Two-Tier Matching Game Design for Wireless Caching in Pico-Cell Networks

Guowei Shi[1], Jun Li[1(✉)], Haijun Zhang[2], Feng Shu[1], and Tingting Liu[1]

[1] School of Electronic and Optical Engineering,
Nanjing University of Science and Technology, Nanjing, China
{guowei.shi,jun.li,shufeng}@njust.edu.cn, liutingting1026@hotmail.com
[2] School of Computer and Communication Engineering,
University of Science and Technology Beijing, Beijing, China
zhanghaijun@ustb.edu.cn

Abstract. Wireless caching brings network content close to mobile users (MU), and has been identified as an effective solution for reducing MUs' transmission delay. In this paper, we concentrate on how to efficiently allocate network files to the storage of pico-cells and associate MUs with the pico-cells. To deal with the series of resource allocation problems, we first separate these problems as two distinct many-to-one matching games. Then, we tackle these problems by proposing different concepts to generate the preference lists respectively. The deferred-acceptance algorithm is designed in this paper to achieve stable matchings in these two separated games. It is shown in numerical results that our proposed design demonstrates a better performance compared to state-of-the-art benchmarks.

Keywords: Matching game · Wireless caching
Content-oriented communications · Pico-cell networks

1 Introduction

Along with the developments of mobile communication technologies, mobile users (MUs) show growing interests in using various multimedia services [1]. To deal with the explosive growth of demands in high speed data transmissions by MUs, an low-cost and low-power pico-cell access architecture is proposed which can coexist with any wireless technology and can be deployed in any area underlaid on Macro-cell network [2]. But when pico-cell network density increases, backhaul capacity limitations would be a big problem [3].

Content caching is one of the efficient solutions to effectively handling this problem. With observations that a large amount of data traffic is caused by a small portion of popular network contents, i.e., movies, as well as the price of storage medium is relatively cheaper compared to the price of backhaul,

K. Long et al. (Eds.): 5GWN 2017, LNICST 211, pp. 663–674, 2018.
https://doi.org/10.1007/978-3-319-72823-0_61

caching in network facilities, such as small-cell, femto-cell, pico-cell and D2D nodes, to release backhaul pressure becomes a potential solution in these years [4–6].

Specially in [4], a fetmo-cell caching system is proposed for assigning files to the femto-cells, in order to minimize the downloading time. Moreover, the authors of [5] proposed a caching system for D2D based cellular network relaying on the MUs' caching of popular files, in order to increase the throughput of networks. In addition, the authors of [6] have proposed a distributed caching scheme using Stackelberg game, in order to deal with resource allocation problems in a small cell network. Although many problems have been solved as above, how to effectively allocate resource in pico-cell networks still remains a great challenge to face.

In this paper, we leverage matching game theory to handle resource allocation problems in cache enabled system. As in [7], authors introduce fundamentals and conventional classification of matching theory for future wireless networks, where matching games are split into three kinds, i.e., one-to-one matching games, many-to-one matching games and many-to-many matching games. In [8], authors utilize many-to-one matching games in wireless small cell networks with a combination of context-aware of information about trajectory profile and quality of service requirements of users, in order to maximum satisfaction ratio and reduce downloading delay. Many-to-many matching games has been used in [9] to reduce backhaul loads and the experienced delay in small cell networks. From discussion above, we are motivated to use matching game theory in pico-cell networks to explore the optimal allocation with caching.

In this paper, we concentrate on proposing a two-tier matching game in wireless caching systems to handle the resource allocation problem. We firstly match network contents with pico-cells and then match the pico-cells with mobile users. The key contributions can be summarized as follows.

1. We model a pico-cell based caching system with the objective to minimize the system transmission delay.
2. We decouple the resource allocation problems into two many-to-one matching games, and address them by using the deferred-acceptance algorithm. We then generate preference lists in the two matching games to obtain a stable matching.
3. We demonstrate that the proposed algorithm has near-optimal outcomes with numerical results.

