future internet

MDPI

Article

Proactive Caching at the Edge Leveraging Influential User Detection in Cellular D2D Networks

Anwar Said [1], Syed Waqas Haider Shah [1], Hasan Farooq [2], Adnan Noor Mian [1,*], Ali Imran [2] and Jon Crowcroft [3]

[1] Department of Computer Science and Electrical Engineering, Information Technology University (ITU), Lahore 54000, Pakistan; anwar.said@itu.edu.pk (A.S.); waqas.haider@itu.edu.pk (S.W.H.S.)
[2] BSON Lab, ECE, University of Oklahoma, Norman, OK 73019, USA; hasan.farooq@ou.edu (H.F.); ali.imran@ou.edu (A.I.)
[3] Computer Lab, University of Cambridge, Cambridge CB2 1TN, UK; jon.crowcroft@cl.cam.ac.uk
* Correspondence: adnan.noor@itu.edu.pk

Received: 28 July 2018; Accepted: 28 August 2018 ; Published: 21 September 2018

check for updates

Abstract: Caching close to users in a radio access network (RAN) has been identified as a promising method to reduce a backhaul traffic load and minimize latency in 5G and beyond. In this paper, we investigate a novel community detection inspired by a proactive caching scheme for device-to-device (D2D) enabled networks. The proposed scheme builds on the idea that content generated/accessed by influential users is more probable to become popular and thus can be exploited for pro-caching. We use a Clustering Coefficient based Genetic Algorithm (CC-GA) for community detection to discover a group of cellular users present in close vicinity. We then use an Eigenvector Centrality measure to identify the influential users with respect to the community structure, and the content associated to it is then used for pro-active caching using D2D communications. The numerical results show that, compared to reactive caching, where historically popular content is cached, depending on cache size, load and number of requests, up to 30% more users can be satisfied using a proposed scheme while achieving significant reduction in backhaul traffic load.

Keywords: D2D communication; proactive caching; community detection; centrality measure; influential users; social network

1. Introduction

Generational shifts in the world of mobile networks have been driven by the unprecedented growth in mobile data traffic. The exponentially growing number of connected devices and the unquenchable thirst for better mobile broadband experience is also a driving force behind this evolution. The story of 5G is no more different as it is envisioned to provide a gigabit experience and virtually zero latency while withstanding an expected 500-fold increase in mobile video traffic over the next ten years. It is predicted that 75% of global mobile data will be video content in which 6.7% will be machine-to-machine (M2M) communication. Such massive growth of multimedia traffic was further fueled by social media feeds, such as Facebook and Twitter (representing 15% of the traffic [1]). This trend is undoubtedly going to stress the capacity of core networks, wireless links and mobile backhauls to their limits eventually leading to poor quality-of-experience (QoE).

In order to cope with this issue, a 5G network is gearing up with various technologies including device-to-device (D2D) communication [2], Massive MIMO (Multiple-input multiple-output) [3,4], Device-centric architecture [5], Small Cells, Caching, mmWave communication [6] all orchestrated by self organizing networks (SON) [7]. Among them, caching through D2D mode of communication is being considered a promising technology aimed for reducing backhaul load and minimizing latency [5].

The key idea of D2D enabled caching is to store the popular content of the network at the user end. The key benefit here is that it brings likely to be accessed content as close as possible to the users. This approach can reduce latency substantially, a key challenging requirement in 5G [8]. It can also increase the effective throughput and reduce the backhaul load [9]. The three key research questions in designing an optimal caching enabled next generation Radio Access Network (RAN) are: *Where to cache? How to cache?* and *What to cache?* Where to cache focuses on the problem of caching the content over the network in such a way that it reduces the network traffic. How to cache focuses on multiple ways of content caching, and what to cache focuses on the problem of finding content having the highest probabilities to be used in the near future. In order to address these questions, various attempts have been made by the researchers such as [9–13]. For a detailed review of caching in RAN, the reader is referred to [14,15]. However, to the best of the authors' knowledge, this is a first study that investigates the potential of the leveraging detection of influential users via centrality measure in cellular networks and then uses that detection for identifying what to cache and pro-actively caching those likely to be accessing content using D2D communications.

Online Social Networks, one of the key sources of mobile data explosion, are known to be scale-free networks, which follow power-law distribution. For example, the Twitter network has millions of users and an enormous flow of information is published daily. There are small number of Twitter users who have a large number of followers while there is a huge number of users who have a small number of followers. In this paper, our key idea to design and evaluate a novel proactive yet low complexity caching solution is as follows. It is highly probable that content accessed or generated by influential users will become popular due to their relatively higher following and connectivity. Furthermore, popular content is expected to be in demand in the same community i.e., the community of the influential user first and then spread slowly in the remaining network. The mapping structure, information diffusion and influence phenomena of the social networks can thus be exploited for proactively caching those content which are expected to be highly demanded in the near future, for reducing the backhaul traffic load [16]. Therefore, we represent a mobile cellular network in the form of a graph and use graph theory and Social Network Analysis (SNA) techniques to improve QoE and reduce the backhaul traffic load.

This study considers social networks aware D2D caching by focusing on influential users, their communities, and content. Initially, we use a preferential attachment model [17] for representing the mobile cellular network. We then perform clustering using a CC-GA algorithm [18], and find influential users using an eigenvector centrality measure [16]. Finally, based on the users' previous history, we proactively cache the content of the influential users with respect to the available cache size. We perform extensive system level experiments to evaluate the gain of the proposed approach. The results reveal that 48% of backhaul traffic load can be reduced (48% satisfied requests) when users generating requests are 100% using the proposed approach.

The rest of the paper is organized as follows. Section 2 presents a brief background to caching methodologies in future cellular networks. Section 3 gives the most important related work. Section 4 discusses the methodology of the proposed approach. Section 5 presents the experimental setup and numerical results. Finally, Section 6 summarizes and concludes the paper.

2. Caching in Future Cellular Networks

We have seen a tremendous growth in cellular networks in the last decade. This growth is mainly driven by the exponentially growing demands of a high data rate by the end users. Various techniques have been employed to achieve a high data rate as well as spectral efficiency. Spectral efficiency can be achieved by efficiently managing cellular spectrum. These techniques range from control data split architecture (CDSA) [19], a cloud-radio access network (CRAN) to ultra-dense cellular networks (UDCN) [20] and D2D communication [21]. Almost all of these technologies are considered as enabling technologies for an upcoming fifth-generation (5G) cellular network. Among these, multi-tier architecture and D2D communication are well thought out as key enablers for providing access to a massive number of users as well as high data rate connectivity. In multi-tier architecture, there is an overlaid macro-cell (MC) comprising multiple micro-cells (mCs) operating in its coverage area. MC provides an overall coverage service and mobility management, whereas mCs can provide local area services and high data rate connectivity.

On the other hand, caching in cellular networks has been introduced to exploit storage capacity of diverse network devices. These devices range from user side (user equipment (UE), edge devices) to the network side (mCs, MC and extended packet core). In the following, we briefly describe how caching can be done on these diverse network devices.

