Distributed Graph Clustering for Application in Wireless Networks

Chia-Hao Yu[1], Shaomeng Qin[2], Mikko Alava[2], and Olav Tirkkonen[1,3]

[1] Department of Communications and Networking, Aalto University, Finland
[2] Department of Applied Physics, Aalto University, Finland
[3] Nokia Research Center, Helsinki, Finland
{chiahao.yu,shaomeng.qin,mikko.alava,olav.tirkkonen}@tkk.fi

Abstract. We consider distributed clustering of weighted graphs. Each node in the graph is represented by an agent, with agents independent of each other. The target is to maximize the sum weight of intra-cluster edges with cluster size constrained by an upper limit. To avoid getting stuck in not-too-good local optima, we approach this problem by allowing bad decision-making with a small probability that is dependent on the depth of local optima. We evaluate performance in a setting inspired by self-organizing coordination area formation for coordinated transmission in wireless networks. The results show that our distributed clustering algorithm cab perform better than a distributed greedy local search.

Keywords: Distributed graph clustering, Flat clustering, Self-organized network, Non-clusterhead-based clustering, Distributed coordination area formation.

1 Introduction

Graph clustering, which by definition groups vertices of a given input graph into clusters based on underlying structures, has grown popular and found its application in e.g., database systems, biological and sociological networks, and information networks [16]. Broadly speaking, we can divide graph clustering methods into *global clustering* [7,13,14] and *local clustering* [2,15]. A global clustering method assigns each vertex on a given graph to a cluster, whereas a local clustering method only assigns a certain subset of vertices on a given graph to clusters. For application in wireless networks, a global clustering output is usually required. Popular examples include Wireless Sensor Networks (WSNs) and Mobile Ad hoc NETworks (MANETs) [12,3,18] whose network structural properties and requirements demand node clustering for efficient operations.

Due to the enormous complexity and potential risks associated with the operation and management of systems nowadays, self-organization is an increasingly important feature for future mobile networks. It applies to WSNs and MANETs whose network operations are based on ad hoc discovery and dynamic routing between network nodes. As self-organization is crucial to the operations of such networks, node clustering, which can be mapped to graph clustering, should also be approached in a self-organizing manner.

In wireless networks, graphs considered for clustering are weighted graphs with interference couplings which are positive real numbers. Nodes in the graph

C. Bettstetter and C. Gershenson (Eds.): IWSOS 2011, LNCS 6557, pp. 92–103, 2011.

consist of transmitters and receivers who communicate with each other, i.e., a transmitter-receiver pair, and each node corresponds to an individual agent in a self-organizing algorithm. Due to the broadcast nature of radio communication, in principle all receivers are disturbed by all transmitters, and the interference graph is in principle *complete*. The graph is, however spatially embedded–the average weight between two nodes decreases with increasing distance.

In MANETs, network *scalability* problems make hierarchical routing more preferred than flat routing. Hierarchical clustering that directly fits the structure of hierarchical routing has been extensively investigated. In principle, the same argument of dynamic routing applies to WSNs. While clusterheads may not be necessary in MANET, they are usually required in WSNs. In WSNs, network lifetime is one of crucial considerations. A clusterhead-based clustering can be more energy efficient by letting clusterheads take charge of inter-cluster transmission and all cluster members to act as clusterhead in turn.

The introduction of Coordinated Multi-Point (CoMP) transmission in mobile networking [1] has found another application of graph clustering in wireless communications. In CoMP transmission, multiple transmitters coordinate their transmission to either avoid interference between transmissions, or to generate a Multi-Input Multi-Output (MIMO) channel between different cellular transmitters and receivers [10]. Since the number of transmitters involved in CoMP transmission cannot go up unlimited, a Coordination Area (CA), in which coordination between transmitters takes place, is defined. Another reason for having CAs is to reduce the burden of interference measurement for mobile devices. This indicates an upper limit on how many transmitters can cooperate, i.e., the CA size, since the size of a CA cannot be larger than the number of measured interferences. The formation of CA is inherently a clustering problem. Below, we elaborate clusterization problems arising from here.

