A Novel Algorithm for Detecting Social Clusters and Hierarchical Structure in Industrial IoT

Jiming Luo, Kai Lin(✉), and Wenjian Wang

School of Computer Science and Technology, Dalian University of Technology, Dalian, China
logicluo@foxmail.com, link@dlut.edu.cn, dlutwwj0boa@mail.dlut.edu.cn

Abstract. The rapid development of IoT has brought life around us with tremendous impact. Especially, industrial IoT as a new research hotspot, has been attracting extensive concern from industry and academia, facilitating many technologies and application in industrial IoT. However, taking full advantage of a large number of resources in industrial IoT is a challenging task. In this article, we present an efficient mobile social cluster algorithm (OMSC) to detect the potential social relationships among mobile devices in industrial IoT. It can discover the overlapping cluster and hierarchical structure in near-line time. We implement this algorithm in the Java Platform and validate the OMSC in synthetic networks and real-world network datasets. The experimental results demonstrate that the presented OMSC algorithm has high performance.

Keywords: IoT · Industrial cloud · Social cluster · Hierarchical structure · Mobile device

1 Introduction

With the rapid development of Internet of Thing (IoT), a splendid Internet blueprint containing all kinds of resources and services appears in front of people [1]. IoT brings economies and societies around us tremendous impact. The obvious feature of IoT is mainly the omnipresent information acquisition and ubiquitous information processing. The intellectualization progress of various field gains the effective pushing from IoT. Especially, IoT has been accelerated the development of next industrial revolution "Industry 4.0" [2]. Industrial Internet of Things (IIoT), as a remarkable feature of "Industry 4.0", is expected to provide high efficient schemes for many existing industrial fields such as manufacturing fields and transportation fields. Much concern from industry and academia around the world have been put into Industrial IoT, which greatly facilitates the progress of Industrial Internet of Things (IIoT).

With the gradually increasing number of mobile devices in industrial IoT, the way of communication among people has a dramatic changing. Worldwide shipments of tablets and mobile phones increasing to 2.072 billion units which

J. Wan et al. (Eds.): Industrial IoT 2016, LNICST 173, pp. 166–178, 2016.
DOI: 10.1007/978-3-319-44350-8_17

is acceding PC-desk based and notebook 315,229 units in 2013 and also 37 % of PC users have switched to cellphone or tablet to surf the Internet or play games [3]. However, how to effectively utilize the resources among the mobile devices has certain challenge.

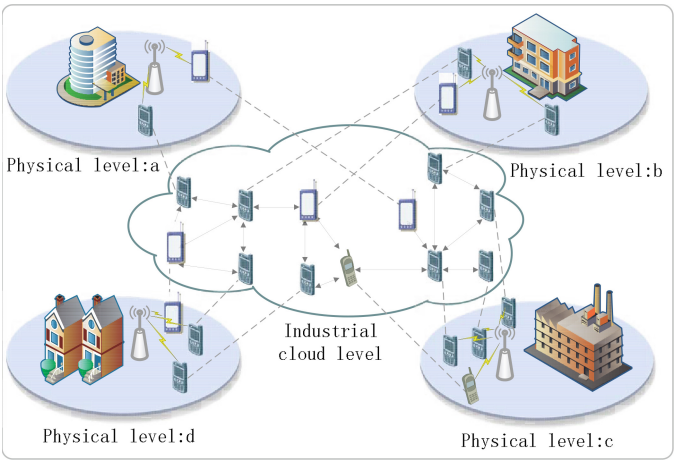


Fig. 1. An illustration of mobile device relationships in the industrial cloud level and physical level.

In industrial cloud applications where the data is shared by multiple mobile users, it is essential to provide consistency among mobile users by means of synchronization algorithms. In particular, if the data is frequently updated and the number of mobile users sharing the data is large, the synchronization traffic can significantly increase. Virtualisation of industrial cloud networks can be seen as a way to substantially reduce the complexity of processes. Due to the limitations of mobile devices, using mobile databases on mobile devices encounters certain scalability issues in mobile data accesses and storage. One alternative approach is to use databases based on industrial cloud to support mobile users and applications on mobile devices. To support the fast increasing needs of mobile data accesses and mobile computing services, social cluster can play an important role in industrial IoT.

