Self-Organizing Resource-Aware Clustering for Ad Hoc Networks

Tales Heimfarth, Peter Janacik, and Franz J. Rammig

Heinz Nixdorf Institute, University of Paderborn
Fuerstenallee 11, 33102 Paderborn, Germany
{tales,pjanacik,franz}@uni-paderborn.de

Abstract. This paper proposes an efficient heuristic for solving the *minimum-intracommunication clustering* problem in energy- and resource-constrained ad hoc networks. The heuristic organizes the network in clusters aiming to minimize a given cost function. The function used measures the total communication cost between all nodes within the cluster, keeping a minimum amount of resources per cluster.

The clusterhead selection of the proposed heuristic is based on the division of labor encountered in social insects. The idea is that each node has probabilistic tendencies to assume a determined role in the network. For example, nodes with good connectivity and high energy level are good candidates for being clusterheads. The probability of assuming a determined role is based on a node's fitness for the specific role and the actual *necessity* (reflected by stimulus) of the role in the current network context. After becoming clusterhead, a node starts recruiting members in order to reach a minimum amount of resources that have to be available in the cluster. The procedure is based on a membership fitness function that evaluates the suitability of a node for the cluster.

The realized simulations demonstrate that the proposed heuristic performance was about in average 25% inferior to the global optimum.

1 Introduction

Wireless ad hoc networks enable a myriad of novel applications ranging from human-embedded sensing to ocean data monitoring. Given current hardware limitations of wireless nodes, e.g. commercial off-the-shelf sensor nodes, approaches for the management of ad hoc networks have to be designed to function using only a low amount of resources and communication overhead.

In general, there are two heuristic design approaches for management of ad hoc networks at different levels (e.g. topology control, network layer, application). The first method is to have all nodes maintain knowledge of the network and manage themselves. This encurse a large amount of overhead. An alternative is to *clustering* the nodes, identifying a subset of nodes, and vest them with the extra responsibility of being a leader (clusterhead) of certain nodes in their proximity. The aim of this approach is to reduce communication and memory overhead.

In this paper, we present a new heuristic to organize an ad hoc network into clusters. Our proposal addresses the problem of partitioning the nodes of the network in multi-hop groups with a guaranteed minimum amount of resources q in each one of them. This kind of clustering is useful in various scenarios. An example is the operating system (OS) we are currently developing in our research group, called NanoOS [1].

NanoOS is a small distributed OS for sensor networks. In order to provide more functionality on hardware constrained nodes, we are distributing the OS and application services among the nodes of the network. We use the heuristic presented here to organize the network in clusters. After this, each OS and application instance (a set of services) is distributed inside one cluster. We set the resource requirement (q) to the worst-case resource utilization of one instance of the OS and application. Therefore, it is guaranteed that each cluster has enough resources for an instance of our distributed OS.

This paper is organized as following: Section 2 reviews the state-of-the-art in clustering algorithms for ad hoc networks. Section 3 describes the proposed architecture, before Section 4 discusses simulation results. Finally, Section 5 presents the conclusions.

2 Related Work

There are several clustering algorithms that aim to find the *Maximum independent set* (MIS) of a network modeled as an undirected graph. This is often combined with the *dominance* property, which means that the following properties should be satisfied: independence (no two clusterheads can be neighbors) and dominance (every ordinary node has at least a clusterhead as direct neighbor). There are several algorithms that satisfy these properties ([2–6]). Different from our approach, they result in a 1-hop distance to the clusterheads of clusters.

Several approaches have been proposed for multi-hop clustering with different construction objectives. Here we have the *Max-Min D-Cluster Formation* [7] that aims to construct the cluster with nodes at most d hops away from the clusterhead.

Other heuristics that pursue different objectives are the *Budget Approach* [8] which tries to divide the ad hoc network in a set of clusters whose size is close to a given one. Beyond this, the *Upper and under bound approach* [9] works with superior and inferior size limits. It tries to construct clusters that respect these limits. Nevertheless, the approach allows a small overlap among the clusters.

In contrast to above heuristics, our clustering algorithm pursues a different objective: all clusters should posess a minimum amount of resources (i.e., the under bound limit is not given by a size in nodes, but by an amount of resources), and we try to minimize the internal cluster communication cost. This will be discussed in detail in the next sections.

