

# Routing in Wireless Networks of Varying Connectivity

Andrew Grundy, Milena Radenkovic  
Department of Computer Science  
University of Nottingham  
Jubilee Campus, Wollaton Road  
Nottingham, NG8 1BB, UK  
{amg, mvr}@cs.nott.ac.uk

**Abstract**—We are concerned with routing data in networks where the topology ranges from dense to sparse and mostly connected to mostly disconnected. Connection orientated routing algorithms developed for connected environments fail in disconnected environments, due to the instability of these networks. Similarly, forwarding strategies designed to disseminate data hop-by-hop in disconnected environments fail in connected environments, as they send more packets than is required, resulting in congestion. Our algorithm exploits the re-occurring patterns in connectivity arising from the typical routine structure of day-to-day life. We present a functionality overview of the three components of our proposal: contact driven source routing, disconnection tolerant data forwarding and packet scheduling for energy efficiency. We evaluate our early emulations (simulations in ns-2 with real world data) of disconnection tolerant data forwarding, our results show that source routing can be extended to improve its performance in disconnected environments.

**Index Terms**—Routing, Delay Tolerant Networks, Mobile Networks, Wireless Networks

## I. INTRODUCTION

Over recent years, ad hoc networking has been a popular research topic, and as such many routing protocols have been developed and extended in this area, such as Ad hoc On-demand Distance Vector routing (AODV)[2] and Dynamic Source Routing (DSR)[3].

The subject of routing in Delay Tolerant Networks (DTN) is said to have almost become an independent area of research [4]. DTNs use strategies for data dissemination, whereby each node uses various techniques to decide whether a packet should be dropped or replicated, in order to reach the destination node. It is clear that the overheads of epidemic dissemination would not suit dense well connected environments.

Ad hoc protocols, such as AODV and DSR, have not been designed to be tolerant to disconnection, delay or disruption, as they follow the Mobile Ad hoc Networking (MANET) ideology. This ideology, similarly to the internet ideology, adheres to the assumption of contemporaneous end-to-end connectivity. The reality faced by the mobile user is that a more resilient, opportunistic protocol is required.

Integrating DTN and MANET routing has been discussed in the work by Ott et al. [1]. They propose a hybrid approach to communicating in mobile ad hoc networks, utilizing AODV for IP layer routing, with DTN-based forwarding, as detailed in the DTNRG bundle specification [5]. This approach suggests that nodes use AODV not only for end-to-end routing, but to assess the availability of an end-to-end path and discover delay tolerant routing nodes. The delay tolerant routing nodes act as a gateway to the disconnected environment when the end-to-end path does not exist. The proposal [1] requires two different classifications of node, one type that is typically

connected and another that is suited to disconnections, the aim being to exploit each node's characteristics.

We propose the use of one classification of node, following one method of routing, that aims to function in environments connected, partially-connected or disconnected.

The previous work implies that the nodes are either generally connected or disconnected by nature, but this is not necessarily the case, as Figure 1 shows.

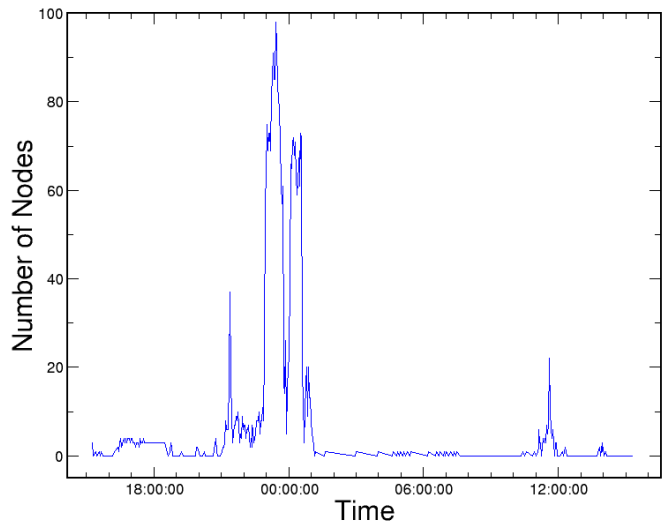


Fig. 1. A graph illustrating the level of connectivity a mobile phone has with other mobile phones by means of Bluetooth communication

Figure 1 illustrates an excerpt from our real life experiments with students logging bluetooth connectivity continuously over multiple days. These experiments result in unidealized data that show variable disconnections throughout the day.

