

Looking Around First: Localized Potential-Based Clustering in Spontaneous Networks

Fehmi Ben Abdesslem, Artur Ziviani, Marcelo Dias de Amorim, and Petia Todorova

Abstract—We propose a new budget-based clustering algorithm for self-organizing networks. The basic idea behind our solution is that nodes first sense the environment using inherent Hello packets before starting forming clusters. In contrast with previous solutions that blindly distribute the budget to nearby nodes, our Potential-Based Clustering algorithm applies a proportional budget distribution based on the connectivity degree of the nodes. This approach matches the principle that nodes in real networks are not uniformly distributed. Our simulation results show that the proposed approach outperforms previous ones.

Index Terms—Networks, Distributed algorithms, Clustering methods.

I. INTRODUCTION

Among the multiple existing techniques for achieving nice scalability properties in self-organizing networks, clustering is perhaps the one that has received the most attention from the research community [1]. In this letter, we are interested in investigating a particular class of clustering techniques called budget-based clustering [2]. This approach has the advantage of producing bounded-size clusters, in contrast with many other existing solutions ([3], [4], [5], [6]). In budget-based clustering algorithms (further detailed in Section II), an initiator node (the cluster-head) is assigned an initial budget. The cluster-head starts then to distribute the budget among its neighbors, which in turn do the same until the budget is exhausted. By controlling the allocated budget for cluster formation, one is able to control the intended cluster size, i.e., the upper bound on the cluster size.

Producing bounded clusters whose sizes are close to the allocated budget is a key optimization issue. Furthermore, obtaining clusters of a predefined size is a desired characteristic, especially in networks where bandwidth is the main performance bottleneck. The problem with previous budget-based clustering algorithms, such as the Rapid and Persistent ones [2] (also refer to Section II for further details), is that they either do not achieve this optimization or generate too much message overhead in doing so. This is fundamentally because nodes adopting these algorithms for budget allocation disregard the nature of neighbors.

In this letter, we propose a new budget-based clustering algorithm that care about the environment that nodes are surrounded by in order to avoid blind budget distribution among

neighbors. This algorithm, called *Potential-Based Clustering*, or *PBC*, aims at producing bounded clusters closer to the allocated budget. Basically, nodes distributing a budget apply a weighted assignment policy that considers the neighbors' connectivity degree. We propose two variants of our algorithm, namely PBC-simple and PBC-persistent. The first one is based on the Rapid algorithm, whereas the second is based on Persistent algorithm. Through simulations with random topologies, we show that both versions provide better performance than the algorithm they are based on, producing clusters that are closer to the initial budget.

II. BUDGET-BASED CLUSTERING

Krishnan and Starobinski [2] propose two algorithms for budget-based clustering: Rapid and Persistent. In the Rapid algorithm, the initiator node A is assigned a budget β_A , of which it accounts for itself and distributes $\beta_A - 1$ to its neighbors. These neighbors do the same and so on until the budget is exhausted. If a node receives a budget that it cannot propagate because there are no neighbors in its subtree, this node simply discards its extra budget. The overhead generated is then minimum, but it may result in clusters much smaller than the initial budget would allow.

As a consequence of such an observation, a second budget-based algorithm is proposed [2]. This algorithm, called Persistent, aims at obtaining cluster sizes as close as possible to the initial budget. When a node receives a message, it does not immediately send an acknowledgment to the initiator. When either the budget is met or when further growth is not possible, an acknowledgment is sent. In the case the budget cannot be completely distributed (e.g., because there are not enough neighbors), the node sends an acknowledgment indicating the exceeding budget. The initiator tries then to redistribute those remaining tokens. The algorithm terminates either when the budget is completely distributed or when no further growth is possible. In summary, the persistent algorithm tries to distribute the tokens by browsing the whole tree and seeking for available nodes.

III. POTENTIAL-BASED CLUSTERING

Both Rapid and Persistent algorithms apply a blind method, i.e., nodes distributing budgets disregard the nature of their neighbors. This leads to wasting tokens distributed to nodes without enough available neighbors, whereas other nodes still have neighbors available to join the cluster.