The rest of this paper is organized as follows. We describe the system model and formulate system objective in Sect. 2. We then decouple the optimization problems and propose matching games to solve the optimization problems in Sect. 3. Our numerical results are illustrated in Sect. 4, while our conclusions are draw in Sect. 5.

2 System Model and Problem Formulation

Let us consider a pico-cell based caching network composed of M MUs which have priorities to connect to pico-cells. As shown in Fig. 1, there exists a Macro-cell serving MUs in its coverage assisted by N pico-cells. Also, we define the set of M MUs by $\boldsymbol{\mathcal{M}} = \{\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_M\}$ and the set of pico-cell by $\boldsymbol{\mathcal{N}} = \{\mathcal{N}_1, \mathcal{N}_2, \cdots, \mathcal{N}_N\}$ which are both randomly located in system. We assume that each pico-cell has the same storage size that could cache only one file.

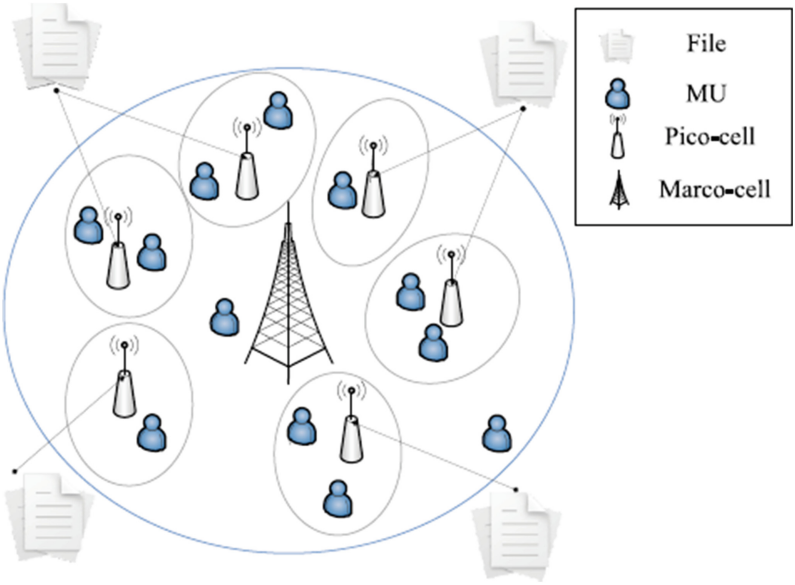

Fig. 1. Pico-cells based caching system model

There are two resource allocation problems in this network. In the first stage, files are allocated to pico-cells and then MUs are associated to pico-cells in second stage. In what follows, we will introduce some key concepts to facilitate the problem formulation.

2.1 File Popularity

Firstly, we model the popularity of files. Let us denote the file set by $\boldsymbol{\mathcal{V}} = \{\mathcal{V}_1, \mathcal{V}_2, \cdots, \mathcal{V}_V\}$ consisting of V popular files which belong to content providers (CP). Intuitively, the majority of MUs request popular files more frequently than the others. Thus, we assume that the popularity profile of files characterized by the Zipf-like distribution as the requests of any files are inversely proportional

to file's rank in the request table [6]. Then, the popularity distribution q_v of $\mathcal{V}_v$ is characterized as

$$q_v = \frac{1/v^{\beta}}{\sum_{j=1}^{V} 1/j^{\beta}}, \quad \forall v, \tag{1}$$

where the exponent β is a positive skewness parameter and v means v-th file. Also, the file with a small index v corresponds to a high popularity.

2.2 Transmission Rate of MUs

Secondly, we define the transmission rate of each pico-cells to MUs as

$$R_{mn} = W \log_2 \left(1 + \frac{P_n d_{mn}^{-\alpha} h_{mn}^2}{\sum_{n' \in \boldsymbol{\mathcal{N}} \setminus n} P_{n'} d_{mn'}^{-\alpha} h_{mn'}^2 + \sigma^2}\right), \tag{2}$$

where W means the transmission bandwidth of downlink channels and P_n is the transmission power at pico-cell $\mathcal{N}_n$. d_{mn} represents distance between pico-cell $\mathcal{N}_n$ and MU $\mathcal{M}_m$ and α is the path-loss exponent. The random channel between $\mathcal{N}_n$ and $\mathcal{M}_m$ is Rayleigh fading, whose coefficient h_{mn} has the average power of one. σ^2 is the variance of the Gaussian noise.