Edge Caching: When an UE generates a request for specific content, first it will search for that content in its own memory and if that content is cached locally (edge), the UE will access that without any delay. This kind of caching can drastically reduce the backhaul traffic as well as access delay at the cost of large UE memory requirements.

Cluster-Head Caching: If the requested content is not available at UE itself, it will then ask its peers for that content through D2D communication. One way is to search for the content sequentially in every peer device. The other way could be to search for an influential user where it is highly probable that content accessed or generated by this influential user will become popular due to its relativity higher connectivity [22]. D2D communication will enable this kind of caching in the network. This mechanism also reduces backhaul traffic and access delay.

Micro-Cell Caching: If the requested content is not found using D2D communication, then mC will provide that content to the user if it is cached there. mC will utilize a radio access network (RAN) for delivering that content and not affect the backhaul channel.

Macro-Cell Caching: If the content can not be accessed using the above-mentioned ways, then MC will provide that content by downloading it from EPC (Evolved Packet Core) or from the cloud itself. It will ensure that content is delivered successfully. This mechanism can cause large access delays and an increase in backhaul traffic.

In the above, we answer the question *where to cache?*; however, in order to understand *how to cache?*, we might have two different approaches such as reactive and proactive. In the following, we describe these approaches in the context of cellular communication.

Reactive Caching: In the reactive caching approach, files are stored in cache if they were repeatedly requested in the past based on the history of the files accessed. In this caching, any content that remains in high demand for a specific period of time can be cached—for example, viral videos, popular tweets and other highly requested social media content.

Proactive Caching: It is a mechanism to predict future content that might be requested by the users. Based on previous content and user histories, certain content might become popular in the near future, which will be cached before they are actually requested, e.g., popular entities in various societies might trigger similar kinds of trends such as top trends on Twitter, etc. Due to the exponential increase of the demand of cellular data, it is very difficult to maintain the user satisfaction rate. Therefore, the importance of backhaul is increasing dramatically. Various studies [11,12] have shown that proactive caching in D2D communication can reduce a significant amount of backhaul traffic load, and is a smarter way to exploit D2D cellular networks. Using this approach in D2D communication in small cellular networks, the popular content is cached at the UE.

In the next section, we focus on the related work that deals with proactive caching using D2D communication.

3. Related Work

In recent years, a large number of studies have investigated caching on the edge in small cell networks. In [9], a proactive caching approach has been proposed. The authors studied two cases and utilized the spatial and social structure of the network. First, based on correlations among users and file popularity, files are cached during off-peak time. Secondly, influential users are detected and strategic content is cached using D2D communication and social networks. The authors used an eigenvector centrality measure for detecting influential users of the social network and model the content dissemination as a Dirichlet Process (CRP). In the experimental setup, a preferential attachment model is used to map the D2D small cellular network. The numerical results show a significant improvement in gain ratio; however, the proposed technique does not consider content of influential users in a specific community, which makes it less suitable for D2D communication in future cellular networks. The study in [11] introduced a new QoE metric for satisfying a given file request using proactive caching and then they proposed an optimization algorithm (*propCaching*) to maximize QoE. This algorithm is based on the popularity statistics of the requested files and cache files with highest popularity. However, unlike our proposed work, the algorithm has a limitation in that it chooses the popular files for caching based on their statistics or previous history. The work in [23] considers UE devices as cooperating nodes on which distributed cache is implemented for efficient downloading. However, this approach has certain drawbacks. Firstly, it does not consider the distance between D2D devices for cooperation and, secondly, it stores random content without prioritizing high in demand content. Golrezaei et al. [10] introduces a collaborative architecture which uses the distributed storage of popular content. In order to choose and cache random files at the user end, authors compute an average number of D2D links that can coexist without interference. However, this work also does not consider content of influential users to cache.

A probabilistic approach for optimizing scheduling policies and D2D caching is considered in [24]. The authors first derived approximated uploading probability for uploading gain and then they optimize the scheduling factor and caching distribution to optimize the successful offloading probability. In [25], authors investigate an optimal caching strategy by formulating an optimization problem and showing the relationship between D2D caching distribution and demand distribution for homogeneous Poisson Point Process models with different noise levels. With the exponential increase in multimedia traffic, content caching at every edge node might become a challenging task. To overcome this problem, authors in [26] proposed an idea of peer-to-peer content delivery networks. The authors have exploited the benefits of distrusted fog nodes for content delivery among the networks. In order to efficiently deploy enabling 5G technologies, authors in [27] proposed an algorithm to reduce total installation costs. The proposed algorithm considers both the hardware and cloud enabled softwarized components (reusable functional blocks) to ensure the users' good performance as well as reduction in computation times. Kennington et al. [28] gives a comprehensive overview of modelling and solving optimization problems arising in wireless networks. Moreover, for a detailed review of applications of big data and caching techniques in mobile computing, the reader is referred to [9,29]. Table 1 summarizes caching strategies in comparison with the one proposed in this paper.

Table 1. Different caching techniques and their methodologies.

Objective	**References**	**Cache Location**	**Methodology**
Cache/Contents Placement	[9]	UE	Using SNA and CRP
	[10,11]	UE	Files popularity statistics
	[23]	UE	Random content
	[24]	UE	Probabilistic approach
	[25]	UE	Optimization
	[30]	Small Cell	Files instantaneous popularity learning and multi-armed bandit (MAB) technique
	[31,32]	Small Cell	Coded placement strategy
	[33]	Small Cell	Problem formulation using cost of files retrieval and bandwidth
	This paper	UE	SNA and influential users content as file popularity index

Finding social hubs in social networks remains an active research area of Social Network Analysis (SNA). Various measures including Eigenvector Centrality measure, Betweenness measure, Closeness measure, Degree centrality measure and PageRank algorithm have been used extensively for finding the nodes importance in social graphs. Among these measures, Eigenvector centrality measure is the most widely used measure for detecting social hubs in social graphs. This measure has been used by three popular methods: PageRank [34], HITS (Hypertext Induced Topic Search) [35], and SALSA (Stochastic Approach for Link Structure Analysis) [36] for retrieving web information.

It is worth mentioning that several prior studies exist that propose various design approaches for proactive caching schemes [9,16,37]. However, the novelty of the proposed work stems from the simple yet under-explored idea of considering the content of influential users as popular content. Furthermore, our work uses a new community detection algorithm for detecting communities in the network and then k influential users are found, one for each community. In the next section, we will explain our proposed methodology for proactive caching in socially aware D2D networks.

4. Proposed Methodology

We exploit the social and spatial structure of social networks and use SNA approaches to find the content of the network that is expected to become viral in the future and proactively cache them in order to reduce the backhaul load. As the number of active users increases in the network, load on Small Base Station (SBS) increases. Therefore, D2D communication can play a vital role in reducing the load and increasing the user satisfaction rate. The D2D communication paradigm can be exploited to perform proactive caching and store highly probable content in the users' caches. In this paper, we introduce a new proactive caching approach, which uses SNA techniques to address this problem. In the following subsection, after describing the system model and the essentials, we shall present our proposed algorithm.