In this context, we propose a new methodology to obtain a more strategic distribution of the allocated budget. We assume that nodes exchange Hello packets and, most importantly,

Fehmi Ben Abdesslem and Marcelo Dias de Amorim are with the LIP6/CNRS Laboratory of the Université Pierre et Marie Curie – Paris 6, France. Emails: {fehmi,amorim}@rp.lip6.fr. Artur Ziviani is with the National Laboratory for Scientific Computing (LNCC), Brazil. Email: ziviani@lncc.br. Petia Todorova is with Fraunhofer FOKUS, Germany. Email: Petia.Todorova@fokus.fraunhofer.de

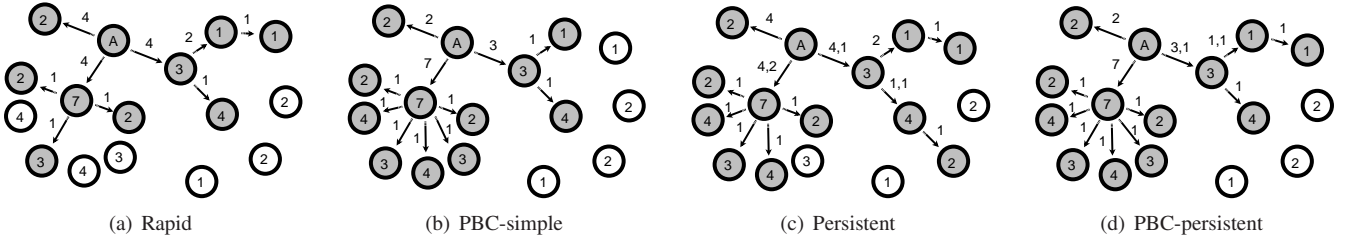


Fig. 1. Example of topology comparing the different algorithms. Flooding is initiated by node A . Links with multiple values are subject to multiple consecutive distributions and the label of each node corresponds to its potential.

learn from each other using these packets. The algorithm we propose in this paper relies on the following assumption.

Assumption 1: If node A has more neighbors than node B , then the subtree associated to A is likely to be larger (in numbers of nodes) than the subtree associated to B .

We call the number of neighbors of a node the *potential* of this node. Thus, relying on Assumption 1, we consider that the number of neighbors of a node is likely to be proportional to its distribution potential. Every node maintains a database containing the potentials of its neighbors and uses this database to determine the amount of budget assigned to each one of them. For any node X in the network, let $\mathbf{N}(X) = \{x_1, x_2, \dots\}$ denote the set of X 's neighbors and $|\mathbf{N}(X)|$ be the size of this set. The potential of node X , denoted $\pi(X)$, is then given by:

$$\pi(X) = |\mathbf{N}(X)|. \quad (1)$$

Let A be a node handling some budget β_A to be distributed among its neighbors. Node A first accounts for itself and then distributes the remaining tokens to its neighbors based on their potentials – a neighbor with higher potential receives more tokens than a neighbor with lower potential. The amount of budget assigned to neighbor $a_i \in \mathbf{N}(A)$ is thus:

$$\beta_{a_i} = \left\lfloor \frac{(\beta_A - 1)\pi(a_i)}{\sum_{j=1}^{|\mathbf{N}(A)|} \pi(a_j)} \right\rfloor. \quad (2)$$

We show through simulations in Section IV that this assumption is verified frequently enough to lead to good results in random topologies.

From Eq. 2, we see that some tokens may be unassigned if $(\beta_A - 1)$ is not a multiple of $\sum_{j=1}^{|\mathbf{N}(A)|} \pi(a_j)$. In this case, the remaining tokens can be evenly distributed among the β_r neighbors with the lowest potentials. We define β_r as:

$$\beta_r = (\beta_A - 1) - \sum_{j=1}^{|\mathbf{N}(A)|} \beta_{a_j}. \quad (3)$$

For the β_r lowest-potential neighbors, node A performs $\beta_{a_i} \leftarrow \beta_{a_i} + 1$ before distributing the budget. This guarantees that no tokens are wasted. Experimentally, we observed that attributing the remaining tokens to lowest-potential neighbors provides better performance to PBC algorithms. Without this

design choice, a low-potential node frequently remains without any tokens until it initiates its own cluster. However, at this point, no neighbors are then available to accept the tokens. In other words, the tokens propagate through the topology avoiding low-potential nodes and clustering highest-potential ones, which results in small clusters in low-density areas and excessive lost tokens in high density areas. Distributing the remaining tokens to lowest-potential nodes attenuates this counterproductive phenomenon by giving more chances to low-potential nodes to join growing clusters.

