# Socially Aware Distributed Caching in Device-to-Device Communication Networks

Chuan Ma$^{\star\dagger}$, Ming Ding$^{\dagger}$, He Chen$^{\star}$, Zihuai Lin$^{\star}$, Guoqiang Mao$^{\dagger\sharp}$, Xu Li$^{\S}$
$^{\star}$ School of Electrical and Information Engineering, University of Sydney, Sydney, NSW, Australia
$^{\dagger}$ Data61, CSIRO, Australia
$^{\sharp}$ School of Computing and Communications, University of Technology, Sydney, NSW, Australia
$^{\S}$ State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing, China
Email: {chuan.ma, he.chen, zihuai.lin}@sydney.edu.au, Ming.Ding@data61.csicro.au, g.mao@ieee.org, xli@bjtu.edu.cn

***Abstract*—Content caching in user devices with device-to-device (D2D) communication capacities is becoming a promising technique to address the data traffic explosion problem in the next generation mobile networks. In this paper, we develop a socially aware distributed caching strategy based on a decentralized learning automaton, referred to as the Discrete Generalized Pursuit Algorithm (DGPA), to optimize the cache placement operation in D2D networks. Different from existing caching schemes, the proposed algorithm not only considers the file request probability and the closeness of devices as measured by their distance, but also takes into account the social relationship between D2D users. Furthermore, we characterize the mutual impact between the contents cached in different D2D users. Simulation results show that the proposed algorithm converges quickly and outperforms its counterparts using deterministic caching and random caching. Our work sheds new insights on the optimal design of D2D cache placement operations.**

## I. INTRODUCTION

With the increasing popularity of tablets and smart phones, mobile data traffic has been increasing dramatically in the past few years. According to Cisco, this unprecedented worldwide growth of mobile data traffic is expected to continue at an annual rate of 45% and exceed 30 exabytes per month by 2020 [1]. Traditional cellular networks alone cannot support such dramatic increase of traffic demands [2]. Distributed data caching, which becomes viable due to the availability of high capacity and low-cost storage devices, has been proposed as an efficient way to offload a significant amount of traffic from cellular networks to other networks [3].

Caching schemes in the literature can be broadly classified into two categories, i.e., small Base Station (BS) caching [4], [5] and device-to-device (D2D) caching [6], [7]. Caching content at small BSs can increase the quality of experience (QoE) of the users and alleviate congestion in the small BSs' backhaul connection, by means of storing data that may be possibly requested by a user at the nearest BS to such user. For example, Marini *et al.* in [4] proposed to use a discrete generalized pursuit algorithm (DGPA) to optimize the cache placement in the small BS caching. Nevertheless, the small BS caching may suffer from long latency and slow update of popular contents.

Compared with the small BS caching, D2D caching provides an alternative solution, where the contents are cached in the storage of D2D users and shared via D2D transmissions. D2D transmissions in cellular networks allow one or multiple pairs of nearby users to communicate directly without going through the BS. D2D users can either use the same bandwidth as the cellular users (underlaying) or use dedicated bandwidth reserved for D2D communication only (overlaying) [8]. Different from small BS caching, in D2D networks, social relationships among users are key factors that encourage successful D2D transmissions [9]. In this case, besides the common factors that have been considered in the small BS caching (e.g., file popularity and physical distance), the social relationship among users should also be taken into account in the design of D2D caching strategy. Therefore, it is interesting and challenging to investigate which users should be selected for content caching that can benefit as many users as possible. Furthermore, compared with small BSs, the storage capacities at users are much smaller. Thus, the optimization of the content placement (i.e., which file should be cached) among the selected users becomes more critical in the design of D2D caching strategies. In [7], a learning automaton was used to solve the content placement problem and optimize the download delay in the D2D underlaying networks. However, different challenges and objectives need to be considered if caching is used in the D2D overlaying networks. Particularly, dedicated bandwidth resources are occupied by the D2D users, and therefore, it is important to maximize the throughput of the D2D links. Moreover, the commonly used assumption that all the D2D links generate the same transmission rate in the D2D underlaying scenario is no longer practical in the D2D overlaying scenario, as the D2D users can communicate with each other with their distances varying from 10 to 1000 meters [10]. As such, the physical distances between transmitter and receiver pairs should to be considered when evaluating their transmission rates.

