MEGHADOOT: A PACKET RADIO NETWORK ARCHITECTURE FOR RURAL COMMUNITIES

B. S. Manoj, K. R. Bharath Bhushan, S. S. Doshi, I. Karthigeyan, and C. Siva Ram Murthy

High Performance Computing and Networking Laboratory
Department of Computer Science and Engineering, IIT Madras
{bsmanoj,bharath,ssdoshi,ikarthik}@cs.iitm.ernet.in
murthy@iitm.ernet.in

ABSTRACT

In this paper, we present a packet based wireless network architecture for low cost rural community networks. We also present the early results of performance studies made on our architecture, and the measurements taken over the implementation.

I. INTRODUCTION

Meghadoot[1] is a packet based wireless network architecture for low cost rural community networks. Traditional wireless networks for rural telephony, such as Wireless in Local Loop, or satellite telephone systems, require extensive infrastructure for service deployment. The high investments required for such networks and the low revenue prospects in rural regions, discourage commercial service providers from providing communication services in the rural regions. Packet based radio networks are considered as an ideal alternative for low cost community networks, both in the urban developed environments and in the rural environments.

II. RELATED WORK

Lin and Hsu have suggested a prototype implementation for Multi-hop Wireless LANs (MWLANs) in [1]. The protocol proposed by them, called Base-driven Multi-hop Bridging Protocol (BMBP), is implemented both in the Mobile Stations (MSs) as well as in the Access Points (APs) in order to enable multi-hop routing and roaming. The AP that computes the bridging table for a particular MS is known as the *Associated AP* of the MS. When a new packet arrives at an MS, and the MS has a routing entry for the packet's destination, then the packet is forwarded to the next hop node on the path to the destination. Otherwise, the MS sends that packet to its *Associated AP*, possibly through multiple hops. The protocol works by building bridging tables at each node (MS or AP). In the bridge table, the destination sequence number field is used for preventing the formation of loops while routing packets. A node additionally time-stamps each entry in the table in order to avoid stale entries. For each destination in the bridge table, the node records the next hop node on the path to the destination, and the path length (hop count on that path).

[1]Name derived from the Cloud Messenger depicted in Kalidasa's epic love story.

III. MEGHADOOT ARCHITECTURE

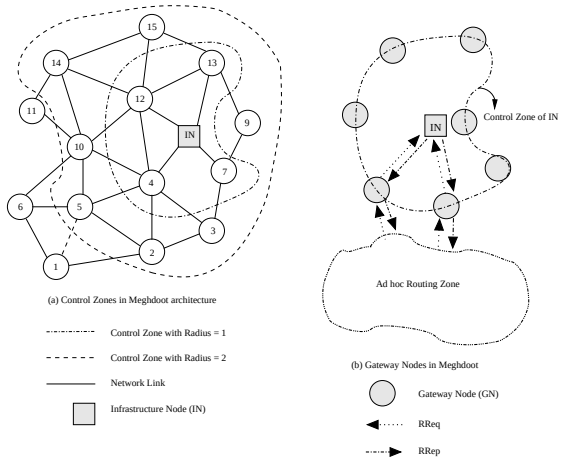


Fig. 1. *An illustration of Meghadoot architecture.*

An illustration of routing process in our Meghadoot architecture is shown in Fig. 1. The Infrastructure Node (IN) controls the routing process in its k-hop neighborhood. Any node registered to the IN assumes that the

routing and other control activities would be taken care of by the IN, and hence it stays away from initiating its own path finding process.

Nodes that are not under the control of the IN operate in the Ad hoc mode, and hence are required to perform self-organization, path finding, and path reconfiguration by themselves. Meghadoot requires the Gateway Nodes (GNs) to hold the additional responsibility of interfacing the nodes in the Ad hoc routing zone (operating in the Ad hoc mode) to the IN in order to efficiently find routes to the nodes inside the control zone of the IN.

Nodes in the Ad hoc routing zone broadcast *Route Request (RReq)* packets in order to find a path to the destination. Every intermediate node that receives the packet forwards it further until the packet reaches the destination. When the destination node receives the packet, it responds by sending back a *Route Reply (RRep)* packet. The disadvantage of this protocol is the high control overhead generated by the broadcast packets used for path finding. Meghadoot aims at reducing this routing overhead with the help of INs. The INs use a protocol called Controlled-Zone routing protocol, which is an extension of the Single Interface Multihop Cellular Networks (SMCN) [2] routing protocol. By using the Controlled-Zone routing protocol, the IN maintains the approximate topology of the nodes within its zone. Whenever a source node (say node S) in the Controlled-Zone needs to send a packet to a destination node (say node D), it sends a *RReq* packet over multiple hops to the IN. The IN runs a shortest path algorithm to find a path to node D, and returns the path found to node S. Node S can now start using this path provided by IN. When a path break is detected by the source node, it sends a new *RReq* packet to the IN for reconfiguring the broken path. If an intermediate node detects a path break, it sends a *Route Error (RErr)* packet to the IN, upon reception of which the IN obtains a new path and informs the source node.

