SDC: A Distributed Clustering Protocol for Peer-to-Peer Networks

Yan Li[1], Li Lao[2], and Jun-Hong Cui[1]

[1] Computer Science & Engineering Dept., University of Connecticut, CT 06029
[2] Computer Science Dept., University of California, Los Angeles, CA 90095
yan.li@uconn.edu, llao@cs.ucla.edu, jcui@engr.uconn.edu

Abstract. Network clustering can facilitate data discovery and peer-lookup in peer-to-peer systems. In this paper, we design a distributed network clustering protocol, called SCM-based Distributed Clustering (SDC), for peer-to-peer networks. In this protocol, clustering is dynamically adjusted based on Scaled Coverage Measure (SCM), a practical clustering accuracy measure. By exchanging messages with neighbors, peers can dynamically join or leave a cluster so that the clustering accuracy of the whole network is improved. SDC is a fully distributed protocol which requires only neighbor information, and it can handle node dynamics locally with very small message overhead while keeping good quality of clustering. Through extensive simulations, we demonstrate that SDC can discover good quality clusters very efficiently.

1 Introduction

In a peer-to-peer system, there are usually large numbers of peers. And the knowledge of each peer about the network topology is usually limited to its immediate neighbors. Due to the large scale and the lack of knowledge about the complete network structure in each peer, a main challenge in peer-to-peer system design is to effectively perform data discovery and peer look-up. The *network clustering* technique can significantly facilitate these operations [1] [2].

Network clustering is the procedure of partitioning a network topology into groups or clusters. It can be performed in both centralized and distributed ways. Centralized network clustering is an off-line procedure, in which complete network topology information need to be obtained before clustering. In our work, we focus on the latter one. We are interested in the network clustering of large-scale peer-to-peer networks.

There are several characteristics of a good distributed clustering protocol. First of all, as a natural requirement of network clustering, nodes in the same clusters should be highly connected, and less connected between clusters. Secondly, the protocol should well control the cluster size (or cluster diameter). Thirdly, the protocol should result in a minimum number of "orphan" nodes. Lastly, a good clustering protocol should take node dynamics into account, since the target networks (peer-to-peer networks) are highly dynamic with frequent entry and exit of nodes.

In the literature, there have been considerable research efforts addressing the problem of network clustering, but very few of them studied the problem of clustering in peer-to-peer networks. Among the existing approaches, MCL [5] is well accepted as an efficient and accurate network clustering algorithm. However, this approach assumes that the complete network topology is available at one central point, which is not realistic in peer-to-peer systems. CDC [4], on the other hand, is a distributed algorithm. It forms clusters based on node connectivity. The main issue with this algorithm is that it can not handle node dynamics in a decent way, as limits its utility in peer-to-peer networks.

With these problems in mind, we design a novel network clustering protocol called **SCM-based Distributed Clustering (SDC)**, which satisfies all the design criteria discussed above. In this protocol, clustering is dynamically adjusted based on Scaled Coverage Measure (SCM) [6], a practical clustering accuracy measure. By exchanging messages with neighbors, peers can dynamically join or leave a cluster so that the clustering accuracy of the whole network is improved. To control the cluster size, TTL (Time-To-Live) is piggybacked in exchange messages to guarantee the cluster diameter will never exceed a predefined threshold. SDC is a fully distributed protocol which requires only neighbor information, and it can handle node dynamics locally with very small message overhead while keeping good quality of clustering. Through extensive simulations, we demonstrate that our proposed protocol, SDC, is able to discover good quality clusters in a very efficient way.

2 Network Model and Scaled Coverage Measure

Network Model We assume each peer-to-peer network is represented by a connected, undirected graph $G = (V, E)$, where V is the set of nodes corresponding to the set of peers in the system and E is the set of links, which are the logical connections between peers. We denote $|V| = n$ and $|E| = m$. Then the partition $\mathcal{C} = \{C_1, C_2, ..., C_l\}$ of V is called a *clustering* $\mathcal{C}$ of graph G, and C_is are called *clusters*. Each cluster should be a non-empty subset of V. Obviously, $\bigcup_{i=1}^{l} C_i = V$. The *diameter* of a cluster C_i is defined as the maximum length of the shortest paths among all pairs of nodes in C_i. Then if a cluster has only one node, it has a diameter of 0. We call the clusters with diameter 0 as *orphan nodes*. In this paper, we also define *cluster size* as the number of nodes in a cluster to represent the cluster scope. Clearly, cluster size and cluster diameter are closely related. In most context, "control cluster size" and "control cluster diameter" have the same meaning of "control cluster granularity". We only differentiate these two concepts in the protocol description.