4.1. System Model

Assume that there are n number of users who generate number of requests $R = \{r_1, r_2, ...r_n\}$, number of files $F = \{f_1, f_2, ...f_n\}$ with lengths $L = \{l_1, l_2, ...l_n\}$, and bit rate $B_r = \{b_1, b_2, ...b_n\}$. Note that bit rate of the file represents the transferring rate of the file per unit time. The requests are generated by the users in different time frames $T = \{t_1, t_2, ...t_n\}$. A request $r_1 \in R$ generated by user n is said to be satisfied iff D2D link capacity is higher than the bit rate of the file. The user satisfaction can be formulated as follows:

$$B_r \leq \frac{l_i}{t'_i - t_i}, \tag{1}$$

where l_i is the size(length) of the file, t'_i and t_i is the end and starting time of delivery of file i. B_r represents the bit rate of file f. Based on Equation (1), the user satisfaction rate S_r can be defined as:

$$S_r = \frac{1}{R} \sum_{r \in R} x_r, \tag{2}$$

where

$$x_r = \begin{cases} 1, & \text{if } \dfrac{L}{t'_z - t_z} \geq B_r, \\ 0, & \text{otherwise.} \end{cases}$$

In order to reduce backhaul traffic load, our main goal is to maximize S_r.

The users are considered to be connected with other users of their community in a D2D mode of communication. Thus, the request r of the user i for a file f is directed to its nearest user to search the requested file and will be served if the file exists in the nearby cache. Thus, a good proactive caching strategy can satisfy more requests at the user end, which results in reducing the backhaul traffic load.

The architecture considered in this work assumes the D2D communication enabled small cellular network. Every community in the network finds a central node that can serve a maximum number of D2D users with a minimum distance as shown in Figure 1. There are a total of n number of users that are connected with each other via D2D communication with a link capacity C_b. The users are also connected with the small cell base station with a link capacity C_w. Assume that the $\sum C_w \leq \sum C_b$, then the average backhaul traffic load due to each file of length l_i can be formulated as follows [13]:

$$B_l(r) = \frac{1}{R} \sum_{r \in R} \frac{1}{l_i} \sum_{t=tr}^{t=t'} \lambda_r(t), \tag{3}$$

where $\lambda_r(t)$ is the backhaul data rate during the content delivery for request r.

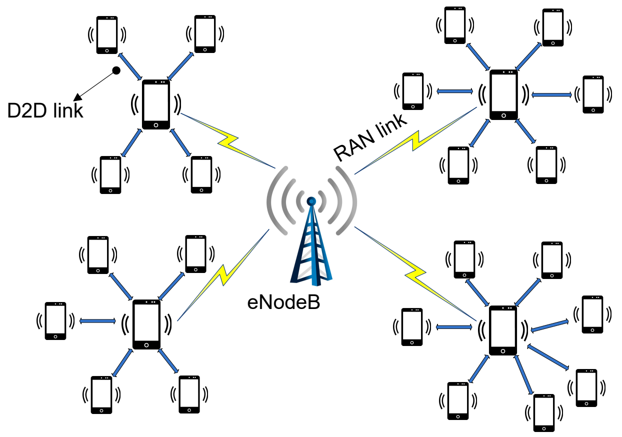

Figure 1. Small Cell Social Network Aware D2D communication

We consider the network as a graph $G = (V, E)$ where V represents set of users $V = \{v_1, v_2, v_3, ...v_n\}$ and E is set of social links $(E \subseteq \binom{v}{2})$ between the users. For generating the social network for our experimental setup, we used a well known model called preferential attachment proposed by Albert Barabasi [17] to form the small cellular network. This model basically represents degree distribution of the random generated network i.e., how users connect to each other. The core idea behind the preferential attachment model is that new users prefer to connect to well-connected users over less-connected users proportional to the probability of their degree. The generated network using this model follows power law distribution with exponent $\alpha = 3$. This process is also known as rich-get-richer or cumulative advantage or Matthew effect. The process starts with some initial subgraph. Each new user comes with some initial m links. The probability $\Pi(u)$ of connecting user u with other users j in the network is:

$$\Pi(u) = m \frac{k_u}{\sum_j k_j}, \tag{4}$$

where $\Pi(u)$ represents the probability of connecting user u with other users j in the network. m represents the number of links to which the new user joins the network and k represents the degree of each user in the network. This process results in a network with a power-law distribution with exponent = 3. Figure 2 shows a network of 256 users generated using a preferential attachment model where nodes represent users while links represent the D2D communication. We see that the network contains very few users having a large number of links while the rest of the users have very few links. The degree distribution of the generated network on a log–log scale is plotted in Figure 3. We can see that the network degree distribution follows power-law distribution.

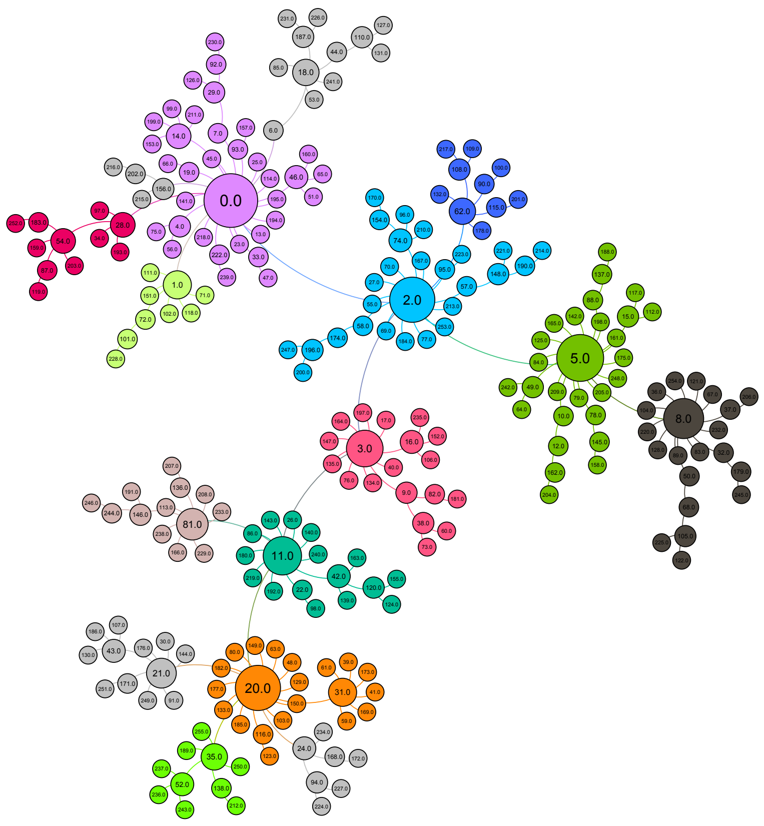

Figure 2. Synthetic network consisting of 256 users generated using the Barabasi Albert Model.