In Figure 1, the node has a mixture of high, moderate and low connectivity during the trial, which illustrates the vast range of environments a mobile node may need to adapt to. At times the user experiences well connected environments with a stable topology which are a quintessential MANET scenario. When the user leaves these areas, the topology might vary considerably, and connectivity might be unstable, representing a more DTN-like scenario.

DTN, MANET, Wireless Sensor Networking (WSN) and Pocket Switched Networking (PSN) concepts are starting to converge within the literature, as each of these areas are concerned with ad hoc networking, mobile or static and connected or otherwise. The different research areas emphasize the importance of different concepts, and as a result a rich knowledge base is developing.

In this paper we propose an algorithm that:

- 1) Considers the effects disconnections have on source routing (Section III-A)
- 2) Reacts to disconnections when forwarding data (Section III-B)
- 3) Schedules packets based upon predicted encounters, to save energy and reduce overheads (Section III-C)

These attributes help in climates of varying connectivity, because, for only a small additional overhead in dense and connected environments, we see a high success ratio in the more sparse DTN topologies, where previously source routing would fail. Section IV details our evaluation.

Firstly, this paper outlines the various research areas that influence our proposal (Section II). Following this, we detail an overview of our protocol (Section III). We continue by discussing each component of our proposal individually (Sections III-A, III-B and III-C). We finish by evaluating our simulations (Section IV), giving a conclusion and detailing future work (Section V).

## II. RELATED WORK

DTN, MANET, WSN and PSN communities are all researching wireless ad hoc networks. Each group focuses on different features and a general solution does not exist, due to conflicting interests.

### A. Mobile Ad hoc Networking

MANETs are an ad hoc wireless network ideology that attempt to establish and maintain end-to-end connectivity, using domain specific routing algorithms to allow wireless technology to behave similarly to wired networks.

This area has developed Proactive and Reactive routing algorithms. Proactive algorithms make a distributed attempt to map the entire network, endeavoring to keep an up to date copy of the map at each node. Reactive algorithms map the routes to destinations that each node sends to, which reduces the size of the map. Furthermore, the node only seeks to find a route should one not already be known, and communication is required. We incorporate reactive routing into our proposal.

### B. Delay / Disconnection Tolerant Networking

The DTN research group (DTNRG) is working with scenarios where TCP/IP networking is either unsatisfactory or simply fails. The recent review by Zhang [6] gives an overview of the main work in this area.

The majority of the routing protocols developed for DTNs have stemmed from store and forward strategies, for instance DakNet [7], and epidemic routing protocols, such as ZebraNet [8].

The simplest DTN routing protocol is epidemic flooding, where messages are simply copied to any node that becomes reachable and does not already have a copy of the message stored. Epidemic routing overheads grow exponentially with the number of nodes, and the amount of traffic that is in the network, and for this reason it is unsuitable for larger networks.

DieselNet, another DTN deployment scenario, is a network of buses in Amherst, MA. Maxprop [9] and RAPID [10] protocols have been developed for use in this domain, optimizing delivery using constrained replication. A packet is routed by means of replication until a copy reaches the desired destination. However, because of limited resources such as bandwidth, the algorithm has to decide how these packets should be replicated in the network in order to optimize

a specified routing metric, such as minimizing the average packet delay.

PROPHET [11] uses the probability of success with its contacts to determine if a data packet should be replicated to them or simply dropped, and is used in a DTN deployment providing communication to reindeer herders in northern Sweden [12].

Building delay tolerance into existing routing algorithms has been investigated successfully in the past, in projects such as Delay Tolerant Link State Routing (DTLSR)[13]. Link state routing worked for this scenario as the nodes were static. Without mobility changing the topology, the focus of this protocol was dealing with large delays.

We too wish to take an existing routing concept, which in our case is source routing, and extend it to succeed in networks that become disconnected, whilst still performing well in connected environments.