3 The *minumum-intracommunication clustering* problem

In this section, a formal definition of our clustering problem (we call it *minumum-intracommunication clustering*) is described.

The ad hoc network is modeled by an undirected graph $G = (V, E)$, where V is the set of wireless nodes and an edge $\{u, v\} \in E$ if and only if a communication link is established between nodes $u \in V$ and $v \in V$. Each node $v \in V$ has a unique identifier (ID_v).

For each link, a weighting function attributes a positive weight $w : E \to [0, 1]$ that represents the quality of a wireless link. In the work [10], we presented a method to estimate the quality of a wireless link based on our combined metric. We call this metric *virtual distance* and smaller values represent better connection links. We define for each edge not in the graph $(\{u, v\} \notin V)$, $w(u, v) = \infty$.

For each node, an additional weighting function r is responsible to characterize the amount of resources available in the node. $r : E \to \Re^*$. This models the resource capacity of the node.

We aim to create multihop clusters with a minimum amount of resource per cluster minimizing the intra-cluster communication cost.

The considered optimization problem is modeled as follows:

Input: A graph with weighted nodes and links (G, w, r) and a resource requirement $q \in \Re^*$ that must hold in each cluster

Constraints: For every input instance (G, w, r, q), $\mathcal{M}(G, w, r, q) = \{C_1, C_2, .., C_k | C_k$ is the kth cluster configuration, where the following proprieties holds $\}$
$C_k = \{c_{k1}, c_{k2}, .., c_{k(nk)}\}$ is the kth possible cluster configuration of the graph, where $k = \{1, 2, .., n\}$ (n is the number of possible configurations, nk is the number of clusters in the kth configuration, $nk = \#C_k$)
$c_{ki} = \left\{v_{ki}^1, v_{ki}^2, .., v_{ki}^{\#c_{ki}}\right\} \in Pot(V)$ is the ith cluster of the kth configuration, where v_{ki}^j is the jth element of the cluster c_{ki}
For each configuration C_k, $k = 1, 2, .., n$, the following proprieties must hold:

1. $\bigcup_{i=1,2,..,nk} c_{ki} = V$ (cluster definition constrain)
2. $\bigcap_{i=1,2,..,nk} c_{ki} = \emptyset$ (no overlapping constraint)
3. Let $P(u, v) = \left\{p_1^{(u,v)}, p_2^{(u,v)}, .., p_m^{(u,v)}\right\}$ be the set of all possible paths between nodes u and v. $p_h^{(u,v)} \in Pot(E)$ is the hth possible path where $p_h^{(u,v)} = \{\{u, x_1^h\}, \{x_1^h, x_2^h\}, .., \{x_{g-1}^h, x_g^h\}, \{x_g^h, v\}\}$, $x_f^h \in V$, $f = 1, 2, .., g$, $g \in \mathbb{N}$
   For each $\{u, v\} \in E \wedge u, v \in c_{ki}$, $i = 1, 2, ..., nk$, $\exists p_h^{(u,v)} \in P(u, v) | x_f^h \in c_{ki}$ for $f = 1, 2, .., g$. (Connectivity constraint)
4. $\sum_{j=1}^{\#c_{ki}} r(v_{k_i}^j) \geq q$, for each $i = 1, 2, ..., nk$ (minimum amount of resource per cluster)

Costs: For every cluster configuration $C_k = \{c_{k1}, c_{k2}, .., c_{k(nk)}\} \in \mathcal{M}(G, w, r, q)$, the cost is $cost(C_k, (G, w, r, q)) = \sum_{i=1}^{nk} \sum_{u,v \in c_{ki}} D_{c_{ki}}(u, v) \cdot (\alpha \cdot r(u) + (1-\alpha))$, where $D(u, v)$ is the virtual distance between $u, v \in V$. $D_{c_{ki}}(u, v)$ is the

virtual distance between u, v using the shortest path that includes just links that are inside the cluster c_{ki} (sum of the link weights of the shortest path inside the cluster) . $\alpha \in [0, 1]$ controls how much the amount of resources influences the distance metric. For $\alpha = 0$, just the distances between cluster members are incorporated into the metric; $\alpha = 1$ means that nodes with n times more resources have an n times stronger influence.

