\quad + \sum_{o_k \in O} u_{i,k}(x_{i,k} - C_i) \\ &\quad + \sum_{v_j \in \eta_i} v_{j,k}(y_{i,j,k} - 1) + \sum_{o_k \in O} \alpha_{i,k}(x_{i,k} - 1) \\ &\quad - \sum_{o_k \in O}\sum_{v_j \in \eta_i} \beta_{i,j,k} y_{i,j,k}\Big] \end{aligned} \tag{15}$$

where $\alpha \geq 0$, $\beta \geq 0$ are nonnegative Lagrangian multipliers.

The objective function is concave and continuously differentiable. All constraints on the variables are affine. Hence, Karush-Kuhn-Tucker (KKT) conditions which are necessary and sufficient for the existence of an optimal solution apply [48]. Thus, the optimal caching strategy X and Y can be derived by solving the set of equations below for all nodes:

$$\begin{cases} \lambda_{i,j,k}(y_{i,j,k} - x_{j,k}) = 0, & \forall\, v_i,\ v_j \in V,\ \ o_k \in O \\ \sum\limits_{o_k \in O} u_{i,k}(x_{i,k} - C_i) = 0, & \forall\, v_i \in V,\ \ o_k \in O \\ \sum\limits_{v_j \in \eta_i} v_{j,k}(y_{i,j,k} - 1) = 0, & \forall\, v_j \in V,\ \ o_k \in O \\ \alpha_{i,k}(x_{i,k} - 1) = 0, & \forall\, v_i \in V,\ \ o_k \in O \\ \beta_{i,j,k} y_{i,j,k} = 0, & \forall\, v_i,\ v_j \in V,\ \ o_k \in O \end{cases} \tag{16}$$

The centralized solution for Eq. (5) requires the information from all nodes to compute the optimal solution. A standard solver needs the demand matrix of each node, cache size, content set, node failure probability set and network topology as inputs. Equation (15) has $3|O| \times |V|^2 + 2|O| \times |V| + |V|$ variables and the same number of equations. For a large $|O|$ and N, the computation overhead is prohibitive and generally infeasible for standard off-the-shelf processors. Thus, a distributed solution is next proposed as a more scalable method.

V. DISTRIBUTED SOLUTION

A. DISTRIBUTED COLLABORATIVE CACHING WITH CONSTRAINTS

Equation (16) gives the centralized solution. In this section, by decomposition, we derive a distributed solution where each node optimizes its utility locally as a subsystem, to supply an approximated solution with significant lower complexity. Our solution simultaneously considers the effect of collaboration with the neighboring nodes in η_i for each node.

From the set of constraints (6) to (11), constraint (11) is a complicating or coupling constraint [49]. We apply Lagrangian relaxation as before to simplify constraint (11) and obtain (13). Then we define the Lagrange dual function g as:

$$g(\lambda) = \inf_{x \in X, y \in Y} L(x, y, \lambda) \tag{17}$$

When the Lagrange dual is unbounded below in x and y, the dual function takes on the value -∞. Since the dual function is the point-wise infimum of a family of affine functions of λ, it is concave. The dual function Eq. (17) yields lower bounds on the optimal p^* of the problem Eq. (5), for $\lambda \geq 0$, we have:

$$g(\lambda) \leq p^* \tag{18}$$

Hence, we have a lower bound that depends on λ. To seek the best lower bound from the Lagrange dual function, the optimization problem below needs to be addressed.

$$\max\ g(\lambda), \quad \text{s.t. } \lambda \geq 0. \tag{19}$$

Eq. (5) is the primal problem and Eq. (13) is the Lagrange dual problem associated with Eq. (5). The problem (13) is a convex optimization problem since the object to be maximized is concave and the constraints are convex. This is the case regardless whether the primal problem is convex [50], [51].

We denote the optimal value of the Lagrange dual problem as d^*. By definition, the best lower bound on p^* can be obtained from the Lagrange dual function. Therefore, the following inequality can be obtained:

$$d^* \leq p^* \tag{20}$$

which holds even if the original problem is not convex.

The difference $d^* - p^*$ is referred to as the duality gap. If $d^* = p^*$, then strong duality holds, which means that the best bound obtained from the Lagrange dual function is tight. The Slater's theorem states that strong duality holds if duality gap is zero and the problem is convex [52]. Therefore, we have:

$$\max\ g(\lambda) \underset{d^*=p^*}{\Longleftrightarrow} \inf_{x \in X, y \in Y} L(x, y, \lambda) \quad \text{s.t. } \lambda \geq 0. \tag{21}$$

By standard sub-gradient optimization method, we iteratively seek the best lower bound [52]. The sub-gradient method minimizes a non-differentiable convex function. It uses step lengths that are fixed ahead of time, instead of an exact or approximate line search as in the gradient method. By combining the sub-gradient method with primal or dual decomposition techniques, we develop a simple distributed caching algorithm for our problem (detailed in Section V-B).

After decomposition, each node only needs to optimize its utility locally for a given λ by calculating:

$$\min L_i(x, y, \lambda) = -\ln(U_i) + \sum_{o_k \in O} \sum_{v_j \in \eta_i} \lambda_{i,j,k}(y_{i,j,k} - x_{j,k}) \tag{22}$$

Problem (13) is convex and convex relaxation technique is used to produce an approximated solution to the primal problem. However, this solution may not satisfy the constraints (8) and (9) of the original problem as it may include solutions with fractional values rather than binary values [53]. To resolve this, we adopt the rounding scheme introduced in [39]. The approximation process consists of two steps: we first relax the integer program to a convex optimization problem and produce a solution within a constant approximation from the optimal. In our problem, we relax the constraints (8) and (9) as $x_{i,k} \in [0, 1]$ and $y_{i,j,k} \in [0, 1]$. Then, the (possibly) fractional solution is rounded by rational approximation to produce a solution to the original integer program. To solve the problem (13), a projected sub-gradient method is used. Each node computes individually to achieve the optimal solution and the utility function converges to the optimal point. Hence, given a solution $x^*_{i,k}$ and $y^*_{i,j,k}$ with fractional values, we approximate the solution with binary values, rounding the fractional variable to its closest integer value 0 or 1.

B. DISTRIBUTED CACHING ALGORITHM

We detail our distributed caching algorithm with the corresponding pseudo code presented in Algorithm 1. At the initialization stage, a node, v_i, takes the W and λ matrices and its search radius, r_i as inputs. The outputs are caching decision

X_i and collaboration decision Y_i. The algorithm searches outwards from v_i until the search radius, r_i is reached. As shown in Lines 1-2 in Algorithm 1, the search path distance gradually increases and in each iteration (indexed by n), the distributed optimal solution for problem (13) is derived subject to constraints (Lines 4-5). δ is a constant of small value. In each iteration, $y^*_{i,j,k} - x^*_{j,k}$ is calculated (Line 6). If the condition $y^*_{i,j,k} - x^*_{j,k} \leq \delta$ is satisfied, then the defined objective function is convergent where the updated optimal solution is achieved.