III-A. Infrastructure Based Ad hoc Routing Protocol

In this section we propose an efficient routing mechanism for Meghadoot, called Infrastructure Based Ad hoc Routing Protocol (IBAR). IBAR provides mechanisms for routing of both control and data packets, possibly through multiple hops.

III-B. Issues in Routing for Seamless Multi-hop Relaying

The fundamental problem that we encounter when we consider Meghadoot is the transfer of control information between the MSs and the INs. The MCN routing protocol proposed in [3] assumes that all control information can be reliably sent in a single hop from or to the Base Station (BS) through the control interface. In the routing protocol suggested in [3], each MS can identify its BS as it receives a *Beacon* packet transmitted with a power corresponding to the cell radius. However in Meghadoot, all nodes transmit data using the same transmission power. We have addressed the issues of how to route the control packets efficiently, and also how an MS can find the BS (IN in Meghadoot) that is nearest to it and register with that BS over multiple hops. The main protocol messages used in IBAR are Registration Request (*RegReq*), Registration Acknowledgment (*RegAck*), Route Request (*RouteReq*), Route Reply (*RouteRep*), Neighbor Message/*Beacon*, and Neighbor Update (*NeighUpdt*). Each node (both IN and MS) will periodically generate *Beacon* messages. This message will contain data regarding the set of MSs that the node has a route to, and also the hop count to each of those MSs. When the IN generates its *Beacon* packets, it will send as part of each *Beacon* packet, its own address and the hop count set as 0. When a *Beacon* originated by an IN reaches an MS, the MS processes the received packet, and determines using the hop count metric whether the corresponding IN is the nearest IN. If so, it proceeds on to register with this new IN. Each node keeps track of data such as the list of INs that are accessible to it, the hop count to each such IN, and the next hop nodes to those INs. When a node receives a *Beacon* from the current next hop node to the IN it is registered to, it simply updates the contents of its local data with the new data. It then proceeds to compute the new IN to register to by finding the IN with the smallest hop count. It also keeps track of the current next hop to its nearest IN. In order to reduce the vulnerability of the control path, nodes will not register if the hop count exceeds a particular threshold (k).

The MS then sends a *RegReq* to the nearest IN computed, by forwarding the request to the current next hop to reach that IN. Each MS has the additional responsibility of forwarding such control packets on be-

half of other nodes also. Further, as the request is being propagated towards the IN each node appends its address to the *Path* field of the request packet in order to facilitate routing of the *RegAck* packets. When the *RegReq* reaches the intended IN, that IN generates a *RegAck* packet addressed to the MS that originated the request, through the path specified in the received *RegReq* packet. Thus, the acknowledgment proceeds in a path that is reverse of the path taken by the request packet. This path is copied into the *RegAck* packet such that each MS receiving the *RegAck* packet would know where to forward it. An MS is said to have completed the registration when it receives the acknowledgment sent by the IN. Only then can the MS start participating in routing and data transmission.

An MS on receiving a *Beacon* packet from its neighbor node, records the received power of the *Beacon*. If the difference between the newly received power and the previously recorded power exceeds a particular threshold, the MS will have to send an update message, *NeighUpdt*, to the IN informing it of the new power to be used. Each MS will periodically check to see if the update needs to be sent to the IN. Along with the received power the MS also time-stamps the recorded power with its local system clock. If it does not receive another *Beacon* from the same neighbor within a prescribed time interval then it comes to a conclusion that the neighbor has moved too far away from it, and indicates this to its IN with a certain large negative value for the received signal power field in the update message it sends. During this periodic check that the MS performs, it also checks if any of the nodes that it has recorded as a next hop to some IN have moved away from it. A node is assumed to have moved away if either the signal strength of a packet from that node is negligibly small, or no *Beacon* from that node has been received. In such cases the MS has to update the local data to reflect the new status.

The *NeighUpdt* is forwarded to the IN in much the same way as the *RegReq*, in the sense that each MS will forward the update through its next hop node to the IN. An MS does not have any routing information locally available except for the address of next hop node to the IN it is registered to, which it uses for transmitting control information. Whenever a packet arrives from the higher layers the MS will send a *RouteReq* to its IN requesting the IN to send the path to the destination. The INs are assumed to know the set of MSs that are registered to each of the INs. This information can be transmitted over the wired network that interconnects the INs. The *RouteReq* and *RouteRep* transmission/reception mechanism is similar to the registration mechanism. On receiving the *RouteRep*, the source MS starts sending packets to the destination, using the source routing mechanism.