Motivated by the aforementioned observations, in this paper we develop a socially aware distributed caching framework for the D2D overlay networks. Specifically, the main contributions and novelties of this paper compared with the existing works are described as follows. First, a subset of users, referred to as important users (IUs) are selected to pre-cache files. Then, inspired by the DGPA, we proposed a distributed caching algorithm for the IUs to learn their caching strategies in a decentralized manner. The learning process of each IU decides which files to cache according to its local and aggregate environment feedback. In the proposed algorithm, we design a new and practical feedback scheme by taking into account three key factors: (i) file request probability, (ii) physical distance between D2D transmitters and receivers and (iii) social influence. Moreover, in order to apply the proposed scheme in large-scale networks, we also characterize the mutual impact of the cached files among nearby IUs. Specifically, when a new IU starts to cache, the files which have not been cached by its nearby IUs will be cached with a higher probability, so as to increase the diversity of the cached contents in the network. Finally, simulation results are provided to show that the proposed algorithm not only outperforms its counterparts using deterministic and random caching, but also exhibits a performance gain compared with the algorithm in [7].

## II. SYSTEM MODEL AND PROBLEM STATEMENT

### A. System Model

We consider a content downloading scenario assisted by D2D overlay communications, where dedicated bandwidth are allocated for D2D communications. As such, there is no interference between the cellular and the D2D links. We further assume that there are $N$ users randomly distributed in the network, each of which carries a smart device with D2D communication and multi-radio capability. A D2D link can be established only if the transmission rate of such link is above a predefined threshold $R_t$ and these two users have positive social relationship. Throughout the paper, a user is called a neighbor of another user if there is a positive social relationship between them.

According to [11], in social networks, the distribution of the node degree, i.e., the number of neighbors of a node, decays according to a power law distribution given by

$$p(k) = c_k \times k^{-\omega}, \tag{1}$$

where $\sum_{k=0}^{\infty} c_k k^{-\omega} = 1$, and $p(k)$ is the probability that a randomly chosen node has $k$ neighbors, and $\omega$ is the decaying coefficient. Let $M$ be the number of nodes in a total of $N$ nodes that have at least $k$ neighbors. Using the aforementioned power law degree distribution, $M$ can be approximately calculated as

$$M = \lfloor N \times \sum_{i=k}^{N-1} p(i) \rfloor, \tag{2}$$

where $\lfloor x \rfloor$ is the floor function, retrieving the largest integer that is equal or smaller than $x$. We assume that these $M$ users can download contents directly from BSs and they are regarded as the important users (IUs). Moreover, these $M$ IUs are sorted by their equipment's available storage capacity $C_i$. We denote the sequence of the IUs by the list $\mathcal{C}$ ($\mathcal{C} = \{C_1, C_2, ..., C_M\}$).

The rest of users who request data via the D2D communications are treated as content downloaders, denoted by $\mathcal{D} = \{1, 2, ..., D\}$. Note that, BSs are treated as content providers that host all contents, obtainable via the Internet connection to the content servers. To maximize the traffic offloading for cellular networks, we consider the "D2D-first" strategy, where any downloaders will first ask nearby IUs for help via the D2D links, and then turn to small BSs for downloading if the required files cannot be provided by the IUs.

### B. Problem Formulation

The D2D propagation channel model is characterized as a frequency-flat Rayleigh fading channel. The additive white Gaussian noise (AWGN) at each user is assumed to be i.i.d. and with the same variance $\sigma^2$. Let $\zeta_{m,d}$ represent the distance between the $m$th IU and the $d$th content downloader, and let $\lambda_{m,d}$ denote the path loss exponent of this D2D link.