In the context of Self-Organizing Networks (SONs), clustering methods to our concern should be *online* and *iterative*, so that the cluster assignments are made based on only the knowledge of previously encountered information, and the clustering process is continuous and cluster assignments made now can be changed later. Considering network scalability, *distributed* clustering where each node runs the same algorithm with local information ensures the efficiency of clustering process. In addition, since the interaction between farther away nodes is weak in wireless networks, taking into account only local information should not allow significant loss from global information. We assume our clustering process is run *asynchronously* among nodes, i.e., an algorithm is run by one node at one time, to allow proper reaction of each node to the environment. Nevertheless, since interference situation is changing gradually, the clustering should be done within relatively short time compared to the coherent time of mutual interference. It is noted that there should be a working output from the distributed clustering algorithm for each node to follow, for any given time.

The requirements on clustering methods for CA formation problem suggests a clustering method to cut weak interference coupling is to our flavor. This connects CA formation to minimum-cut problems, and therefore maximum-flow problems with the well-known connection with minimum-cut problems [4,9]. However, due to scalability issue, global clustering is in general not preferred

for a very big graph. Secondly, for minimum-cut algorithms, a source and a sink are required for the algorithms to proceed. This is in general hard in our application. In addition, minimum-cut algorithms use divisive clustering, meaning a graph is sequentially divided into partitions. Due to coordination overhead and growing complexity with respect to involved transmitters for CoMP transmission, a big cluster is not practically favored. This indicates a longer delay before a working clustering can be reached by this approach. Similar argument on divisive clustering methods is applied to the famous Kernighan-Lin bisection algorithm [11]. In addition, Kernighan-Lin methods in general divide nodes into two equal-sized subsets. As we have no prior information on the size of the network, special cares are required if bisection methods are used.

With the properties to be fulfilled above, we consider to reach global clustering via local clustering [2,15]. This can be done by, e.g., initiating the procedure multiple times using each vertex as the seed vertex once, and then combine the local clusters into a global clustering according to some quality measure. While some local search procedures, such as hill-climbing and simulated annealing, are required for local clustering algorithms [2,15] because of big cluster size, this can be avoided in concerned CA formation.

In this paper, we study distributed graph clustering with application to CA formation for CoMP transmission, and we classify it as distributed self-organized clusterization. We assume each vertex can learn local information within its neighborhood. The distributed clustering is negotiation-based so that each node can invite other nodes to join its cluster. By exchanging local information, the inviting and invited nodes can negotiate their local clustering. The clustering does not need to be hierarchical since CoMP transmission in each cluster is independent. We do not consider clusterheads election since it is not compulsory in CoMP transmission. In fact, a clustering output with clusterheads will jeopardize the negotiation process since clusterheads, who usually have dominant right, can decide its action regardless of opposing opinions. This leads to local optima easily because of more limited information for local decisions. We note that clustering algorithms in literature do not allow immediate application to our problem from our previous discussion.

As we have mapped a CA formation problem for CoMP transmission in wireless networks to a node clustering problem, we will refer to CA and cluster as the same concept and use them interchangeably. Although the clustering problem is emerged from the original requirement on CoMP transmission, an evaluation of the system performance requires detailed description on CoMP type as there are different variants for CoMP transmission. Due to the space limit, the proposed distributed graph clustering method will be evaluated using common figure of merit in clustering problems.

The rest of the paper is organized as follows. In Section 2, the distributed graph clustering problem is introduced, its connection to CA formation in wireless networks together with the modeling of cellular networks with weighted graphs are discussed. In Section 3, we present the proposed distributed clustering algorithm. In Section 4, we present results concerning the proposed distributed clustering algorithm on interference graphs. Conclusions and future work are summarized in Section 5.

2 System Model

2.1 Distributed Graph Clustering

The graph clustering problem can be formulated as follows. Let a graph $G = G(V, E)$ of vertices $v \in V$ and edges $e \in E$ connecting two vertices be given. The vertices consist, for example, of collections of wireless network elements, which may be interchangeably called nodes, or cells. We assume all nodes use the same resources so that edge $(v, u) \in E$ for every $v \in V$, $u \in V$, and $v \neq u$ (i.e., our graph is complete). To each edge $(v, u) \in E$ there is a corresponding weight $w(v, u) \in \mathbb{R}_+$ indicating the strength of interference from vertex u to vertex v. The objective in graph clustering is to find a clustering so that we have minimum inter-cluster cut, where the sum weight of edges between different clusters is minimized, and the cluster size is limited to less than a fixed number.