III-C. The Long Haul Access

Meghadoot is designed to suit remote rural areas and other terrains where an infrastructure based network is not available or is difficult to set up. Hence Meghadoot would require a long haul access network to communicate with the rest of the world. Several mechanisms, such as satellite links, microwave point to point links, and packet based long haul multi-hop relaying, could be used for this purpose. In this work we do not focus on the long haul access scheme.

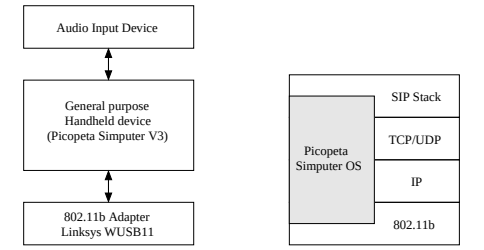
III-C.1. Issues in using IEEE 802.11

Use of IEEE 802.11b frequency bands for long haul multi-hop wireless links, requires special permissions from the regulatory authorities. Even with special permissions, it can lead to problems of interference with the indoor usage of the same band where it is license-free. Since long haul 802.11 links are operating across distances of 20-30 kilometers, the transmission power required is high with appropriate directionality for the transmitting antenna. Explicit licensing of these bands to private long haul operators raises concerns of geographical separation. For example, among the 802.11b channels permitted in USA (11 bands) only 3 channels can be used simultaneously. In Europe, only 4 simultaneous channels can be used. Given this scenario, the long haul operation of 802.11 bands raises concerns of licensing the otherwise license-free bands.

III-D. The 802.11phone

The end user equipment in Meghadoot is an IEEE 802.11 enabled device, either a laptop computer with an 802.11 adapter, or a small handheld device with an 802.11 interface. Meghadoot aims at deployment in rural areas where other communication infrastructure is not available, we envision an 802.11phone (an inexpensive handheld device with 802.11b card). The 802.11phone could either be a general-purpose palm-

top device or a dedicated processor based device similar to a GSM handset. The actual usage of Meghadoot not only aims at voice communication; it aids the rural community to utilize the other applications, such as data gathering, accounting, limited data processing, and for using local language based applications. Therefore, Meghadoot utilized the Picopeta Simputer (Version 3) as one of the devices, for implementation of the 802.11phone. Fig. 2 shows the architecture used in the 802.11phone, implemented in Meghadoot.



(a) Schematic diagram of 802.11 phone (b) Software architecture of 802.11 phone

Fig. 2. *Architecture of 802.11phone.*

III-D.1. Pros and Cons of using a General-purpose Handheld Device

The advantages of using a general-purpose handheld device such as Picopeta Simputer for the implementation of 802.11phone include the following. In addition to voice communication over 802.11b, the village community can use the device for limited computation, data gathering and exchange, and other multimedia applications. The main disadvantage is the processor load by way of the operating system overhead which limits the capability of the 802.11phone to exploit the high bandwidth provided by the 802.11b interface. We notice a degradation in TCP throughput achieved, which was attributed to the operating system overhead.

III-D.2. Choice of Power Source for 802.11phone

Meghadoot aims at providing community networking for voice or data over multi-hop wireless networks, with or without the use of infrastructure nodes. The deployment scenarios in the remote rural communities adds to the additional pressure on the choice of power source for powering the 802.11phone. Meghadoot envisions the usage of cycle dynamo for powering the 802.11phone. The users on the move can charge the 802.11phone by connecting it to the cycle dynamo through a *CB8 charger* (Cycle Based 802.11phone charger) that is part of Meghadoot vision. This enables the village community, including the vegetable sellers, fish sellers, and others who use a bicycle as part of their daily life, to use Meghadoot 802.11phone for their communication. The other options available for charging include solar cell panels which are costlier than the CB8 charger. Fig. 3 shows an illustration of the *CB8 charger*.

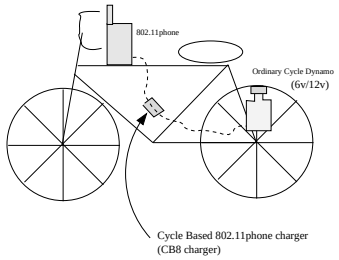


Fig. 3. *An illustration of CB8 charger.*

IV. PERFORMANCE RESULTS

We have simulated the IBAR and BMBP protocols using GloMoSim[4]. We have used the free space propagation model and no-capture model for the radio layer. We have simulated the protocols for different values of node densities and also varying traffic loads. The simulations also include various mobility values ranging from 2 m/s to 10m/s using the random waypoint model.