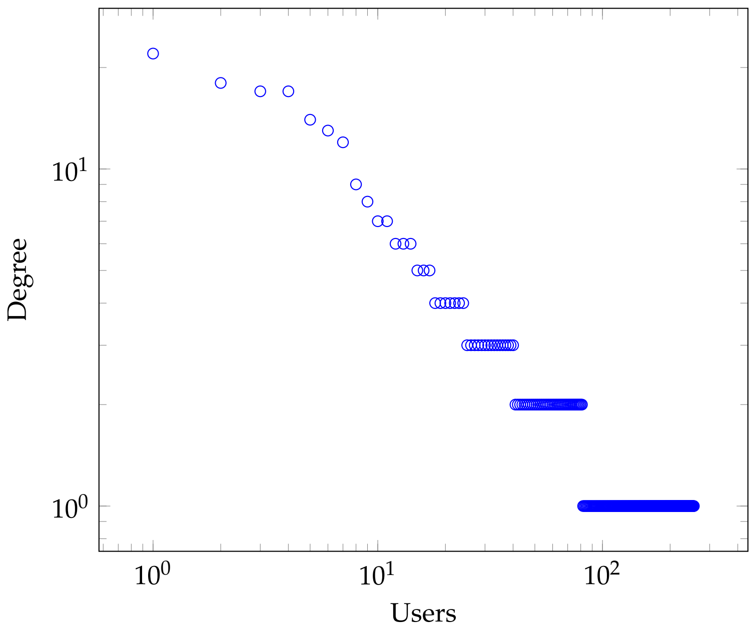

Figure 3. Users' degree distribution of a generated synthetic network.

4.2. Cluster Formation in a Social Network

The first step of our proposed approach is to detect the cluster of each user in the social network as discussed in the previous subsection. For a given graph $G = (V, E)$, there exist partitions

$C = \{c_1, c_2, ...c_k\}$ of V, where $|c_i| > 0$. Our goal is to identify a set of clusters that are densely intra-connected and sparsely interconnected. The process of identifying clusters in social networks is known as community detection.

Various kinds of community detection algorithms have been proposed. In this paper, we use the CC-GA algorithm [18]. The main reason for using CC-GA is its promising results when compared to other state-of-the-art algorithms such as the information theoretic algorithm (Infomap) [38] and the label prorogation algorithm (LPA) [39] on various kinds of networks including power-law and Forest fire models. This algorithm uses a well-known SNA measure called a Clustering Coefficient (CC) for generating an initial population. CC gives a better population for a genetic algorithm and results in better clustering of the network. Additionally, the algorithm using CC has shown that it converges very quickly [18]. In the following equation, C_{v_i} represents the value of clustering coefficient of node v_i:

$$C_{v_i} = \frac{2L_{v_i}}{k_{v_i}(k_{v_i} - 1)}, \tag{5}$$

where L_{v_i} represents the total number of links between the k_{v_i} neighbors. The higher value of C_{v_i} represents the dense neighborhood of a node. The main intuition behind using CC is that if the neighborhood of a node is densely populated, then it is more likely that all of the neighborhood nodes belong to the same cluster. Furthermore, CC-GA uses Modularity measure [40] as an objective function that is a quality index for clustering of social networks. It can be formally defined as follows:

$$Q = \sum_{C \in c} \left[\frac{|E(C)|}{m} - \left(\frac{|E(C)| + \sum_{C' \in c} |E(C, C')|}{|2m|} \right)^2 \right], \tag{6}$$

where $|E(C)|$ represents the number internal links within the cluster while $E(C, C')$ represents the links between clusters. m represents the total number of links in the whole network. Equation (6) can be rewritten into a precise mathematical formulation as given below:

$$Q = \sum_{C \in c} \left[\frac{|E(C)|}{m} - \left(\frac{\sum_{v \in c} deg(v)}{|2m|} \right)^2 \right]. \tag{7}$$

In Equation (7), the first fraction represents the true intra-connectivity of the users, while the second fraction represents the expected number of links within the cluster. The value of modularity ranges between -1 and 1. Greater value of modularity indicates better clustering of the network while a negative value represents an absence of true clustering in the network.

CC-GA tries to maximize the value of modularity in order to get nearly optimal clustering of the network. It is to be noted that modularity optimization is an NP-hard problem, which can not be solved with a guaranteed optimal solution. Algorithm 1 presents the pseudocode of CC-GA. Initially, the algorithm requires a graph consisting of vertices (V) and edges (E) as inputs. It also requires a set of input parameters including: the termination criteria *(r)*, population size (P_n), crossover rate (P_s), mutation rate (P_m), mutation extension rate (α) and percentage of population (T_c) to be selected for the next iteration. Initially, the algorithm computes CC of all nodes (steps 2–4) and generates the initial population (step 5). Afterwards, the genetic operators (crossover and mutation) are performed iteratively until the termination criterion is satisfied (steps 5-25). The algorithm also uses extension of mutation to merge small communities (step 19), and uses modularity measure as an objective function (steps 9, 15, 21). In the end, the algorithm returns the community structure of the input network. For more details about this algorithm, readers are referred to [18]. The interested readers are referred to [41,42] for more about genetic algorithms.

Algorithm 1 CC-GA (G)

```
Inputs:
$P_s$ = Crossover rate;
$P_m$ = Mutation rate;
$P_n$ = Population size;
$\alpha$ = Rate of mutation extension;
$r$ = Termination criteria;
$T_c$ = Selection criteria;
Output:
$C^* = \{c_1, c_2, .....c_k\}$;

1:  procedure CC-GA($G, P_n, T_c, P_s, P_m, r, \alpha$ )
2:      for $i \leftarrow 1, |V|$ do
3:          $C_{v_i} = \frac{2L_{v_i}}{K_{v_i}(K_{v_i} - 1)}$
4:      end for
5:      $P = \{P_1, P_2, P_3......P_n\} \leftarrow$ Initialize population $(g, P_n, C_v)$;
6:      repeat
7:          $O \leftarrow$ Apply Crossover $(P, T_c, P_s,)$;
8:          $Q$ = Compute fitness ($P$);
9:          $Q' \leftarrow$ Compute modularity ($O$);
10:         if $Q' > Q$ then
11:             P $\leftarrow$ update population ($O$);
12:         end if
13:         $O \leftarrow$ Apply mutation ($O$, $P_m$);
14:         $Q$ = Compute fitness ($P$);
15:         $Q'' \leftarrow$ Compute modularity ($O$);
16:         if $Q'' > Q$ then
17:             P $\leftarrow$ update population ($O$);
18:         end if
19:         $O \leftarrow$ Apply mutation extension ($O$, $\alpha$);
20:         $Q$ = Compute fitness ($P$);
21:         $Q''' \leftarrow$ Compute modularity ($O$);
22:         if $Q''' > Q$ then
23:             P $\leftarrow$ update population ($O$);
24:         end if
25:     until (termination-criterion(r))
26:     $C^* \leftarrow$ Chromosome having highest fitness value
27:     Return $C^*$
28: end procedure
```

4.3. Finding Influential Users and Popular Content

After clustering of the social network, our task is to find the Influential Users (IUs) of the network and then find their content. An influential user can be defined as the most connected and active user of the network [43]. In order to find the IUs, we use the notion of centrality [44], which is a well-known concept in SNA. The centrality measure provides a way to quantify the importance of nodes with respect to their positions and neighborhood connectivity within a social network. While various centrality measures exist, eigenvalue centrality is the most successful measure for detecting the social hubs or IUs within a social network. It is a widely used measure for finding the nodes centrality in the networks [16,45,46].

