Towards an Approach for Mobile Profile Based Distributed Clustering

Christian Seitz and Michael Berger

Siemens AG, Corporate Technology, Information and Communications,
81730 Munich, Germany,
Christian.Seitz@mchp.siemens.de
m.berger@siemens.com

Abstract. We present a new application for mobile ad hoc networks, which we call *Mobile Profile based Distributed Clustering*(MPDC), which is a combination of mobile clustering and data clustering. In MPDC each mobile host is endowed with a user profile and while the users move around, hosts with similar profiles are to be found and a robust mobile cluster is formed. The participants of a cluster are able to cooperate or attain a goal together. We adapt MPDC to a taxi sharing application, in which people with similar destinations form a cluster and could share a taxi or other public transportation resources.

1 Introduction

Technology developments like mobile devices and mobile communication have formed a new computing environment which is referred as mobile computing, in which an entire new class of distributed applications has been created. An mobile ad hoc network consist of hosts travelling through physical space and communicating in an opportunistic manner via wireless links. In the absence of a fixed network infrastructure, the mobile hosts must discover each other's presence and establish communication patterns dynamically. The structure of an ad hoc mobile network is highly dynamic. In an ad hoc network two hosts that want to communicate may not be within wireless transmission range of each other, but could communicate if other hosts between them are also participating in the ad hoc network and are willing to forward packets for them. The absence of a fixed network infrastructure, frequent and unpredictable disconnections, and power considerations render the development of ad hoc mobile applications a very challenging undertaking.

We present a new mobile ad hoc network application area, which me call *Mobile Profile based Distributed Clustering* (MPDC). In MPDC each mobile host is endowed with a user profile and while the users move around, hosts with similar profiles are to be found and a mobile cluster is formed.

We apply MPDC to a taxi sharing scenario. If a train arrives at a railway station or an airplane at an airport the different passengers may have the same destination e. g. a hotel. This destination address is part of a user profile and

H. Kosch, L. Böszörményi, H. Hellwagner (Eds.): Euro-Par 2003, LNCS 2790, pp. 1109–1117, 2003.

stored on a mobile device. While people are waiting at the baggage terminal the mobile devices exchange the profiles and try to find small groups with similar destinations.

The rest of the paper is organized as follows. Section 2 gives an overview of related work of other clustering problems. In section 3 we outline the architecture of MPDC, define the ad hoc network model and make some assumption. The next section describes the used algorithms in our approach to MPDC and finally, section 5 concludes the paper.

2 Problem Classification and Related Work

Mobile Profile based Distributed Clustering comprises three main problems. The first problem is the dynamic behavior of an ad hoc network, where the number of hosts and communication links permanently changes. Furthermore, an expandable profile has to be defined and a mechanism must be created to compare profile instances. Finally, similar profiles have to be found in the ad hoc network and the corresponding host form a robust cluster, in spite of the dynamic behavior of the ad hoc network.

The term clustering is used in the research areas databases, data mining and in the mobile networks are. Clustering in mobile networks describes the partitioning of a mobile network in several, mostly disjoint, clusters [1,2]. The clustering process comprises the determination of a cluster head in a set of hosts. A cluster is a group of hosts, all able to communicate with the clusterhead. This clustering takes place at the network layer and is used for routing purposes. Clustering in the database or data mining area encompasses the search for similar data sets in huge data bases. In the surveys of Fasulo [4] or Fraley and Raftery [5] an overview of many algorithms for that domain can be found. Maitra [8] and Kolatch [7] examine data-clusters in distributed databases.

In *Mobile Profile based Distributed Clustering* mobile hosts are equipped with wireless transmitters, receivers, and a user profile. They are moving in a geographical area and form an ad hoc network. In this environment hosts with similar profiles have to be found. Therefore, MPDC must combine the two aforementioned clustering approaches to accomplish its objective. The problems, arising by means of the motion of the hosts could be solved by methods used in the mobile network area. Searching for similar profiles is based on algorithms of data clustering. Both methods must be adapted to MPDC, e. g. while in the database area millions of data sets must be scanned, in the MPDC application at the utmost one hundred other hosts are present. In contrast to data sets in databases ad hoc hosts move around and are active, i. e. they can publish their profile by their own.

There is other work that analyzes ad hoc clustering algorithms. Roman *et al.* [10] deals with consistent group membership. They assume that the position of each host is known by other hosts and two hosts do only communicate with each other, if it is guaranteed that during the message exchange the transmission

range will not exceed. In our environment obtaining position information is not possible, because such data is not always available, e. g. inside of buildings.