$PCost(p_h^{(u,v)}) = w(u, x_1^h) + \sum_{f=1}^{g-1} w(x_f^h, x_{f+1}^h) + w(x_g^h, v)$

$D(u, v) = PCost(p_h^{(u,v)})$, where $PCost(p_h^{(u,v)}) = \min\left(PCost(p_b^{(u,v)})\right)$ for

$b = 1, 2, .., m$ $D_{c_{ki}}(u, v) = PCost(p_h^{(u,v)})$, where $p_h^{(u,v)} \in P(u, v)|x_f^h \in c_{ki}$

and $PCost(p_h^{(u,v)}) = \min\left(PCost(p_b^{(u,v)})\right)$ for $b = 1, 2, .., m$

Goal: *Minimum* (i.e. $min\{cost(C_k, (G, w, r)) \mid$ for $k = 1, 2, .., n\}$)

It is important to note that we are trying to minimize the sum of the link costs over all clusters. In each cluster, this cost is given by the sum of the link costs from every node to all other ones.

4 Emergent Clustering

The heuristic presented aims to find good clustering configuration in a network with a low amount of mobility. It reacts by stronger changes through re-execution.

4.1 Clusterhead selection

In the initial state, all nodes of the network are ordinary nodes, i.e., there is no cluster structure in the network. The variable $state_v$ describes the actual state of a node v ($state_v \in \{CH, Me, Nm\}$) and c_i is set of the current members of the cluster $i \in \mathbb{N}$. For the sake of simplicity, we define that $clusterID = i$. Initially, for $i = 0, 1, .., n$, $c_i =$. The response function $T_{\theta_{CH_v}}(s_{CH_v}) = \frac{s_{CH_v}^{\beta}}{s_{CH_v}^{\beta} + \theta_{CH_v}^{\beta}}$ is responsible for the transition from an ordinary (Nm) node $v \in V$ to clusterhead. θ_{CH_v} is the threshold of the node v to become clusterhead and s_{CH_v} is the stimulus of v to assume the clusterhead role.

The threshold indicates how appropriate a node is for a role. Smaller θ_{CH_v} means that the node v is very well suited to carry out the role of a clusterhead. The definition of the threshold can be seen in (1).

$$\theta_{CH_v} = k_1 \left(\frac{\sum_{u \in Ngb_{Nm}(v)} w(u, v)}{\#Ngb_{Nm}(v)}\right) + k_2(1 - E_v) + k_3 \left(1 - \frac{\#Ngb_{Nm}(v)}{Max\_Nm}\right) \quad (1)$$

Where $E_v \in (0, 1)$ describe the energy level of the node v, such that 1 means the battery is full and 0 that it is depleted. Let $Ngb(v)$ be the set of nodes that are directly connected with v, i.e. $u \in Ngb(v)$ iff $\{u, v\} \in E$. A node u is in the

set $Ngb_{Nm}(v)$ iff $u \in Ngb(v)$ and $state_u = Nm$. This means that $Ngb_{Nm}(v)$ is the set of neighbors of v that do not yet belong to any cluster.

The idea of this threshold function is that nodes with high energy level and high connectivity are good candidates for becoming elected as clusterhead. The energy is an important factor because clusterheads assume administrative (among other) tasks within the cluster and have a special status in the network. Good connectivity comes from the greedy assumption that starting a cluster from well-connected nodes will result in a relatively small clustering cost.

The stimulus function is given (for $k_1 + k_2 = 1$) by $s_{CH_v} = k_1 \frac{t_{elapsed}}{t_{required}} + k_2 \left(1 - \frac{Ngb_{Me}(v) + Ngb_{CH}(v)}{Ngb(v)}\right)$. The elapsed time is $t_{elapsed}$ and $t_{required}$ is the maximum running time of the algorithm.

A node u is in the set $Ngb_{Me}(v)$ iff $u \in Ngb(v)$ and $state_u = Me$. Similarly, $u \in Ngb_{CH}(v)$ iff $u \in Ngb(v)$ and $state_u = CH$. With simple words, $Ngb_{Me}(v)$ is the set of neighbors of v that are members of some cluster. $Ngb_{CH}(v)$ is the set of neighboring nodes that are already clusterheads.