SCM is a practical measure to evaluate the accuracy of connectivity based clustering proposed by S.Van Dangon [5]. We assume $\mathcal{C} = \{C_1, C_2, ..., C_l\}$ is a clustering on network $G = (V, E)$. Given a node $v_i \in V$, we have the following notations: $\mathbf{Nbr}(v_i)$ is the set of neighbors of node v_i; $\mathbf{Clust}(v_i)$ is the set of nodes in the same cluster as node v_i (excluding v_i); Then, two special sets of nodes associated with v_i are defined as follows: $\mathbf{FalsePos}(v_i, \mathcal{C})$ is the set of nodes in the same cluster as v_i but not neighbors of v_i; **FalseNeg**$(v_i, \mathcal{C})$ is the set of neighbors of v_i but not in the same cluster as v_i. The SCM of node v_i is defined as follows:

$$SCM(v_i) = 1 - \frac{\mid FalsePos(v_i, \mathcal{C}) \mid + \mid FalseNeg(v_i, \mathcal{C}) \mid}{\mid Nbr(v_i) \cup Clust(v_i) \mid}. \quad (1)$$

For graph G, $SCM(G)$, is defined as the average of the SCM values of all the nodes, that is, $SCM(G) = (\sum_{v_i} SCM(v_i))/n$, which lies in $[0, 1]$.

SCM well reflects the significance of clustering features in a given network. First of all, it is easy to see that the higher the SCM, the smaller the connectivity between clusters and the higher the connectivity within clusters. For graphs containing only isolated clusters/subgraphs that are themselves fully connected, the SCM value is 1. Secondly, for any graph, there exists a highest SCM value which is determined solely by the network structure. If the network does not contain significant clustering substructures, this highest "available" SCM value can be very small. However, if we evaluate two clustering techniques on the same network, the one which results in a higher SCM value discovers more accurate clustering substructures than the one with smaller SCM value, although both resultant SCM values could be very small. Lastly, the SCM value of an orphan node is 0, which matches our goal of minimizing the number of orphan nodes.

Based on the definition of SCM, the network clustering problem can be simplified as partitioning a network topology so that its SCM is maximized. Our proposed SDC protocol exactly follows this idea, adaptively forming clusters in an aggressive manner.

3 The SDC Protocol

3.1 Protocol Description

The SDC protocol performs in a fully distributed way. Each node v_i only needs to maintain some basic information about its neighbors and the cluster it belongs to, such as the cluster id *clust_id*, the cluster size *clust_size* (which is the total number of nodes in the cluster).

Given a network, each node v_i is initialized as an orphan node with its own *clust_id* (any unique *id* is sufficient) and *clust_size* (1 in this case). Then all nodes start to exchange messages with their neighbors, conduct some simple computation, and form clusters in a greedy manner. After a number of rounds of communication, the clustering procedure becomes stable without further message exchange and the network is finally clustered.

In SDC, we define a set of **Clust_** type of messages. Suppose node v_i wants to be clustered. The following clustering messages may be involved.

- **Clust_Probe**. Node v_i first sends the message *Clust_Probe* to every node $v_j \in Nbr(v_i)$ to find out other clusters in the neighborhood. Each node which receives *Clust_Probe* will send its *clust_id* and *clust_size* back to v_i.