The nominal transmission rate of this link between the $m$th IU and the $d$th downloader (if the IU $m$ and the downloader $d$ has positive social relationship) can be expressed as :

$$r_{m,d} = BW \times \log_2 \left( 1 + \frac{P_m |h_{m,d}|^2}{\sum\limits_{m' \neq m, m' \in M} P_{m'} |h_{m',d}|^2 + \sigma^2} \right), \tag{3}$$

where $BW$ denotes the bandwidth allocated by the BS, $|h_{m,d}|^2 = (\zeta_{m,d}^{-\lambda_{m,d}}) \times |h_0|^2$ denotes the channel coefficient of the D2D link and $|h_0|^2$ is the multi-path channel gain of the Rayleigh fading. Besides, $P_m$ represents the transmission power of IU $m$.

In order to calculate the system throughput, each nominal D2D transmission rate will settle down as an effective D2D transmission rate on condition that (i) the transmission rate is above the threshold $R_t$ and (ii) the IU has cached the required file of the downloader. Note that, it is possible that more than one adjacent IUs cache the same files and what they cache may influence each other. Let $g_m$ denote the hitting rate of the $m$th IU, which is essentially the probability of the event that a downloader can find the required file from its neighbor IU $m$. Therefore, based on the above definitions, the throughput of the entire D2D transmission system using the dedicated resources can be expressed as:

$$T_{D2D} = \sum_{m \in M, d \in D} \{r_{m,d} \geq R_t\} \times r_{m,d} \times g_m, \tag{4}$$

where $\{\varrho\}$ is the indicator function, which takes the value of 1 if the condition $\varrho$ is satisfied. .

As can be observed from (4), the system throughout can be improved by increasing either the hitting rate of each IU $g_m$ or the transmission rate $r_{m,d}$. The resource allocation problem has been well studied to satisfy the requirement of transmission rate, such as the algorithms proposed in [8]. In this paper, we focus on the cache placement optimization at the IUs to improve the hitting rate, which in turn improves the system throughput.

## III. DISTRIBUTED AND SOCIALLY AWARE STRATEGY FOR CACHING

In this section, we propose a decentralized learning automaton, which helps each IU to optimize its cache placement according to its local demands. The proposed algorithm is inspired by the DGPA. In the following, we first provide some preliminaries of the DGPA before formally presenting the proposed algorithm. Then, we design a scheme to characterize the mutual impact of cache placement between nearby IUs, which enables the proposed algorithm to be applied in large-scale networks.

### A. Discrete Generalized Pursuit Algorithm

The goal of the DGPA is to determine an optimal action out of a set of allowable actions $\mathcal{F} = [1, 2, ..., F]$. The DGPA has a probability vector $\mathbf{P}(t) = [p_1(t), p_2(t), ..., p_F(t)]$, where $p_i(t)$ is the probability that the automaton will select the action $i$ at iteration $t$ with $\sum_{i=1}^{F} p_i(t) = 1$. This learning algorithm will converge when any action $i$ achieves the probability of one, i.e., $p_i = 1$. The updating of the probability vector is performed based on the reward estimation $\mathbf{d}(t) = [d_1(t), d_2(t), ..., d_F(t)]$ and each reward estimation is determined by the environment feedback [12]. In the considered D2D caching system, at each learning process, an action of each IU is to choose one popular file from the file library to cache. This action is taken considering the file request probability. A certain action will get a positive reward from the aggregate environment feedback if it is beneficial to the system.

The DGPA generalizes the concepts of the pursuit algorithm by "pursuing" all the actions that have higher reward estimates than the current chosen action. In this algorithm, the action probability vector $\mathbf{P}(t)$ is recursively updated by the following equation:

$$\mathbf{P}(t+1) = \mathbf{P}(t) + \frac{\Delta}{K(t)} \times \mathbf{e}(t) - \frac{\Delta}{F - K(t)} \times [\mathbf{u} - \mathbf{e}(t)], \quad (5)$$

where $\mathbf{u}$ is a vector in which $u_i = 1, i = 1, 2, ..., F$, and $\mathbf{e}$ is a direction vector given by:

$$e_i(t) = \begin{cases} 1, & \text{if } d_i(t) = \max\{d_j(t)\},\ j \in 1, ...F; \\ 0, & \text{otherwise.} \end{cases}$$
$$e_j(t) = \begin{cases} 0, & \text{if } d_j(t) \le d_i(t); \\ 1, & \text{if } d_j(t) > d_i(t). \end{cases} \quad (6)$$

