Segment-Based Scheduling Algorithm in Cache-Enabled Device-to-Device Wireless Networks

Shaoqin Peng[1(✉)], Bo Chang[1], Liying Li[2], Guodong Zhao[1], Zhi Chen[1], and Qi Wang[1]

[1] National Key Laboratory of Communication, UESTC, Chengdu, China
psq20110607@163.com, changb3212@163.com, gdngzhao@163.com, chenzhi@uestc.edu.cn, wqforward@163.com
[2] School of Automation Engineering, UESTC, Chengdu, China
liyingli0815@gmail.com

Abstract. In this paper, we study the link scheduling problem in cache-enabled *device-to-device* (D2D) wireless networks considering the *quality-of-service* (QoS) requirement of each scheduled D2D link. We propose a segment-based link scheduling method which consists of two phases, user pairing and link scheduling, to maximize the overall system throughput. We designed a segment factor to control interference among the established D2D links in the link scheduling phase. With the proposed method, interference among different scheduled D2D links can be significantly reduced and the QoS of each link can be guaranteed. Simulation results show that the proposed method outperforms the existing ones in terms of overall system throughput and the number of scheduled D2D links.

Keywords: Scheduling · Cache-enabled · D2D · Wireless networks

1 Introduction

The widely used smart phones and the popularity of tablets have greatly boosted the demand for wireless video services. The videos services will soon occupy most of the wireless data traffic, and lead to a rapid growth in the wireless data traffic [1]. The heavy wireless data traffic has been imposing a significant burden on the current wireless infrastructure. Introducing the caching technique has been a promising solution to solve this problem [2]. Through the caching technique, users can obtain their desired popular files from nearby users via D2D communication links, rather than the *base station* (BS) [3].

Currently, some contributions discuss the link scheduling problem in cache-enabled D2D networks. In [1], the macro-cell is divided into some small clusters

This work was supported in part by the National Natural Science Foundation of China under Grant 61631004 and 61601094.

J. Zheng et al. (Eds.): ADHOCNETS 2018, LNICST 258, pp. 191–201, 2019.
https://doi.org/10.1007/978-3-030-05888-3_18

to reduce intra-cluster interference. However, it ignores the co-channel interference among different clusters. In [4], the authors made some modifications over the conventional *information-theoretic link scheduling* (ITLinQ), called *cached ITLinQ*, where each destination user is connected to the closest source user that caches its requested file. However, the oversimplified user pairing without considering the interference causes performance loss. The *content-centric link scheduling* (CTLinQ) algorithm was proposed in [5] to obtain the maximum number of activated links. In the first phase of CTLinQ, priorities are given to potential D2D links according to their channel gains. This makes the schedule links set settled to a large extent, even though the co-channel interference is considered in the second phase of CTLinQ. However, the latter consideration of interference will degrade the performance of this algorithm in terms of the number of activated links. In [6], the caching and scheduling policies were jointly designed to maximize successful offloading probability. The authors designed a scheduling factor which divided the given duration into some equal-length time slots to control interference among the established D2D links. However, the oversimplified partition limits the performance.

In this paper, we propose a segment-based link scheduling method which consists of two phases for the link scheduling problem in cache-enabled D2D networks. In the first phase, we obtain the set of schedule D2D links according to users' cached file and requested file. In the second phase, we design a segment factor to control interference among the scheduled D2D links. With the proposed method, we can make both the overall system throughput and the number of scheduled D2D links improved. Our results demonstrate the advantages of the proposed method by comparing with the cached ITLinQ method in [4] and the CLTinQ method in [5].