IV-A. Performance Comparison of IBAR and BMBP

The simulation parameters that were used in the following comparison study are presented in Table I.

TABLE I
SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
Terrain X range	2010m	Transmission range	250m
Terrain Y range	2610m	*Beacon* period	1s
Number of INs	11	Simulation time	5 mins
Cell radius	500m		

In Fig. 4, we compare the overall performance (packet delivery ratio) of the IBAR and the BMBP schemes. IBAR not only out performs BMBP at various node densities, but also shows greater scalability, in the sense

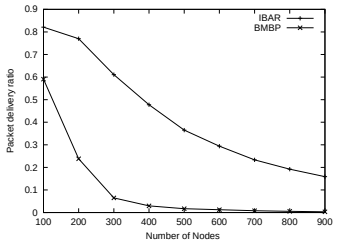


Fig. 4. Performance Vs Node density (no mobility).

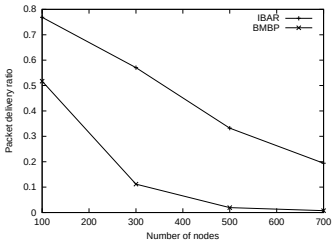


Fig. 5. Performance Vs Node density (mobility 2 m/s).

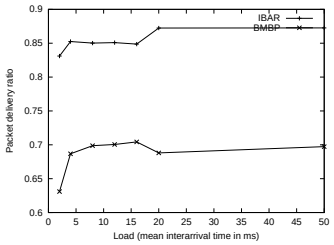


Fig. 6. Performance Vs Load (100 nodes, no mobility).

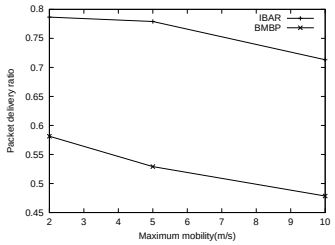


Fig. 7. Performance Vs Mobility (100 nodes, inter arrival time 30ms).

that the performance degradation at larger node densities is not as high as in BMBP. The possible reason for this could be the broadcast nature of the control packets in the BMBP protocol. As a result, the network bandwidth available for data traffic is negligible at high node densities. In contrast, the IBAR scheme tries to send the control packets as much as possible in the unicast mode and hence performs significantly better.

We have also compared the performance of the two protocols at different mobility values, in addition to the no mobility scenario. Fig. 5, compares performance at low mobility of 2m/s. Fig. 7 compares the performance against mobility with a network size of 100 MSs.

Fig. 6 shows the performance of the two schemes at varying load values. The locality represents the percentage of traffic that is intended for a destination within the same cell. In both cases the performance improves at higher locality values, an expected trend, since the length of the wireless path is lesser and traffic directed towards the BS (IN) is lesser. The load values shown represent the mean inter arrival time of UDP packets (with payload 1900 bytes) in milliseconds. We have compared the performance at loads ranging from 2ms inter-arrival time (reasonably high load) to 50ms inter-arrival load (low load). We expect the performance to improve slightly with decreasing load, and the trend is observed for both the schemes.

IV-B. Performance of Picopeta Simputer for 802.11 phone

Table II shows the power consumption on the Picopeta Simputer based 802.11phone with 1024 Byte *ping* packets.

TABLE II
POWER CONSUMPTION OBSERVATIONS

Ping Packet size	Battery Duration
1024 B	1 hour 48 minutes
1024 B (with flooding option)	1 hour 34 minutes

V. CONCLUSION

We had presented in this paper, Meghadoot, a packet based wireless network architecture. Results of our initial simulation experiments were also presented. The low cost and the reasonably good performance of our architecture make it ideally suitable for rural community networks. We do hope that Meghadoot comes into a reality in the near future.

REFERENCES

[1] Y. D. Lin, Y. C. Hsu, K. W. Oyang, T. C. Tsai, and D. S. Yang, "Multi-hop Wireless IEEE 802.11 LANs: A Prototype Implementation", *Journal of Communications and Networks*, vol.2, no.4, December 2000.

[2] V. Sekar, B. S. Manoj, and C. Siva Ram Murthy, "Routing for a Single Interface MCN Architecture and Pricing Schemes for Data Traffic in Multi-hop Cellular Networks", *Proc. IEEE ICC 2003*, vol. 2, pp. 969-973, May 2003.

[3] R. Ananthapadmanabha, B. S. Manoj, and C. Siva Ram Murthy, "Multi-hop Cellular Networks: The Architecture and Routing Protocols", *Proc. IEEE PIMRC 2001*, vol. 2, pp. 78-82, October 2001.

[4] "GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Networks", *Proc. PADS-98*, May 1998.