According to (5), the probabilities of the chosen action $i$ and other action $j$ are updated as follows:

$$\begin{cases} p_j(t+1) = \min\{p_j(t) + \frac{\Delta}{K(t)}, 1\}, \text{if } d_j(t) > d_i(t); \\ p_j(t+1) = \max\{p_j(t) - \frac{\Delta}{F-K(t)}, 0\}, \text{if } d_j(t) < d_i(t); \\ p_i(t+1) = 1 - \sum_{j \neq i} p_j(t+1). \end{cases} \quad (7)$$

At each iteration of the DGPA, the number of actions which has a higher reward estimation $d(t)$ than the current chosen one is counted, denoted by $K(t)$. At the end of iteration, the probability of all actions with a higher reward estimation $d(t)$ will increase by an amount of $\Delta/K(t)$, and the probability of all the other actions except the chosen one will decrease by an amount of $\Delta/(F - K(t))$, where $F$ is the action library size. Besides, $\Delta = 1/F\delta$ and it is a resolution step and $\delta$ is the resolution parameter.

In order to update the probability of each action, the reward estimation $\mathbf{d}(t)$ should be estimated at first. The updating equations of reward estimation $\mathbf{d}(t)$ for the chosen action $i$ are given as follows:

$$\begin{cases} Z_i(t+1) = Z_i(t) + 1; \\ W_i(t+1) = W_i(t) + \beta(t); \\ d_i(t+1) = \frac{W_i(t+1)}{Z_i(t+1)}, \end{cases} \quad (8)$$

where $Z_i(t)$ represents the number of times that action $i$ has been chosen, and $W_i(t)$ represents the number of times that action $i$ has been rewarded. $\beta(t) \in \{0, 1\}$ is a binary factor reflecting a positive or negative feedback. If the feedback is positive (i.e., $\beta = 1$), then this action $i$ is rewarded.

In the next subsection, based on the preliminaries of the DGPA, we will design the functions of the aggregate environment feedback in the proposed socially aware D2D networks.

### B. Aggregate Environment Feedback

In our model, we assume that BSs can get the position of every IU and every downloader, thus BSs can provide each IU with its relevant downloaders' information, such as the file request probability, and each IU can broadcast the list of cached file to its neighboring downloaders. Thus, different cached files (actions) at a certain IU would have different impacts on its neighbors and other IUs.

In the process of learning, when the $m$th IU caches the file $f$ according to its downloader neighbor $d$'s request, we define the aggregate environment reward $R_{m,d}^f$ as a weighted sum of the request probability ($p_d^f$) of file $f$, the physical distance influence ($\frac{1}{1+\zeta_{m,d}^{\lambda}}$) between the IU $m$ and its neighbor $d$, and the social influence ($s_{m,d}$) between them, which is intuitively defined as:

$$R_{m,d}^f = \tau \times p_d^f + \theta \times h_{m,d} + \eta \times s_{m,d}, \quad (9)$$

where $\tau$, $\theta$ and $\eta$ are tunable parameters and they satisfy $\tau + \theta + \eta = 1$. We provide detailed explanation for each term in (9) as follows.

*1) The request probability $p_d^f$:* We use the $Zipf$ distribution, which has been commonly used in the literature (e.g., [6]), to model the file request probability. Specifically, for the $f$th file, its file request probability $p_f$ for the downloader $d$ is written as

$$p_d^f = \frac{\frac{1}{f^{\gamma}}}{\sum_{i=1}^{F_d} \frac{1}{i^{\gamma}}}, \quad (10)$$

where $F_d$ is the file library size of downloader $d$ and $\gamma$ is the discounted rate in the $Zipf$ distribution. Note that, each downloader has different file library size $F$ and different discounted rate $\gamma$, where $\gamma \in [0.4, 1]$ [6].