### C. Pocket Switched Networking

PSNs [14] mainly target highly mobile devices such as personal digital assistants and mobile phones. PSNs are a special case of delay tolerant networking, with most of the work originating from the HAGGLE project [15]. The work in this area has explored the available forwarding opportunities presented by human mobility and social patterns, such as "Pocket Switched Networks and Human Mobility in Conference Environments" [16], where the concept of contacts is introduced.

Contacts are further utilized in Bubble rap[17], where custodian node compare centrality values when encountering other nodes, to determine whether or not to forward their stored packets to them. The concept aims to bubble the message up through a social hierarchy, in order for it to stand a greater chance of meeting the destination node.

We use this contact concept in order to develop a scheduling system, which predicts contact encounters based on previous history, in order to avoid superfluous packet transmissions.

PSNs use contacts to selectively disseminate data packets, in a controlled form of epidemic routing. We monitor contacts to schedule transmissions and broadcasts, rather than to make routing decisions.

### D. Wireless Sensor Networking

WSNs also target ad hoc networking; with a strong focus on energy efficiency through routing, such as Energy Efficient Route Discovery (EERD) [18], and through power management, such as Efficient Node Discovery (END) [19].

EERD also extends DSR [3], incorporating passive clustering with delayed intelligence (PCDI) [20], with additional variables for the speed of the nodes in order to prevent the most mobile nodes receiving RouteRequest packets. As a result nodes that move less are more likely to become part of a route, increasing route stability.

END exploits the node's pattern of contact encounters to determine the node's duty cycle. Reinforcement learning techniques are used to detect and dynamically change the times at which a node beacons its existence, and listens for other beaconing nodes.

In order to ensure that a node detects other nodes, and is detected effectively, END spends more of the node's energy budget at the time slots predicted to be busier. END still spends some of the energy budget monitoring the non-active time slots, in order for the schedule to evolve.



Contact logging (as described in Section III-C) informs a source node when a particular contact is going to be in transmission range, which we call  $T_{wait}$ . Each route a node has to a particular destination has an RTT value, which we refer to as  $T_{route}$ . We calculate the estimated delivery time to a particular destination via each each potential first hop with the formula  $T_{delivery} = T_{wait} + T_{route}$ . This calculation requires us to have both a knowledge of routes, and ideally a reasonably mature contact history.

A RouteRequestAck packet is returned to a node that has sent a RouteRequest, as an acknowledgement that the node receiving the RouteRequest packet is attempting to acquire a path to the specified destination. In the case where the node already has a route to the specified destination, or the node happens to be the destination, a RouteReply message is returned. The RouteReply informs the requester of the route, and the time taken for the custodian to acquire that route.

If a route is found to be broken, a RouteFail packet is sent back along the route. Each node receiving the route failure information removes the route from its routing table, or re-costs it, and removes the node from which the RouteFail packet was received from the set of contacts that have performed the source routing process to the given destination, allowing the contact to acquire a new route.

### B. Disconnection Tolerant Data Forwarding

In DSR, when a node receives a MAC level failure, it sends a RouteFail packet back along the route. When a node receives a RouteFail message, the given route is removed from its routing table, in order to prioritize the use of alternative routes (should any be known). If no other routes are known, the route acquisition process begins again.

Our protocol attempts to resend data packets that failed in transmission, scheduling retransmission at a later time. The node stores a packet retry (PR) value with the packet, and decrements this each time the packet is resent, as illustrated in Figure 2.

Discovering routes costs the network valuable time and energy, and so we feel that a route should not be discarded without sufficient opportunity to succeed.

We must balance giving a route additional opportunities to succeed, against the cost of delays due to following erroneous paths. As such, when the PR value reaches zero, we then return a RouteFail packet, in order to limit the total quantity of resends and to reduce the overall delay.

In Section III-C, we discuss packet scheduling, and how scheduling makes retransmitting the data packet less costly and more likely to succeed. Without packet scheduling, the delay tolerant data forwarding detailed here is brute force and, as such, may soon make the network congested.