In addition to the formulation above, we want to perform the node clustering in a self-organized and distributed manner so that each cell decides which cluster to go for based on local information (gathered either through passive interference observation or through a negotiation protocol) only. In general, these decisions take place inside a routine, which each node executes independently an identical copy of. It is assumed to have access to all the local information necessary, such as the interference caused by the neighboring nodes together with any data the nodes choose to broadcast to their neighbors. Any information sent by a node to its neighbors is assumed to be immediately available to the neighboring nodes, i.e., no transmission delays are modeled.

A common measure for clustering results involves *density* of inter-cluster and intra-cluster. With weighted graph, the definition of density in [16] involves number of edges, in addition to edge weights of a clustering output. Since our graph is complete, we ignore the number of edges and define a density measure as the ratio of summed intra-cluster weights to total weights by

$$Q = \frac{\sum_{\mathcal{C} \in \mathfrak{C}} \sum_{(v,u) \in E, v \in \mathcal{C}, u \in \mathcal{C}} w(v, u)}{\sum_{(s,t) \in E} w(s, t)}, \tag{1}$$

where $w(v, u)$ is the weight of edge (v, u), $\mathfrak{C}$ is the output of clustering, and $\mathcal{C}$ is a cluster in $\mathfrak{C}$.

The motivation of distributed graph clustering with such settings is for application in CoMP transmission in wireless networks. With precious spectrum resources, one should make efficient use of them spatially (i.e., the same frequency bands should be reused in space as often as possible). This makes mutual interference between co-channel users stronger. Our transmission environment would become interference-limited, rather than noise-limited. In an interference-limited environment, raising transmit power does not increase Signal-to-Interference plus Noise Ratio (SINR), which is a crucial performance metric. The introduction of CoMP transmission is to coordinate the transmissions from different transmitters so that the mutual interference does not jeopardize our transmissions.

Optimally, the best solution for CoMP transmission would need cooperation between all transmitters over a whole communication network. However, the complexity for global cooperation is very high and the expected additional gain

of global cooperation over local cooperation is usually limited. Because of this, CoMP transmission is defined to be within a CA. Transmitters within a same CA shall coordinate their transmissions, whereas transmitters of different CAs shall treat each other as interferers. Our distributed graph clustering is to solve the formation of CAs for CoMP transmission in wireless networks.

For a transmitter to learn the interference situation of its receivers, interferences from different interferers are measured at receivers and then reported to the transmitter through a feedback channel. From practical points of view, interference information fed back to a transmitter should be kept to a reasonable amount. In addition, there is a limit on how many interferers one receiver is capable of measuring. If the receiver is a mobile device, this limit can be set so that the interference measurement task is not too big a burden for the mobile device. This hard limit immediately defines a limit on the cluster size.

2.2 Connection to Wireless Networks

The graphs addressed in this paper can be related to snapshots of networks of mobile radio Transmitter-Receiver elements (TRx). Each TRx consists of a transmitter and its corresponding receivers. While a transmitted signal is sent from transmitter in a TRx, the reception is performed by receivers of the TRx. To describe the interference between different TRxs, a statistical number based on the observation of all related receivers is defined. In this paper, the terms TRx and node are used interchangeably.

In a snapshot, each TRx is located at a specific location x_i. From the transmission by each TRx, all others receive a (differently) attenuated version due to the broadcast nature of the radio channel. Being indistinguishable from thermal noise, weak signals can be neglected. The snapshot of the network is a weighted directional graph with edges starting from a TRx and ending in another with the interference powers between the corresponding transmitter and receiver as the edge weights. Typically, these are modeled as consisting of a deterministic, distance-dependent component, and a random component due to shadowing and fading, e.g.,

$$I_{jk} = P_k \|\mathrm{x}_j - \mathrm{x}_k\|^{-\alpha} \xi_{jk}, \tag{2}$$

where I_{jk} is the interference power from node k to node j, P_k is the transmission power of node k, α is the path loss exponent and ξ_{jk} is the random component of the path loss [17], which is correlated in the spatial domain. Typically $\alpha \geq 2$.