In order to find the IUs, we exploit the EigenVector Centrality measure. This measure uses the notion of eigenvector and eigenvalue of the adjacency matrix. It quantifies centrality of nodes based on centrality of its neighbors. Based on this measure, a node having more central neighbors will have greater centrality value. It can be formally defined as follows:

$$x_{v_i} = \frac{1}{\lambda} \sum_{N \in G} a_{i,j} x_{v_i}, \tag{8}$$

where λ is a normalization constant and N represents neighbors of node v_i. $a_{i,j}$ is 1 if nodes i and j are directly connected to each other, and 0, otherwise. The greatest value of x_{v_i} represents the highest influence of the user in the network. Equation (8) can be rewritten in a vector form as follows:

$$\lambda x = Ax, \tag{9}$$

where A represents the adjacency matrix of the graph, and x is the vector of the eigenvalue scores. Based on eigenvector centrality measure, if our clustering algorithm returns K clusters of the network, then we find K influential users, one for each cluster in our social network. As we are considering D2D communication, it is necessary to identify at least one influential user in each cluster in order to communicate with other users directly. In our social network model shown in Figure 2, different clusters of the network have been shown in different colors after applying our clustering algorithm. There is one central influential user (nodes with greater sizes) found by using an eigenvector centrality measure. For example, user 11 is the influential user of its community represented with the green color, and there are 16 members of its community. Overall, the network is divided into 15 different communities and each community has one influential user. Furthermore, users who join the network earlier have more of a chance to become more connected and are hence more popular and influential (users 0–20).

Once we know the influential users and their communities, our next step is to determine the popular content of each community. For this purpose, inspired by the pervasive use of online social media, we assume that the content requested, generated, or accessed by the influential users is going to be the popular content for the same community, and we proactively cache this content based on the available cache size.

Algorithm 2 shows a pseudocode for computing the centrality values. We initialize an empty matrix M^N for storing the centrality value of each node. Then, the algorithm recomputes the centrality score of each node as a weighted sum for centralities of all nodes in a node neighborhood. In order to normalize the centrality score, λ is used. We set the value of λ to the largest value of x_u. The procedure repeats until the values of each node converge.

After describing all the essentials in the following, we shall now present the proposed algorithm.

Algorithm 2 Compute Centrality

```
1: procedure COMPUTE CENTRALITY($A$)
2:     Initialize matrix $M^N$ by 1
3:     repeat
4:         for $v \in V$ do
5:             $x_u = \frac{1}{\lambda} \sum_{u \in |A|} A_{u,v} x_u$
6:             Update $M_u$
7:         end for
8:     until values of $x_u$ converge
9: end procedure
10: Return $M$
```

4.4. Proposed Algorithm

Algorithm 3 presents our proposed approach for centrality based D2D proactive caching. Initially, we model the system based on the procedure presented in Section 4.1. After designing the model (step 2), we apply the CC-GA algorithm (step 3) for identifying clusters in the network that returns a set C depicting the communities of the network (communities and clustering have the same meaning and we use them interchangeably). Afterwards, we initialize a matrix M (step 4) having the dimensions $(C \times U)$, where C represents the total number of clusters (C_i row contains the centrality values of cluster C_i) and U equals $max(|C_i|)$. This matrix stores the centrality value of each node with respect to its community. Note that C_i has $|C_i|$ values and the remaining indexes which are $max(|C_i|) - (|C_i|)$ will have 0 entries. We also initialize an IU matrix to find and store the influential users. The dimensions of this matrix are $(C \times 2)$. Then, we call the compute centrality algorithm (step 5), presented in Algorithm 2, which returns a vector of centrality values (M') indexes by node labels that are in the form of integers. As we need matrix M for computing the influential users, we therefore compute it from M' (step 6). We compute the primary influential users and secondary influential users from the matrix in each cluster and store them in the matrix (steps 7–10). The reason for finding secondary influential users is to ensure reliability of the system. In the end, we compute the popular content of influential users as described in Section 4.3 and load it into the cache (step 12).

We see that the proposed Algorithm 3 relies on one primary influential user from one cluster, which raises important questions such as how many clusters the network has and how many users each cluster contains. To answer these questions, we present the distribution of community sizes in power-law networks in Figure 4. The figure shows the community sizes distribution of the network shown in Figure 2. We can see that the community size also follows a power law distribution that implies that there are no such communities who have either a large number of users that cover a great part of the network and neither there are such communities who have very few numbers of nodes. Taking this into consideration, we also follow network community size distribution to load the content in the cache of influential users. The user that has a greater community size has more content in the cache and vice versa.

The cellular networks suffer from a network outage problem, which can be in the form of power failure, signal loss or link break down. For maintaining the reliability of the network, we also maintain a list of secondary influential users as shown in Algorithm 3. In case of network outage problems, the secondary influential users will be used for D2D communication.

Algorithm 3 Proactive Caching

```
1:  procedure D2D PROACTIVE CACHING
2:      $G$ = Model the system
3:      $C$ = CC-GA (G)
4:      initialize $M^{C \times U}$ and $IU^{C \times 2}$matrix by 0
5:      $M'$ = Compute Centrality(G)
6:      compute matrix $M$ from $M'$
7:      for $i \leftarrow 1, |C|$ do
8:          $IU_0 = max(M_i)$
9:          remove($max(M_i)$)
10:         $IU_1 = max(M_i)$
11:     end for
12:     Compute the popular content of influential users and load into cache
13: end procedure
```

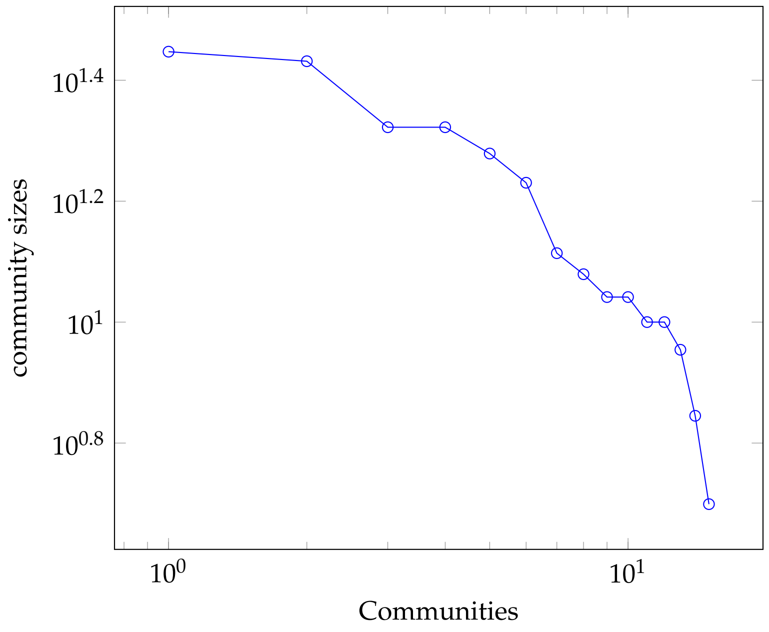

Figure 4. Distribution of community sizes in power law networks.