For the sake of simplicity, we assume that the Macro-cell will support a fixed download rate, denoted by R_{ma}, for the MUs in the channels which are orthogonal to those spanning from the pico-cells to MUs.

2.3 Caching Related Issue

Next, we introduce the pico-cell caching system with details. In the first stage, each pico-cell cache one file of $\boldsymbol{\mathcal{V}}$. The file placement commence during off-peak time on backhaul links. It is clear that MUs show different preferences towards different files. Thus, we could define the preference from MU $\mathcal{M}_m$ to file $\mathcal{V}_v$ as

$$p_{mv} = \alpha_{mv} q_v, \tag{3}$$

where α_{mv} is a factor that influence MUs' preferences to file $\mathcal{V}_v$ and q_v is the popularity of the file $\mathcal{V}_v$. Then, the pico-cells will download a file relying on collecting the serving MUs' preferences. We define the preference p_{nv} of pico-cell $\mathcal{N}_n$ to file $\mathcal{V}_v$ as

$$p_{nv} = \frac{1}{\mathcal{C}_n} \sum_{k \in \mathcal{C}_n} p_{kv}, \tag{4}$$

where k means the k-th serving MU of the pico-cell and $\mathcal{C}_n$ is the serving MUs set of $\mathcal{N}_n$. Pico-cells will cache the most preference file of its serving MUs according to (4) during off-peak time.

In the second stage, MUs start to request files. In general, an MU can be covered by multiple pico-cells. When an MU $\mathcal{M}_m$ request file $\mathcal{V}_v$, it tries to connect to the nearest pico-cell which cached file $\mathcal{V}_v$. Otherwise, the MU $\mathcal{M}_m$

will download the requesting file from Macro-cell directly. Thus, the transmission delay τ_{ml}^{v} of MUs can be represented as follows.

$$\tau_{ml}^{v} = \begin{cases} \tau_{ma}^{v} = \frac{S}{R_{ma}} \text{ if MU connects to Macro-cell,} \\ \tau_{mn}^{v} = \frac{S}{R_{mn}} \text{ if MU connects to pico-cell,} \end{cases} \tag{5}$$

where S is the file size.

2.4 Problem Formulation

In this subsection, we formulate the file allocation problem to minimize the transmission delay of MUs. The problem can be formulated as

$$\begin{aligned} \min_{X,Y} \ & \sum_{n\in\mathcal{N}} \sum_{m\in\mathcal{M}} \sum_{v\in\mathcal{V}} x_{nv} y_{nm} \tau_{ml}^{v}, \\ \text{s.t.} \ & (a) \sum_{v\in\mathcal{V}} x_{nv} \leq 1, \\ & (b) \sum_{n\in\mathcal{N}} y_{nm} \leq 1, \\ & (c) \sum_{n\in\mathcal{N}} x_{nv} \leq Q_1, \\ & (d) \sum_{m\in\mathcal{M}} y_{nm} \leq Q_2, \\ & (e) x_{nv}, y_{nm} \in \{0,1\}, \end{aligned} \tag{6}$$

where x_{nv} is the element of matrix X and $x_{nv} = 1$ represents the pico-cell $\mathcal{N}_n$ caches the file $\mathcal{V}_v$, otherwise $x_{nv} = 0$. y_{nm} is the element of matrix Y, and $y_{nm} = 1$ denotes that the pico-cell $\mathcal{N}_n$ is serving MU $\mathcal{M}_m$, if not $y_{nm} = 0$. Condition (a) guarantees that each pico-cell could only cache one file, condition (b) states a user will be served by one pico-cell, condition (c) concerning file variety is to make sure that file $\mathcal{V}_v$ could only be cached Q_1 duplication in this network, condition (d) assures that each pico-cell can serve Q_2 MUs at most, and condition (e) states that the values of x_{nv} and y_{nm} can be neither 0 or 1.