From a snapshot of a mobile network, an interference graph can be constructed by collecting a group of closely coordinated elements to a node. Examples of a node in such an interference graph are given by a Base Station (BS) in a cellular network, together with the mobile stations it is serving, or a WLAN Access Point, together with its clients. The edge weights in such graphs are based on statistical numbers describing the statistics of the interference experienced at receivers at a node from transmitters in other nodes, according to Eq. (1). To measure the relevance of interference experienced, it should be compared to the signal level experienced in communications inside the node. In the cellular and WLAN settings, a natural measure of interference experienced at a mobile station/client, is the ratio of the interference power caused by the BS in another

cell to the signal power of the own BS. Both of these quantities would follow (1). Considering a mobile station m at distance d_{om} from its serving BS o, and at distance d_{nm} from an interfering BS n, a measure of the interference power caused by this neighbor to m would be

$$I_{on,m} = (\frac{d_{om}}{d_{nm}})^{\alpha} \frac{\xi_{nm}}{\xi_{om}}, \tag{3}$$

assuming the same transmit power for the interfering and serving BSs.

If there is only one receiver and transmitter per node, the interference coupling between o and n is directly determined by (3). If there are multiple transmitters and/or receivers, a statistical measure of the corresponding interferences (3) for different m served by o should be used. We shall simply use the maximum experienced interference as an interference coupling so that $I_{on} = \max_{m \in \mathcal{S}_o} I_{on,m}$, where $\mathcal{S}_o$ is the set of mobile stations served by BS o. Usually, the strongest BS is chosen to serve a mobile station, so that $I_{on,m} \leq 1$. An important consequence of this model of interference coupling is that generically, the further away an interferer is, the weaker is its interference.

It is assumed here that the interference couplings have been symmetrized leading to symmetric real-value interference network and therefore a symmetric, i.e., undirected, graph with $I_{on} = I_{no}$. Unless guaranteed by other properties of a system, such symmetrization can be achieved by inter-node signaling.

Up till now, an interference matrix I with entries I_{on} describing the interference price between node n and node m can be constructed. The interference prices are used as the weight of edges and therefore $w(o, n) = I_{on}$ when calculating the density measure defined in (1).

2.3 Weighted Graphs Modeling Cellular Networks

As a study case for distributed clustering, we construct a weighted graph describing a network according to typical system simulation prescriptions used for evaluation of wide area cellular networks [5]. Base stations are placed on vertices of a triangular grid, so that the corresponding cells are hexagonal. We consider a single-slope path-loss model (2) with path loss exponent $\alpha = 3.76$. Shadow fading standard deviation is assumed to be 3 dB, and shadowing correlation between cells is 0.5. Note that shadow fading in a wide area cellular network is to take into account the impact of terrain, obstructions etc. on the path loss. We consider an interference limited network, where the cell radius is small enough to render thermal noise irrelevant. In simulations we have used 1 km cell radius.

We uniformly drop 8 users per cell on average to the coverage area of the cellular system. Cell selection is performed for all mobile stations; the serving cell is selected based on the smallest total path loss including the deterministic distance-dependent path loss and shadow fading. The interference coupling is then based on symmetrized worst couplings, i.e., the coupling is the worst of the relative interferences from transmissions of base station n experienced by the users served by o and from transmissions of base station o experienced by users served by n.

Note that in a wireless communication system, a more straightforward interference measure should be Carrier-to-Interference-plus-Noise Ratio (CINR) on the interference links. In 3, the additive while gaussian noise and the interference from other BSs are ignored. The use of (3) here is for better illustration while still capture the essence of CINR.

3 Neighborhood Negotiation with Monte Carlo Method Clustering

Here, we describe our clustering algorithm denoted by Neighborhood Negotiation with Monte Carlo Method (NNMCM). Each node in our problem runs this algorithm in a fully asynchronous manner, indicating that the time it takes for communicating and negotiating with the neighboring nodes and to decide the local clustering are small compared to the time of iteration, which is the period between the time a node decides on its clustering. Due to our application, we assume an initial clustering where each node belongs to a unique cluster. As clustering routine runs, nodes are merged into same cluster so that the total number of clusters decreases. It is also possible that one node decides to stay in one cluster in current iteration but changes to another cluster in later iteration. In the classification proposed in [16], NNMCM is a localized version of agglomerative clustering.

In real-value weighted graph, it is unlikely that we run into a plateau during clustering routine, meaning two local clustering options with same performance measure. However, it is so possible that the clustering process is running toward a local optimum because lacking global information. In [6], a random walk through a graph is introduced to grab an idea on global structure to avoid bad local merging. Lacking the capability of exploring global structure of a graph, we incorporate in NNMCM algorithm a feature of allowing bad merging all the time to enable breakout from local optima. This idea is common in distributed coloring problem on conflict graphs. In the following, we will start by describing a simple distributed clustering algorithm without bad merging, a Greedy Distributed Minimum Cut (GDMC) algorithm. Then, the feature of allowing bad merging in NNMCM is elaborated.