2 System Model and Problem Formulation

2.1 System Model

We consider a cache-enabled D2D wireless network, where a BS with coverage radius R_B is located at the center of a cell. In this paper, we do not consider the mobility of users [7]. We assume that N users distribute uniformly in the coverage of BS. For notational simplicity, we assume that each user has cached a popular file from a file library with M identical size files [8]. Suppose that each user requests one file from the library independently, whose popularity follows Zipf distribution [9]. If we rank M files in terms of popularity in a decreasing order, then the requested probability of the i-th ranked file is given by

$$p_r(i) = \frac{\frac{1}{i^{\gamma_r}}}{\sum_{j=1}^{M} \frac{1}{j^{\gamma_r}}} \quad , \quad i = 1, 2, ..., M, \tag{1}$$

where γ_r is the requested Zipf exponent. Here, the large value of γ_r means that the requests concentrate on the high ranking files.

In the system, a user can obtain its requested file through three ways: self-serve, D2D-Serve or BS-Serve. Here, we mainly discuss the D2D-Serve. We label D2D transmitter (DT) as the user who transmits file, and D2D receiver (DR) as the user who receives file. In our paper, point-to-point link and half-duplex operation modes are assumed [10]. That is, a user can communicate with at most one user, and it cannot transmit or receive files simultaneously. For any DR, only the nearest user who cached its requested file serves as a DT. For any DT, only transmits the file to the nearest user who requests its cached file. In addition, we also assume that all D2D links operate in the same frequency bandwidth, and the cellular links and D2D links operate in orthogonal frequency bandwidth [10].

2.2 Problem Formulation

We define $U = \{u_1, u_2, ..., u_N\}$ as a set of N users, and $V = \{(u_t, u_r) \,|\, u_t \in U, u_r \in U, u_t \neq u_r\}$ as the set of D2D pairs, where (u_t, u_r) is a D2D pair. Here, u_t and u_r represent a DT and a DR respectively. In addition, a D2D pair (u_t, u_r) needs to satisfy the two conditions as follows:

(a) D2D-Serve condition: u_r cannot cache the file requested, it needs to obtain the requested file from other users via D2D link, i.e., $f_c(u_r) \neq f_r(u_r)$, where $f_c(u_r)$ and $f_r(u_r)$ represent the cached file and requested file of u_r, respectively. To obtain the D2D pair (u_t, u_r), u_t needs to cache the requested file of u_r. This means $f_c(u_t) = f_r(u_r)$, where $f_c(u_t)$ represents the cached file of u_t.
(b) D2D range condition: We define d_{u_t,u_r} as the distance between u_t and u_r. In order to establish a D2D communication link, d_{u_t,u_r} should be less than or at least equal to the D2D help distance, i.e., $d_{u_t,u_r} \leq R_{D2D}$, here R_{D2D} is the D2D help distance.

If a D2D pair (u_t, u_r) satisfies the two conditions, it is a potential D2D link. Denote the set of all these potential D2D links as $S = \{(u_t, u_r) \,|\, (u_t, u_r) \in V, (u_t, u_r)$ satisfies conditions (a) (b)$\}$. Then, the set of potential D2D links S is a subset of V.

To indicate the schedule process, we define a binary variable v_{u_t,u_r} as below:

$$v_{u_t,u_r} = \begin{cases} 1, & (u_t, u_r) \in S \textit{ and } (u_t, u_r) \textit{ is scheduled}, \\ 0, & (u_t, u_r) \in S \textit{ but } (u_t, u_r) \textit{ is removed} \\ & \quad \textit{or } (u_t, u_r) \notin S, \end{cases} \tag{2}$$

where $(u_t, u_r) \in V$. Then, the set of schedule D2D links, which is a candidate set for scheduling, can be expressed as

$$L = \{(u_t, u_r) | (u_t, u_r) \in V, v_{u_t,u_r} = 1\}. \tag{3}$$

Then L is a subset of set S. Assume that the number of D2D links in the set L is l, i.e., $|L| = l$.