3 Proposed Matching Algorithm

3.1 Decoupling the Optimization Problem

The optimization in (6) is a generalized knapsack problem which is proved to be an NP-hard combinatorial problem [10]. It is hard to find a global optimal solution for these association problems. Hence, to solve the optimization problem in (6), we resort to a suboptimal approach and split the optimization problem into two independent sub-problems: (i) optimal file selection problem, (ii) optimal pico-cell selection problem.

Optimal File Selection Problem

$$\begin{aligned} \min_{X} \ & \sum_{n\in\mathcal{N}}\sum_{v\in\mathcal{V}} x_{nv}\tau_{ml}^{v}, \\ \text{s.t.} \ & (a)\sum_{v\in\mathcal{V}} x_{nv} \le 1, \\ & (b)\sum_{n\in\mathcal{N}} x_{nv} \le Q_1, \\ & (c) x_{nv} \in \{0,1\}. \end{aligned} \tag{7}$$

Optimal Pico-Cell Selection Problem

$$\begin{aligned} \min_{Y} \ & \sum_{n\in\mathcal{N}}\sum_{m\in\mathcal{M}} y_{nm}\tau_{ml}^{v}, \\ \text{s.t.} \ & (a)\sum_{n\in\mathcal{N}} y_{nm} \le 1, \\ & (b)\sum_{m\in\mathcal{M}} y_{nm} \le Q_2, \\ & (c) y_{nm} \in \{0,1\}, \end{aligned} \tag{8}$$

These two suboptimal problems are still NP-hard combinatorial problems [10]. Since they both contain one binary variable, they can be modeled as two distinct many-to-one matching problems respectively [11].

3.2 Matching Related Definition

Matching has been introduced in [12] as a effective way to solve the allocate resource problem. These resources will be divided into two finite and disjoint sets of players. Different sets of players have different preferences over the opposite sets. To construct the preference lists in our model, we use the symbol $\succ$ to represent that the players prefer one player to another player in the opposite set. For example, when a pico-cell $\mathcal{N}_n$ shows $\mathcal{V}_1 \succ \mathcal{V}_2$ in its preference lists, it means that $\mathcal{N}_n$ prefers file $\mathcal{V}_1$ than file $\mathcal{V}_2$. In this paper, we use μ_1 to represent the first many-to-one matching of sub-problem 1.

For $\mathcal{V}_v \in \boldsymbol{\mathcal{V}}$ and $\mathcal{N}_n \in \boldsymbol{\mathcal{N}}$, a matching μ_1 is $\mathcal{V}\bigcup\mathcal{N} \to 2^{\mathcal{V}\cup\mathcal{N}}$, which satisfies

1. $\mu_1(\mathcal{N}_n) \in \boldsymbol{\mathcal{V}}$ and $|\mu_1(\mathcal{N}_n)| \le 1$,
2. $\mu_1(\mathcal{V}_v) \in \boldsymbol{\mathcal{N}}$ and $|\mu_1(\mathcal{V}_v)| \le Q_1$,
3. $\mu_1(\mathcal{V}_v) = \mathcal{N}_n \Longleftrightarrow \mu_1(\mathcal{N}_n) = \mathcal{V}_v$.

The second many-to-one matching μ_2 of sub-problem 2 has the following properties.