*2) The physical distance influence $h_{m,d}$:* Intuitively speaking, there will be little influence if the physical distance between IU $m$ and downloader $d$ is large. In this case, the physical distance influence can be model as

$$h_{m,d} = \frac{1}{1+\zeta_{m,d}^{\lambda}}, \tag{11}$$

where $\zeta_{m,d}$ represents the distance between IU $m$ and downloader $d$ and $\lambda$ is the path loss exponent.

*3) The social influence $s_{m,d}$:* The degree of similarity among users has an important impact in information dissemination [13]. For example, when the degree of similarity between two users is lower, more time would be required for transmitting information of the same length. As a result, we use the degree of similarity to determine the social influence $s_{m,d}$.

The degree of similarity can be measured by the ratio of common neighbors between individuals. According to [13], we suppose that IU $m$ is connected with downloader $d$. Let $V(m)$, $V(d)$ denote the set of neighbors of user $m$ and $d$, respectively. Let $z$ be one of the common neighbors of them and let $X(z)$ denote the number of $z's$ neighbor, including $m$ and $d$. We then can define the similarity between IU $m$ and its neighbor $d$ as [13]:

$$q_{m,d} = \sum_{z\in V(m)\cap V(d)} \frac{1}{X(z)}. \tag{12}$$

If $m$ and $d$ have no common neighbors, such that $V(z) = 0$, then $q_{m,d} = 0$. In order to make the three factors of the environment feedback comparable, we normalize the similarity $s_{m,d}$ as follows:

$$s_{m,d} = \frac{q_{m,d}}{\sum\limits_{m\in M} q_{m,d}}. \tag{13}$$

Now, we are ready to calculate the environment feedback using the reward functions. We first denote the neighbor set of IU $m$ by $N_m$. Then IU $m$ will choose a file $f$ to cache according to its request probability, and its neighbors will also ask a file to download according to their own file request probabilities. If IU $m$ and one of its neighbor $d$ choose the same file, such as the file $f$, we can define this action as a positive one, which brings a positive reward. If not, this action will be determined as a negative action. Mathematically, the reward functions are defined as:

$$\begin{cases} \Psi_P = R_{m,d}^{f}, \text{ if } m \text{ and } d \text{ choose the same file;} \\ \Psi_N = -R_{m,d}^{f}, \text{ if } m \text{ and } d \text{ choose different files.} \end{cases} \tag{14}$$

Thus, for IU $m$, the aggregate environment feedback function of choosing file $f$ can be expressed as :

$$\digamma_m^f = \sum_{d=1}^{N_m} (\Psi_P + \Psi_N). \tag{15}$$

If $\digamma_m^f > 0$, then $\beta = 1$ and this action that IU $m$ cache file $f$ can get a positive feedback from the environment. In this case the estimation vector $\mathbf{d}(t)$ can be updated.

IU $m$ will keep learning and get the optimal request files from the BS according to the aggregate environment feedback until its available storage is full.

### *C. The Mutual Impact of Nearby IUs*

The decision of content placement for IU $m$ will impact its nearby IUs, which have common neighbors with IU $m$. If there are two IUs, the content placement of these two IUs should be made different as much as possible to serve different requests of their common neighbors. In this case, the BSs should update the file request probability of the common neighbors according to the former IUs who have already cached contents.

IUs start learning in the order determined by the list $\mathcal{C}$. To update the file request probability of the common neighbors, all the IUs should report the cached files to the BS after learning. This updating information can be considered as a combined information of the cached files and the physical distance. For example, if two previous IUs $m$ and $m'$ have already cached files $f$ and $f'$, respectively, then for the next IU $n$, it should first estimate which IU has a larger physical influence (a shorter distance) to IU $n$. If IU $m$ has a larger physical influence than IU $m'$, i.e., $\zeta_{n,m}^{-\lambda} > \zeta_{n,m'}^{-\lambda}$, then the file $f$ cached by IU $m$ should be considered when updating the file request probability of their common neighbors. Let $N_{m,n}$ denote the set of the common neighbors of IU $m$ and IU $n$, then the request probability ($Y_f^{N_{m,n}}$) of file $f$ for the common neighbors ($N_{m,n}$) can be updated as

$$Y_f^{N_{m,n}} = Y_f^{N_{m,n}} \cdot \frac{1}{1+\zeta_{n,m}^{\lambda}}, \tag{16}$$

where $\zeta_{n,m}$ represents the physical distance between the $n$th IU and the $m$th IU, and $\lambda$ is the path loss exponent.