Social cluster discovering is an essential tool to understand the architecture of industrial IoT, which gives a new perspective to create applications. In the cluster, as illustrated in Fig. 1, a key observation is that mobile devices are coupled not only in the physical level owing to the physical relationship, but also in industrial cloud level due to the social ties among them [4]. In the physical level, mobile devices may change their location in a short time, but the social relationships are not changed. Mobile devices in the same cluster usually have the similar function or characteristic in industrial IoT. For this reason, cluster in industrial IoT can be seen as a way of tackling challenges.

Numerous techniques have been developed for social cluster detection in industrial IoT. However, most of them only considered one mobile device belongs to one social cluster [5,6], or hierarchical and non-overlapping clusters [7], or non-overlapping and overlapping clusters [8–10]. Even though some solutions can detect hierarchical and overlapping clusters in industrial IoT, they need to acquire the whole networks information, and the computational complexity of them is high which makes them unsuitable for real-world networks.

In this paper, we are based on the observation of industrial IoT, and propose a novel algorithm to detect the mobile social cluster. The proposed algorithm can find the hierarchical structure and overlapping mobile social clusters with reasonable time and space complexity. In addition, the proposed method takes full advantages of local resources in industrial IoT and there is no need to acquire the whole network information. To summarize, the contributions of our paper are summarized as follows:

1. We present an overlapping mobile social cluster algorithm (OMSC), the thought of which is detecting the relationship among mobile devices in industrial IoT. It can find the overlapping mobile social cluster in nearly-linear time.
2. We also design a mobile hierarchical structure method, which makes use of the results of OMSC and illustrates the relation of mobile social clusters.
3. To validate the efficiency and effectiveness of the proposed method OMSC, we have implemented our algorithm in the Java Platform and evaluated it in synthetic and real-world network datasets.

The remainder of this paper is organized as follows. Section 2 presents some related works. In Sect. 3, we describe the cluster method in industrial cloud networks. Experiments and results are presented in Sect. 4. In the end, we draw our conclusions in Sect. 5.

2 Related Work

In industrial IoT, social clusters are the essential local structures, which involve a group of mobile devices that intercommunicate with each other densely but communicate to the others sparsely. A large number of mobile social cluster discovery methods have been proposed, especially in the past decade, which fall in two broad categories: (1) matrix transformation; (2) non-matrix transformation.

As for matrix transformation mobile social clusters detection methods, I. Psorakis et al. [11] propose a probabilistic clusters detection approach where a Bayesian non-negative matrix factorization model is utilized to form an industrial cloud networks through extracting overlapping modules. The integration of soft-partitioning solutions, allocation of node concernment scores for modules, and an intuitive basis is achieved in this scheme. Based on the matrix factorization approach, Y. Zhang et al. [12] present a bounded nonnegative matrix tri-factorization (BNMTF) method, which can detect the overlapping social clusters with reasonable time complexity.

On the other hand, U.N. Raghvan et al. [13] introduce a simple label propagation algorithm in which the network structure is viewed as the only guidance. Moreover, in the algorithm, a predefined objective function and the preferential information of the cluster don't need optimizing. In the process of the algorithm, a unique label is used to initialize every mobile device, and then each mobile device receives the label coming from its neighbors. In this iteration, the groups of mobile devices that densely connected each another reach a consensus on a unique label to form clusters. To contain the information about more than one cluster, G. Steve extends the label and propagation step: every node can currently belong to v clusters, where v is the number of clusters [14]. Z. Zhang et al. investigate an functional and efficient cluster detecting algorithm MOHHC, where the overlapping and hierarchical organization in industrial IoT can simultaneously is discovered. In the algorithm, all maximal cliques are first extracted from the original Industrial IoT, and these extracted maximal cliques are merged into a dendrogram based on the aggregative framework from MOHCC. Finally, the dendrogram is cut through and achieve a network partition by using maximum extended partition density [15].