For $\mathcal{M}_m \in \boldsymbol{\mathcal{M}}$ and $\mathcal{N}_n \in \boldsymbol{\mathcal{N}}$, a matching μ_2 is $\mathcal{M}\bigcup\mathcal{N} \to 2^{\mathcal{M}\cup\mathcal{N}}$, which satisfies

1. $\mu_2(\mathcal{M}_m) \in \boldsymbol{\mathcal{N}}$ and $|\mu_2(\mathcal{M}_m)| \le 1$,
2. $\mu_2(\mathcal{N}_n) \in \boldsymbol{\mathcal{M}}$ and $|\mu_2(\mathcal{N}_n)| \le Q_2$,
3. $\mu_2(\mathcal{M}_m) = \mathcal{N}_n \Longleftrightarrow \mu_2(\mathcal{N}_n) = \mathcal{M}_m$.

3.3 File Selection Algorithm

Next, Deferred Acceptance (DA) algorithm [11] is proposed to solve these two many-to-one matching games. We first focus on modeling the preference lists in the first matching.

Definition 1. *For MUs $\mathcal{C} \subseteq \boldsymbol{\mathcal{M}}$, the corresponding pico-cell's preference over file $\mathcal{V}_v \in \boldsymbol{\mathcal{V}}$ is*

$$\mathbf{\Gamma}^v = p_{nv}, \tag{9}$$

where $\mathbf{\Gamma}^v \in \mathbb{C}^{N*V}$ is pico-cells' preference matrix over files.

Also, the files of CPs have certain preference towards different pico-cells considering their transmission delay. A pico-cell can serve $\mathcal{C}$ MUs so that we take the average transmission delay of $\mathcal{C}_n$ serving MUs as CP's preference over this pico-cell. We define it as follows.

Definition 2. *For $\mathcal{V}_v \in \boldsymbol{\mathcal{V}}$, its preference over pico-cell $\mathcal{N}_n \in \boldsymbol{\mathcal{N}}$ can be given as*

$$\mathbf{\Gamma}^n = \frac{1}{\mathcal{C}_n} \sum_{k \in \mathcal{C}_n} \tau_{kn}, \tag{10}$$

where $\mathbf{\Gamma}^n \in \mathbb{C}^{V*N}$ is the files' preference matrix over pico-cells.

The detailed algorithm of sub-problem 1 is shown in Algorithm 1. We propose a distributed algorithm where both files and pico-cells selfishly and rationally interact in a way that the MUs sum-transmission delay is minimized. As shown in Algorithm 1, we use $\mathbf{\Gamma}^v$ and $\mathbf{\Gamma}^n$ as preference lists. We assume that the number of files is far more larger than the number of pico-cells. At first, the pico-cells send their first choices according to their preference lists. Since a file can be

Algorithm 1. File Selection Matching Algorithm

Require: $\mathbf{\Gamma}^v$ $\mathbf{\Gamma}^n$

Initialize:

Set Q_1 and construct lists of Unmatched pico-cells to files as sets of $Unmatch_1 = \{\mathcal{N}_1, \mathcal{N}_2, \cdots, \mathcal{N}_n\}$

Main Process:

(1)Each pico-cell sends offer to first file in its preference list.

(**a**)If the number of offers is more than Q_1, then the CP choose most preferred pico-cells and remove these Q_1 pico-cells from $Unmatch_1$ and the others will be rejected.

(**b**)Else the offers will all be accepted.

(2)The rejected pico-cells will remove the most preferred files from its preference lists and go on to take part in $Unmatch_1$.

(3)The algorithm continues until $Unmatch_1$ is empty.

Ensure: stable matching μ_1.

cached in Q_1 pico-cells, we need judge whether the requests from pico-cells for a file are more than the quota Q_1. If there are more than Q_1 pico-cells requesting the same file, CP will select its most preferred Q_1 pico-cells. If the requests for a file are less than or equal to the quota Q_1, files accept these requests all. Then, the remaining pico-cells will be rejected and continue to take part in next round of offers until each pico-cell gets a file to cache.

3.4 Pico-Cell Selection Algorithm

After the file allocation problem solved by Algorithm 1, we turn- to handle the pico-cell selection problem. The transmission delay from the pico-cell to each MU is employed to construct the preference of this pico-cell. Thus, the preference of pico-cell over MU is written as follows.