Algorithm 1: Distributed Caching Algorithm of Node v_i

```
Input  : Demand matrix $W$, Dual variables matrix $\lambda$,
         Path distance starts at $l = 0$, Search radius $r_i$
Output: Caching decision $X_i$, Collaboration decision $Y_i$
1  while $l \leq r_i$ do
2      $l = l + 1$; $n = 0$; Set $f_i(\lambda^0)$;
3      while $f_i(\lambda^n) > \delta$ do
4          Solve for $x_i^n, y_i^n = \text{argmin}_{x,y}(L_i(x, y, \lambda^n))$
5          Subject to
             $\sum_{o_k \in O} x_{i,k} \leq C_i$,
             $\sum_{v_j \in \eta_i} y_{i,j,k} \leq 1$,
             and $x_{v_{i,k}}\ y_{i,j,k} \in [0, 1]$;
6          $f_i(\lambda^n) = y^*_{i,j,k} - x^*_{j,k}$ ;
7          for $v_j \in \eta_i$ do
8              Retrieve $f_j$;
9              $f_i = f_i + \sum_{v_j \in \eta_i} f_j$;
10             $\lambda = (\lambda + \gamma_n f_i(\lambda^n))_+$;
11         end
12         $n = n + 1$;
13     end
14 end
```

The projected sub-gradient solves the following (Line 3-13),

$$\text{Solve for } x_i^n, y_i^n = \text{argmin}_{x,y}(L_i(x, y, \lambda^n))$$
$$\lambda^{n+1} = (\lambda^n + \gamma_n f_i(\lambda^n))_+, \quad i = 1, \dots m \tag{23}$$

where $f_i(\lambda^n) = y^*_{i,j,k} - x^*_{j,k}$ is the sub-gradient of L_i which is given by: $f_i(\lambda^n) \in \nabla g(\lambda)$ and $\gamma_n > 0$ is the n^{th} step size which can be determined by several standard methods [52]. In this work, a non-summable diminishing step size is used [52]. Further, $(.)_+$ is the projection. In each iteration, node v_i need to solve the subsystem to update dual variable λ. Projected sub-gradient method projects λ on its constraint ($\lambda \geq 0$) in each iteration. The primal solution can be constructed from optimum λ. We assume that Slater's condition holds (convex problems with the Slater's condition), and each λ^n is a unique minimizer. Then the limit point of λ^n is primal feasible, which is also the optimal. The dual variable matrix λ is taken into account in the algorithm. λ is updated after each iteration. For each f_i, v_i retrieves o_k from neighboring node v_j where the 'cache hit' occurred and updates λ while l grows. After each iteration locally, the adjustment f_i will be updated by removing information that is not included in λ. It is noted that v_i exchanges the updated f_i within its neighborhood and λ contains the aggregated popular content in the neighborhood. This process is shown in Lines 7-11.

C. COMPUTATION COMPLEXITY AND OVERHEAD ANALYSIS

For the centralized solution, Equation (15) has $3|O| \times |V|^2 + 2|O| \times |V| + |V|$ variables and the same number of equations. This optimal centralized solution comprises of $|O|$ matrices of size $|V|^2$, which has a computation complexity of $\Theta(|O||V|^2)$. However, since we consider only r_i-hops neighbourhood of each node, our proposed Algorithm 1 does not scale with increasing network size but rather scale with the expected number of nodes within their neighbourhood. Specifically, in Algorithm 1, the collaboration distance of the neighborhood is restricted by r_i, as shown in Line 1 of Algorithm 1. We denote with N_i as the neighbourhood of node v_i with a limited collaboration distance r_i and $|\overline{N}_i|$ is the expected size of N_i. Further, following [38], let $|O'|$ be the truncated content set size according to the Zipf-like distribution. For instance, for a Zipf distribution with $\alpha = 1.0$, with a content set of 10^6, we are able to cover 72.8% of the requests by caching only 2% of $|O|$, which translate to 98% (i.e., $|O'| = 0.02|O|$) reduction. The complexity of Algorithm 1 is then significantly reduced to $\Theta(|O'||V||\overline{N}_i|)$. Comparing with the centralized solution, Algorithm 1 has reduced the computation complexity by $\frac{|O'|}{|O|} \times \frac{|\overline{N}_i|}{|V|}$.

In collaboration schemes, there is additional communication overhead for nodes to gain knowledge on the content distribution within its neighborhood to make better caching decisions. Having nodes to collect information from all nodes in the network is clearly prohibitive for network with large size. Therefore, it is important for the collaboration to be restricted to a small neighborhood. In our proposed Algorithm 1, we can see that the communication overhead due to computing λ originates from two parts: (1) replies to queries from nodes having v_i in their neighborhood, N_i^+ and (2) information collection from nodes in v_i's own neighborhood, N_i. The communication overhead is measured by the number of exchanged messages, and the overhead ϕ_i of node v_i is given by [38]:

$$\phi_i = c \times |O| \times (|N_i^+| + |N_i|) \tag{24}$$

Scalar c represents a constant factor for communication overhead, and the system communication overhead Φ due to collaboration for calculating optimal caching strategy can then be written as follows [38]:

$$\Phi = 2c \times |O| \times \sum_{v_i \in V} |N_i| \tag{25}$$

where node v_i has a neighborhood N_i uniquely determined by its search radius r_i. In a network $G(V, E)$ where a node's average neighborhood size equals $|\overline{N_i}|$, system communication overhead equals:

$$\Phi = 2c \times |O| \times |V| \times |\overline{N_i}| \tag{26}$$

For a node v_i, we can organize its neighborhood N_i into r_i concentric circles according to the neighbor's distance to v_i. We denote z_r as the average number of r-hop neighbors on the r^{th} circle, where, $|\overline{N_i}| = z_1 + z_2 + \ldots + z_r$. In a random network where nodes have average search radius r, the induced system overhead $\Delta_r^{r+1}\Phi$ by increasing the average search radius by 1 is given by [38]:

$$\Delta_r^{r+1}\Phi = 2c \times |O| \times |V| \times [\frac{z_2}{z_1}]^r \times z_1 \tag{27}$$

It shows that the increase in overhead depends on the ratio between the number of two-hop and one-hop neighbors. This applies to any general network with arbitrary degree distribution. The overhead only converges if there are less two-hop neighbors than first-hop ones, i.e., $[\frac{z_2}{z_1}] < 1$, which actually implies the graph is not connected and has multiple components [54]. This is important for collaborative caching, and shows that the collaboration overhead grows exponentially on general connected topologies. Therefore, collaboration is suggested to be restricted to a very small neighborhood to keep overhead reasonable [52].

VI. EVALUATION

In our evaluation, we use both synthetic topologies based on well-known graph models as well as real-world networks. For synthetic topologies, we use the Erdos-Renyi (ER) random graph model [55] and the Barabasi-Albert (BA) scale-free (SF) graph model [56]. We set the synthetic network size, $N = 100$ nodes. For ER model, given N, a link randomly connects a pair of nodes with probability p_r^{ER} independent of other links. This results in binomial degree distribution. For generating the ER graph, we set $p_r^{ER} = 2p_c^{ER} = 2 \times \ln(N)/N$ (i.e., two times the sharp threshold for connectedness, p_c^{ER}). This ensure the generated topology is connected at the start of the evaluation while simultaneously having low link density to avoid highly meshed topology. For SF graph, preferential attachment [56] is used and in each step, three new nodes are attached based on the degree of current nodes whereby the probability of an existing node chosen is proportional to its current degree. This results in power-law degree distribution. Hence, ER and SF graphs offer two different topological structures with distinct degree distributions. For real-world networks, we use the dataset from [57] and extract three topologies with L3 (AS1), Sprint (AS1239) and AT&T (AS7018) as the root domain. Their respective sizes are 42, 52 and 113 nodes.