3 Social Clusters Detection in Industrial Cloud Network

This section presents a novel overlapping and hierarchical mobile social clusters detection (OMSC) algorithm for industrial IoT. Social cluster is an important characteristic of Industrial IoT. Although the concept of mobile social clusters dose not have clear definition, most industrial IoT emerges cluster structures. For instance, a group of mobile devices densely intercommunicate and sparsely communicate with others in industrial IoT. Furthermore, mobile cloud networks also contain some hierarchical organizations. As shown in Fig. 2, cluster 1 and cluster 2 contain some mobile devices and they are components of cluster a. Cluster a and cluster b belong to the cluster A. The proposed algorithm include two parts: overlapping mobile clusters detection (OMSC) and hierarchical structures detection. To describe the algorithm more clearly, we use the vertex or node to replace the mobile device and the edge to replace the relationship of two different mobile devices.

3.1 Overlapping Mobile Clusters Detection

According to the potential social relationship in industrial IoT, we define a cluster set $C(c_a, c_b, ..., C_\varepsilon)$ which represents social clusters result of industrial IoT. Based on the analysis of social relationship, all the mobile devices in industrial IoT can be divided into clusters. We also define a vector P_ξ^λ for node ξ (λ represents the iterations which will be described detailly later), which storages some key-value pairs $(label, proportion)$. In the key-value pair, the $label$ represents the node (mobile device) belonging to the cluster (C_{label}) and the $proportion$ is the possibility of the node belonging to the cluster (C_{label}). When the algorithm runs over, the remains of labels in the vector P_ξ^λ denote the node should be divided to which clusters.

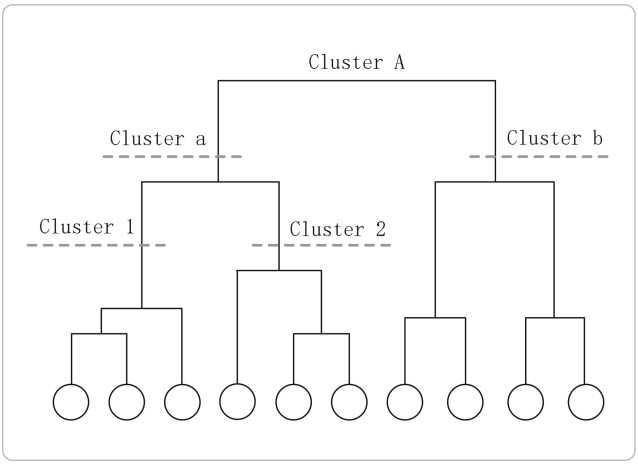


Fig. 2. An example of hierarchical structure in the industrial cloud networks.

Based on our observation, we find that a set of nodes in a complete subgraph are more likely to belong to a same cluster. So we come up with a new concept named primitive cluster (PC) in industrial IoT. Each pair of nodes in the PC is connected with each other and every nodes have the maximum number of the same neighbors existed in the same PC, and all the number of same neighbors should be equal to or less than this maximum value. The first step of the proposed overlapping mobile social clusters discovers the algorithm is to extract PCs from the industrial IoT.

In each node, a sequence of labels are maintained which can be broadcasted to its neighbors. We define n $1 \times n$ vectors P_ξ^λ (n is the number of nodes in the industrial cloud networks). For each key-value pair $(label, proporation)$ in the vector P_ξ^λ, we denote $P_\xi^\lambda(label) = proporation$ for convenience. In every broadcasts, each neighbor of node ξ send its vector to node ξ. Then, node ξ's vector will be computed by following equation:

$$P_\xi^{\lambda'+1}(label) = \sum_{i \in Nb(\xi)} \frac{P_\xi(label)}{k_\xi}, \quad \forall label \in L \tag{1}$$

where $Nb(\xi)$ is the neighbor set of node ξ and $k_\xi = |Nb(\xi)|$ is the number of neighbors. $L = \{a, b, c, ...\}$ (same as node id's) and $|L| = n$. As for some labels in the node ξ's neighbors not in ξ, we add these labels and their proportions on node ξ's vector directly. And we normalized ${P_\xi}^{\lambda'+1}$ using follow equation so that the proportions in $P_\xi^{\lambda'+1}$ sum to 1:

$$P_\xi^{\lambda''+1}(label) = \frac{P_\xi^{\lambda'+1}(label)}{\sum_{label \in P_\xi^{\lambda'+1}} P_\xi^{\lambda'+1}(label)} \tag{2}$$