In GDMC, each node executes periodically a routine. From the point of view of one node, we denote the time of executing the routine as active phase, and the rest of the period as passive phase. In active phase, a node can initiate changes to local clustering, and is denoted as an initiator. On the other hand, a node can only react to invitations for changes in passive phase. A node in active and passive phases of GDMC works as follows.

Active Phase

1. If my cluster is full, select one cluster member as kick-out candidate at random.
2. Invite one of neighboring nodes to join my cluster at random.
3. Calculate my new intra-cluster metric by summing up all intra-cluster weights.

4. Exchange new intra-cluster metric with the neighboring node.
5. Calculate sum of intra-cluster metric, q_{new}, of clusters related to this node and the invited node.
6. If $q_{\text{new}} > q_{\text{old}}$, add the neighboring node and remove the kick-out candidate (and the kick-out node forms a single-member cluster temporarily) from my cluster (if any) . Otherwise, do nothing.

Passive Phase

– Upon receipt of an invitation.

1. Calculate intra-cluster metric of my cluster without me by summing up all intra-cluster weights.
2. Exchange intra-cluster metric with the neighboring node.
3. Calculate sum of intra-cluster metric, q_{new}, clusters related to this node and the node that sends out the invitation.
4. If $q_{\text{new}} > q_{\text{old}}$, accept the invitation. Otherwise, do nothing.

– Upon receipt of kick-out request, leave original cluster and form a single-member cluster.

A new decision can be made after the initiator exchanges interference metric with the invited node. It is noted that as both the initiator and the invited node have the same information after exchanging interference metric, the decision made is also consistent. However, the kick-out candidate has to be informed if the decision is to remove it from its current cluster.

From the last rule of active phase, the clustering based on GDMC is improving all the time. The fact that GDMC gets stuck in the global optimum is a desirable feature; once a good clustering is found, discarding that clustering does not make sense. It is also favored to look for improvement in each iteration for quick convergence. On the other hand, GDMC does get stuck in local optima from which the progress to the global optimum is unlikely.

The clustering output of GDMC is evidently affected by which nodes get to run the routine earlier. In other words, the ordering of nodes impact on the clustering results much. By setting up an ordering, one almost at the same time decides which local optimum the graph clustering will go. To avoid this, one of course can run the routine with different node orderings for many times and pick the best. However, this kind of search for optimal node ordering requires global mechanism to some degree, which apparently conflicts with our definition of distributed clustering.

To remedy being stuck in local optima, one should add the possibility of making bad decisions. However, simply adding such decisions does not work satisfactorily since then the global minimum would no longer be an absorbing state. We address this issue in NNMCM algorithm. It is the same as GDMC except the last step in active phase and the last step of upon receipt of an invitation in passive phase. These steps are modified as follows.

Active Phase

6. If $q_{\text{new}} > q_{\text{old}}$, add the neighboring node and remove the kick-out candidate (and the kick-out node forms a single-member cluster temporarily) from my

cluster. Otherwise, with probability p add the neighboring node and remove the kick-out candidate (and the kick-out node forms a single-member cluster temporarily) from my cluster, and with probability $1-p$ do nothing.

Passive Phase

– Upon receipt of an invitation.
  4. If $q_{\text{new}} > q_{\text{old}}$, accept the invitation. Otherwise, with probability p accept the invitation, and with probability $1-p$ do nothing.

If $p = 0$, the method corresponds exactly to GDMC. When $0 < p < 1$ however, the method is able to do bad decisions locally and end up in better local and global state than when only minimizing local interference.

In this paper, we use the following definition for p

$$p = \exp\left(\beta(q_{\text{new}} - q_{\text{old}})\right). \tag{4}$$

It is noted that in (4), p is a variable depending on metric difference $(q_{\text{new}} - q_{\text{old}})$ rather than a fixed number. The bigger the metric difference, the lower p is. The rationale behind this is that the smaller the metric difference $(q_{\text{new}} - q_{\text{old}})$ is, the more likely it is to jump over the barrier to find another optima. This idea has been applied in e.g., simulated annealing [8], although in a slightly different manner. Note that we have GDMC scheme when $\beta = \infty$.