Definition 3. *For a pico-cell $\mathcal{N}_n \in \boldsymbol{\mathcal{N}}$, its preference over MU $\mathcal{M}_m \in \boldsymbol{\mathcal{M}}$ can be given as*

$$\mathbf{\Gamma}^m = \tau_{mn}^v, \tag{11}$$

where $\mathbf{\Gamma}^m \in \mathbb{C}^{N*M}$ is the pico-cells' preference matrix over MUs.

Moreover, the preference over pico-cells by MUs will be effected by sub-problem 1. When the pico-cells cached files, they broadcasts information about the cached files to all serving MUs. Accordingly, the MUs' preference over pico-cell is also based on p_{mv}, since different files have different attraction for MUs. Then, the MU's preference over pico-cells is written as follows.

Definition 4. *For $\mathcal{M}_m \in \boldsymbol{\mathcal{M}}$, its preference over pico-cell $\mathcal{N}_n \in \boldsymbol{\mathcal{N}}$ can be given as*

$$\mathbf{\Gamma}^{n'} = \tau_{mn}^v p_{mv}, \tag{12}$$

where $\mathbf{\Gamma}^{n'} \in \mathbb{C}^{M*N}$ is the MUs' matrix preference over pico-cells.

The specific algorithm of sub-problem 2 is shown in Algorithm 2. The distributed Algorithm 2 has been proposed to minimize sum-transmission delay by allocate the MUs to pico-cells effectively. The distributed Algorithm 2 progresses as MUs send their offers to most select pico-cells. Then, if the requests for a pico-cell are more than the quota Q_2, the pico-cell will decide to accept the more preferred MUs according to preference lists, and reject the others. Else, pico-cells will accept all requests. The rejected MUs remove the most select pico-cells and continue to take part in progress till all pico-cells could not construct more links to MUs. At the last, if there still exists a MU without links, it will connect to Macro-cell.

Algorithm 2. Pico-cell Selection Matching Algorithm

Require: $\mathbf{\Gamma}^m$ $\mathbf{\Gamma}^{n'}$

Initialize:

Set Q_2 and construct lists of Unmatched MUs to pico-cells as sets of $Unmatch_2 = \{\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_m\}$

Main Process:

(1)Each MU makes an offer to pico-cell which is in first place in MUs' preference list.

(**a**)If the number of offers is more than Q_2, then the pico-cell choose most preferred MUs and remove these MUs from $Unmatch_2$. The others are rejected and still be active in $Unmatch_2$.

(**b**)Else, the all offers will be accepted by pico-cells.

(2)The rejected ones delete the most preferred pico-cell from its preference lists.

(3)The process repeats (1) and (2) until $Unmatch_2$ is empty or all pico-cells can not accept any other MUs.

(4)The unmatched MUs will have a connection with Macro-cell.

Ensure: stable matching μ_2.

3.5 Stability of Matching

At last, it is critical to find a stable matching between two opposite sides, because it guarantees that none of players have motivations to change their matched players. Since the matching μ_1 obeys the similar matching rules with μ_2, we will analysis the stability of the second matching μ_2 for brevity.

We prove it by contradiction. Given a blocking pair $(\mathcal{M}_1, \mathcal{N}_1)$ for μ_2. In this case, $\mathcal{N}_1$ prefers $\mathcal{M}_1$ to $\mu_2(\mathcal{N}_1)$ and $\mathcal{M}_1$ prefers $\mathcal{N}_1$ to at least one element in $\mu_2(\mathcal{M}_1)$, i.e., the MU matches with another pico-cell which is in a higher order in preference list than its previous matched player, and hence both opposite sides will improve their outcomes. But it is contrary to the original definition of matching, i.e., the opposite two sides will change its order in preference list firstly in order to get a better outcome. Thus, there exists no blocking pair in matching games and μ_2 is proved to be a stable matching.