5. Experimental Setup and Results

In this section, we present simulation setup and experimental results for evaluating the effectiveness of the proposed approach.

5.1. Simulation Setup

Currently, simulations are performed in Python 3. For community detection, CC-GA is implemented using Python's library NetworkX to interact with the system model. Initial parameters for the genetic algorithm are initialized with the values shown in Table 2a. Typically, for network clustering problems, GA parameters like population size, termination criteria and mutation probability are set as 200, 50, and 0.15, respectively [18,47,48]. However, we performed experiments with different parameter settings to select values that perform best in our case. We set population size, termination criteria and mutation probability values as 500, 50 and 0.2, respectively. Note that the higher values of population size and mutation probability may increase the running time of the algorithm but give better results.

Table 2b shows a parameter setup for performing simulations. Two parameters, R and D2D cache size *(D)*, are considered for performance evaluation as we are interested in evaluating backhaul traffic load with respect to cache size and the number of requests. For simulation purposes, we selected values of bitrate and size of each file, link capacity, and total number of requests from well-studied and well-cited papers on proactive caching [9,13]. Moreover, for a total number of files, number of users, and cache size, we selected higher values, in order to evaluate our proposed algorithm in more scalable scenarios. Table 2b presents simulation parameter values.

Table 2. Simulation parameters.

(a) Genetic Algorithm Parameters		
Parameter	**Description**	**Value**
P_n	Population size	500
P_m	Mutation rate	0.2
P_s	Crossover rate	0.2
T_c	Selection criteria	0.1
α	Mutation extension rate	0.02
r	Termination criteria	50
(b) Proposed Approach Parameters		
Parameter	**Description**	**Value**
K	Total of communities	15
N	Users	256
$\|F\|$	Total number of files	1024
l_i	Length of each file	1 Mbits
b_i	Bitrate of each file	1 Mbits/s
C_w	Total small base link capacity	64 Mbits/s
C_b	Total D2D link capacity	128 Mbits/s
R	Maximum number of requests	10,000
D	Total D2D cache size	256 Mbits

5.2. Simulation Results

In order to evaluate the performance of proposed approach, results are compared with a reactive caching approach, in which files are stored in the cache if they were repeatedly requested in the past based on the history of files accessed. Initially, a system model is generated using a preferential attachment model [17], and uniform distribution is considered for user requests' generation. Based on the parameters setup shown in Table 2, the proposed algorithm is evaluated for a different number of requests and cache size.

In Figure 5, we plot satisfaction rate with respect to the number of requests generated by the users and the user's cache size. Our experimental tests are based on number of active users in the network. We considered two main scenarios, first high load scenario when 100% users are active i.e., 256 users and second, and a low load scenario when 50% of the users are active i.e., 128 users. In Figure 5 (left), we measured satisfied requests of the proposed approach based on proactive caching with low and high load against reactive caching with respect to the number of requests. We see that the performance of reactive caching decreases more as compared with proactive caching when the number of requests increases. We see that with 100% requests, the number of satisfied requests is about 48%. We also note that, with 50% of requests, the proposed approach satisfies 52% requests on average with low load and satisfies 45% requests with high load, whereas these are 36% and 31%, respectively, in the case of the reactive approach. In Figure 5 (right), the effect of cache size on satisfaction rate is shown. We can see that with caching 20% of files, 65% of requests are satisfied by the proposed approach while caching 80% files increases the satisfied requests to 100% with low load, whereas these are 30% and 100%, respectively, in the case of reactive caching.

In Figure 6, we plot backhaul traffic load, with respect to the number of requests generated by the users and also with respect to the user's cache size. It can be seen from Figure 6 (left) that, as we increase the number of requests, the backhaul traffic also increases. High load scenarios cause more backhaul traffic to increase as compared to low load scenarios. With the proposed approach, we see that on 100% requests, the backhaul load is 52% on low load and 65% on high load, whereas these are 75% and 79%, respectively, in the reactive approach. The proposed approach thus obtains significant reduction in backhaul traffic load as compared with the reactive caching approach. The effect of cache size on backhaul traffic is shown in Figure 6 (right). We see that with low values of cache size, the backhaul traffic load is significantly lower in the case of the proposed approach as compared to the reactive approach. For instance, with cache size of 10%, the backhaul traffic load is more than 90% in the case of reactive caching, whereas it is half of this in the proposed approach. As we increase cache size of the influential user, backhaul traffic decreases. For a low load scenario, using the proposed approach with 60% of the files cached, backhaul traffic load drops down to less than 10% and, when 80% of the files are cached, there is no backhaul traffic.

Figures 5 and 6 also reveal the effect of increasing the number of users. Specifically from plots in Figures 5 (left) and 6 (left), we see that as the number of active users increases, the number of satisfied requests decreases, leading to an increase in backhaul traffic. On the other hand, reducing the number of active users increases the number of satisfied requests and decreases the backhaul traffic load.

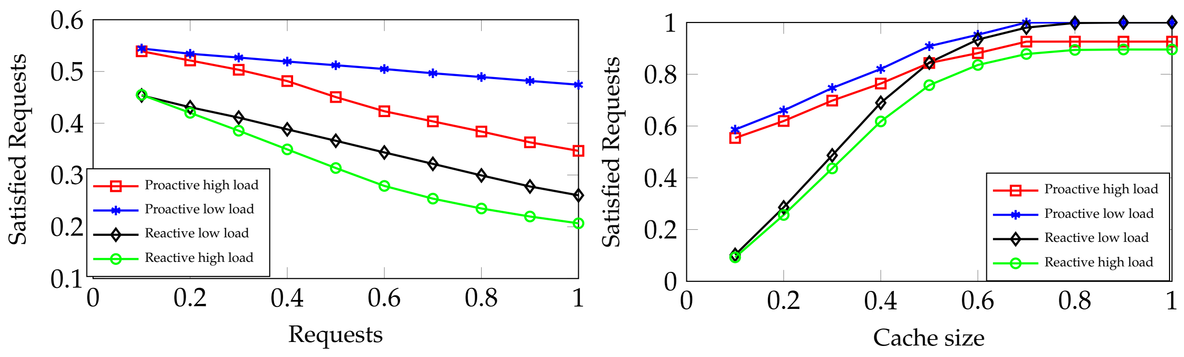

Figure 5. Proactive vs. Reactive caching showing satisfied requests vs. no. of requests (**left**) and cache size (**right**).