After updating the probability of every cached files by IU $m$, the file probabilities $Y^{N_{m,n}}$ of the common neighbors between IU $m$ and IU $n$ will be normalized, and IU $n$ can start its learning process.

### *D. Convergence*

Due to the paper limitation, the proof of the convergence will be relegated to the journal version of this paper.

## IV. PERFORMANCE EVALUATION

In this section, numerical and simulation results of the proposed scheme are presented for various scenarios.

### *A. Simulation Scenario*

A wireless network consisting of one omnidirectional BS and a number of D2D users is considered. The D2D users are randomly distributed in an area of $5 \times 5km^2$. The social similarities among users are randomly generated according to the power law distribution with a parameter $\kappa = 2.42$ [11] and the file request probability follows the $Zipf$ distribution with a discounted rate $\gamma = 0.5$. Note that the physical layer parameters in our simulations, such as the path-loss exponent, noise power and transmit power of the IUs and the BS, are chosen to be practical and in line with the values set by 3GPP

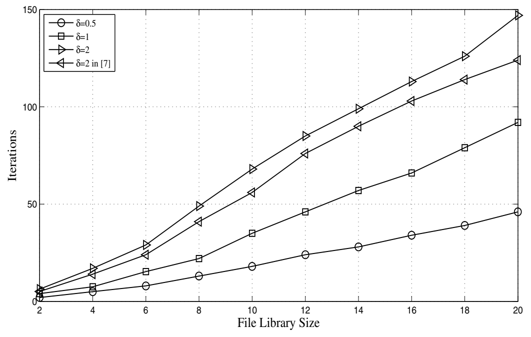

Fig. 1. The average convergence iterations for various resolution parameters $\delta$

standards. For instance, the transmission power of IUs is 25 dBm. Unless specified otherwise, we set the path loss exponent $\lambda = 3$, and the noise to $\sigma^2 = -95$ dBm. All the simulations are executed using MATLAB.

## B. Convergence and Reward Probability

We first verify the convergence of the proposed algorithm. A small-scale mobile network is considered, which consists 3 IUs and each of them has 6 neighbor downloaders. Each IU chooses one file to cache from a file library of at most 20 files. The algorithm is considered to converge when the probability of taking one action (caching one file) is larger than $0.999$ and the number of the required iterations has been recorded.

Fig. 1 shows the average convergence iterations for various resolution parameters, i.e., $\delta$. As shown in Fig. 1, the average number of iterations of 3 IUs are recorded in different file library sizes. We consider various resolution parameters $\delta$ to verify the convergence. As the file library size grows, the average number of iterations for the 3 IUs increase. For example, when the resolution parameter $\delta = 1$, the average iteration number for 20 files is about $2.6$ times of that number for 10 files, which means that if there are more requested files, more iterations will be needed to finish the learning. Moreover, different resolution parameters $\delta$ show different increasing trends, and require different numbers of iterations to converge. For example, when $\delta = 0.5$, the average number of iterations is nearly half ($49.2\%$) of that when $\delta = 1$. We also compare the proposed algorithm with [7] when $\delta = 2$. As can be observed from Fig. 1, the algorithm presented in [7] requires fewer iterations on average compared with the proposed algorithm in this paper. This is because the algorithm in [7] was based on a simple environment feedback functions, in which the physical distance influences were not considered.