4 Numerical Results

In this section, we evaluate the performance of the proposed two algorithms by some numerical results. We assume the quota of $\boldsymbol{\mathcal{V}}$ is 20 and $Q_1 = 2$. Moreover, the quota of $\boldsymbol{\mathcal{M}}$ is $\in [16, 34]$ and the quota of $\boldsymbol{\mathcal{N}}$ is $\in [8, 20]$. In this simulations, we assume all MUs and pico-cells are randomly located and the physical layer parameters are set to be practical. The transmit power of a pico-cell is typically 2W, the pass loss α is 4 and the noise power is set to 10^{-10}.

In the following figures, we compare our proposed algorithms with no caching, random allocation and exhaustive searching algorithms. In the no caching algorithm, the pico-cell caches no file. If MUs connect to the pico-cells, the pico-cells would download the requesting files via backhaul channels firstly and then

transmit the files to MUs. Thus, no caching algorithm will lead to high latency. In the random allocation algorithm, we assign that files are randomly cached. In the exhaustive searching algorithm, the problems are addressed by centralized solution with high complexity.

In Fig. 2, the number of MUs is 60 and the pico-cell number varies from 8 to 20 with each pico-cell serving at most 2 MUs. As observed from Fig. 2, with the increasing of pico-cell number, the average delay of both the exhaustive searching algorithm curve and proposed algorithm curve have a decreasing trend. Though the exhaustive searching algorithm shows better performance to the proposed one, the proposed algorithm has less computation complexity $\mathcal{O}(\mathcal{M} \times \mathcal{N})$ while the exhaustive searching algorithm complexity increases exponentially over network size. It is easy to verify that with low complexity, the proposed algorithm will reduce a big cost of server processing power. Also, it can be observed that no caching algorithm's average downloading delay is fixed with the 0.5 s delay. And we find that random allocation algorithm exhibits a inferior performance compared with proposed algorithm.

In Fig. 3, we fix the pico-cell number to be 10 and vary the MUs' number from 16 to 34. Figure 3 displays that the proposed one has similar delay performance with the exhaustive searching algorithm, and they both descend as the increase of MUs' numbers. In random allocation algorithm, it shows inferior performance compared with proposed algorithm.