### C. Energy Efficiency and Packet Scheduling

We have extended the exponentially weighted moving average (EWMA) smoothing filter (as described in END [19] and illustrated in figure 4) to operate on a per contact basis, in order to predict both the best time to schedule packets to be transmitted (as shown in Formula 1), and when to beacon and listen for contacts (as shown in Formula 2 and illustrated in figure 4).

$$M_{tn} = C_{tn} * \alpha + M_{tn} * (1 - \alpha), t = 0, \dots, T-1, n = 0, \dots, N-1 \quad (1)$$

In Formula 1,  $M_{tn}$  is the estimated encounter frequency (EEF) for the time period  $t$ , for the contact  $n$ . The value  $C_{tn}$  is the count of connections to node  $n$ , during the time period  $t$ .  $T$  is the total number of time periods, and  $N$  is the total number of contacts. The  $\alpha$  value dictates how responsive the node is to changes in its pattern of daily encounters.

In END, the value assigned to  $C_t$  is the number of encounters that have finalized in time period  $t$ , gives limited knowledge about the connections in the network. We instead count the number of beacons received per node throughout each time period, giving us a better understanding of the connectivity a node has with its neighbors.

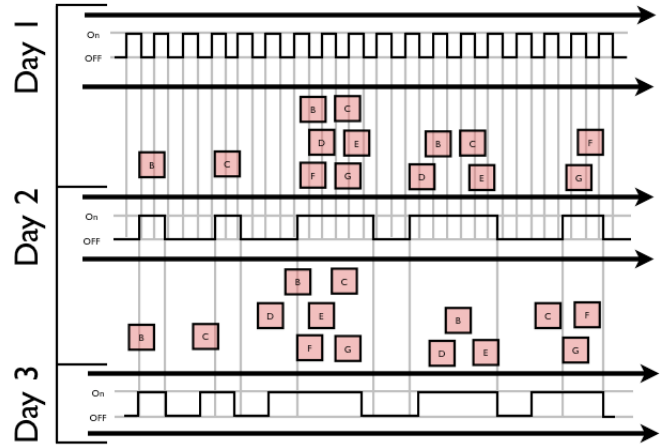


Fig. 4. Duty cycling example (Learning when contacts occur)

We use these previously generated EEF values in order to schedule packet transmissions, as shown in Figure 2. We also use this information to calculate  $T_{beacon}$  (beaconing frequency) for the current time period, as shown in Figure and in Formula 2.

If a device beacons too frequently, energy will be wasted. If a device does not beacon enough, it is likely that other nodes will not encounter it. Figure 5 shows an example node's contact log, in which contact B does not know about this node, and similarly, this node does not know about contact B, as this node's beacon frequency interval is greater than the duration of contact with B.

$$T_{beacon} = 1 / \left( \frac{\sum_{n=0}^{N-1} M_{tn}}{\sum_{n=0}^{N-1} \sum_{t=0}^{T-1} M_{tn}} B \right) \quad (2)$$

In order to calculate the  $T_{beacon}$  value, we use the fraction 1 over the product of the total number of broadcasts, and the sum of EEF values for the current timeslot, over the sum of all EEF values. Should the  $T_{beacon}$  value be greater than the  $T_{timeslot}$  value, then, to allow the node to detect contacts outside of the routine experienced up to this point, the node beacons with a probability of  $P = \frac{T_{timeslot}}{T_{beacon}}$ .

If a node beacons and a contact responds, the node will then instead beacon at  $T_{keepalive}$  intervals, for as long as at least one contact is in range. A node uses these beacon messages to build a profile of the contacts it meets, as shown in Figure 5.

Figure 5 details a rigid approach to beaconing, that has serious negative effects. If the  $T_{beacon}$  frequency was adaptive, it would be likely that contact node B would have been discovered and entered into the contact log.

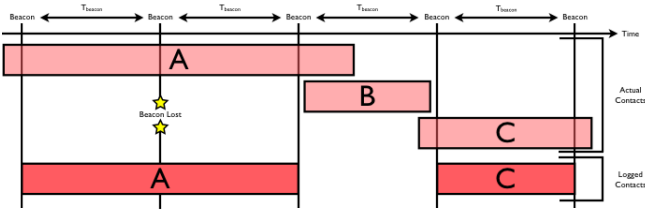


Fig. 5. Contact logging example (taken from [21] and extended)

We decided to use a contact logging technique similar to that of the seal2seal paper [21], as illustrated in Figure 5. We consider a connection to a given contact to be intact, provided that the node has received a packet from the contact within the last  $3 * T_{keepalive}$  intervals. This allows for minor disruptions to the connection, and as a result we keep our findings realistic, as depicted in Figure 5 with contact A.