Since we consider the point-to-point link, the following condition needs to be satisfied,

$$\sum_{u_r \in U, u_r \neq u_t} v_{u_r,u_t} + \sum_{u_r \in U, u_r \neq u_t} v_{u_t,u_r} \leq 1, \forall u_t \in U. \tag{4}$$

In addition, the half-duplex operation mode can be expressed as

$$\sum_{u_r \in U, u_r \neq u_t} v_{u_r,u_t} \sum_{u_r \in U, u_r \neq u_t} v_{u_t,u_r} = 0, \forall u_t \in U. \tag{5}$$

After we obtain the set of schedule D2D links L, the *signal-to-interference-plus-noise ratio* (SINR) of $(u_t, u_r) \in L$ can be expressed as

$$SINR_{u_t,u_r} = \frac{P_t d_{u_t,u_r}^{-\alpha}}{\sum_{u_m \in U_T} P_t d_{u_m,u_r}^{-\alpha} + P_n}, \tag{6}$$

where P_t is the transmission power, P_n is the power of noise, α is the path loss exponent, and $\sum_{u_m \in U_T} P_t d_{u_m,u_r}^{-\alpha}$ is the total interference from all the other DTs (constituting the set U_T). In addition, we assume that the transmission power P_t and the power of noise P_n are known to all users [11], thus the DRs of D2D links can obtain their SINR.

To satisfy the users' QoS, the data rate of D2D link $(u_t, u_r) \in L$ has to be greater than or equal to the data rate threshold R_0, i.e.,

$$R_{u_t,u_r} = W \log_2(1 + SINR_{u_t,u_r}) \geq R_0, \tag{7}$$

where W denotes the bandwidth.

This paper aims at maximizing the total system throughput. We define the overall system throughput as S_T, then the problem can be formulated as

$$\begin{aligned} \max \quad & S_T = \sum_{(u_t,u_r) \in L} W \log_2(1 + SINR_{u_t,u_r}) \\ s.t. \quad & \sum_{u_r \in U, u_r \neq u_t} v_{u_r,u_t} + \sum_{u_r \in U, u_r \neq u_t} v_{u_t,u_r} \leq 1, \forall u_t \in U, \\ & \sum_{u_r \in U, u_r \neq u_t} v_{u_r,u_t} \sum_{u_r \in U, u_r \neq u_t} v_{u_t,u_r} = 0, \forall u_t \in U, \\ & R_{u_t,u_r} \geq R_0, \forall (u_t, u_r) \in L, \\ & v_{u_t,u_r} \in \{0, 1\}, \forall (u_t, u_r) \in V. \end{aligned} \tag{8}$$

Next, we will develop a segment-based link scheduling method to obtain the solution.

3 Segment-Based Scheduling Algorithm

The proposed method consists of two phases: The first phase considers the D2D-serve condition to obtain the schedule D2D links. In the second phase, we design a segment factor to divide a given duration T into two different time slots. We rank the files in library in terms of popularity in a decreasing order. In the meantime, the segment factor also divides the popular files into two segments correspondingly. Thus, the set of schedule D2D links can be divided into two subsets according to the D2D links' transmitting file. The links in two different subsets will be scheduled in different time slots.

3.1 Phase 1: User Pairing

In this phase, we first obtain the set of potential D2D links S. Then, according to (4) and (5), we obtain the set of schedule D2D links. Algorithm 1 shows the details about this user pairing algorithm to obtain the set of schedule D2D links.

Algorithm 1. User Pairing Algorithm.

1: Step 1: The BS collects the users' requested and cached file, i.e., $f_r(u_i)$ and $f_c(u_i)$ $(i = 1, 2, ..., N)$.
2: Step 2: The BS chooses the D2D links which satisfy the conditions (a) and (b), then obtains the set of potential D2D links S.
3: Step 3: For any DR, only the nearest user who cached its requested file can serve as a DT. For any DT, only transmits the file to the nearest user who requests its cached file, the set of schedule D2D links L is obtained finally.

3.2 Phase 2: Link Scheduling

After we obtain the schedule D2D links in the previous step, these links have not been scheduled at this time. Some of them may not meet the requirement on QoS.