The performance of NNMCM is comparable to that of GDMC with the important difference that NNMCM converges to the global optimum even in difficult cases when GDMC does not. The convergent speed of both algorithms are similar. Both of them converges within few iterations (e.g., around $10-20$ in our simulation cases) where one iteration is defined as the time it takes for each node in network to run the corresponding routines once. This indicates a linear scalability of complexity and network size.

4 Simulation Results

Here, we evaluate the efficiency and performance of NNMS algorithm to solve distributed interference graph clustering problem. The figure of merit is the interference density measure defined in (1). With CoMP transmission, only interference generated by nodes residing in same cluster could be canceled. Therefore, the interference density measure reflects a percentage of total interference that could be potentially eliminated. This is the figure of merit most pertinent for wireless network operation. Complete interference graphs in systems with 400 nodes in a rectangular 20×20 triangular grid have been constructed according to the principles of Section 2.3. Edge weights are calculated using (3). We assume a fixed limit on cluster size of 3.

Fig. 1 illustrates the relationship between converged density measure Q and parameter β for one realization of interference graph. A higher β indicates a lower probability of making bad decisions. With very small β, bad decisions are taken all the time and thus, disappointing Q are observed. Density measure Q is improved as β increases, until it hits the maximum achieved by β value around 5. It is noted that Q converges to a constant for $\beta > 10$. When $\beta > 10$, the

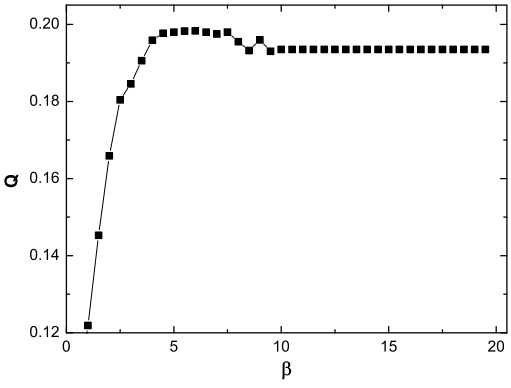

Fig. 1. Density measure Q with respect to β in NNMCM algorithm

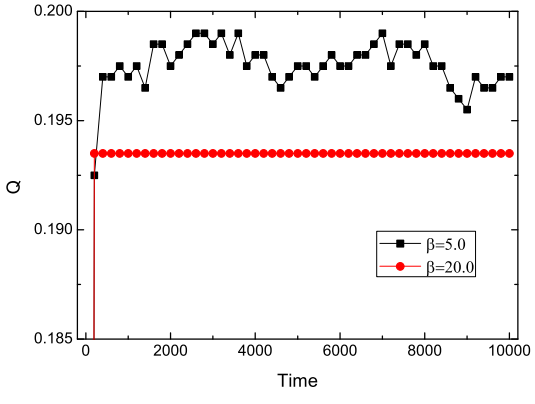

Fig. 2. Illustration of convergence time for NNMCM($\beta = 5.0$) and GDMC($\beta = 20.0$) algorithms

probability of taking bad decisions is low enough so that it does not affect the results in sensible way, and therefore can essentially be treated as the same as taking $\beta = \infty$.

Fig. 2 presents convergence behavior for NNMCM and GDMC algorithms. We use $\beta = 5.0$ and $\beta = 20.0$ for NNMCM and GDMC algorithms, respectively. The selected β values is justified by Fig. 1. The convergence time of both algorithms are comparable, with NNMCM showing better results due to its capability of escaping from local optima. Using GDMC, not any kind of random walk among local optima is possible, so that the whole network converges to a local optimum quickly and never escape from it. It is noted that the time unit here is the time period it takes for all nodes to run the corresponding routine one times for the whole network.

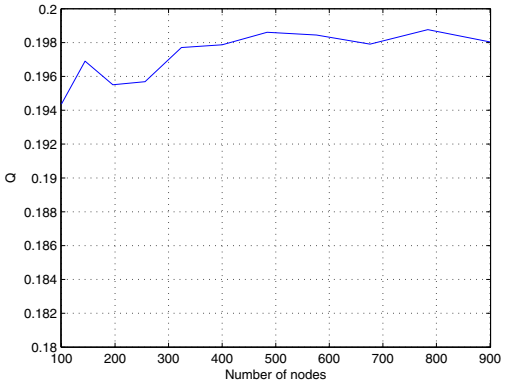

Fig. 3. Density measure Q with respect to network size in NNMCM algorithm. For each network size, optimized β value has been used.