The content population is randomly distributed in the network with each content object persistently hosted in one server. We set the entire content catalogue to be 100,000. Content popularity follows Zipf-distribution with popularity factor, $\alpha = 0.9537$ [58]. In a similar manner as in [38], we truncate the content population and consider the top 500 most popular content to reduce computational complexity. Further, without loss of generality, we assume all nodes have the same cache capacity, i.e., $C_i = C_j; \forall\{i, j\} \in V$ and C_i is defined as a fraction of the overall content population. We set it to 10% of the truncated content population. For the rest of the paper, we also assume unit object size, $s_k = 1$. Finally, unless otherwise specified, the default flooding scope is set to three.

We investigate different node failure probability distributions as detailed in Section III.C. We compare our solution (labeled as `Optimized`) with two common caching approaches:

- `Random` – A node caches a content at random. This algorithm is simplistic and does not require keeping any information about the content access history.
- `Greedy` – A node caches a content without collaborating with its neighbor nodes. The caching decision is aimed at maximizing its own utility. The algorithm makes its caching decisions based on its local knowledge of content popularity. This then involves the overhead of tracking the access frequency of content at the node. This algorithm closely resembles the popularity-based caching approach studied in the information-centric networking (ICN) literature (e.g., [59]).

A. CENTRALIZED SOLUTION

Using the centralized solution described in Eqs. 15 and 16, the joint utility of the problem in Eq. 5 is evaluated. Constraints 8 and 9 are relaxed to $x_{i,k} \in [0, 1]$ and $y_{i,j,k} \in [0, 1]$. Then the caching decision variables X and Y are further rounded up/down to obtain the approximated solution.

As the centralized solution has high computational complexity, we are restricted to evaluate its performance on a small network. We chose to use the L3 network with 42 nodes. In Figure 2, we compare our solution (`Optimized`) against `Random` and `Greedy` caching schemes across three failure probability distributions (i.e., Uniform, Degree-based Linear and Betweenness-based Linear). All achieved utilities are normalized against the utility achieved by our `Optimized` in a no failure scenario (i.e., when $p^{\max} = 0$). As such, in an unperturbed network, the normalized utility achieved by `Optimized` equals 1.0.

We see that `Optimized` consistently achieves the best utility against other caching schemes across different failure distributions, with `Random` performing the worst in all cases. The utility achieved by `Optimized` is up to 68% and 11% better against `Random` and `Greedy` respectively. In general, when the failure probability is increased (i.e., the network suffers more perturbations), the utility achieved decreases. This applies to all schemes. This is most obvious for the case when failures are set proportional to node degrees (Figure 2(b)). The performance gain achieved by `Optimized` is consistent for the uniform and betweenness-based failure probability distribution cases but decreases for degree-based failure probability distribution.

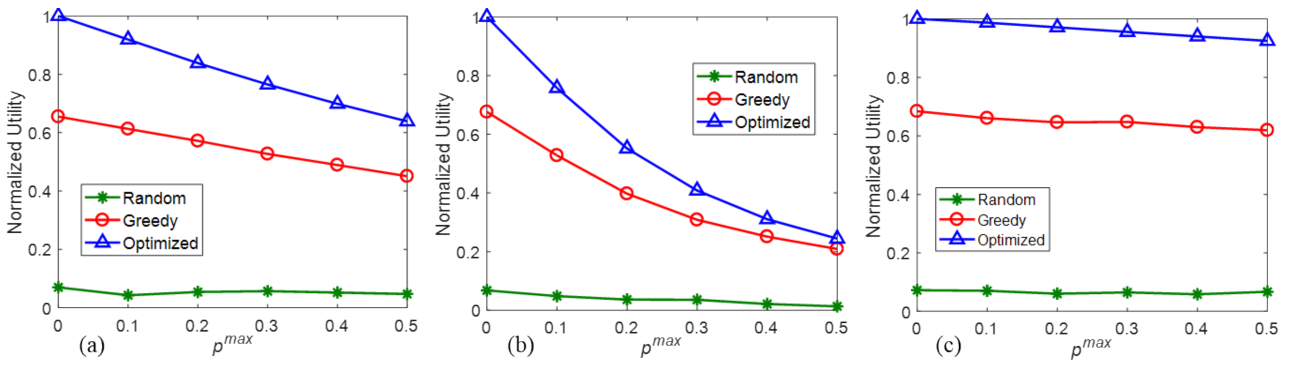


FIGURE 2. The utility in L3 with different cache schemes for (a) uniform, (b) degree-, (c) betweenness-based linear failure probability distributions.

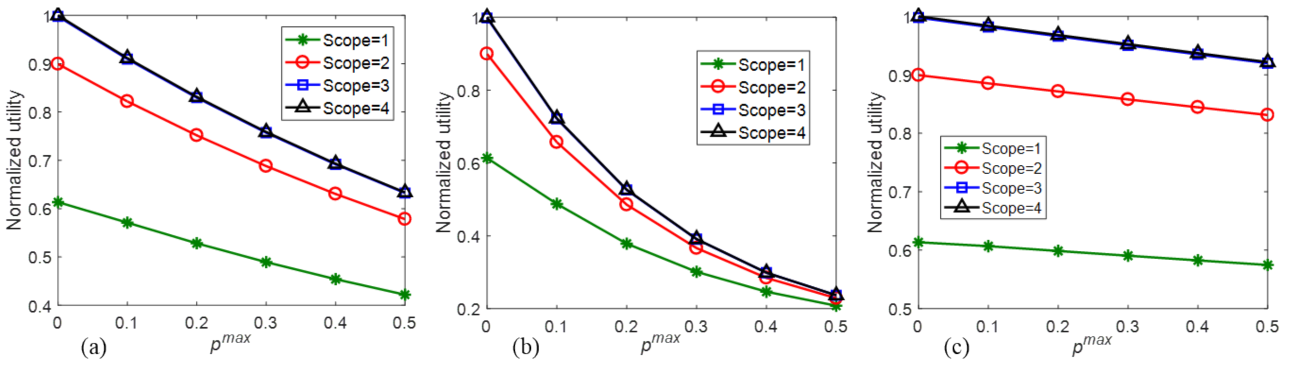


FIGURE 3. The utility in L3 with different scope-flooding radii for (a) uniform, (b) degree-, (c) betweenness-based linear failure probability distributions.

This behavior could be understood from studying Figure 1. In the case of betweenness-based linear failure probability distribution, most node failure probabilities concentrate to small values (e.g., ≈75% of nodes have failure probability less than 0.1 for the L3 network) while comparatively, the failure probabilities under the degree-based linear failure probability distribution have higher values.

Next, we investigate the impact of scope flooding content request on the achieved utility (see Figure 3 for L3 network) where we vary the flooding scope between one and four from each node to its neighbors. This experiment is computed using the same centralized approach in Eq. 2. The achieved utility increases as the flooding scope increases. However, the normalized utility for `Scope` $=$ 3 and `Scope` $=$ 4 overlaps for all three different failure probability distributions. This means that the caching gain has diminishing return and saturates at three hops (i.e., expanding the flooding scope further will not significantly improve the achieved utility). This observation validates the findings reported in [22]. This also indicates that there is no necessity to increase the node collaboration neighborhood beyond three hops away since the additional benefit will be minimal while the overhead increases.

We repeat the experiment on L3 with non-linear failure probability distributions as detailed in Section III-D3. Figure 4 shows the results. Our solution consistently achieves the best utility. Compared to the results in Figure 2, we note that with increasing $p^{\max}$, the utility of the degree-based linear model reduces more rapidly than the non-linear one. However, for the betweenness-based failure models, both linear and non-linear cases have similar performance. The improvement by `Optimized` is generally stable for different $p^{\max}$.

