Dynamic Load Balancing for Position-Based Routing

Matthias Transier  
University of Mannheim  
Holger Füßler  
University of Mannheim  
Martin Mauve  
University of Düsseldorf  
Jörg Widmer  
DoCoMo Euro-Labs  
Wolfgang Effelsberg  
University of Mannheim

Categories and Subject Descriptors: C.2.2 Network Protocols: Routing Protocols  
Keywords: Ad-hoc Networks, Position-based Routing, Load Balancing.

1. INTRODUCTION

Position-based routing algorithms [3] promise an effective delivery of data packets by making use of the geographic positions of the nodes in the network. When developing new algorithms, different factors have to be taken into account. Besides achieving high packet delivery ratios and low delays, the network load is an important characteristic which has to be kept as low as possible. Distributing the load within the network is desirable for different reasons. One of these is the effective usage of the available resources. A good load balancing mechanism helps to raise the packet delivery rate of the transmissions through the network as well as the throughput of the data.

Another point is that through an equally distributed load the nodes’ energy consumption is balanced. Typically, mobile nodes only have limited energy available and if these energy resources are exhausted, the nodes will have to leave the network and will not be available for the routing of packets anymore. Load balancing thus prolongs the lifetime of the nodes and consequently, the lifetime of the ad-hoc network.

2. THE ALGORITHM

In this section we will describe the construction of a position-based routing algorithm which tries to avoid routing data through loaded areas if alternate paths are possible. In general, it should be evaded that a node reaches its forwarding capacity limit because in this case it will have to queue and finally drop data packets. To detect areas which are already heavily loaded, it is required that each node has the ability to sense its current load and to publish its knowledge to other nodes. Hence, the routing decisions will not only be based on the nodes’ positions but also on the corresponding load values.

2.1 Load Detection

To determine the current network load at a node, it is possible to scan the medium in regular intervals and determine how often it is busy—either because the node is sending or receiving or because communicating neighbors are interfering. The scanned values can then be used to calculate the percentage of time in which the medium was busy respectively idle by summing up all measured states and averaging them over a certain period of time. Responsible for this task is the medium access control (MAC) layer—which is also assuring that the medium is idle when a packet is to be transmitted.

There are two different parameters which have to be defined: the sampling interval and the sampling period. Assuming a node is able to send with a sending rate of 2 MBit/s and the average packet size is 250 bytes, the time a packet is on the channel amounts to approximately 1 millisecond. In the case that each packet has a size of about 250 bytes it is thus a good choice to sample once every millisecond. The samples are then transferred into an integer percentage value. For this task, a reasonable number of readings is between one and a few hundred. This means, an actual load figure is available every or every few 100 milliseconds.

2.2 Load Balancing

Let us assume, the algorithm to find the suitable next forwarding hop determines the distances from all neighbors to the target and chooses the one which is located closest to the destination. This node yields the Maximum Forward progress within the transmission Radius and is therefore called MFR neighbor.

The information about the available neighbors is exchanged through beacon messages to all hosts in a node’s transmission range. A beacon is a small packet containing status information like the network address, the physical coordinates and the load value. It is sent out at regular intervals or piggybacked on data packets.

Now the load information has to be included into this decision.

The goal is to choose the node with the minimal load value among the nodes situated closer to the destination than the node itself. As these two functions have to be minimized at the same time, this is a bi-objective optimization problem. If both functions have a common minimum, i.e., a node which is the least loaded and MFR neighbor simultaneously, this would represent a ‘perfect solution’ in terms of multi-objective optimization. Otherwise, a trade-off between an MFR neighbor and a node that is less loaded is necessary.

An intuitive and commonly used way for combining two or more objectives into one is to use a weighted sum. Here, the weight will be used to decide how intense the load information should be included into the routing decision. In order to work out a convex combination which combines distance and load, these values have to be normalized, so that they are of the same dimension. For this purpose, the distance and the load have to be expressed through a real number between 0 and 1.

The distance progress can be expressed as a number between 0 and 1 by using the relation between the distances before and after a data packet was sent. Let us assume, the algorithm to find the suitable next forwarding hop determines the distances from all neighbors to the target and chooses the one which is located closest to the destination. This node yields the Maximum Forward progress within the transmission Radius and is therefore called MFR neighbor.

The information about the available neighbors is exchanged through beacon messages to all hosts in a node’s transmission range. A beacon is a small packet containing status information like the network address, the physical coordinates and the load value. It is sent out at regular intervals or piggybacked on data packets.

New the load information has to be included into this decision. The goal is to choose the node with the minimal load value among the nodes situated closer to the destination than the node itself. As these two functions have to be minimized at the same time, this is a bi-objective optimization problem. If both functions have a common minimum, i.e., a node which is the least loaded and MFR neighbor simultaneously, this would represent a ‘perfect solution’ in terms of multi-objective optimization. Otherwise, a trade-off between an MFR neighbor and a node that is less loaded is necessary.

An intuitive and commonly used way for combining two or more objectives into one is to use a weighted sum. Here, the weight will be used to decide how intense the load information should be included into the routing decision. In order to work out a convex combination which combines distance and load, these values have to be normalized, so that they are of the same dimension. For this purpose, the distance and the load have to be expressed through a real number between 0 and 1.

The distance progress can be expressed as a number between 0 and 1 by using the relation between the distances before and after a data packet was sent.
Load information is only propagated over one hop. Simulations show that using this one-hop strategy is not sufficient to deliver all packets to their destination even when $\alpha$ is nearly 1. Therefore, the load values are calculated based on the own MAC load and the values in the neighbor table. Namely, the current load is weighted by two thirds and the average of the neighbors by one third. In order to avoid that neighbors which are less loaded than a specific node tear this node’s high load value down, the calculated value is only used if it is higher than the node’s currently measured load. Through this technique a noticeable improvement of the packet delivery rate is achieved as the spatial distribution of the load is smoothed and the packets are diverted even before they reach an area that is really loaded.

3. ONGOING WORK

We have proposed a dynamic load balancing algorithm as addition to greedy position-based routing algorithms in mobile ad-hoc networks. Along with the current positions of the neighbors, it uses the current load of these nodes to decide about the next forwarding hop. The load is determined through scanning the medium at the medium access control layer in regular intervals.

The evaluation (see also [5]) shows that the packet delivery rate can be increased significantly through this load balancing algorithm. Additionally, the average connection delay is reduced. These advantages come at the expense of higher path lengths.

There are several possibilities to run further simulations. A first step will be to add some connection patterns which use other load levels than the scenarios that were used so far. Furthermore, the influence of motion within the scenarios has to be simulated.

The main problem of the algorithm is that it seems to increase the likeliness of packets being routed to local optima of the routing conditions where they get lost. On one hand, the combination of our algorithm with existing recovery strategies [2][1] has to be investigated and on the other hand, our algorithm should be adapted to cope with the described problem.

4. REFERENCES