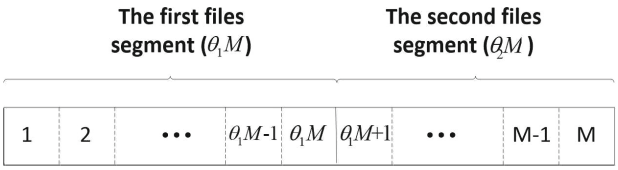

Fig. 1. Illustration of the division of the popular files, where popular files have been ranked in terms of popularity in a decreasing order.

In order to divide the set of schedule D2D links into two subsets, we design a segment factor θ, $\theta \in \{\frac{1}{M}, \frac{2}{M}, ..., 1\}$. This segment factor divides the duration T into two different time slots. The length of the first time slot is $\theta_1 T$, and another is $\theta_2 T$, where $\theta_1 = \theta$, $\theta_2 = 1 - \theta$. After we rank the files in library in terms of popularity in a decreasing order, the segment factor θ also divides the popular files into two segments with length $\theta_1 M$ and $\theta_2 M$ respectively, as shown in Fig. 1. For example, if a file is ranked the top $\theta_1 M$, it will be at the first popular files segment.

Figure 2 illustrates the segment-based link scheduling method, which operates periodically with a given duration T. From the figure, we can observe that k D2D links are scheduled in the first time slot, and another $l - k$ D2D links are scheduled in the second time slot.

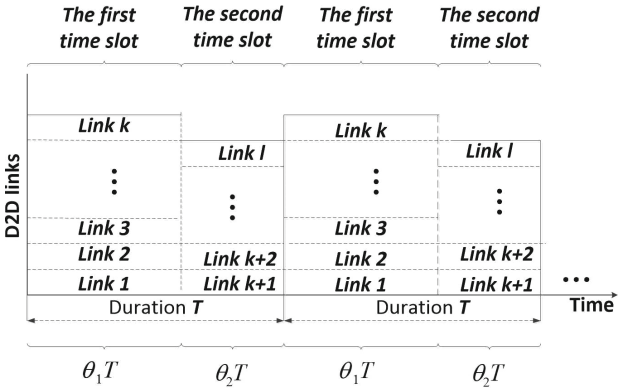

Fig. 2. Illustration of the segment-based link scheduling method for schedule D2D links.

For each established D2D link, there is a file transmitting from the DT to its corresponding DR. Only the D2D link whose transmitting file is in the first files segment can be scheduled in the first time slot, and stays muting in another time slot. While the D2D link whose transmitting file is in the second files segment will be scheduled in the second time slot. That means, the transmitting file of a D2D link decides the time slot that the D2D link will be scheduled. For example, assume that a D2D link $(u_t, u_r) \in L$ transmits the file f_i, $(i = 1, 2, ..., M)$. Then, according to the partition of the popular files, if the file f_i is the top $\theta_1 M$ files, the link (u_t, u_r) will be scheduled in the first time slot, otherwise it will be scheduled in the second time slot.

We assume that the links scheduled in the first time slot belong to the link set L_1, and the others belong to the link set L_2. Both of these sets are subsets of the schedule links set L. The relationships between these sets are given by

$$L = L_1 \cup L_2, \tag{9}$$

and

$$L_1 \cap L_2 = \emptyset. \tag{10}$$

Based on the aforementioned contents, we can rewrite $SINR_{u_t,u_r}$ of $(\mathrm{u}_t,\mathrm{u}_r) \in \mathrm{L}_i$ $(i = 1, 2)$ as

$$SINR_{u_t,u_r} = \frac{P_t d_{u_t,u_r}^{-\alpha}}{\sum\limits_{u_m \in U_{T_i}, i=1,2} P_t d_{u_m,u_r}^{-\alpha} + P_n}, \tag{11}$$

where U_{T_i} is the set of DTs in i-th time slot.