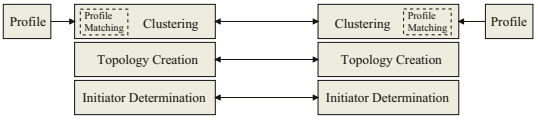

Fig. 1. Architecture

3 Architecture

In this section the architecture of MPDC is presented, which consists of three layers. With each layer at least one agent is associated. Figure 1 envisages the layered architecture of MPDC. The lowest layer is the *Initiator Detection* layer, which assigns the initiator role to some hosts. An initiator is needed in order to guarantee, that the algorithm of the next layer is not started by each host of the network. This layer does not determine one single initiator for the whole ad hoc network. It is sufficient, if the number of initiator nodes is only reduced. The *Virtual Topology* layer is responsible for covering the graph G with another topology, e. g. a tree or a logical ring. This virtual topology is necessary to reduce the number of messages, that are sent by the mobile hosts. First experiences showed, that a tree is the most suitable virtual topology and therefore we will only address the tree approach in this paper. The next layer is the most important one, the *Clustering* layer, which accomplishes both, the local grouping and the decentralized clustering. Local grouping comprises the selection of hosts which are taken into account for global clustering. Decentralized clustering encompasses the exchange of the local groups with the goal to achieve a well defined global cluster. The *Profile Matching* module, depicted with a dashed box in figure 1 is responsible for comparing two profiles.

Below, two definition are given to distinguish local grouping and decentralized clustering.

Definition: A *local group* g_i in a graph G(V,E) is a subset of vertices $V_g^i \subseteq V$ with *similar* profiles according to a node v_i. Furthermore, there is a similarity operator σ, with $\sigma(v_i, V) = V_g^i$. Graph G consists of $|V|$ local groups, which together build a set, denoted with $\mathcal{G}$.

Definition: A *decentralized cluster* $\mathcal{C}$ is the intersection of all local groups g_i in a graph G(V,E). A decentralized cluster $\mathcal{C}$ is obtained, if σ is applied to all V_g^j.

4 Algorithms

In this section the network model is defined and the algorithms for each layer are presented. The used middleware for MPDC is an ad hoc multi agent platform

which is described in Berger and Watzke [3], where each agent platform hosts an ad hoc management agent that makes itself known to its neighbors by generating a beacon at regular intervals and by listening to signals from other hosts around.

4.1 The ad hoc Network Model

An ad hoc network is generally modelled as an undirected graph $G_0 = (V_0, E_0)$ as depicted in figure 2a. The vertices v_i of G_0 represent mobile hosts. If the distance between v_i and v_j is below the transmission range r_t, the two vertices are connected by an edge e_{ij}. Due to the motion of the vertices, a graph G_0 as shown in figure 2a is only a snapshot of an ad hoc network, because in consequence of the mobility of the hosts G_0 will change.

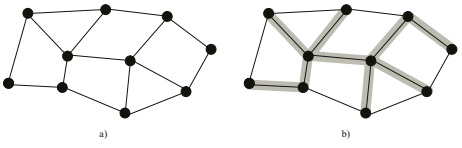

Fig. 2. Graph and a possible Spanning tree

Assumptions on the the mobile nodes and network are:

- Each mobile device has a permanent, constant unique ID.
- The transmission range of *all* hosts is r_t.
- Each host knows all its neighbors and its associated IDs.

4.2 Initiator Determination

At first, initiators must be determined who are allowed to send the first messages. Without initiators all hosts start randomly sending messages with the result that the algorithm in the next layer cannot start. We are not in search of one single initiator, we only want to guarantee, that not all hosts start the initiation.

There are active and a passive methods to determine an initiator. The active approach starts an election algorithm (see Malpani *et al.* [9]). These algorithms are rather complex and a lot of messages are sent. They guarantee that only one leader is elected and in case of link failures that another host takes the initiator role. This is not necessary for MPDC, because the initiator is only needed once and it matters little if more than one initiator is present. Therefore, we decided for a passive determination method, which is similar to Gafni and Bertsekas [6]. By applying the passive method no message is sent to determine an initiator. Since each host has an ID and knows all neighbor IDs, we only allow a host being an initiator, if its ID is larger than all IDs of its neighbors. The initiator is in charge of starting the virtual topology algorithm, described in the next section.

4.3 Virtual Topology Creation

Having confined the number of initiators, the graph G_0 can be covered with a virtual topology (VT). Simulations showed that a *spanning tree* is a promising approach for a VT and therefore we will only describe the spanning tree VT in this paper.
A spanning tree spT(G) is a connected, acyclic subgraph containing all the vertices of the graph G. Graph theory guarantees, that for every G a spT(G) exists. Figure 2b shows a graph with one possible spanning tree.