The underlying idea is that nodes that are not belonging to any cluster for a longer period of time and nodes without clusters in their vicinity should have a higher stimulus to become clusterhead. With the response function given by $T_{\theta_{CH_v}}(s_{CH_v})$, spontaneously, some nodes will start to change the role to clusterhead based on the stimulus function. When a node decides to be clusterhead, it selects a random $ClusterID$.

4.2 Members Selection

Influencing parameters During membership selection by the clusterheads the following paramenters help to evaluate the suitability of a node b:

1. *The distance to the closest node already in the cluster*: This parameter helps to reduce the communication cost within the cluster. It is given by $D^b_{c_i} = min\{w(b,e)|e \in Ngb(b) \cap c_i\}$, i.e., the smallest vertice weight that is adjacent to node b and to a member of the cluster c_i. If a node is not directly connected to a cluster member, $D^{c_i}_d = \infty$.
2. *The distance to the clusterhead*: This parameter is responsible for shaping the cluster in order to constrain its diameter.
3. *Connectivity to nonmembers*: This parameter is important when there is a lot of resources still missing in the cluster. Given by $Cn^b_{Nm} = \sum_{e \in Ngb_{Nm}(b)}(1 - w(b,e))$, i.e., the sum of the "proximity" $(1 - w(b,e))$ of the set of the neighbors of b that have nonmember status. Figure 1 illustrates the effect of this term.
4. *Connectivity to members of the cluster*: This parameter helps to reduce the communication cost within the cluster. It is given by $Cn^b_{c_i} = \sum_{e \in \{Ngb(b) \cap c_i\}}(1 - w(b,e))$ where c_i is the current set of members of the cluster $i \in \mathbb{N}$.
5. *The resource availability of the node*: Aim of communication cost reduction based on the idea that nodes with higher resource availability will reduce the cost of the cluster to a greater extent since fewer of them are needed.

These aspects will be explicitly or implicitly considered by the *Membership-Select* algorithm presented here.

Membership-Select algorithm For the membership selection, we use the $state_v$ variable describing the actual state of a node v with an additional state: the deciding (Dd) state ($state_v \in \{CH, Me, Nm, Dd\}$). Let Δq be the amount of additional resources needed by a cluster in order to fulfill the requirement q at a certain point of time. The *Membership-Select* algorithm is an incremental process, i.e., at beginning, the cluster has just the clusterhead (CH) node and during the clustering process, more and more nodes are added until the cluster achieves an appropriate size ($\sum_{v \in c_i} r(v) \geq q$). At the beginning of the clustering process of the cluster i, just one node belongs to the cluster: the clusterhead. We will call it node h_i ($h_i \in c_i, state_h = CH$). When a node becomes part of the cluster (including the clusterhead), immediately a message is broadcasted to the neighboring nodes signalizing the new status and requesting new members (`Call_Members` message). Each nonmember and deciding node d ($status_d \in \{Nm, Dd\}$) that receives this message changes its state to deciding ($status_d = Dd$).

Deciding nodes are the potential new members of the cluster. Nevertheless, not all nodes are the best choice to be included into the cluster. In order to privilege nodes potentially contributing to a low global cluster cost, each node b in the decision state estimates its own fitness value $0 \leq Fitness_{c_i}^b \leq 1$. This value will be defined later. $Fitness_{c_i}^b$ describes the suitability of the inclusion of node $b \in V$ into the cluster c_i.

At this point, the node b waits using a delay which is proportional to the $1 - Fitness_{c_i}^b$ value. When the waiting time has elapsed, the node sends a `Membership_Request` message to the clusterhead, informing it that it is willing to be included into the cluster. Now the clusterhead, based on Δq and the availability of resources of the candidate, can decide whether the node will be accepted as member. If accepted, the clusterhead includes the new node in a table with all members of the cluster. A message is sent back to the node confirming/refusing the entrance into the cluster. When receiving the response message, the requester changes its status accordingly ($state_b = Me$, if accepted and $state_b = Nm$ if refused). If accepted, this new status is broadcasted immediately in a message calling for new members (`Call_Members`) to the neighborhood of b, starting the process again.