B. CENTRALIZED vs. DISTRIBUTED SOLUTIONS

We compare in Figure. 5 the normalized utility achieved by both our centralized solution and distributed caching algorithm for L3 network. We are restricted to use a small network because the computation for centralized solution is not feasible for large networks. For the distributed solution, we show the utility achieved for one sample node as an example (v_4 in this case). In the figure, the normalized utility achieved by the centralized and distributed approaches across the three different failure distributions (`Set1` - uniform, `Set2` - degree-based linear and `Set3` - betweenness-based linear failure probability distribution). Both approaches achieve results that closely agree with each other. Our distributed solution mostly underestimates the utility by a small margin. The difference is consistently small (i.e., greater than 96% accuracy).

C. DISTRIBUTED CACHING ALGORITHM WITH UNIFORM AND LINEAR FAILURE PROBABILITY DISTRIBUTION

In this section, we evaluate our distributed caching algorithm. We first focus on both uniform (cf. Section III-D1) and linear (cf. Section III-D2) failure distributions. We follow the

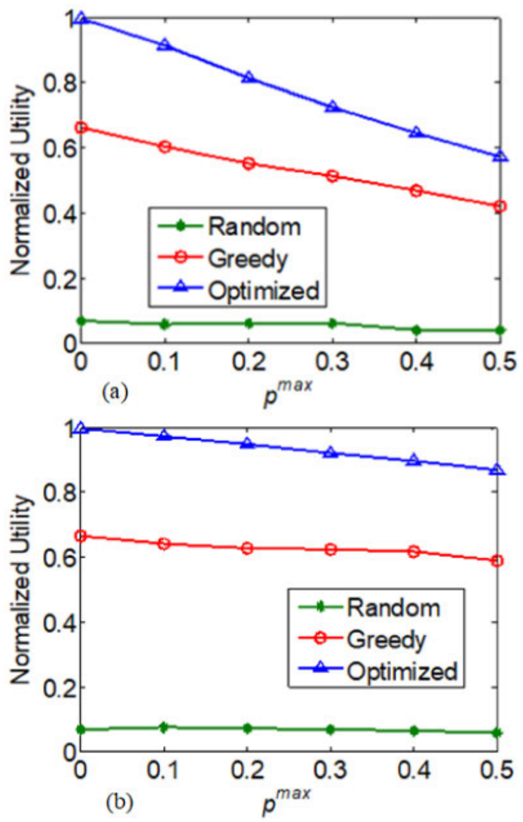


FIGURE 4. Normalized utility achieved in L3 network: (a) degree- and (b) betweenness-based non-linear failure probability distributions.

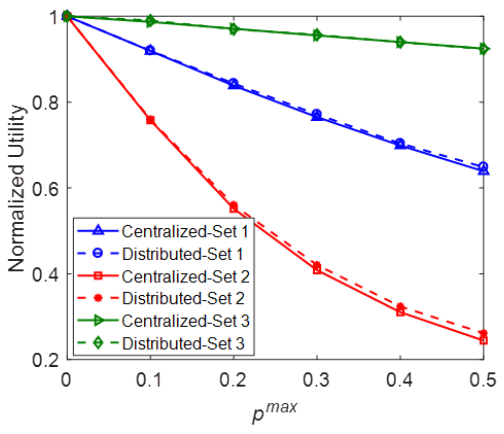


FIGURE 5. Centralized and distributed results closely match.

proposed algorithm (cf. Section V-B) by using a sub-gradient method to find an optimized distributed caching solution for each individual node. We repeat our analysis with the same input parameters over five networks (i.e., ER, SF, L3, Sprint and AT&T networks). The normalized caching utility achieved are given in Figure 6 and Figure 7 for synthetic and real world networks respectively (Solid lines for our `Optimized` solution and dashed lines for `Greedy` caching scheme.).[2] Our distributed solution achieves highest normalized utility for all the cases. With our distributed caching algorithm, we have the additional benefit of having lower complexity compared to the centralized solution yet obtaining similar caching gain.

We further investigate the impact of cache size, C_i, on our proposed distributed caching algorithm. Figure 8 shows the normalized utility achieved with cache sizes ranging between 2% and 12% of the size of the total content for the L3 network. The failure probability is set based on uniform, degree-based linear and betweenness-based linear distributions respectively with $p^{\max} = 0.5$. For this set of results, we normalize all achieved utilities against the highest utility achieved by `Optimized` scheme. The figure shows the normalized utility of a sample node in the network. The results shows that the utility achieved using distributed optimized algorithm is much higher than the `Greedy` and `Random` caching schemes. The gain in utility achieved by `Optimized` increases initially for smaller cache sizes (e.g., between 2%–8%) after which the gain achieved stabilizes, all the time maintaining better caching utility against `Greedy` and `Random` schemes.

Figure 9 presents the normalized utility achieved with different flooding scope for $p^{base} = \{0.0, 0.02, 0.04\}$ across uniform and linear failure probability distributions. The utility does not increase significantly when flooding scope increases beyond three hops, which corroborates with the results for centralized solution (see Figure 3). Further expanding the flooding scope will significantly increase the computational complexity. Hence, we recommend that the flooding scope is limited to three when `Optimized` is applied. Moreover, the highest increase in the achieved utility happens when the flooding scope is increased from one hop to two hops. This property could be exploited for the cases when computation resource is highly constrained whereas sub-optimal utility could be accepted. The overall achieved utility is negatively affected when the failure probability increases (from $p^{base} = 0$ to $p^{base} = 0.04$).

D. DISTRIBUTED CACHING ALGORITHM WITH NON-LINEAR FAILURE PROBABILITY DISTRIBUTION

In the previous section, the failure probability of a node, p_i, is either uniform or linearly proportional to its degree or betweenness centrality. We now proceed to study the performance of our distributed caching algorithm when the node failure probability distributions follow those described in Section III-D3 (i.e., node failure probability distribution follows a non-linear *Sigmoid* function).

We first present the normalized utility for degree- and betweenness-based non-linear models for ER and SF networks with $p^{\max}$ between 0.1 and 0.5. The settings of the *Sigmoid* function are listed in Table 2. σ_0 is set as $\{0.2, 0.4.0.6, 0.8, 1.0\}$ according to different $p^{\max}$. Figure 10 shows the results for degree-based and betweenness-based

[2]We omit `Random` here as it consistently performs significantly worse.

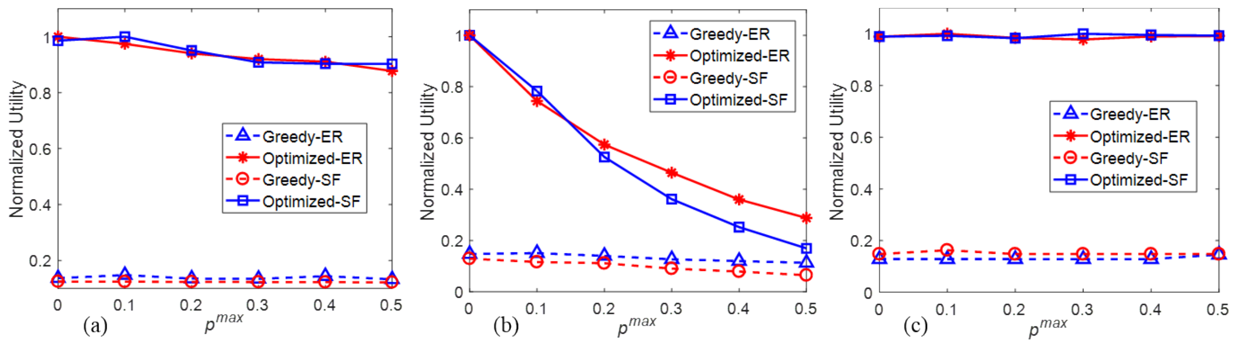


FIGURE 6. The utility of `Greedy` and `Optimized` in ER and SF for (a) uniform (b) degree and (c) betweenness based linear failure probability distributions.