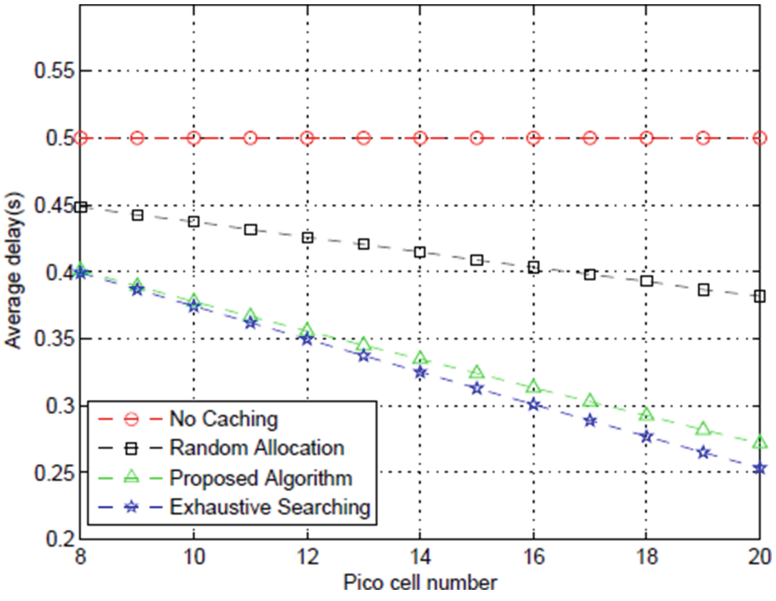

Fig. 2. Average delay vs Number of pico-cells

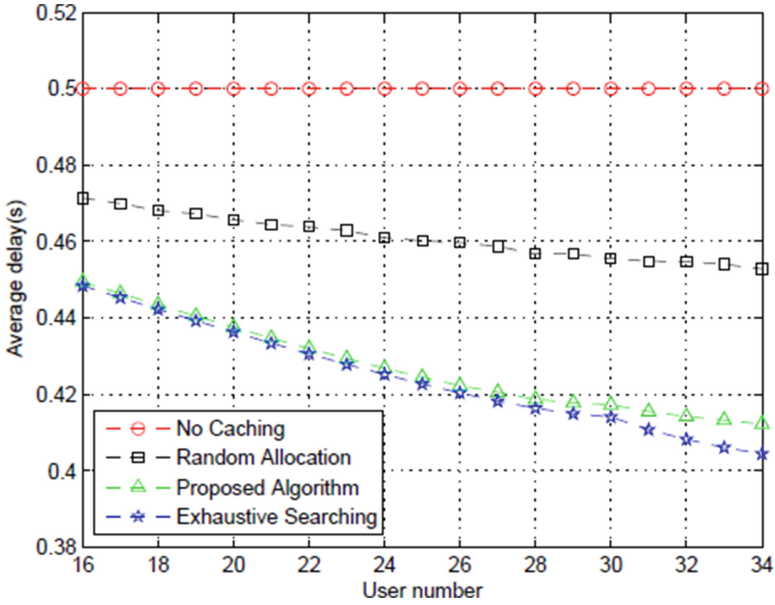

Fig. 3. Average delay vs Number of MUs

5 Conclusion

In this paper, we solve the resource allocation problems in a pico-cell based caching network. The problems are firstly decoupled into two sub-problems. Then, we construct the preference lists of pico-cells, MUs and CPs, respectively. Then, two algorithms is proposed to find stable matchings in these two matching games. At last, simulation results are shown to demonstrate that the proposed algorithm has similar performance with the exhaustive searching algorithm in reducing average transmission delay and shows better performance than the random allocation and no caching algorithms.

References

1. 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. **51**(4), 142–149 (2013)
2. Zhang, H., Liu, H., Jiang, C., Chu, X.: A practical semidynamic clustering scheme using affinity propagation in cooperative picocells. IEEE Trans. Veh. Technol. **64**(9), 4372–4377 (2015)
3. Quek, T., De, G., Guvenc, I., Kountouris, M.: Small Cell Networks - Deployment, PHY Techniques, and Resource Management. Cambridge University Press, New York (2013)
4. Shanmugam, K., Golrezaei, N., Dimakis, A.G., Molisch, A.F., Caire, G.: Femtocaching: wireless content delivery through distributed caching helpers. IEEE Trans. Inf. Theor. **59**(12), 8402–8413 (2013)

5. Golrezaei, N., Mansourifard, P., Molisch, A.F., Dimakis, A.G.: Base-station assisted device-to-device communications for high-throughput wireless video networks. Comput. Sci. **13**(7), 7077–7081 (2012)
6. Li, J., Chen, H., Chen, Y., Lin, Z., Vucetic, B., Hanzo, L.: Pricing and resource allocation via game theory for a small-cell video caching system. IEEE J. Sel. Areas Commun. **34**(8), 2115–2129 (2016)
7. Gu, Y., Saad, W., Bennis, M., Debbah, M., Han, Z.: Matching theory for future wireless networks: fundamentals and applications. IEEE Commun. Mag. **53**(5), 52–59 (2015)
8. Namvar, N., Saad, W., Maham, B., Valentin, S.: A context-aware matching game for user association in wireless small cell networks. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 439–443, May 2014
9. Hamidouche, K., Saad, W., Debbah, M.: Many-to-many matching games for proactive social-caching in wireless small cell networks. In: 2014 12th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), pp. 569–574, May 2014
10. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, New York (2004)
11. Roth, A., Sotomayor, M.: Two Sided Matching: A Study in Game-Theoretic Modeling and Analysis, 1st edn. Cambridge University Press, Cambridge (1989). Ideas Help Page
12. Gale, D., Shapley, L.S.: College admissions and the stability of marrige. Am. Math. Mon. **69**(1), 9–15 (1962)