When $\Delta q \leqslant 0$, i.e., the cluster is complete, all additional receiving requests will be rejected.

Consider the example depicted in Figure 3. We colored the nodes according the *state*; white nodes are nonmembers, black nodes members (or clusterhead, i.e. $status \in \{Me, CH\}$) of the cluster c_i being formed and gray members are deciding nodes.

In Figure 3 (a), the initial state with a one member cluster (clusterhead, selected by the response function) is shown. The clusterhead broadcasts the `Call_Members` message transmitting its state (3 (b)). At this point, all nodes
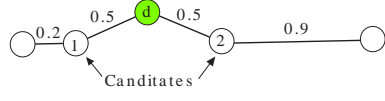
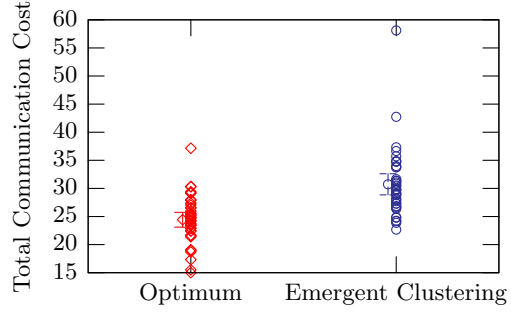**Fig. 1.** Example of two candidates with different neighborhood.

Fig. 2. Resulting communication cost of a randomly-placed wireless network using the signal strength as link metric. For this trial, $q = 3$ was used.
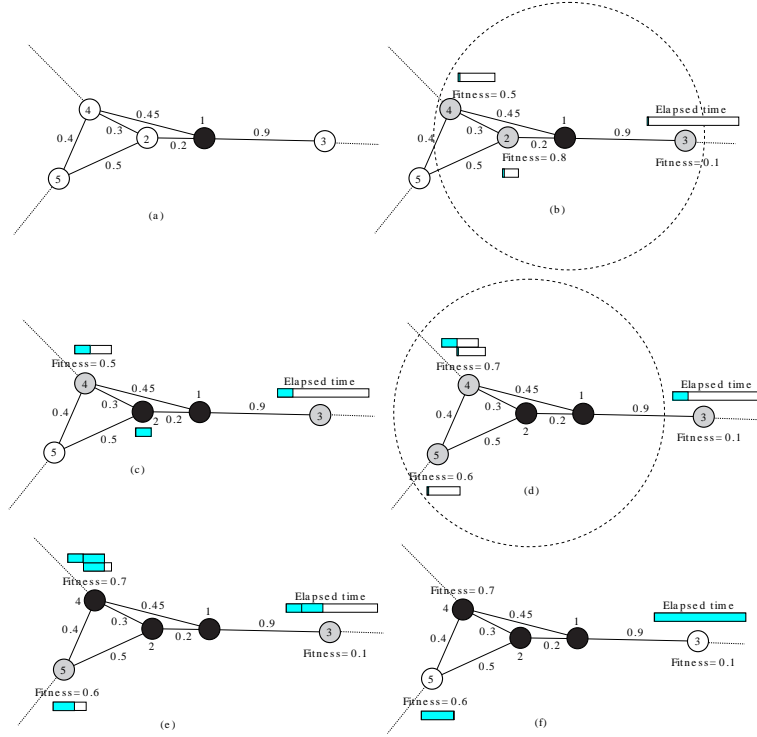

Fig. 3. Example of member selection in a partial network. All nodes have unitary amount of resource ($r(b) = 1, b \in V$) and $q = 3$.

$$Fitness_{c_i}(v) =$$

$$= \begin{cases} 1 - \left(k_1 \cdot D_{c_i}^v + k_2 \cdot min\{\frac{D(v, Clusterhead_{c_i})}{Max\_dist}, 1\} + \right. \\ \left. + k_3 \cdot min\{\frac{Cn_{Nm}^b}{Max\_connect}, 1\} + k_4 \cdot min\{\frac{Cn_{c_i}^b}{Max\_connect}, 1\} + \right. & \text{if} \quad r(v) < q \\ \left. + k_5 \frac{r(v)}{q}\right) \\ 0 & \text{if} \quad r(v) \geqslant q \end{cases}$$