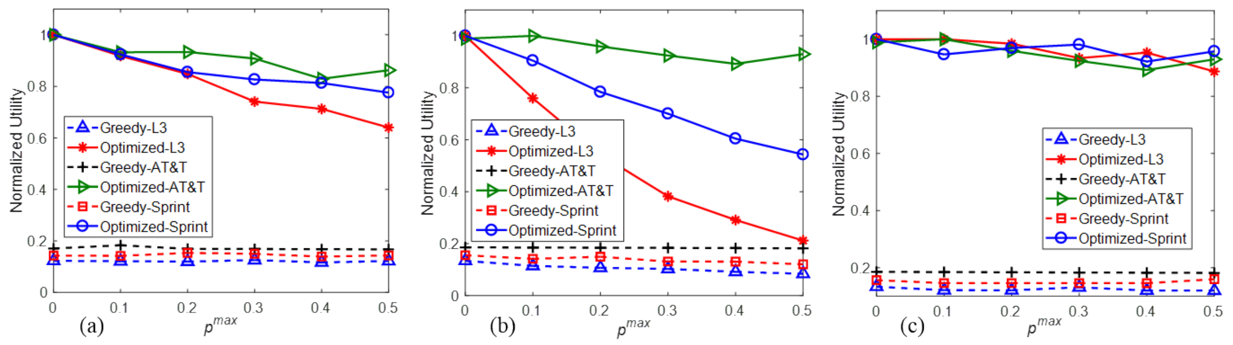


FIGURE 7. The utility of `Greedy` and `Optimized` in L3, Sprint, AT&T for (a) uniform, (b) degree, (c) betweenness-based linear failure probability distributions.

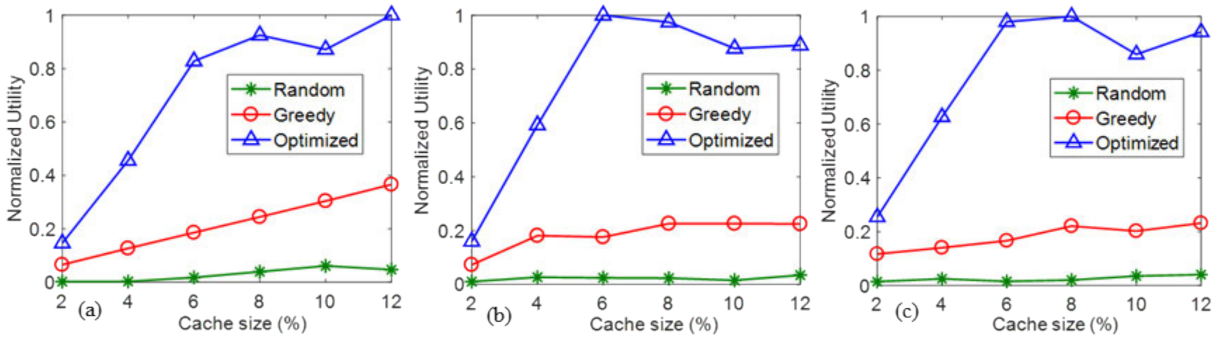


FIGURE 8. The utility with different cache sizes in L3 for (a) uniform (b) degree and (c) betweenness-based linear failure probability distribution ($p^{max} = 0.5$).

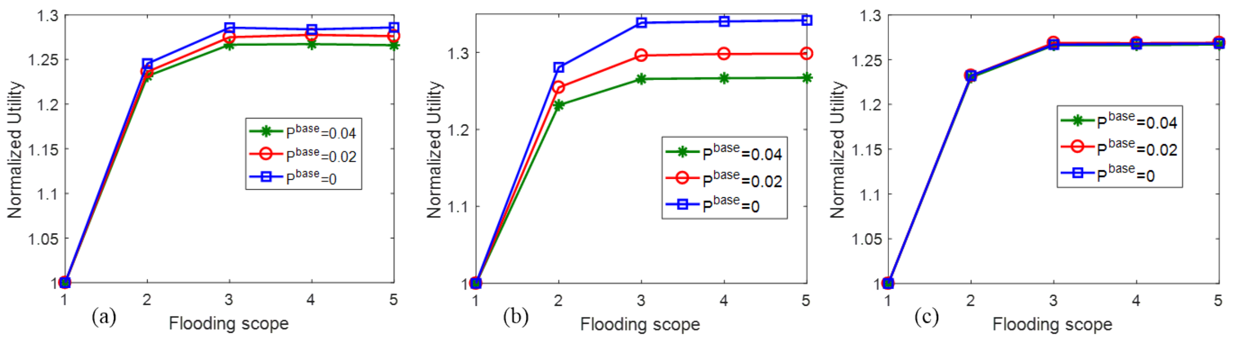


FIGURE 9. The utility with different flooding scope in L3 network for (a) uniform, (b) degree-based and (c) betweenness-based failure probability distributions.

non-linear failure models. We see that the `Optimized` caching scheme has better utility than the `Greedy` algorithm for both ER and SF topologies. Furthermore, the improvement is stable with increasing $p^{\max}$. This is because with the non-linearity, only a few nodes have very high failure probability while the rest have small failure probability.

TABLE 2. ***Sigmoid* function settings for experiments with different $p^{\max}$.**

	degree-based		betweenness-based	
Topology	μ_0	μ_1	μ_0	μ_1
ER	-10	0.5	-10	200
SF	-15	0.5	-10	40
L3	-6.4	0.8	-8.75	12.5
Sprint	-6	0.4	-5	12.5
AT&T	-5	0.2	-8.75	12.5

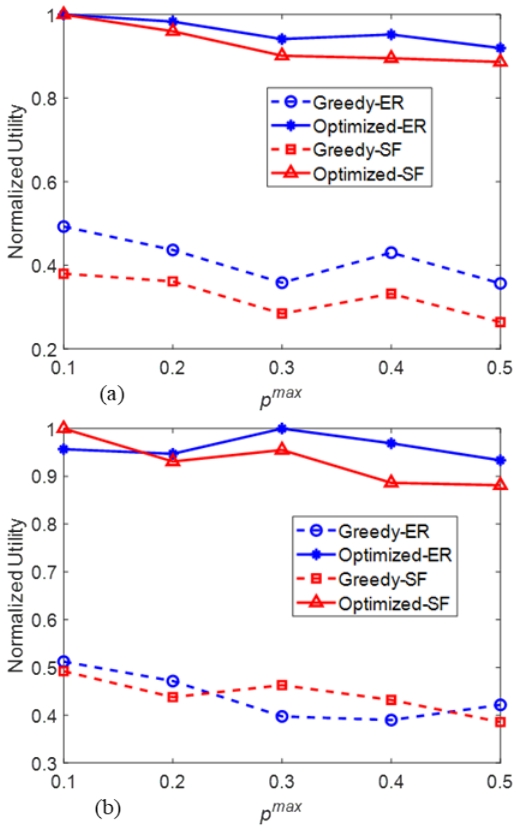


FIGURE 10. **The utility in ER and SF: (a) degree- and (b) betweenness-based non-linear failure probability distributions with different $p^{\max}$.**

For real-world networks (i.e., L3, Sprint and AT&T), we show the results in Figure 11 for degree-based and betweenness-based non-linear models. Our `Optimized` algorithm still consistently achieves higher utility. However, compared to their linear counterparts (see the linear case in Figure 7(b)–(c)), the utility achieved for L3 network decreases at a rate much faster in the linear case. This is due to the different failure distribution where in the non-linear case, there are only several nodes with very high failure probability but the rest of the network has low failure probability.