Fig. 3 shows the relation between performance of NNMCM and network size. For each network size, the best numerically-searched β value is used. From Fig. 3, the converged density measure Q is stable with respect to network size, indicating NNMCM algorithm is robust to network scalability.

5 Conclusions

In this paper, we studied weighted graph clustering in a distributed manner. We have attempted to find the minimum cut clustering in a distributed manner, with restriction on cluster size. Due to a stated application scenario in CoMP transmission of wireless communication, the clustering algorithm is also required to provide a working clustering output in any moment.

The proposed Neighborhood Negotiation with Monte Carlo Method (NNMCM) clustering is capable of escaping from local optima by allowing a bad decision-making mechanism. The probability of making bad decision is adaptive to the depth of current local optimum. Our results show comparable efficiency of NNMCM to a Greedy Distributed Minimum Cut (GDMC) clustering where no local optima escaping mechanism is available. In addition, NNMCM algorithm outputs clustering results which show higher interference density measure than GDMC clustering. A linear scalability between complexity and network size is also featured in NNMCM.

For practical applications in CA formation problems, the proposed distributed clustering problem should be evaluated in a dynamic network where part of the entries of the interference matrix between nodes may change from time to time. In such a dynamic environment, another important issue would be how fast can the clustering method adapts to a good enough clusterization. In addition, it is likely that there are multiple radio resource shifts in the network, with each of the resource shift requiring an individual CA solution. Mobile stations in one cell need to be grouped to different resource shifts before the interference matrices

for different resource shifts can be constructed. This indicates the CA formation is coupled with the grouping of mobile stations, which is a challenging task. This is left for future work.

References

1. 3GPP: Feasibility study for further advancements of E-UTRA (LTE-Advanced). Tech. Rep. TR 36.912 (2009)
2. Clauset, A.: Finding local community structure in networks. Phsical Review E 72(2), 26132 (2005)
3. Deosarkar, B., Yadav, N., Yadav, R.: Clusterhead selection in clustering algorithms for wireless sensor networks: A survey, pp. 1–8 (December 2008)
4. Elias, P., Feinstein, A., Shannon, C.E.: Note on maximum flow through a network. IRE Transactions on Information Theory IT-2, 117–119 (1956)
5. 3rd Generation Partnership Project; TSG RAN: Physical layer aspects for evolved universal terrestrial radio access (UTRA). Tech. Rep. TR 25.814 (September 2006)
6. Harel, D., Koren, Y.: On clustering using random walks. In: Hariharan, R., Mukund, M., Vinay, V. (eds.) FSTTCS 2001. LNCS, vol. 2245, pp. 18–41. Springer, Heidelberg (2001)
7. Hopcroft, J., Khan, O., Kulis, B., Selman, B.: Natural communities in large linked networks. In: KDD 2003: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 541–546. ACM, New York (2003)
8. Johnson, D.S., Aragon, C.R., McGeoch, L.A., Schevon, C.: Optimization by simulated annealing: an experimental evaluation. part i, graph partitioning. Oper. Res. 37, 865–892 (1989)
9. Ford Jr., L., Fulkerson, D.: Maximum flow through a network. Canadian Journal of Mathematics 8, 399–404 (1956)
10. Karakayali, M.K., Foschini, G.J., Valenzuela, R.A.: Network coordination for spectrally efficient communications in cellular systems. In: IEEE Wireless Commun., pp. 56–61 (August 2006)
11. Kernighan, B.W., Lin, S.: An Efficient Heuristic Procedure for Partitioning Graphs. The Bell System Technical Journal 49(1), 291–307 (1970)
12. Kulkarni, V., Forster, A., Venayagamoorthy, G.: Computational intelligence in wireless sensor networks: A survey. Communications Surveys Tutorials, IEEE PP(99), 1–29 (2010)
13. Newman, M.E.J.: Fast algorithm for detecting community structure in networks. Physical Review E 69, 066133 (2004)
14. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Physical Review E 69(026113) (2004)
15. Schaeffer, S.E.: Stochastic local clustering for massive graphs. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 354–360. Springer, Heidelberg (2005)
16. Schaeffer, S.E.: Graph clustering. Computer Science Review 1, 27–64 (2007)
17. Stüber, G.L.: Principles of Mobile Communication (1996)
18. Yu, J., Chong, P.: A survey of clustering schemes for mobile ad hoc networks. IEEE Communications Surveys Tutorials 7(1), 32–48 (2005)