- **Clust_Request**. Once receiving the *clust_ids* from its neighbors, node v_i can determine its "neighbor clusters". Suppose v_i discovers that a cluster Cl is connected with it, it issues a *Clust_Request* message which is flooded in Cl and v_i's current cluster $Clust(v_i)$. This is a well-controlled flooding, since upon receiving *Clust_Request*, a node can forward this message to others only if it is in Cl or $Clust(v_i)$. For any node v_j in cluster Cl, upon receiving *Clust_Reqest*, a very simple computation is performed to obtain $\Delta SCM(v_j)$, the gain in $SCM(v_j)$ assuming node v_i joins Cl. This computation only requires the information of whether v_i is v_j's neighbor or not. Similarly, for any node $v_k \in Clust(v_i)$, it needs to compute $\Delta SCM(v_k)$ as if v_i leaves its current cluster.

  To control the number of exchanged messages, a TTL is carried in *Clust_Request*. Once receiving *Clust_Request*, any node should check the TTL value first and will discard the message without forwarding to others if TTL expires. TTL is also used to control the cluster diameter.
- **Clust_Reply**. Upon receiving *Clust_Request* from v_i ($TTL \neq 0$), node v_j sends back a *Clust_Reply* message carrying $\Delta SCM(v_j)$ and v_j's *clust_id* back to node v_i.
- **Clust_Reject**. Based on the TTL in *Clust_Request*, node $v_j \in Cl$ can determine whether or not the cluster diameter will exceed the predefined threshold due to the joining of node v_i. If this is the case ($TTL = 0$), v_j simply stops forwarding *Clust_Request* to other nodes and a *Clust_Reject* message will be sent back to v_i. Once receiving *Clust_Reject*, node v_i will not join Cl.
- **Clust_Update**. After node v_i receives *Clust_Reply* messages from all the nodes in its current cluster and the neighbor cluster Cl (in the case that no *Clust_Reject* is received from Cl), it computes the overall gain $\Delta SCM(G)$ based on the received information, assuming it leaves its original cluster and joins Cl. If $\Delta SCM > 0$, v_i should join Cl. Once v_i determines which cluster to join, a *Clust_Update* message containing v_i's node id and its original *clust_id* is flooded in its original cluster and the new cluster it will join. Then, v_i and any node receiving this message will update the *clust_size* and their own SCM.

After node v_i joins the new cluster, its neighbors in the original cluster are affected and should check whether they should join other clusters, in the same way as node v_i does. The whole procedure will end if no node can join any cluster based on $\Delta SCM(G)$ and the cluster diameter control.

3.2 Handling Node Dynamics

Peer-to-peer networks are dynamic systems. With node entry and exit at arbitrary points, the network structure is changed and the existing clusters are affected. Re-do the whole clustering procedure may keep good clustering accuracy. However, it is very inefficient and the procedure may never stabilize if node entry and exit happens frequently. Therefore, designing an effective and efficient scheme to handle node dynamics is critical in peer-to-peer network clustering.

Our SDC protocol can naturally handle node dynamics. Whenever a new node v_i joins the system, it is first initialized as an orphan node and gets its own *clust_id* (any unique *id* is sufficient) and *clust_size* (which is 1). Since the network structure between node v_i and its neighbors is changed, a **Join** message carrying v_i's *clust_id* is issued by v_i to all of the neighbors so that they can update their

SCM. As v_i's joining changes its neighbors' connectivity, the affected neighbor nodes should perform a new round of clustering procedure. When a node wants to leave, it sends a **Leave** message to each of its neighbors as well as every other node in its cluster through flooding so that the *clust_size* and SCM values of the affected nodes can be updated. This will also activate a new round of clustering procedure at these affected nodes. The idea behind this scheme comes from the fact that node entry and exit are localized events and only a few nodes are affected and need to be re-clustered.

It is clear that some overhead is introduced when SDC handles node dynamics. Nevertheless, this overhead is very small since only neighbors and/or the nodes in the same cluster are directly affected. In next section, we will show that SDC can achieve very good clustering accuracy while with low overhead in the presence of node dynamics. In contrast, CDC has to re-do the complete clustering procedure for any node join or leave in order to maintain good clustering accuracy, which introduces a lot of overhead.

3.3 Simulation Evaluations

In this section, we conduct simulations to evaluate the performance of SDC, comparing it with CDC, in dynamic systems.