The Algorithm
Each host keeps a spanning tree sender list (STSL). The STSL contains the subset of a host's neighbors belonging to the spanning tree. The initiator, determined in the previous section, sends a `create`-message furnished with its ID to all its neighbors. If a neighbor receives a `create`-message for the first time, this messages is forwarded to all neighbors except for the sender of the `create`-message. The host adds each receiver to the STSL. If a host receives a message from a host which is already in the STSL, it is removed from the list. To identify a tree, the ID of the initiator is always added to each message. It may occur that a host already belongs to another tree. Under these circumstances the message is not forwarded any more and the corresponding host belongs to two (more are also possible) trees.

In order to limit the tree size a hop-counter c_h is enclosed to each message and is each time decremented, the message is forwarded. If the counter is equal to zero, the forwarding process stops.

By using a hop-counter it may occur that a single host does not belong to any spanning tree, because all tree around are large enough, i. e. c_h is reached. The affiliation of that host is not possible, because tree nodes do not send messages in case the hop-counter's value is zero. When time elapses and a node does notice it does still not belong to a tree, an initiator determination is started by this host. Two cases must be distinguished. In the first one the host is surrounded only by tree nodes, in the other case a group of isolated hosts are existing. In both cases, the isolated host contacts all its neighbors by sending an `init`-message, and if a neighbor node already belongs to a tree it answers with a `join`-message. If no non-tree nodes are around, the single node chooses arbitrarily one of the neighbors and joins the tree by sending an `join-agree`-message, to the other hosts a `join-refuse`-message is sent. If another isolated host gets the `init`-message, a `init-agree`-message is returned and the host sending the init-message becomes the initiator starts creating a new tree.

Evaluation
The reason for creating a virtual spanning tree is the reduction of messages needed to reach an agreement. Let n be the number of vertices and let e be the number of edges in a graph G. If no tree is built each messages must be forwarded to all its neighbors, which results in $2e-n+1$ messages. Overlaying a graph with a virtual spanning tree, the number of forwarded messages is reduced to $n-1$ plus $2e-n+1$ messages for tree creation. Determining the factor A,

when a tree becomes more profitable leads us to $A = \frac{2e-n+1}{2(e-n+1)}$. If on the average $e = 2n$, the amortization A results in $\frac{3n+1}{2n+2}$.

Table 1. Relation of edges and vertices in an arbitrary graph

#vertices n	#edges e	amortization A
9	26	1.22
19	45	1.33
26	71	1.27
50	135	1.28

To confirm this formal work, some arbitrary ad hoc networks were investigated (see table 1). The amortization value A does never exceed 1.4 (theoretical value), that means if each host sends only two messages to all of its neighbors, less messages are sent then without the spanning tree.

In the equation above, the tree maintenance cost are not taken into account. If a new host comes into transmission range or an other host goes away, additional messages must be sent to re-establish the virtual topology.

4.4 Local Grouping: Optimizing the Local View

In this section the subset of neighbor hosts are determined, which initially belong to a host's local cluster, called a *group*. The algorithms presented in this and the following section depend on the used profile, which on its part depends on the application. We describe the grouping and clustering algorithms using the taxi sharing application with a very simple profile that only consists of a X- and Y-value, representing the destination of a person. These points are plotted in figure 3a in a coordinate system, which is the local view of the black point. The grey points are the X, Y-values of the points the black one is able to communicate with. Besides, all hosts are currently located around the origin of the coordinate system, which can always be achieved by rotating and translating the coordinate system.

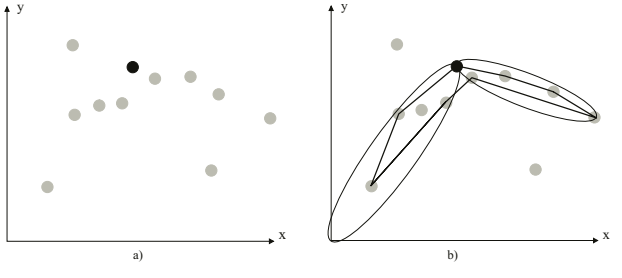

Fig. 3. Local Grouping of a mobile host

The algorithm starts by creating an ellipse from the origin of the coordination system to the local's point destination (P_d), see the left ellipse in figure 3b. An ellipse was chosen because it is a continuous shape, whereas a rectangle is not. The height (the semi minor half-axis) of the ellipse is in fixed relation to the length of the ellipse. All points inside this oval already belong to the local group. In order to interchange local groups, a more compact shape, e. g. polygon, including all points is desirable.