#### IV. EVALUATION

##### A. Protocol emulation and Scenario Definitions

Here, we evaluate the Disconnection Tolerant Data Forwarding property of our protocol, using the ns-2 [22] network simulator.

The nodes in the experiment move according to the mobility traces generated by the bovine movement emulator, as detailed in 'Practical MANETs for Pervasive Cattle Monitoring' [18].

The nodes move around an area of 35.75m x 29.25m. The whole emulation lasts for three hours and is repeated five times over networks with 10, 25, 50, 75, 80 and 100 nodes. We compare DSR with disconnection tolerant data forwarding (DSRR), as detailed in Section III-B.

Our current implementation does not have the packet scheduling aspect of our proposal implemented, and so we currently replay the data packets at five second intervals. We restrict the number of retransmissions by decrementing a packet's TTL value every time it is re-sent, and sending a RouteFail message back along the path when the TTL reaches zero. The current implementation is suboptimal, and will be superseded in future work, as discussed in Section V.

Because of the amount of time a node may spend retrying a route that may well be broken, we expect the delays to be high in comparison with DSR, as DSR simply rejects a route at the first sign of failure. Furthermore, due to the simplifications in our implementation, our expectation is that the delays will be even greater in our initial trials than we expect in the finished protocol. We also predict that, because our proposal is more persistent, the success ratio should be higher than with DSR.

##### B. Emulation Results

The outcome of our emulations are shown in Figures 6 and 7. Each Figure has four groups of results. These four groups consist of DSR over 802.11, DSR over Zigbee, DSRR over 802.11 and DSRR over Zigbee. Each group has five points per network size, each point being the value of a single run. In these graphs, the triangle represents DSR over 802.11, and the ex denotes DSRR over 802.11; the circle signifies DSR over Zigbee, and the plus represents DSRR over Zigbee.

Figure 6 details the per run mean average delay of data packet transmissions. The results for 802.11 show that DSRR does not noticeably increase the delays in the network when the network is well connected. We expected the delays to increase with the density of the network, as we expected

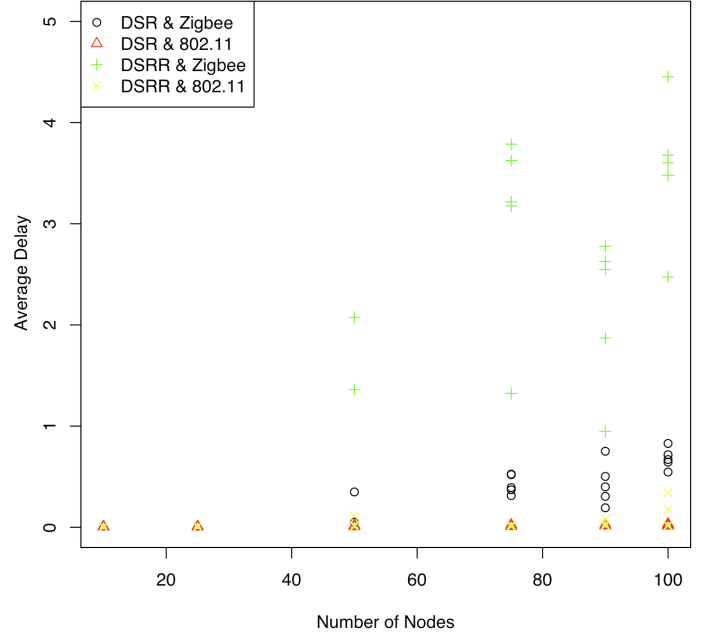


Fig. 6. DSR and DSR with Disconnection Tolerant Data Forwarding (DSRR) Mean Average Delay (Seconds) over networks of 10, 25, 50, 75, 90 and 100 nodes

packet collisions to occur. The graph shows that, in fact, DSRR only marginally increase delays in the dense networks.

The Zigbee results show that in networks of 10 and 25 nodes, both DSR and DSRR fail to successfully transmit any packets. We assume that the networks are so disconnected that no end-to-end route can be established.