Then, according to [6], the data rate of the D2D link (u_t, u_r) in the i-th time slot is

$$R_{u_t,u_r} = \theta_i W \log_2(1 + SINR_{u_t,u_r}). \tag{12}$$

With the growth of θ, the data rate of the D2D links in the first time slot increases according to (12). Due to the higher ranking files have a higher probability to

be requested, the number of links in the first time slot is more than the second time slot. Thus, in order to obtain the maximum system throughput, we have the following expression as

$$\theta_1 > \theta_2. \tag{13}$$

When θ reaches a certain point, the number of D2D links scheduled simultaneously in the first time slot is too large, leading to more interference. This will cause the data rate of the first time slot to decrease, so the optimal θ needs to be obtained.

Based on the schedule D2D links that we obtained in the phase 1, the problem can be expressed as

$$\begin{aligned} \max \quad & S_T = \sum_{i=1,2} \sum_{(u_t,u_r)\in L_i} \theta_i W \log_2(1 + SINR_{u_t,u_r}) \\ s.t. \quad & R_{u_t,u_r} \geq R_0, \forall (u_t, u_r) \in L, \\ & L{=}L_1 \cup L_2, L_1 \cap L_2{=}\emptyset, \\ & \theta_1 = \theta, \theta_2 = 1 - \theta, \theta \in \{\tfrac{1}{M}, \tfrac{2}{M}, ..., 1\}, \\ & \theta_1 > \theta_2. \end{aligned} \tag{14}$$

The overall system throughput consists of the throughput of the two subsets. As mentioned above, the file library has M files, due to (13), we start the search at the $(\frac{M}{2} + 1)$-th file, i.e.,

$$\theta = \frac{\frac{M}{2} + 1}{M}. \tag{15}$$

Although D2D links are scheduled in different time slots, there are still some links may not meet the requirement on QoS. If this happens, they cannot be scheduled finally and have to be removed. However, instead of removing these links together, we remove them one by one. The link that has the minimum data rate will be removed from the set of schedule links successively, until all scheduled D2D links can meet the QoS requirement. Suppose that there are Q_i D2D links in L_i cannot meet the Qos requirement, then the D2D link $(u_t, u_r)_j \in L_i$ which should be removed according to the following expression,

$$j = \underset{1\leq j\leq Q_i}{\arg} \min\{\mathrm{R}_{(u_t,u_r)_j}\}, \ (u_t, u_r) \in L_i. \tag{16}$$

In this way, the co-channel interference can be reduced. After removed some links, the links that originally cannot be scheduled may finally meet the QoS requirement.

Algorithm 2 provides the details on how to obtain the optimal θ to maximize the system throughput. Specifically, we start searching at a certain θ, under which we calculate the data rate of scheduled D2D links that meet the QoS requirement. In the meantime, we remove the D2D links which cannot satisfy the requirement one by one based on (16), until all D2D links satisfy the QoS requirement. Then, we obtain the overall system throughput and the number of scheduled D2D links. Next, we conduct the search by repeating the above steps. As a result, we can obtain the optimal θ that maximizes the system throughput.

Algorithm 2. Link Scheduling Algorithm.

After user pairing phase: $(u_t, u_r) \in L$
Initialization: R_0, $S_{Temp} = 0$, $N_{Temp} = 0$
For $\theta = (\frac{\frac{M}{2}+1}{M}) : \frac{1}{M} : 1$ **do**
$\theta_1 = \theta$, obtain the links set L_1
While $\exists(\mathrm{u}_t,\mathrm{u}_r) \in \mathrm{L}_1, R_{\mathrm{u}_t,\mathrm{u}_r} < R_0$ **do**
calculate $R_{\mathrm{u}_t,\mathrm{u}_r} = \theta_1 W \log_2(1 + SINR_{u_t,u_r})$
remove $(\mathrm{u}_t,\mathrm{u}_r) \in \mathrm{L}_1$ that has the minimum data rate
end while
$\theta_2 = 1 - \theta$, obtain the links set L_2
While $\exists(\mathrm{u}_t,\mathrm{u}_r) \in \mathrm{L}_2, R_{\mathrm{u}_t,\mathrm{u}_r} < R_0$ **do**
calculate $R_{\mathrm{u}_t,\mathrm{u}_r} = \theta_2 W \log_2(1 + SINR_{u_t,u_r})$
remove $(\mathrm{u}_t,\mathrm{u}_r) \in \mathrm{L}_2$ that has the minimum data rate
end while
calculate the system throughput S_T and the number of scheduled links $N_{Schedule}$
If $S_T > S_{Temp}$ **then**
$S_{Temp} = S_T$
$N_{Temp} = N_{Schedule}$
end if
end for