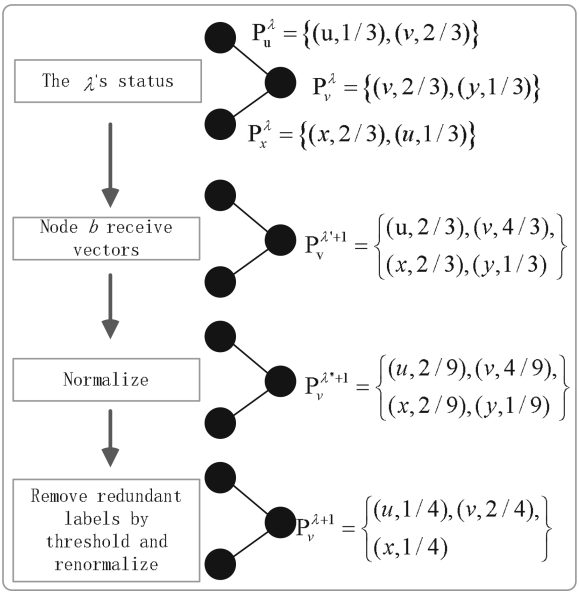


Fig. 3. Node v's vector changes from λ to $\lambda + 1$ iteration.

After normalizing the vector $P_{\xi}^{\lambda'+1}$, there are some redundant key-value pairs in the $P_{\xi}^{\lambda''+1}$. To solve this problem, we set a threshold ψ which value is initialized at the begin of algorithm. At each propagation step, we remove these key-value pairs in which the proportion is less than the threshold.

When the redundant key-value pairs are removed, the proportions in the vector $P_{\xi}^{\lambda''+1}$ may not satisfy the condition that the proportions in the vector $P_{\xi}^{\lambda''+1}$ are sum to 1. So we should use the Eq. 2 to re-normalize the vector. Finally, the step of propagation is finished completely.

$$P_{\xi}^{\lambda+1}(label) = P_{\xi}^{\lambda'''+1}(label) \tag{3}$$

There is an problem that when the algorithm stop or what is the algorithm terminate condition. In [13], they have been proved that if the number of iterations is larger than twenty, the algorithm has high performance. So when it run twenty times, the algorithm will be stopped.

To deeply comprehend the proposed algorithm, there is an example to explain it. As shown in Fig. 3, node v has two neighbors which denote u and x respectively. At the λ's iteration, $\mathrm{P}_u^{\lambda} = \{(u, 1/3), (v, 2/3)\}$, $\mathrm{P}_v^{\lambda} = \{(v, 1/3), (y, 2/3)\}$ and $\mathrm{P}_x^{\lambda} = \{(x, 2/3), (u, 1/3)\}$ (u, v, x and y, are *label*s). In the next iteration, the neighbors of node v send their vectors to node v, and node v use the Eq. 1 to compute its vector $P_v^{\lambda'+1}$, and the result is $P_v^{\lambda'+1} = \{(u, 2/3), (v, 4/3), (x, 2/3), (y, 1/3)\}$. Because the proportions in each vector sum to 1, we use equation to normalize the vector $P_v^{\lambda'+1}$, and it is

Algorithm 1. OMSC

T: the maximum iteration user defines.
ψ: threshold to cut off redundant labels.