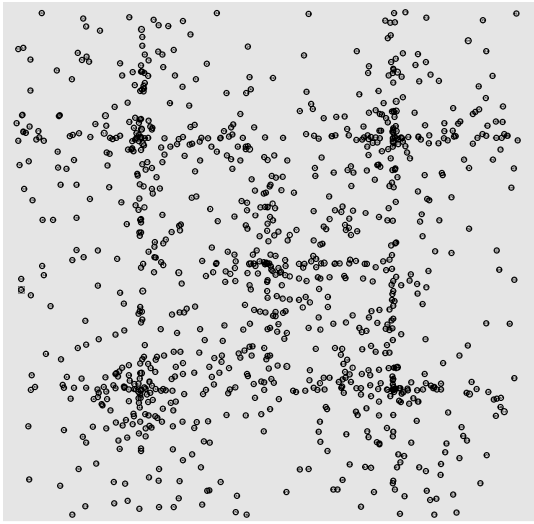
The same distribution scheme is applied by the nodes in the subtrees until the budget is exhausted or no further growth is possible. Both PBC algorithms use this method to distribute tokens. The difference is that in the PBC-simple algorithm, if a node is still assigned with extra tokens, the exceeding tokens are lost, as they are in the Rapid algorithm. As for the PBC-persistent algorithm, the exceeding tokens are sent back to the distributor and assigned to an available node if possible, using the same mechanism as Persistent.

As a basic example, we propose a simple topology on which we execute the 4 algorithms, with an initial budget of $\beta = 13$. Figure 1(a) presents the Rapid algorithm running on this topology. Observe that only 10 nodes joined the cluster initiated by A . PBC-simple optimizes this algorithm, reaching 12 joining nodes, as represented in Figure 1(b). Figure 1(c) illustrates the Persistent algorithm. The second value on some links represents the number of exceeding tokens assigned in a second distribution. Figure 1(d) shows that PBC-persistent provide the same maximal cluster size than Persistent, but it tries to assign tokens proportionally to the degree of connectivity of each neighbor. In this way, the remaining 3 nodes are less sparse and can produce another cluster. On the contrary, Persistent produces more frequently isolated nodes on the leaves which are likely to produce separated clusters, lowering the average size of the clusters.

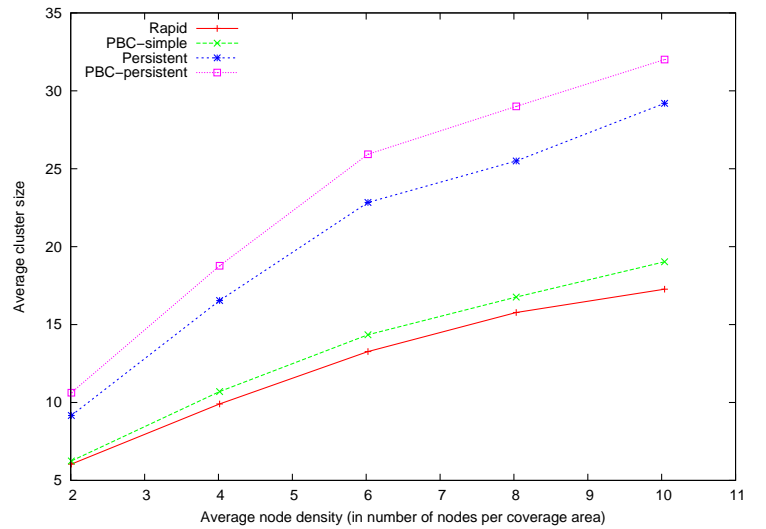
To consider the potential of a neighbor more deeply, we use in our algorithm the notion of aggregate potential, which corresponds to the potential plus the sum of the neighbors' potentials.