Experiment Settings We implement both the SDC and CDC algorithms and run them on different topologies. The configurable parameters used in the CDC scheme are carefully tuned so that we can get the best results for CDC. For implementation details, please refer to our technical report [3]. We use two metrics: *clustering accuracy* and *message overhead*. We compute the clustering accuracy using SCM, and measure the overhead in term of the number of exchange messages between peers.

Results and Analysis In this set of experiments, we use power-law topologies. We fix the average degree as 10, and vary the topology size (i.e., the number of nodes) from 200 to 5000. We run each experiment more than 100 times so that all the results have a standard deviation of less than 0.1%. We measure the message overhead and clustering accuracy for arbitrary node join (and leave).

We first study node leaving. In SDC, when a node leaves the network, the affected nodes (its neighbors and the nodes in the same cluster) need to "re-cluster" in order to maintain good clustering accuracy. In CDC, upon a node entry or exit, the whole network has to be re-clustered. For comparison, we also run "SDC Reclustering", in which the whole topology redoes SDC clustering after each node exits the network. The results are plotted in Fig. 1 and Fig. 2. We observe that SDC can maintain a higher clustering accuracy than CDC while only much smaller overhead is introduced. Moreover, compared with "SDC Reclustering", SDC yields almost same accuracy values, which further demonstrates the effectiveness of SDC for node leaving. We conduct similar experiments for node joining, and obtain similar results. Thus, we conclude that SDC can handle node dynamics very effectively.

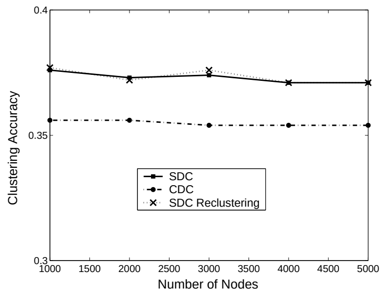

Fig. 1. Clustering accuracy on node exit

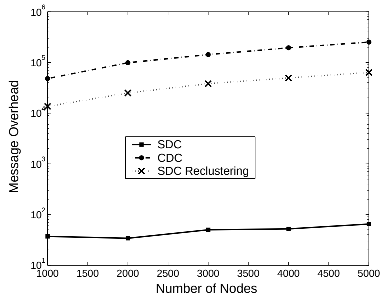

Fig. 2. Message overhead on node exit

Besides the performance evaluation of SDC in dynamic systems, We also study the influence of node degree and TTL on the performance of SDC. Due to space limit, we do not show those results in this paper. Interested readers can find the complete simulation study in [3].

4 Conclusion Remarks

We have presented a distributed clustering protocol, SDC, for peer-to-peer networks. SDC can satisfy all the criteria for a good clustering algorithm: it considers node connectivity; it well-controls the cluster size; it minimizes the number of orphan nodes; and it can locally handle node dynamics with small overhead. Through simulations, we demonstrate that SDC can achieve much better performance than CDC in terms of both clustering accuracy and message overhead.

References

1. L. Garces-Erice, E. W. Biersack, K. W. Ross, P. A. Felber, and G. Urvoy-Keller. Hierarchical p2p systems. In *Proceedings of ACM/IFIP International Conference on Parallel and Distributed Computing (Euro-Par)*, 2003.
2. G. Kwon and K. D. Ryu. An efficient peer-to-peer file sharing exploiting hierarchy and asymmetry. In *SAINT*, pages 226–233, 2003.
3. Y. Li, L. Lao, and J.-H. Cui. SDC: A distributed clustering protocol for peer-to-peer networks. *UCONN CSE Technical Report: UbiNet-TR06-02*, February 2006.
4. L. Ramaswamy, B. Gedik, and L. Liu. A distributed approach to node clustering in decentralized peer-to-peer networks. *IEEE Transactions on Parallel and Distributed Systems*, 16(9), Sept. 2005.
5. S. van Dongen. A new cluster algorithm for graphs. *Technical report INS-R9814, Centrum voor Wiskunde en Informatica (CWI), ISSN 1386-3681*, Dec. 1998.
6. S. van Dongen. Performancde criteria for graph clustering and markov cluster experiments. *Technical report, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam*, 2000.