4 Simulation Results

In the simulation, we assume that each user caches a file according to Zipf distribution with the exponent γ_c. The main simulation parameters are listed in Table 1. For comparison, we provide the simulation results to demonstrate the performance of the cached ITLinQ algorithm in [4] as well as the CTLinQ algorithm in [5].

Table 1. Parameter settings

Parameters	Values
The coverage radius of the BS: R_B	600 m
The D2D help distance: R_{D2D}	100 m
The number of users in the coverage of the BS: N	500
The number of files in the networks: M	1000
The cached Zipf exponent: γ_c	1.2
The requested Zipf exponent: γ_r	0.6
The path loss exponent: α	4
The D2D transmit power: P_t	20 dBm
The noise power density: P_n	-170 dBm\Hz
The bandwidth: W	1 MHz

Figure 3 compares the number of scheduled D2D links versus the data rate threshold. Here, we assume that the number of files is 50, i.e., $M = 50$. From the figure, the number of scheduled D2D links decreases as the data rate threshold increases. This is because if the QoS requirement get stricter, the number of the D2D links which meet the requirement would decrease, then less D2D links can be obtained. We can observe that the performance of the three schemes are better than the CTLinQ and the cached ITLinQ schemes, and the schemes of different segment reach the similar performance. In particular, the complexity of the algorithm will increase as the number of segments grows. Then, it is more efficient to choose the two-segment scheme than the others.

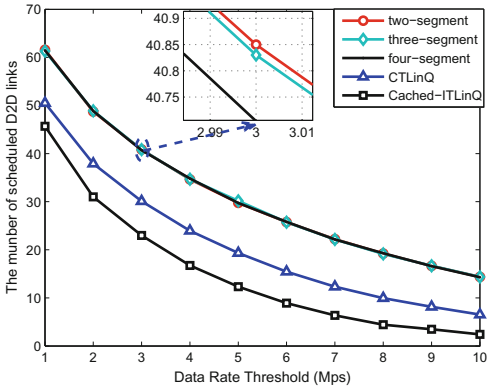

Fig. 3. Performance comparison of different segment schemes in terms of the number of scheduled D2D links, where $R_{D2D} = 100\,\mathrm{m}$, $\gamma_r = 0.6$.

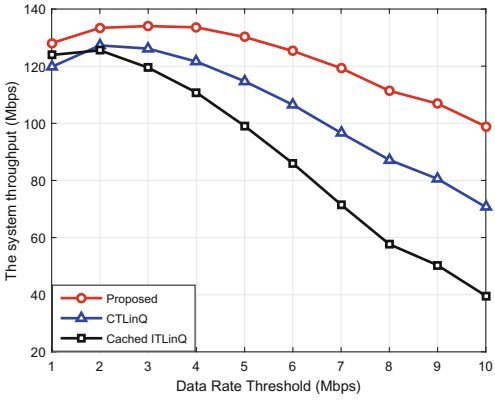

Fig. 4. Performance comparison of different schemes in terms of the overall system throughput, where $R_{D2D} = 100\,\mathrm{m}$, $\gamma_r = 0.6$.

Figure 4 shows the comparison of the overall system throughput between the three methods. From the figure, as the data rate threshold R_0 increases, the system throughput in three methods first increases and then decreases. This is reasonable because as the data rate threshold increasing, there are more D2D links cannot meet the QoS requirement which will be removed in the phase 2. The interference to the existing links will decrease, then the system throughput first increases. As the data rate threshold increasing continually, the number of D2D links which can meet the Qos requirement is decreasing. As the figure shows, while $R_0 \geq 2$, the system throughput starts to drop.