IV. PERFORMANCE ANALYSIS

To compare the different algorithms, we performed simulations over random topologies. The simulator used is NS-2 [7], and the topologies were generated on a square surface of 2000×2000 meters, with a radio range of 100 meters for each node. The number of nodes ranges from 250 to 1300 in



(a) Example of random 1000-node topology generated by the simulator.



(b) Average cluster size depending on the node density.

Fig. 2. Simulation results over random topologies.

order to simulate different average node densities. For a more realistic topology, we fixed some high-density areas through the topology, as illustrated in Figure 2(a). Previous works (e.g., [8], [9], [10]) consider such heterogeneous topologies. For each average node density, a new random topology was generated. The algorithms were compared on a same topology for each average node density.

The efficiency of a budget-based clustering algorithm is evaluated according to the average cluster size produced. The best case would be to get cluster sizes equal to the initial budget. Nevertheless, the token propagation does not always allow reaching such a size, and the average cluster size is lower than this value in general. Nevertheless, the closer this average size to the initial budget, the more efficient the budget-based clustering algorithm. Figure 2(b) presents the average size of clusters produced by each algorithm depending on the average node density with a budget set at $\beta = 50$. Each presented value corresponds to an average obtained through 6 simulations. The Rapid algorithm is the less efficient, as expected. This is due to the blind distribution and the tokens lost by nodes without enough neighbors available. Using the same approach, our algorithm PBC-simple provides better results by distributing tokens according to the potentials, i.e., avoiding token waste. By sending back the exceeding tokens, the Persistent algorithm reaches better performance, whereas the average cluster size is even better when using our PBC-persistent algorithm. For both approaches, taking into account the potentials of neighbors when distributing tokens provide significant optimization of clustering, which confirms Assumption 1 for random topologies generated through our simulations.

V. CONCLUSION

In this letter, we propose to consider the degree of connectivity of neighbors when distributing tokens in budget-

based clustering algorithms. We propose two algorithms, PBC-simple and PBC-persistent, to produce clusters with sizes closer to the initial budget. We show through simulation that on random topologies, the average size of clusters obtained using our approach is larger than in previous work.

Our current work focuses on determining the best set of nodes that should initiate flooding to optimize the algorithms, instead of assigning random timers to the nodes.

REFERENCES

- [1] S. Basagni, M. Mastrogianni, A. Panconesi, and C. Petrioli, "Localized protocols for ad hoc clustering and backbone formation: A performance comparison," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 4, pp. 292–306, Apr. 2006.
- [2] R. Krishnan and D. Starobinski, "Efficient clustering algorithms for self-organizing wireless sensor networks," *Ad Hoc Networks*, vol. 4, no. 1, pp. 36–59, Jan. 2006.
- [3] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *Proc. of the IEEE Infocom*, San Francisco, CA, USA, Apr. 2003.
- [4] O. Younis and S. Fahmy, "Distributed clustering for ad-hoc sensor networks: A hybrid, energy-efficient approach," in *Proc. of the IEEE Infocom*, Hong Kong, Mar. 2004.
- [5] K. Sohrabi, W. Merrill, J. Elson, L. Girod, F. Newberg, and W. Kaiser, "Methods for scalable self-assembly of ad hoc wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 317–331, Oct. 2004.
- [6] J.-S. Liu and C.-H. R. Lin, "Energy-efficiency clustering protocol in wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 3, pp. 371–388, May 2005.
- [7] "The Network Simulator NS-2," <http://www.isi.edu/nsnam/ns/>.
- [8] L. Lazos and R. Poovendran, "Coverage in heterogeneous sensor networks," in *WiOpt*, Boston, MA, USA, 2006.
- [9] D. Miorandi and E. Altman, "Coverage and connectivity of ad hoc networks presence of channel randomness," in *Proc. of the IEEE Infocom*, Miami, FL, USA, 2005.
- [10] C. Guo, L. C. Zhong, and J. M. Rabaey, "Low power distributed mac for ad hoc sensor radio networks," in *Proc. of the IEEE Globecom*, San Antonio, TX, USA, Nov. 2001.