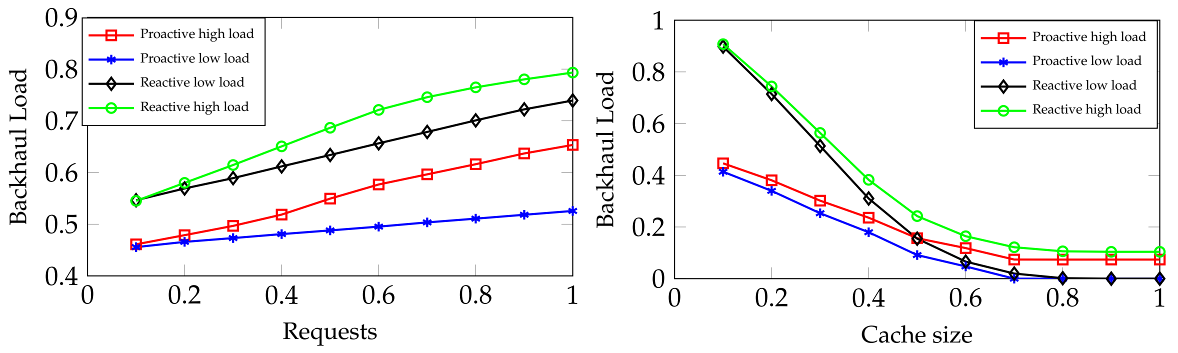

Figure 6. Proactive vs. Reactive caching showing backhaul load vs. no. of requests (**left**) and cache size (**right**).

6. Conclusions and Future Directions

In this paper, we present a proactive caching approach for D2D in small cellular communication networks. We exploit the social and spatial structure of social networks and represent the D2D cellular network as a social graph. In order to find the influential users and clusters formation, we use an eigenvector centrality measure and CC-GA clustering algorithms, and cache the content of influential users of each community. Based on the experimental results, we conclude that caching content of influential users at the user cache can reduce a significant amount of backhaul load and increase the user satisfaction rate, compared to reactive caching or no caching. The key advantage of the proposed scheme is the simplicity of its popular content identification process, which is a key high complexity challenge in both reactive and proactive caching.

This work can be extended in the following directions:

- *Event-driven influential users detection:* When there is a change in the network being detected, it is possible that current influential users may not be available or as influential as they were before the change. In such scenarios, the system should be able to dynamically detect the influential users through re-computing the centrality scores.

- *Joint optimization techniques for D2D enabled caching:* Another interesting future work would be to investigate joint optimization of proactive content caching, scheduling techniques, and interference management.
- *Machine learning for network optimization:* For optimizing D2D caching, effective machine learning approaches can be exploited for clustering the network users and detection of influential users.
- *Mobility aware proactive caching:* Most of the work discussed in the literature focuses on cache placement and delivery strategy while the user is stationary. Caching becomes challenging when a user is moving. For example, when a user with a live streaming session moves from the connectivity of one influential user to another, the user will not be able to stream if the content is not available at the new influential user cache. In such scenarios, deep learning and big data analytics are essential tools that can be leveraged.

Author Contributions: A.S., S.W.H.S. and A.I. discussed and confirmed the idea. A.S. and S.W.H.S. carried out the experiments and wrote the main paper. H.F., A.N.M., A.I. and J.C. provided the technical guidance for proactive caching in D2D small cellular networks with proofreading the paper. All authors provided comments on the manuscript at all stages.

Acknowledgments: The authors would like to thank Saeed Ul Hassan for the insights provided and Tayyaba Liaqat for her help in reviewing the manuscript.

Conflicts of Interest: The authors declare that there is no conflict of interests regarding the publication of this paper.

References

1. OBILE, W. Ericsson Mobility Report. Available online: https://mypresswire.com/log/pm_files/file_31840.pdf (accessed on 15 January 2018).
2. Shen, X. Device-to-device communication in 5G cellular networks. *IEEE Netw.* **2015**, *29*, 2–3. [CrossRef]
3. Larsson, E.G.; Edfors, O.; Tufvesson, F.; Marzetta, T.L. Massive MIMO for next generation wireless systems. *IEEE Commun. Mag.* **2014**, *52*, 186–195. [CrossRef]
4. Shah, S.W.H.; Amin, S.; Iqbal, K. An adaptive algorithm for mu-mimo using spatial channel model. *Intern. J. Eng.* **2016**, *10*, 1.
5. Boccardi, F.; Heath, R.W.; Lozano, A.; Marzetta, T.L.; Popovski, P. Five disruptive technology directions for 5G. *IEEE Commun. Mag.* **2014**, *52*, 74–80. [CrossRef]
6. Andrews, J.G.; Buzzi, S.; Choi, W.; Hanly, S.V.; Lozano, A.; Soong, A.C.; Zhang, J.C. What will 5G be? *IEEE J. Sel. Areas Commun.* **2014**, *32*, 1065–1082. [CrossRef]
7. Imran, A.; Zoha, A. Challenges in 5G: How to empower SON with big data for enabling 5G. *IEEE Netw.* **2014**, *28*, 27–33. [CrossRef]
8. Ban, T.W. A Practical Resource Management Scheme for Cellular Underlaid D2D Networks. *Future Internet* **2017**, *9*, 62. [CrossRef]
9. Bastug, E.; Bennis, M.; Debbah, M. Living on the Edge: The role of proactive caching in 5G wireless networks. *IEEE commun. mag.* **2014**, *52*, 82–89. [CrossRef]
10. Golrezaei, N.; Dimakis, A.G.; Molisch, A.F. Wireless device-to-device communications with distributed caching. In Proceedings of the 2012 IEEE International Symposium on Information Theory Proceedings, Cambridge, MA, USA, 1–6 July 2012; pp. 2781–2785.
11. Bastug, E.; Guénégo, J.L.; Debbah, M. Proactive small cell networks. In Proceedings of the 2013 20th International Conference, Casablanca, Morocco, 6–8 May 2013; pp. 1–5.
12. Gregori, M.; Gómez-Vilardebó, J.; Matamoros, J.; Gündüz, D. Wireless content caching for small cell and D2D networks. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 1222–1234. [CrossRef]
13. Bastug, E.; Bennis, M.; Debbah, M. Social and spatial proactive caching for mobile data offloading. In Proceedings of the 2014 IEEE International Conference on Communications Workshops (ICC), Sydney, NSW, Australia, 10–14 June 2014; pp. 581–586.
14. Wang, S.; Zhang, X.; Zhang, Y.; Wang, L.; Yang, J.; Wang, W. A survey on mobile edge networks: Convergence of computing, caching and communications. *IEEE Access* **2017**, *5*, 6757–6779. [CrossRef]