As can be observed from Table II, after each learning process, each IU will cache one popular file from 10 files. Then we collect the number of times that each IU has been rewarded with different resolution parameters. Thus the average reward probability can be calculated for each file library size. Here, the reward probability represents the probability that the learning result can get a positive feedback from each learning process. Fig. 2 depicts the average reward probability of different resolution parameters $\delta$. It is shown that with the increasing size of the file library, the average reward probability decreases. Also it can be observed from this figure that the proposed algorithm can get a higher reward probability than [7]. Considering both Fig. 1 and Fig. 2, although a larger resolution parameter $\delta$ implies more time to converge, it can achieve a higher reward probability. For example, when $\delta = 1$, its reward probability is about $1.22$ times of that when $\delta = 0.5$, while it needs $3 \sim 7$ more iterations to converge. Moreover, for the proposed algorithm, although it takes more time to converge compared with [7] with the same $\delta$, the reward probability is much better. Finally, it can be observed that the proposed algorithm strikes a more beneficial balance between performance and complexity compared with the algorithm presented in [7]. This is because the proposed algorithm requires less iterations than the algorithm in [7] to achieve a similar reward probability performance. For example, the proposed algorithm only needs about 26 iterations to converge while the algorithm in [7] needs about 52 iterations to get a similar reward probability.

TABLE I
REWARD TIMES AFTER REPEATING 50 TIMES LEARNING WHEN THE FILE LIBRARY SIZE IS 10

| | IU1 | IU2 | IU3 | Average reward probability |
|---|---|---|---|---|
| $\delta = 0.5$ | 17 | 19 | 17 | 0.362 |
| $\delta = 1$ | 23 | 22 | 22 | 0.446 |
| $\delta = 2$ | 26 | 27 | 26 | 0.514 |
| $\delta = 2$ in [7] | 23 | 23 | 22 | 0.453 |

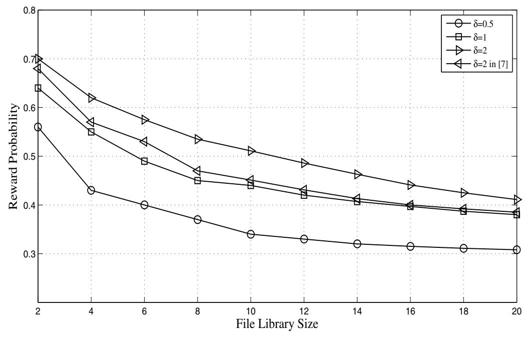

Fig. 2. The average reward probability for various resolution parameters $\delta$

## C. Hitting Rate and System Throughput

In this subsection, we compare our proposed algorithm with [7], as well as determined caching, random caching and the optimal caching algorithms. Here, determined caching means caching the most popular files, which follows the $Zipf$ distribution. Random caching is to cache files randomly from the file library following the probability $p = \frac{1}{F}$, where $F$ is the file library size. The optimal caching is performed by the noncausal algorithm, in which we assume the IUs have the whole knowledge of the network, so they can make the optimal choices. Same as previous subsection, we pose a constraint

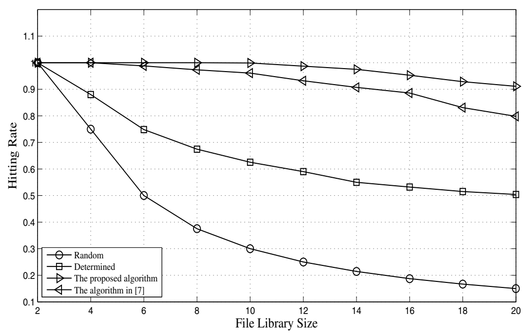

Fig. 3. The hitting rate of the considered caching strategies

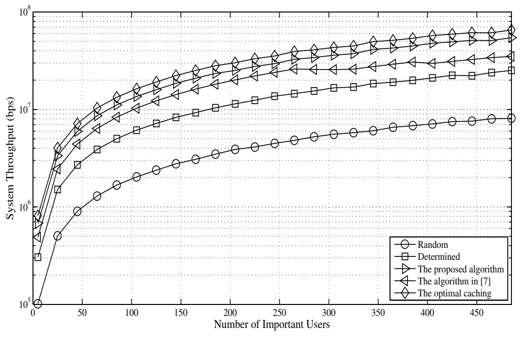

Fig. 4. The system throughput of the considered caching strategies

that each IU can store at most three files. Fig. 3 shows the hitting rate of different caching strategies with different file library size. The simulation environment is similar as the previous scenario and $\delta = 0.5$. From the simulation results, we can observe that the performance gaps between our proposed algorithm with other three strategies are enlarged with the increase of the file library size. Our proposed strategy performs better than the other three benchmark strategies. When there are more files need to be considered, it could bring more flexibility and performance improvement for managing the caching contents.