1. First, extract PC from the industrial cloud networks.
2. Second, in each PC, every nodes are initialized with the same label and the rest of nodes in the network are also initialized with unique labels, and the proportion of each label in the vector is initialized 1.
3. Then, the following steps are repeated until the maximum iteration T is reached:
   (a) Selected one node v as a listener.
   (b) Each neighbor of the listener node v send their vector to node. Node v first compute $P_v^{\lambda'+1}$, and then normalize it to $P_v^{\lambda''+1}$.
   (c) Using the threshold ψ to remove the redundant key-value pairs $(label, propotion)$ and re-normalize the vector to $P_v^{\lambda'''+1}$.
   (d) The listener node v adds vector $P_v^{\lambda'''+1}$ to its memory and the vector as the initial value for next propagation.
4. Finally, in each node, its vector's label denotes that the node belong to which mobile social cluster.

changed to $P_v^{\lambda''+1} = \{(u, 2/9), (v, 4/9), (x, 2/9), (y, 2/9)\}$. The threshold is $1/9$ to cut off redundant labels, and the pair of $(y, 1/9)$ will be removed from the vector $P_v^{\lambda''+1}$. Then we re-normalize the vector and this iteration is complete. $P_v^{\lambda+1} = \{(u, 1/4), (v, 2/4), (x, 1/4)\}$. If the iteration $\lambda + 1$ is the last step of the algorithm, node v belongs to clusters c_u, c_v, c_x simultaneously. The mobile social clustering algorithm is detailly described in Algorithm 1.

3.2 Hierarchical Structure Detection

Hierarchical structure is an potential characteristic of industrial IoT. Based on the results of the last subsection, we propose a mobile hierarchical structure detection algorithm. Firstly, nodes that belong and only belong to a cluster will be removed from industrial IoT. The rest of nodes in the cluster are given a same label (different cluster contains different label). Secondly, randomly select one node ϕ as the listener. All neighbors of node ϕ send their labels to the node ϕ and node ϕ choose the label which has the maximum number from the received labels. If each node in the network has a label with the maximum number from their neighbors, the iteration is stopped. Otherwise, repeat this step. Finally, if the rest of nodes have the same label, the algorithm stops, if not, go to the first step. The flow diagram of this algorithm is described in Fig. 4.

When the algorithm is completed, the dendrogram will be constructed. The potential relationship in industrial IoT can be detected and all mobile devices are divided into different mobile social clusters. As mentioned above, any mobile devices can belong to more than one cluster. We can get the clear perspective for industrial IoT.

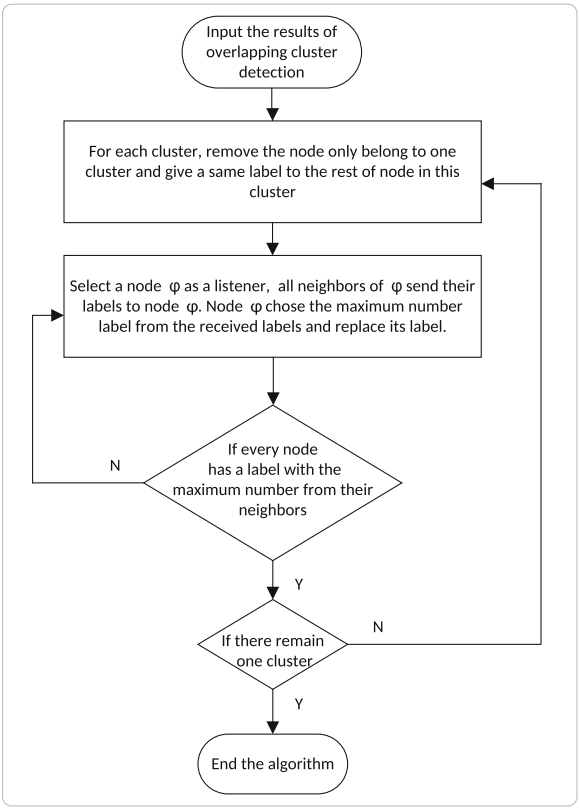


Fig. 4. Flow chart of industrial cloud hierarchical structure detecion.