In the previous results, we gradually increase $p^{\max}$. Since p_i no longer forms a linear relationship with node degree or betweenness, the skewness of the *Sigmoid* function also influences the utility. Hence, we further provide the normalized utility for a set of *Sigmoid* functions by varying μ_0 and μ_1 parameters to get increasing mean failure probability, p^{mean}. The settings are given in Table 3. With a fixed $p^{\max}$,

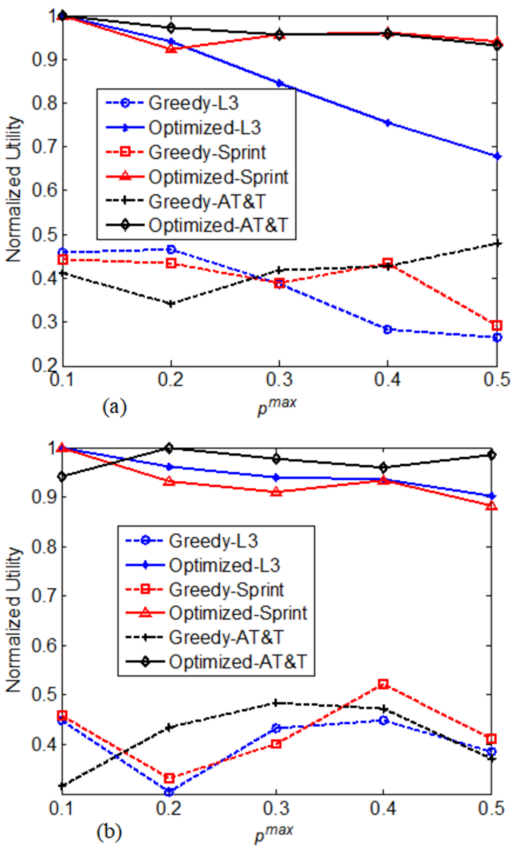


FIGURE 11. **The utility in L3, Sprint, AT&T: (a) degree-, (b) betweenness-based non-linear failure probability distributions with different $p^{\max}$.**

TABLE 3. ***Sigmoid* function settings for analysis with different p^{mean}.**

	degree-based		betweenness-based	
Network	μ_0/μ_1	μ_1	μ_0/μ_1	μ_1
L3	-8	{10, 4, 2, 0.8, 0.35}	-0.5	{100, 50, 25, 12.5, 6 }
Sprint	-15	{2, 0.8, 0.6, 0.4, 0.2}	-0.35	{200, 100, 40, 20, 8}
AT&T	-25	{0.8, 0.6, 0.4, 0.2, 0.1}	-0.63	{100, 50, 30, 10, 5}

the *Sigmoid* function becomes increasingly more skewed (i.e., most nodes having almost zero failure probability while a few nodes with highest degree or betweenness having $p_i \approx p^{\max}$) when μ_0 is increasingly more negative. Conversely, the *Sigmoid* curve increasingly becomes a linear line graph when $\mu_0 \rightarrow 0$ (see Figure 13). Figure 12 shows the normalized utility for degree- and betweenness-based non-linear failure patterns for L3, Sprint and AT&T.

For the degree-based non-linear failure model, our `Optimized` caching scheme achieves better utility than `Greedy`. However, the improvement generally decreases when p^{mean} increases. This implies that when the network is increasingly volatile, the achievable utility tends to converge regardless of the failure pattern. The improvement of

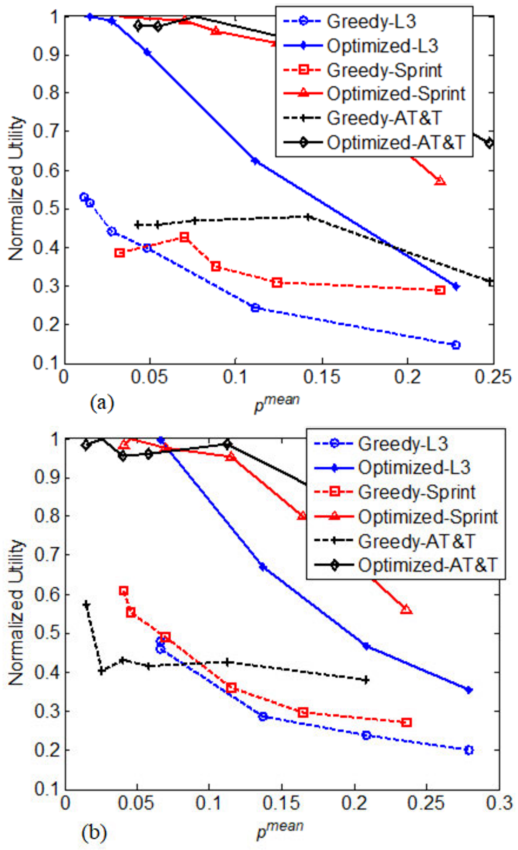


FIGURE 12. The utility in L3, Sprint, AT&T: (a) degree-, (b) betweenness-based non-linear failure probability distributions with different p^{mean}.

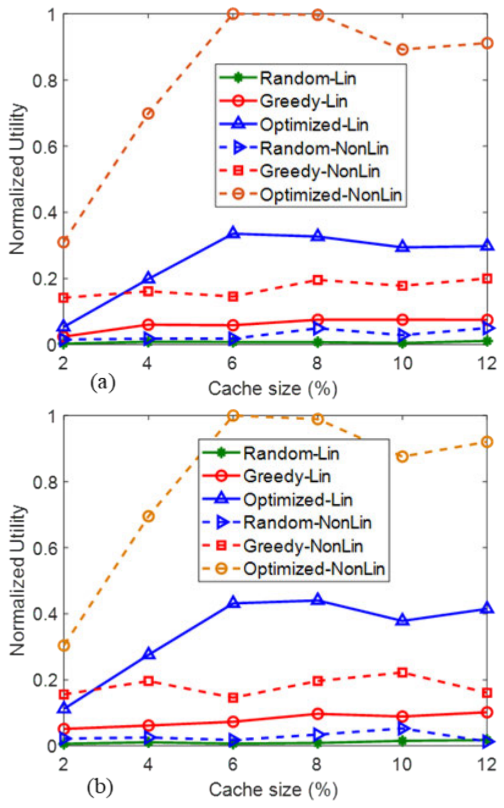


FIGURE 14. The comparison of a (a) degree-based and (b) betweenness-based linear and non-linear failure probability distributions.

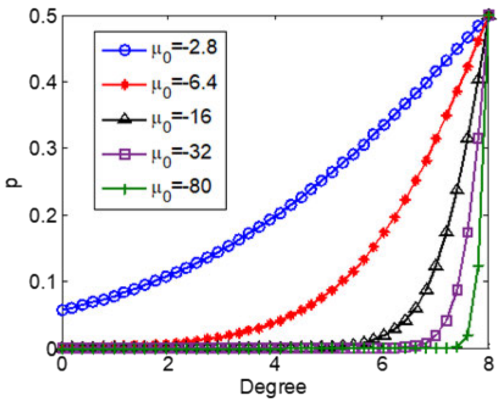


FIGURE 13. Example failure probability curves using a non-linear model for L3, $\sigma_0 = 1$, $\mu_0/\mu_1 = -8$.