Fig. 4 shows the simulation results of the system throughput with the file library size set to $10$. From this figure, we can see that the system throughput increases with the increasing number of IUs $M$, and our proposed algorithm always performs better than the other three investigated strategies except the optimal one. For example, in comparison with the counterparts using determined caching and random caching, the system throughput of the proposed algorithm is increased by $1.24$ and $5.27$ times, respectively, when $M = 400$. Also, the gap between the proposed algorithm and the algorithm in [7] grows as the number of IUs increases. As shown in this figure, the algorithm in [7] is able to achieve almost the same system performance compared with the proposed algorithm when the number of IUs is small. However, it is not the case for a large number of IUs. This is because the algorithm in [7] no mutual impact is considered, thus, nearby IUs may cache similar contents, and cannot provide downloading service for other popular contents. In contrast, the proposed algorithm encourages nearby IUs to cache different content in order to achieve caching diversity. Moreover, the gap between the proposed algorithm and the optimal one is not big and narrowing slowly with the increasing number of IUs. Thus, our caching strategy is effective to increase the hitting rate, which in turn largely boosts the system throughput.

## V. CONCLUSION

In this paper, we developed a distributed and socially aware framework based on a decentralized learning automaton to solve the optimum cache placement problem in D2D overlaying networks. In the process of learning, in order to solve the problem of unsuccessful transmission in D2D communications, we combined our algorithm with the aggregate environment feedback. Also the mutual user impacts were considered in this scheme to enable its application in the large-scale networks. Simulation results showed that our algorithm has fast convergence speed and can achieve considerable system throughput gains when compared with the existing caching strategies.

## REFERENCES

[1] Cisco, "Visual networking index forecast, 2015–2020," 2015.

[2] C. Xu, L. Song, Z. Han, Q. Zhao, X. Wang, X. Cheng, and B. Jiao, "Efficiency resource allocation for device-to-device underlay communication systems: A reverse iterative combinatorial auction based approach," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 348–358, 2013.

[3] U. Niesen, D. Shah, and G. W. Wornell, "Caching in wireless networks," *IEEE Transactions on Information Theory*, vol. 58, no. 10, pp. 6524–6540, 2012.

[4] L. Marini, J. Li, and Y. Li, "Distributed caching based on decentralized learning automata," in *IEEE International Conference on Communications (ICC)*. IEEE, 2015, pp. 3807–3812.

[5] B. Bharath, K. Nagananda, and H. V. Poor, "A learning-based approach to caching in heterogenous small cell networks," *IEEE Transactions on Communications*, vol. PP, no. 99, pp. 1–1, 2016.

[6] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless d2d networks," *IEEE Transactions on Information Theory*, vol. 62, no. 2, pp. 849–869, Feb 2016.

[7] C. Ma, Z. Lin, L. Marini, J. Li, and B. Vucetic, "Learning automaton based distribued caching for mobile social networks," in *IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2016.

[8] A. Asadi, Q. Wang, and V. Mancuso, "A survey on device-to-device communication in cellular networks," *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 4, pp. 1801–1819, 2014.

[9] B. Zhang, Y. Li, D. Jin, P. Hui, and Z. Han, "Social-aware peer discovery for d2d communications underlaying cellular networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 5, pp. 2426–2439, May 2015.

[10] D. Feng, L. Lu, Y. Wu-Yuan, Y. G. Li, S. Li, and G. Feng, "Devive-to-device communications in cellular networks," *Communications Magazine, IEEE*, vol. 52, no. 4, pp. 49–55, 2014.

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

[12] M. Agache and B. J. Oommen, "Generalized pursuit learning schemes: new families of continuous and discretized learning automata," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 32, no. 6, pp. 738–749, 2002.

[13] T. Zhou, L. Lü, and Y.-C. Zhang, "Predicting missing links via local information," *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 71, no. 4, pp. 623–630, 2009.