3.3 Time Complexity

As industrial IoT contains millions of mobile devices, time complexity is an important specification for the mobile social cluster detection algorithm. In our algorithm, the major time consumption is focused on the overlapping mobile social cluster detection. When we extract the PCs for mobile device h, d_h comparisons are made to discover the neighbors of h with the maximum number of common neighbors. The number of comparisons will be at most $\sum_{h=1}^{n} d_h$ where d_h represents the degree of node h and n is the number of mobile devices. On the other hand, $\frac{1}{2}\sum_{h=1}^{n} d_h = m$, where m is the total of links and T is the sum of iteration. In short, the complexity of OMSC is $O(Tm)$.

4 Experiments and Results

In this section, to validate the performance of the algorithm OMSC, we conducted experiments in synthetic networks and real-world network dataset.

We adopted the LFR benchmark graphs to generate the synthetic networks dataset and the Normalized Mutual Information (NMI) to qualify our algorithms performance [16,17]. Given two social clusters A and B, the NMI is defined below.

$$\begin{aligned} NMI(X,Y) &= 1 - \tfrac{1}{2}(H(X|Y)_{part} + H(Y|X)_{part}) \\ H(X|Y)_{part} &= \tfrac{1}{|C_X|}\sum_i \tfrac{\min_{j\in\{1,2,\ldots,|C_Y|\}} H(X_i|Y_j)}{H(X_i)} \\ H(Y|X)_{part} &= \tfrac{1}{|C_Y|}\sum_i \tfrac{\min_{j\in\{1,2,\ldots,|C_Y|\}} H(Y_i|X_j)}{H(Y_i)} \end{aligned} \tag{4}$$

where $H(X|Y)$ and $H(Y|X)$ are conditional entropy, $|C_X|$ and $|C_Y|$ severally represent the number of mobile social cluster in X and Y. The computation of NMI consists of two steps. In the first step, the pairs of clusters whose similarity degree is highest in two social clusters are discovered. In the second step, the mutual information among those pairs of clusters is further averaged. The more similar two clusters are, the higher NMI score is. If two cluster A and B are exactly the same, the value is 1.

For real-world networks dataset, we use EQ (extended modularity) to evaluate the cluster structures. EQ is a variant of the commonly used modularity (Q) [18], which is defined for overlapping communities by Newman et al. [19]. This extended modularity is defined as follows:

$$EQ = \frac{1}{2m}\sum_{\alpha,\beta}(W_{\alpha\beta} - \frac{k_\alpha k_\beta}{2m})\ell(c_\alpha, c_\beta) \tag{5}$$

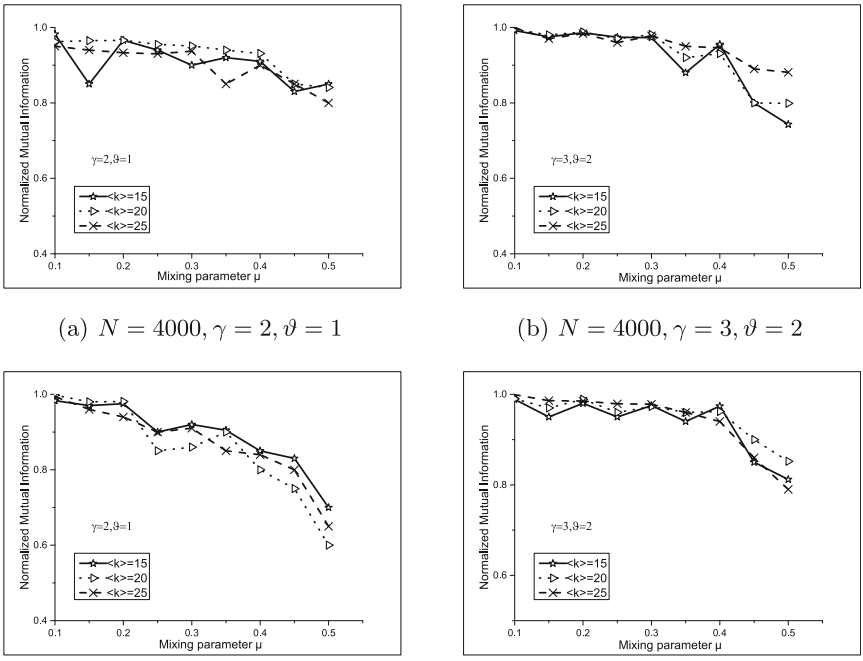
where $W_{\alpha\beta}$ is the weight of the edge between node α and node β. k_α is the degree of vertex α. c_α is the cluster which the node α belongs to. m is the sum of the edges of the industrial IoT,