Sprint and AT&T are more significant than L3. From the failure probability distribution, we notice that this improvement depends on the non-linearity of the *Sigmoid* function. Compared to Sprint and AT&T, L3 has a more uniform failure distribution.

Finally, we show how different failure patterns impact the utility across different cache sizes. Figure 14 shows the normalized utility achieved for the L3 network. For degree-based non-linear failure probability distribution, we use $p^{\max} = 0.5$, $\sigma_0 = 1$, $\mu_1 = 10$ and $\mu_0/\mu_1 = -8$ while for betweenness-based pattern we use $\mu_1 = 20$ and $\mu_0/\mu_1 = -0.7$. For both cases, `Optimized` caching schemes generally achieve higher utility than others. Comparing against itself, `Optimized` achieves higher utility for the non-linear case. This is again due to the non-linearity in the failure probability distribution.

VII. CONCLUSION

In this paper, we focus on enhancing resilience in information/content delivery. Specifically, we investigate how collaborative caching could improve information resilience in perturbed networks (considering both random and targeted failures). We first formulate a convex optimization problem for maximizing the joint utility of caching nodes in serving content requests. We solve the problem in a centralized manner by adopting a sub-gradient Lagrangian relaxation approximation, showing significant improvement in the achieved caching utility over greedy and random caching approaches. For scalabiity, we further developed a distributed caching algorithm and show it approximates the performance of the centralized solution. We study the performance in terms of utility across different networks and failure patterns, the impact of cache size and request flooding scope. While the achievable utility generally decreases when the failures in the network increase, our solution consistently achieves significantly higher caching utility compared to other schemes. Our results show that with the proposed optimized caching scheme, the utility is up to five times higher when compared against a greedy caching scheme for different networks. The performance of our scheme is also robust against increasing failure rate, even for networks with nodes having high failure probability.

REFERENCES

[1] *Akamai Technologies*. Accessed: Jan. 15, 2021. [Online]. Available: https://www.akamai.com/

[2] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5G wireless networks," *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 82–89, Aug. 2014.

[3] B. Yang, W. K. Chai, G. Pavlou, and K. V. Katsaros, "Seamless support of low latency mobile applications with NFV-enabled mobile edge-cloud," in *Proc. 5th IEEE Int. Conf. Cloud Netw. (Cloudnet)*, Oct. 2016, pp. 136–141.

[4] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Commun. Mag.*, vol. 50, no. 7, pp. 26–36, Jul. 2012.

[5] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 2, pp. 1024–1049, 2nd Quart., 2014.

[6] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache 'less for more' in information-centric networks (extended version)," *Comput. Commun.*, vol. 36, no. 7, pp. 758–770, Apr. 2013.

[7] M. Zhang, H. Luo, and H. Zhang, "A survey of caching mechanisms in information-centric networking," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1473–1499, 3rd Quart., 2015.

[8] J. Li, T. K. Phan, W. K. Chai, D. Tuncer, G. Pavlou, D. Griffin, and M. Rio, "DR-cache: Distributed resilient caching with latency guarantees," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, Apr. 2018, pp. 441–449.

[9] V. Sourlas, O. Ascigil, I. Psaras, and G. Pavlou, "Enhancing information resilience in disruptive information-centric networks," *IEEE Trans. Netw. Service Manage.*, vol. 15, no. 2, pp. 746–760, Jun. 2018.

[10] W. K. Chai, V. Sourlas, and G. Pavlou, "Providing information resilience through modularity-based caching in perturbed information-centric networks," in *Proc. Int. Teletraffic Congr. (ITC)*, 2017, pp. 214–222.

[11] E. Rietberg, L. D'Acunto, R. Kooij, and H. van den Berg, "Analyzing information availability in ICN under link failures," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2018, pp. 199–204.

[12] J. P. G. Sterbenz, D. Hutchison, E. K. Çetinkaya, A. Jabbar, J. P. Rohrer, M. Schöller, and P. Smith, "Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines," *Comput. Netw.*, vol. 54, no. 8, pp. 1245–1265, Jun. 2010.

[13] J. T. Park, J. W. Nah, and W. H. Lee, "Dynamic path management with resilience constraints under multiple link failures in MPLS/GMPLS networks," *IEEE Trans. Dependable Secure Comput.*, vol. 5, no. 3, pp. 143–154, Jul. 2008.

[14] G. Tyson, E. Bodanese, J. Bigham, and A. Mauthe, "Beyond content delivery: Can ICNs help emergency scenarios?" *IEEE Netw.*, vol. 28, no. 3, pp. 44–49, May/Jun. 2014.

[15] *Information-Centric Networking (ICNRG)*. Accessed: Jan. 23, 2021. [Online]. Available: https://datatracker.ietf.org/rg/icnrg/documents/

[16] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: A survey," *Comput. Netw.*, vol. 57, no. 16, pp. 3128–3141, 2013.

[17] A. Ioannou and S. Weber, "A survey of caching policies and forwarding mechanisms in information-centric networking," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 4, pp. 2847–2886, May 2016.

[18] W. K. Chai, M. Georgiades, S. Spirou, H. Moustafa, and S. Zeadally, "Towards information-centric networking: Research, standardization, business and migration challenges," in *Media Networks: Architectures, Applications and Standards*. Boca Raton, FL, USA: CRC Press, 2012.

[19] S. Guo, H. Xie, and G. Shi, "Collaborative forwarding and caching in content centric networks," in *Proc. IFIP Netw.*, May 2012, pp. 41–55.

[20] G. Rossini and D. Rossi, "Coupling caching and forwarding: Benefits, analysis, and implementation," in *Proc. 1st Int. Conf. Inf.-Centric Netw. (INC)*, 2014, pp. 127–136.

[21] N. Laoutaris, H. Che, and I. Stavrakakis, "The LCD interconnection of LRU caches and its analysis," *Perform. Eval.*, vol. 63, no. 7, pp. 609–634, Jul. 2006.

[22] L. Wang, S. Bayhan, J. Ott, J. Kangasharju, A. Sathiaseelan, and J. Crowcroft, "Pro-diluvian: Understanding scoped-flooding for content discovery in information-centric networking," in *Proc. 2nd ACM Conf. Inf.-Centric Netw.*, Sacramento, CA, USA, Sep. 2015, pp. 9–18.

[23] V. Sourlas, I. Psaras, L. Saino, and G. Pavlou, "Efficient hash-routing and domain clustering techniques for information-centric networks," *Comput. Netw.*, vol. 103, pp. 67–83, Jul. 2016.

[24] M. F. Al-Naday, M. J. Reed, D. Trossen, and K. Yang, "Information resilience: Source recovery in an information-centric network," *IEEE Netw.*, vol. 28, no. 3, pp. 36–42, May 2014.

[25] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. 5th Int. Conf. Emerg. Netw. Exp. Technol. (CoNEXT)*, 2009, pp. 1–12.

[26] V. Sourlas, L. Gkatzikis, P. Flegkas, and L. Tassiulas, "Distributed cache management in information-centric networks," *IEEE Trans. Netw. Service Manage.*, vol. 10, no. 3, pp. 286–299, Sep. 2013.

[27] S. Saha, A. Lukyanenko, and A. Ylä-Jääski, "Efficient cache availability management in information-centric networks," *Comput. Netw.*, vol. 84, pp. 32–45, Jun. 2015.