Fig. 4. Definition of the fitness function

that receive the message change to the deciding state. A timer is set based on the calculated fitness for each node. In Figure 3 (c), the programmed time of node 2 has already elapsed. The node now asks for membership. As the total resource request (q) is not satisfied by the current cluster size, the node 2 is included in the cluster. Now it also broadcasts a `Call_Members` message to the neighborhood (Figure 3 (d)). When node 4 and 5 receive the broadcasted message, they start a timer related to the computed fitness ($1 - Fitness_{c_i}(4)$ and $1 - Fitness_{c_i}(5)$). Due to the fact that node 4 has already a timer, just the timer with the shortest deadline is kept. In Figure 3 (e), the programmed time of node 4 has elapsed and it requests for membership. It receives permission to enter into the cluster. Since the cluster is already complete, node 4 does not broadcast a new `Call_Members` message. Finally, the waiting time for nodes 5 and 3 is over. They request membership, but due to the fact that the cluster has enough resources, the permission to integrate the cluster is refused.

Now we will integrate the already presented heuristic hints (see Section 4.2) that should guide the member selection. The first point says that the heuristic should privilege nodes with a small distance to some of the nodes inside the actual cluster. In order to observe that, two aspects must be addressed: (1) Include the distance to the next cluster member in the fitness function with the aim of reducing communication costs. (2) An implicit behavior of the heuristic makes that nodes far away from the clusterhead, but with good connection to the cluster are penalized since starting the timer in a later stage.

This should be addressed together with the point number two in our influence parameters list: the distance to the clusterhead. This point is aided by the implicit behavior of algorithm. The two aspects are important for reducing the cluster cost. Nodes near to the cluster are suitable because the connection cost is smaller, nevertheless, to keep clusters with smaller diameter also helps to reduce the total cost.

The distance to clusterhead is also addressed by two points: (1) Including the distance to the clusterhead in the fitness function. (2) Implicit behavior of the heuristic. To show that we can reuse the example shown in Figure 3. The fact that nodes near to the clusterhead started the timer earlier implicitly helps to get small diameter clusters.

Analyzing these two different requisites, the following method was created in order to penalize the distance to the clusterhead and reward the distance to the cluster (i.e. the distance to the closest node in the cluster). We will now count the rounds that the algorithm has already executed. Using the example presented in Figure 3, (b) represents the first round of the algorithm and (d) the second one. Each time that a new member was selected and made a broadcast to the neighborhood, the variable $round_v, v \in V$ is increased.

We define the waiting time of a node v to request to be included in the cluster c_i as $WaitingTime_{c_i}^v = k \cdot (1 - Fitness_{c_i}(v)) \cdot \frac{1}{\kappa \ round_v + (1 - \kappa)}$, where $v \in V, \kappa \in [0, 1], k \in \Re*$ and $0 \leq Fitness_{c_i}(v)) \leq 1$. It uses the fitness function and the current round to calculate the waiting time.

Using this equation, for bigger rounds, the time that should be waited is shortened. With the κ parameter, the amount of reward given to the distance to the cluster versus penalization of distance to clusterhead can be controlled.

The fitness function that takes into account all points presented in the Section 4.2 is presented in Figure 4 (for $\sum_{i=1}^{5} k_i = 1$), where $k_1, ..., k_5$ define how each of the terms influences the fitness metric. It is important to remark that $0 \leq Fitness_{c_i}(v) \leq 1$. For two nodes $v, u \in V$ and $Fitness_{c_i}(v) < Fitness_{c_i}(u)$ means that the node v is less suitable for the cluster c_i than the u. Max_dist describes the minimum distance to the clusterhead that should be considered the maximum penalty, $Max\_connect$ is the same for the connection measurements. We should remark that for nodes with more resources than required, the fitness is always 0 because they should form a cluster with one member.

5 Simulation and Results

We implemented our emergent clustering heuristic using Shox [11], a Java-based wireless ad hoc network simulator. As input, we generated 40 instances of the problem with 13 nodes in a field of 25m by 25m. These instances were generated by random selection of the nodes' position.