$$m = \frac{1}{2}\sum_{\alpha,\beta} W_{\alpha\beta} \tag{6}$$

and the function $\ell(c_\alpha, c_\beta)$ is 1 iff $c_\alpha = c_\beta$, that is, α and β belong to the same cluster and 0 means other situations.

4.1 Synthetic Networks Dataset

As shows in Fig. 5, we tested OMSC algorithm in the networks which were generated by LFR benchmark. N is the number of nodes in the synthetic network. γ is the exponent for the cluster degree distribution, ϑ is the exponent for the cluster size distribution, and $\langle k\rangle$ is the average degree and there is a mixing parameter μ, which is the average fraction of neighboring mobile devices of a mobile device that do not belong to any cluster that benchmark mobile device belongs to. This parameter controls the fraction of edges that are between clusters. As shown in Table 1, we choose the $N = 4000$ and 8000, $\gamma = 2$ and $\gamma = 3$, $\vartheta = 1$ and $\vartheta = 2$ and μ from 0.1 to 0.5 respectively. The result showed that OMSC achieves the most stable and competitive NMI scores in synthetic industrial IoT.



(a) $N = 4000, \gamma = 2, \vartheta = 1$ (b) $N = 4000, \gamma = 3, \vartheta = 2$

(c) $N = 8000, \gamma = 2, \vartheta = 1$ (d) $N = 8000, \gamma = 3, \vartheta = 2$

Fig. 5. Test of our algorithm on the LFR benchmark. The number of nodes is N, γ is the exponent for the degree distribution, ϑ is the exponent for the cluster size distribution, and $\langle k \rangle$ is the average degree.

Moreover, with the increasing of mixing parameter μ, the NMI scores still remains high and balance. we can draw a conclusion that the clusters produced through OMSC maintain the pretty high similarity with the ground-truth, even though many social clusters are overlapped with each other.

4.2 Real-world Networks Dataset

We next utilize OMSC to make an analysis of the real-network datasets which are more irregular than synthetic networks and find their overlapping mobile social clusters. Because industrial IoT and social networks have the same characteristics and there are no real industrial IoT datasets available, we use the four real social network datasets, namely the Zachary's karate club network, the bottlenose dolphin network, the American college football league and the network of users of the Pretty-Good-Privacy algorithm, to verify our algorithm. Among them, the well-known Zachary's karate club network contains 78 edges and 34 nodes [20]. There are 159 edges and 62 nodes in the bottlenose dolphin network [21]. The American college football league is composed of 613 edges and 115 nodes [22]. The network that implements the Pretty-Good-Privacy algorithm

Table 1. The distribution number of nodes γ and ϑ.

Number of Nodes	γ	ϑ
4000	2	1
4000	3	2
8000	2	1
8000	3	2

has 24340 edges and 10680 nodes. Besides, the communicating network based on emails in Enron consists of 367662 edges and 367692 nodes [23]. The information about the networks testing results is detailedly shown in Table 2. It shows that our mobile social cluster detection algorithm has a good performance.

Table 2. Our algorithm performance in the real-world network datasets.

Network	Modularity(Q)	Time (s)
Zacharys karate club	0.45	0.392
Bottlenose dolphin	0.57	0.481
Football	0.63	0.546
Pretty-Good-Privacy	0.82	125.45
Enron via emails	0.60	3256.26

5 Conclusion

In this paper, we propose a mobile overlapping social cluster and hierarchical structure detection algorithm in industrial IoT, which can discover the mobile social clusters with near-line time and the results of this algorithm give a new perspective for Industrial IoT. We apply this algorithm to synthetic network and real-world network dataset, and the experiment result shows that our algorithm has a high performance. The algorithm promotes the development of industrial IoT and offers great help for researchers to intensively study industrial IoT. In future work, we plan to further increase the performance of our algorithm which can be used in the world-class industrial IoT.