[28] W. Jiang, G. Feng, S. Qin, and Y. Liu, "Multi-agent reinforcement learning based cooperative content caching for mobile edge networks," *IEEE Access*, vol. 7, pp. 61856–61867, 2019.

[29] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017.

[30] S. Safavat, N. N. Sapavath, and D. B. Rawat, "Recent advances in mobile edge computing and content caching," *Digit. Commun. Netw.*, vol. 6, no. 2, pp. 189–194, May 2020.

[31] K. Avrachenkov, J. Goseling, and B. Serbetci, "A low-complexity approach to distributed cooperative caching with geographic constraints," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 1, pp. 1–25, Jun. 2017.

[32] L. Wang, S. Bayhan, and J. Kangasharju, "Cooperation policies for efficient in-network caching," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 533–534, Sep. 2013.

[33] O. Hahm, E. Baccelli, T. C. Schmidt, M. Wählisch, C. Adjih, and L. Massoulié, "Low-power Internet of Things with NDN & cooperative caching," in *Proc. 4th ACM Conf. Inf.-Centric Netw.*, Berlin, Germany, Sep. 2017.

[34] H. Wu, J. Li, and J. Zhi, "MBP: A max-benefit probability-based caching strategy in information-centric networking," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 5646–5651.

[35] *AT&T Content Delivery Network: Accelerate, Optimize, and Help Protect Your Apps and Content*. AT&T. Accessed: Feb. 5, 2021. [Online]. Available: http://www.att.com/cdn

[36] A. Wolman, M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. M. Levy, "On the scale and performance of cooperative web proxy caching," *ACM SIGOPS Operating Syst. Rev.*, vol. 33, no. 5, pp. 16–31, Dec. 1999.

[37] M. Hajimirsadeghi, N. B. Mandayam, and A. Reznik, "Joint caching and pricing strategies for popular content in information centric networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 3, pp. 654–667, Mar. 2017.

[38] L. Wang, G. Tyson, J. Kangasharju, and J. Crowcroft, "Milking the cache cow with fairness in mind," *IEEE/ACM Trans. Netw.*, vol. 25, no. 5, pp. 2686–2700, Oct. 2017.

[39] S. Ioannidis and E. Yeh, "Adaptive caching networks with optimality guarantees," in *Proc. Int. Conf. Meas. Modeling Comput. Sci. (ACM SIGMETRICS)*, Jun. 2016, pp. 113–124.

[40] S. Ghemawat, H. Gobioff, and S. Leung, "The Google file system," *ACM SIGOPS Operating Syst. Rev.*, vol. 37, no. 5, pp. 29–43, 2003.

[41] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: Measurement, analysis, and implications," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 350–361, Aug. 2011.

[42] A. Barabasi, *Network Science*. Cambridge, U.K.: Cambridge Univ. Press, 2016.

[43] S. Trajanovski, J. Martin-Hernandez, W. Winterbach, and P. Van Mieghem, "Robustness envelopes of networks," *J. Complex Netw.*, vol. 1, no. 1, pp. 44–62, Jun. 2013.

[44] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 1994.

[45] A. Muthoo, *Bargaining Theory With Applications*. Cambridge, U.K. Cambridge Univ. Press, 1999.

[46] C. H. Papadimitriou, "On the complexity of integer programming," *J. ACM*, vol. 28, no. 4, pp. 765–768, 1981.

[47] M. R. Garey and D. S. Johnson, "'Strong' NP-completeness results: Motivation, examples, and implications," *J. ACM*, vol. 25, no. 3, pp. 499–508, 1978.

[48] J. Borwein and A. S. Lewis, *Convex Analysis and Nonlinear Optimization: Theory and Examples*. New York, NY, USA: Springer, 2010.

[49] D. Niu and B. Li, "An efficient distributed algorithm for resource allocation in large-scale coupled systems," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 1501–1509.

[50] L. Xiao, M. Johansson, and S. P. Boyd, "Simultaneous routing and resource allocation via dual decomposition," *IEEE Trans. Commun.*, vol. 52, no. 7, pp. 1136–1144, Jul. 2004.

[51] G. Wanka and R. Bott, "On the relations between different dual problems in convex mathematical programming," in *Operations Research Proceedings*. Berlin, Germany: Springer-Verlag, 2002, pp. 255–262.

[52] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[53] A. A. Ageev and M. I. Sviridenko, "Pipage rounding: A new method of constructing algorithms with proven performance guarantee," *J. Combinat. Optim.*, vol. 8, no. 3, pp. 307–328, Sep. 2004.

[54] S. Bornholdt, and H. G. Schuster, *Handbook of Graphs and Networks*. Hoboken, NJ, USA: Wiley, 2003.

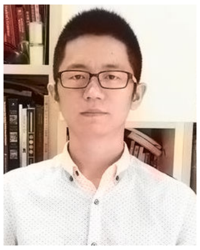
[55] P. Erdős and A. Rényi, "On random graphs I," *Pub. Math. Debrecen*, vol. 6, no. 38, pp. 290–297, 1959, ch. 2.

[56] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[57] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 32, no. 4, pp. 133–145, 2002.

[58] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system," in *Proc. 7th ACM SIGCOMM Conf. Internet Meas. (IMC)*, 2007, pp. 1–14.

[59] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack, "WAVE: Popularity-based and collaborative in-network caching for content-oriented networks," in *Proc. IEEE INFOCOM Workshops*, Mar. 2012, pp. 316–321.



DEHAO WU received the bachelor's degree in optical and information engineering, the M.Sc. degree in microelectronics and telecommunication engineering, and the Ph.D. degree in optical wireless communications from Northumbria University, Newcastle, U.K., in 2007, 2009, and 2013, respectively. He is currently a Senior Lecturer with Bournemouth University (BU), U.K. Before joining BU, he has been holding the role of Sussex Research Fellow and a Senior Research Fellow with the Department of Engineering and Design, University of Sussex, since April 2018. Before joining Sussex, he was a Postdoctoral Research Fellow with Nanyang Technological University, Singapore, and The University of Manchester. He has published more than 45 peer-reviewed papers in international conferences and journals, two book chapters, and two patents in indoor high-accuracy positioning technology. His current research interests include machine-learning algorithms and intelligent management of 5G and beyond-5G systems, artificial intelligence for 5G verticals, and the IoT. His other research interests include the area of optical wireless communications, free space optics, visible light communications, RF communications, indoor high-accuracy positioning, optical sensing and detecting, underwater RF and acoustic communications, information-centric networking, and information resilience. He is an Executive Committee Member of IET RF and Microwave Technical and Professional Networks (TPN), and also service as the technical committee member for a number of IEEE conferences.

WEI KOONG CHAI (Senior Member, IEEE) received the B.Eng. (Hons.) degree in electrical engineering from Universiti Teknologi Malaysia, Johor Bahru, Malaysia, in 2000, and the M.Sc. (Hons.) and Ph.D. degrees from the University of Surrey, Surrey, U.K., in 2002 and 2008, respectively. He is currently a Principal Academic with Bournemouth University (BU), U.K, where he heads the Future and Complex Networks Research Group (FlexNet), Department of Computing and Informatics. He is also a Visiting Academic with the Department of Electronic and Electrical Engineering, University College London (UCL), U.K. Previously, he was a Senior Research Associate at UCL, where he led the research activities of several research projects. He has successfully raised research funding from both EU and U.K. funding bodies. He has published papers in fully refereed international conferences and journals and contributed chapters to books. He serves on the technical program committee of various IEEE/ACM international conferences and workshops.