After creating the first oval, the local group has to be enlarged, if other points are still present. Therefore, a new destination point P_d must be found, which acts as new ending point of another oval (see second oval in figure 3b). This point must meet the following requirements:

- P_n must not harm the structure of the existing oval group, i. e. no P_n is allowed which result in a cycle, loop or a helix.
- To build the local group with as little ovals as possible, P_n must include as many other points as possible.

To guarantee the first requirement, the angle between the first oval and the potential next one has to be considered and may not exceed a specified value. In order to satisfy the second requirement, for each point P, which is not yet in an oval, it is calculated, how many other points would be in that new oval if it may become the next P_d and how many points are excluded in becoming further P_d points. The point with the largest difference becomes the new P_d.

This polygon has to be merged with the polygon of the first oval, because for the clustering we need one single polygon, that contains the whole group. Merging polygons is a tricky undertaking, because the merged polygon should not be larger than the ovals. But simply connecting two polygons does not meet this requirement. Therefore, additionally virtual profile points P_v must be inserted to reduce the size of a merged polygon.

Having determined a local group it is easy to proceed, when a host with a new profile appears. If the new point is inside the polygon, no changes are to be done. If it is outside the polygon and could become a new P_d the polygon is adjusted. In all other cases the point does not become a member of the group.

4.5 Decentralized Clustering: Achieving the Global View

In the previous section each host has identified its neighbor hosts that belong to its local group g_i. These local groups must be exchanged and a global cluster has to be achieved.

The algorithm presupposes no special initiator role. Each host may start the algorithm and it can even be initiated by more than one host contemporaneously. Initially, each host sends a `cluster`-message with its group-polygon enclosed to its neighbors which are element of the spanning tree. If a message arrives, the enclosed polygon is taken and it is intersected with it current local view of the host to get a new local view. This new local view is forwarded to all neighbors except for the sender of the received message. If a node has no other outgoing

edges and the algorithm has not terminated, the message is sent back to the sender. If node receives two messages from different hosts, only one message is forwarded in order to reduce the number of messages. If the algorithm has terminated, each host has the same lokal view, i. e. the global is achieved.

A critical point is to determine the termination of the clustering process. The algorithm terminates in at most $2 \cdot d_G = 4 \cdot c_h$ steps. If a host receives this amount of messages, the clustering is finished. But, if the tree is smaller or larger than it is supposed to be, waiting until $2 \cdot d_G$ are received is no termination criteria. For that reason, the real hop counter must be enclosed to a cluster message. If isolated points are adopted, c_h increases and the new c_h value must be announced.

5 Conclusion

In this paper we presented a new ad hoc applications called *Mobile Profile based Distributed Clustering* (MPDC). Each mobile host is endowed with its user's profile and while the user walks around clusters are to be found, which are composed of hosts with similar user profiles. The ad hoc network is covered with a virtual topology in order to reduce the number of messages. Each host determines a set of hosts, that belong to its local group. Finally, the local groups are exchanged and a global cluster is achieved.

We are simulating MPDC by means of a taxi sharing application and analyzing MPDC with respect to performance issues. But currently, the grouping and clustering depends on the used profile. One major goal will be to find a more generic solution, for other domains with more complex profiles.

References

1. S. Banerjee and S. Khuller. A clustering scheme for hierarchical control in multihop wireless networks. Technical report, Univ. of Maryland at College Park, 2000.
2. S. Basagni. Distributed clustering for ad hoc networks. In *Proceedings of the IEEE International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN), Perth.*, pages 310–315, 1999.
3. M. Berger, M. Watzke, and H. Helin. Towards a FIPA approach for mobile ad hoc environments. In *Proceedings of the 8th International Conference on Intelligence in next generation Networks (ICIN), Bordeaux*, 2003.
4. D. Fasulo. An analysis of recent work on clustering algorithms. Technical report, University of Washington, 1999.
5. C. Fraley and A. E. Raftery. How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal*, 41(8):578–588, 1998.
6. E. M. Gafni and D. P. Bertsekas. Distributed algorithms for generating loop-free routes in networks with frequently changing topology. *IEEE Transactions on Communications*, COM-29(1):11–18, January 1981.
7. E. Kolatch. Clustering algorithms for spatial databases: A survey. Technical report, Department of Computer Science, University of Maryland, College Park, 2001.

8. R. Maitra. Clustering massive datasets. In statistical computing at the 1998 joint statistical meetings., 1998.
9. N. Malpani, J. Welch, and N. Vaidya. Leader election algorithms for mobile ad hoc networks. In *Proc. of the Fourth Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 96–103, 2000.
10. G.-C. Roman, Q. Huang, and A. Hazemi. Consistent group membership in ad hoc networks. In *Int. Conference on Software Engineering*, pages 381–388, 2001.