Acknowledgments. This work was supported by the National Natural Science Foundation of China under Grant No. 61103234 and No. 61272417, the China Scholarship Council and the Fundamental Research Funds for the Central Universities under Grant No. DUT16QY18.

References

1. Liu, J., Li, Y., Chen, M., Dong, W., Jin, D.: Software-defined internet of things for smart urban sensing. IEEE Commun. **53**(9), 55–63 (2015)
2. Wang, S., Wan, J., Li, D., Zhang, C.: Implementing smart factory of Industrie 4.0: an outlook. Int. J. Distrib. Sens. Netw. **2016**, 10 (2016)
3. Al-Saeed, L., Li, M., Al-Raweshidy, H.: Cognitive data routing in heterogeneous mobile cloud networks. In: 2014 2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud), Washington, DC, USA, pp. 194–199 (2014)
4. Xu, C., Gong, X., Yang, L., J. Zhang.: A social group utility maximization framework with applications in database assisted spectrum access. In: IEEE INFOCOM 2014 Proceedings, Toronto, pp. 1959–1967 (2014)
5. Newman, E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **69**(2), 292–313 (2004)
6. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(35), 75–174 (2009)
7. Huang, J., Sun, H., Han, J., Feng, B.: Density-based shrinkage for revealing hierarchical and overlapping community structure in networks. Phys. A Stat. Mech. Appl. **390**(11), 2106–2171 (2011)
8. Souam, F., Aïtelhadj, A., Baba-Ali, R.: Dual modularity optimization for detecting overlapping communities in bipartite networks. Knowl. Inf. Syst. **40**(2), 455–488 (2014)
9. Sun, P., Gao, L., Han, S.: Identification of overlapping and non-overlapping community structure by fuzzy clustering in complex networks. Inf. Sci. **181**(6), 1060–1071 (2011)
10. Yang, J., Leskovec, J.: Overlapping community detection at scale: a nonnegative matrix factorization approach. In: Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, New York, NY, USA, pp. 587–596 (2013)
11. Psorakis, I., Roverts, S., Ebden, M., Sheldon, B.: Overlapping community detection using Bayesian non-negative matrix factorization. Phys. Rev. E **83**(6), 1–9 (2011)
12. Zhang, Y., Yeung, D.: Overlapping community detection via bounded nonnegative matrix tri-factorization. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, pp. 606–614 (2012)
13. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E **76**(3), 1–11 (2007)
14. Steve, G.: Finding overlapping communities in networks by label propagation. New J. Phys. **12**(10), 1–27 (2010)
15. Zhang, Z., Wang, Z.: Mining overlapping and hierarchical communities in complex networks. Phys. A Stat. Mech. Appl. **421**, 25–33 (2015)
16. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Phys. Rev. E Stat. Nonlin. Soft Matter Phys. **78**(4), 561–570 (2008)
17. Estévez, P., Tesmer, M., Perez, C., Zurada, J.M.: Normalized mutual information feature selection. IEEE Trans. Neural Netw. **20**(2), 189–201 (2009)
18. Blondel, V.D., Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. Theor. Exp. **2008**(10), 1–12 (2008)
19. Newman, M.E.: Analysis of weighted networks. Phys. Rev. E Stat. Nonlin. Soft Matter Phys. **70**(5), 1–9 (2004)

20. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3), 75–174 (2010)
21. Psorakis, I., Roberts, S., Ebden, M., Sheldon, B.: Overlapping community detection using Bayesian non-negative matrix factorization. Phys. Rev. E **83**(6), 1–9 (2011)
22. Xie, J., Kelley, S., Szymanski, B.K.: Overlapping community detection in networks: the state-of-the-art and comparative study. ACM Comput. Surv. (CSUR) **45**(4), 1–35 (2013)
23. Leskovec, J., Lang, K.J., Dasgupta, A., Mahoney, M.W.: Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. Internet Math. **6**(1), 29–123 (2009)