15. Hoyhta, M.; Apilo, O.; Lasanen, M. Review of Latest Advances in 3GPP Standardization: D2D Communication in 5G Systems and Its Energy Consumption Models. *Future Internet* **2018**, *10*, 3. [CrossRef]
16. Bastug, E.; Hamidouche, K.; Saad, W.; Debbah, M. Centrality-Based Caching for Mobile Wireless Networks. In Proceedings of the 1st KuVS Workshop on Anticipatory Networks, Stuttgart, Germany, 29–30 September 2014.
17. Barabási, A.L.; Albert, R. Emergence of scaling in random networks. *Science* **1999**, *286*, 509–512. [PubMed]
18. Said, A.; Abbasi, R.A.; Maqbool, O.; Daud, A.; Aljohani, N.R. CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks. *Appl. Soft Comput.* **2018**, *63*, 59–70. [CrossRef]
19. Mohamed, A.; Onireti, O.; Imran, M.A.; Imran, A.; Tafazolli, R. Control-data separation architecture for cellular radio access networks: A survey and outlook. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 446–465. [CrossRef]
20. Peng, M.; Li, Y.; Jiang, J.; Li, J.; Wang, C. Heterogeneous cloud radio access networks: A new perspective for enhancing spectral and energy efficiencies. *IEEE Wirel. Commun.* **2014**, *21*, 126–135. [CrossRef]
21. Tehrani, M.N.; Uysal, M.; Yanikomeroglu, H. Device-to-device communication in 5G cellular networks: Challenges, solutions, and future directions. *IEEE Commun. Mag.* **2014**, *52*, 86–92. [CrossRef]
22. Sastry, N.; Crowcroft, J. SpinThrift: Saving energy in viral workloads. In Proceedings of the first ACM SIGCOMM workshop on Green networking, New Delhi, India, 30 August 2010; pp. 69–76.
23. Golrezaei, N.; Molisch, A.F.; Dimakis, A.G.; Caire, G. Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution. *IEEE Commun. Mag.* **2013**, *51*, 142–149. [CrossRef]
24. Chen, B.; Yang, C.; Xiong, Z. Optimal caching and scheduling for cache-enabled D2D communications. *IEEE Commu. Lett.* **2017**, *21*, 1155–1158. [CrossRef]
25. Malak, D.; Al-Shalash, M. Optimal caching for device-to-device content distribution in 5G networks. In Proceedings of the 2014 IEEE Globecom Workshops (GC Wkshps), Austin, TX, USA, 8–12 December 2014; pp. 863–868.
26. Shojafar, M.; Pooranian, Z.; Naranjo, P.G.V.; Baccarelli, E. FLAPS: Bandwidth and delay-efficient distributed data searching in Fog-supported P2P content delivery networks. *J. Supercomput.* **2017**, *73*, 5239–5260. [CrossRef]
27. Chiaraviglio, L.; D'Andreagiovanni, F.; Siderotti, G.; Melazzi, N.B.; Salsano, S. Optimal Design of 5G Superfluid Networks: Problem Formulation and Solutions. In Proceedings of the 21st Conference on Innovation in Clouds, Internet and Networks (ICIN), Paris, France, 20–22 February 2018.
28. Kennington, J.; Olinick, E.; Rajan, D. (Eds.) *Wireless Network Design: Optimization Models and Solution Procedures*; Springer Science and Business Media: New York, NY, USA, 2010; Volume 158.
29. Laurila, J.K.; Gatica-Perez, D.; Aad, I.; Blom, J.; Bornet, O.; Do, T.M.T.; Dousse, O.; Eberle, J.; Miettinen, M. The Mobile Data Challenge: Big Data for Mobile Computing Research. In Proceedings of the 10th International Conference, Pervasive Computing 2012, Newcastle, UK, 18–22 June 2012.
30. Blasco, P.; Gündüz, D. Learning-based optimization of cache content in a small cell base station. In Proceedings of the 2014 IEEE International Conference on Communications (ICC), Sydney, NSW, Australia, 10–14 June 2014; pp. 1897–1903.
31. Shanmugam, K.; Golrezaei, N.; Dimakis, A.G.; Molisch, A.F.; Caire, G. Femtocaching: Wireless content delivery through distributed caching helpers. *IEEE Trans. Inf. Theory* **2013**, *59*, 8402–8413. [CrossRef]
32. Sengupta, A.; Amuru, S.; Tandon, R.; Buehrer, R.M.; Clancy, T.C. Learning distributed caching strategies in small cell networks. In Proceedings of the 2014 11th International Symposium on Wireless Communications Systems (ISWCS), Barcelona, Spain, 26–29 August 2014; pp. 917–921.
33. Pantisano, F.; Bennis, M.; Saad, W.; Debbah, M. In-network caching and content placement in cooperative small cell networks. In Proceedings of the 1st International Conference on 5G for Ubiquitous Connectivity (5GU), Levi, Finland, 26–28 November 2014.
34. Brin, S.; Page, L.; Motwami, R.; Winograd, T. *The PageRank Citation Ranking: Bringing Order to the Web*; Technical Report 1999-0120; Stanford InfoLab: Stanford, CA, USA, 1999.
35. Kleinberg, J. Authoritative sources in a hyperlinked environment. *J. ACM* **1999**, *46*, 604–632. [CrossRef]
36. Lempel, R.; Moran, S. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. *Comput. Netw.* **2000**, *33*, 387–401. [CrossRef]
37. Li, H.; Nakazato, H.; Ahmed, S.H. Request Expectation Index Based Cache Replacement Algorithm for Streaming Content Delivery over ICN. *Future Internet* **2017**, *9*, 83. [CrossRef]

38. Rosvall, M.; Bergstrom, C.T. Maps of random walks on complex networks reveal community structure. *Proc. Natl. Acad. Sci. USA* **2008**, *105*, 1118–1123. [CrossRef] [PubMed]
39. Raghavan, U.N.; Albert, R.; Kumara, S. Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E* **2007**, *76*, 036106. [CrossRef] [PubMed]
40. Newman, M.E.; Girvan, M. Finding and evaluating community structure in networks. *Phys. Rev. E* **2004**, *69*, 026113. [CrossRef] [PubMed]
41. Anderson-Cook, C.M. Practical genetic algorithms. *J. Am. Stat. Assoc.* **2005**, *100*, 1099. [CrossRef]
42. Gen, M.; Cheng, R. *Genetic Algorithms and Engineering Optimization*; John Wiley: Hoboken, NJ, USA, 2010; Volume 8.
43. Sastry, N.; Yoneki, E.; Crowcroft, J. Buzztraq: predicting geographical access patterns of social cascades using social networks. In Proceedings of the Second ACM EuroSys Workshop on Social Network Systems, Nuremberg, Germany, 31 March 2009; pp. 39–45.
44. Borgatti, S.P. Centrality and network flow. *Soc. Netw.* **2005**, *27*, 55–71. [CrossRef]
45. Everett, M.G.; Borgatti, S.P. Extending centrality. In *Models and Methods in Social Network Analysis*; Carrington, P.J., Scott, J., Wasserman, S., Eds.; Cambridge University Press: New York, NY, USA, 2005.
46. Kas, M.; Carley, L.R.; Carley, K.M. Monitoring social centrality for peer-to-peer network protection. *IEEE Commun. Mag.* **2013**, *51*, 155–161. [CrossRef]
47. Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*; MIT Press: Cambridge, UK, 1992.
48. Naeni, L.M.; Berretta, R.; Moscato, P. MA-Net: A reliable memetic algorithm for community detection by modularity optimization. In Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems, Springer, Cham, 10–12 November 2014; pp. 311–323.

© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).