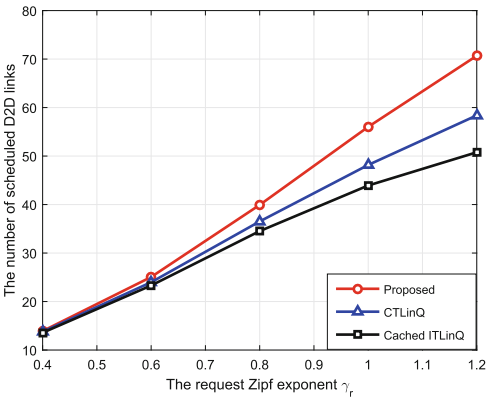

Fig. 5. Performance comparison of different schemes in terms of the number of scheduled D2D links, where $R_{D2D} = 100\,\text{m}$, $R_0 = 1\,\text{Mbps}$.

The impact of the request Zipf exponent is shown in Fig. 5. From the figure, as γ_r increases, the requests are more concentrated on high ranking files, which leads to a higher probability of finding their the requested file in the nearby users' cache. Thus the increasing γ_r raises the number of scheduled D2D links.

5 Conclusions

In this paper, we proposed a segment-based link scheduling method in cache-enabled D2D wireless networks considering the QoS requirement of each D2D link. The first phase obtains the schedule D2D links, and the second phase designs a segment factor to maximize the overall system throughput. Simulation results showed that the performance of the proposed method outperforms the existing cached ITLinQ and CTLinQ methods.

References

1. Golrezaei, N., Mansourifard, P., Molisch, A.F., Dimakis, A.G.: Basestation assisted device-to-device communications for high-throughput wireless video networks. IEEE Trans. Wireless Commun. **13**(7), 3665–3676 (2014)
2. Golrezaei, N., Molisch, A., Dimakis, A., Caire, G.: Femtocaching and device-to-device collaboration: a new architecture for wireless video distribution. IEEE Commun. Mag. **51**(4), 142–149 (2013)
3. Golrezaei, N., Dimakis, A.G., Molisch, A.F.: Wireless device to-device communications with distributed caching. In: Proceedings of Information Theory Proceedings (ISIT), pp. 2781–2785 (2012)
4. Naderializadeh, N., Kao, D.T.H., Avestimehr, A.S.: How to utilize caching to improve spectral efficiency in device-to-device wireless networks. In: Proceedings of Communication, Control, and Computing (Allerton), Monticello, pp. 415–422 (2014)
5. Zhao, G., et al.: CTLinQ: content-centric link scheduling in cache-enabled device-to-device wireless networks. To appear in ICC 2018
6. Chen, B., Yang, C., Xiong, Z.: Optimal caching and scheduling for cache-enabled D2D communications. IEEE Commun. Lett. **21**, 1155–1158 (2017)
7. Akilesh, B., Sathya, V., Ramamurthy, A., Tamma, B.R.: A novel scheduling algorithm to maximize the D2D spatial reuse in LTE networks. In: IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Bangalore, pp. 1–6 (2016)
8. Wang, X., Bao, Y., Liu, X., Niu, Z.: On the design of relay caching in cellular networks for energy efficiency. In: Proceedings of Computer Communications Workshops, pp. 259–264 (2011)
9. Cha, M., Kwak, H., Rodriguez, P., Ahn, Y.Y., Moon, S.: Analyzing the video popularity characteristics of large-scale user generated content systems. IEEE/ACM Trans. Netw. **17**(5), 1357–1370 (2009)
10. Zhang, L., Xiao, M., Wu, G., Li, S.: Efficient scheduling and power allocation for D2D-assisted wireless caching networks. IEEE Trans. Commun. **64**(6), 2438–2452 (2016)
11. Lee, H.S., Lee, J.W.: QoS and channel-aware distributed link scheduling for D2D communication. In: IEEE International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, pp. 1–8 (2016)