Our link metric used the received signal strength (RSSI) that was calculated using the free space model for an isotropic point source in an ideal propagation medium. The limits of the RSSI were determined using two thresholds, having the meaning of maximum signal ($w = 0.1$) strength and no signal ($w > 1$). We adjust the radio power in order to achieve a maximum transmission range of 10m. The *RSSI* was the only metric used to calculate the *virtual distance.* In our simulation, we adjust several parameters of the described equations such that every part of the equation has the same weight. In order to calculate the optimum cost of a problem instance, we model our *minimum-intracommunication clustering* as an integer linear programming problem and for each generated instance, we solve it using the *lp_solve* program.

Figure 2 shows the results of the 40 runs for our distributed heuristic and the respective optimal solution. The picture also shows the confidence interval of the obtained average. The average communication cost of the emergent clustering was 30.72 with a standard deviation of 6.07. The optimum solution has a mean of 24.42 with a standard deviation of 4.26.

6 Conclusion

In this paper, we introduce a useful clustering problem and develop an efficient heuristic to solve it. The heuristic is based on the response functions derived from the division of labor in social insects. On the basis of the response function, the most suitable nodes in terms of connectivity, energy and resources are selected for the clusterhead role. After emerging spontaneously, each clusterhead starts gathering members for the clusters until a resource requirement q is satisfied. The membership candidates are evaluated using a fitness function taking into

consideration their distance to the cluster and its clusterhead, connectivity and resource availability. After evaluating those items, a node delays its response by a time related to its fitness. Therefore, the higher fitness nodes announce themselves earlier, having higher priority for entering the cluster.

Using simulations in the Shox network simulator, we show that our approach performs well in average. It uses just local information and it is capable of starting in an unorganized ad hoc network, finding a cluster configuration that is in average just 25% above the optimum one. Further, the proposed emergent clustering approach obtains a performance of at least 1.5 times the optimal for 60% of the test cases. There are no test cases with a performance inferior to 2 times the optimum result.

Our results demonstrate once again that a principle from natural systems can be successfully transferred to an efficient algorithm for ad hoc networks solving a problem which is NP-complete in good aproximation.

References

1. Heimfarth, T., Danne, K., Rammig, F.J.: An os for mobile ad hoc networks using ant based hueristic to distribute mobile services. (2005) 77
2. Baker, D., Ephremides, A.: The architectural organization of a mobile radio network via a distributed algorithm. Communications, IEEE Transactions on **29**(11) (Nov 1981) 1694–1701
3. Baker, D.J., Ephremides, A., Flynn., J.A.: The design and simulation of a mobile radio network with distributed control. IEEE J. on Selected Areas in Communications **SAC2(1)** (1984) 226–237
4. Gerla, M., Tsai, J.T.C.: Multicluster, mobile, multimedia radio network. Wirel. Netw. **1**(3) (1995) 255–265
5. Basagni, S., Chlamtac, I., Farago, A.: A generalized clustering algorithm for peer-to-peer networks. In: Proceedings of the Workshop on Algorithmic Aspects of Communication (Satelite workshop of ICALP), Bologna, Italy. (1997)
6. Heinzelman, W., Chandrakasan, A., Balakrishnan, H.: An application-specific protocol architecture for wireless microsensor networks. Wireless Communications, IEEE Transactions on **1**(4) (Oct. 2002) 660–670
7. Amis, A., Prakash, R., Vuong, T., Huynh, D.: Max-min d-cluster formation in wireless ad hoc networks. In: INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE. Volume 1. (26-30 March 2000) 32–41vol.1
8. Krishnan, R., Starobinski, D.: Message-efficient self-organization of wireless sensor networks. In: Proc. IEEE Wireless Communications and Networking Conference, New Orleans, USA (2003)
9. Bannerjee, S., Khuller, S.: A clustering scheme for hierarchical control in wireless networks. In: Proc. IEEE INFOCOM, Anchorage, Alaska (2001)
10. Heimfarth, T., Janacik, P.: Ant based heuristic for os service distribution on ad hoc networks. In: 1st IFIP International Conference on Biologically Inspired Cooperative Computing (BICC 2006), volume 216 of IFIP International Federation for Information Processing, Boston, MA, USA, Springer (August 2006) 75–84
11. : http://shox.sourceforge.net Accessed on 28. Jannuary 2006.