Networks of 50 nodes only succeed two out of five runs, and networks of 75, 90 and 100 nodes succeed on all five runs. As we expected, the average delay is much longer with DSRR than with DSR, with the worst case delay being nearly five seconds. This is equivalent to each packet resending approximately once, on average.

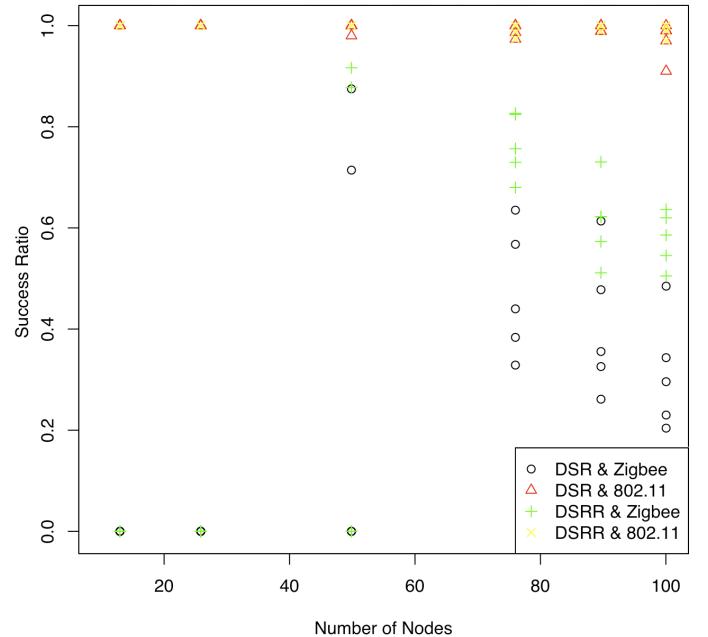


Fig. 7. DSR and DSR with Disconnection Tolerant Data Forwarding (DSRR) Success Ratio (%) over networks of 10, 25, 50, 75, 90 and 100 nodes

Figure 7 shows the success ratio of data packet transmissions, for DSR and DSRR. We expected the success ratio for

802.11 to be similar to the average delay results, with DSRR falling behind in dense scenarios, due to the nature of the implementation.

However, the results for 802.11 show that DSRR has a better success ratio than DSR in each of the different densities of network. This appears to be due to the reduced quantity of routing packets DSRR sends, as we transmit data packets again if the first attempt is unsuccessful.

The Zigbee results show that, in networks of 10 and 25 nodes, both DSR and DSRR fail to successfully transmit any packets, and as a result the success ratio is zero. Networks of 50 nodes only succeed in only two out of five cases, and in these two, DSRR performs better than DSR.

All the runs for networks of 75, 90 and 100 nodes succeed in sending packets over Zigbee, similarly to the networks of 50 nodes. The networks of 75, 90 and 100 nodes all show that DSRR gives a more robust performance than DSR, without the addition of disconnection tolerant data forwarding, even in the current implementation.

The outcome of our emulations, as detailed in Figures 6 and 7, show that DSRR is an improvement on DSR, but that DSRR still requires end-to-end connectivity in order to source route.

An additional concern with source routing that arises from this evaluation is that a data packet could be sent out into a DTN, and be lost under the current protocol. We cannot assume that a RouteFail packet will be received in the event of a failure, and as a result we must devise a way of calculating a timeout value that, on expiration, retransmits the data at the source. If an acknowledgment packet is received confirming successful transmission, this timer can be canceled.

## V. CONCLUSION AND FUTURE WORK

This paper has identified two major concepts – disconnection tolerant source routing (Section III-A), and disconnection tolerant data forwarding (Section III-B) – that when applied to a network of varying connectivity, would perform well in connected and disconnected environments.

We also proposed an extended version of energy efficient node discovery (Section III-C), that tunes duty cycling to make nodes more responsive when their contacts are likely to be around. We schedule data and routing control transmissions to our contacts, storing a higher level of granularity of information about each individual contact, resulting in a reduced number of superfluous transmissions.

We have evaluated disconnection tolerant data forwarding (Section IV). We assessed our implementation, finding that the average transmission had increased delays in the network, in comparison with DSR, but that the data packets were more likely to successfully reach the destination node.

In addition to assessing our proposal for disconnection tolerant data forwarding, we would also like to complete our software implementation, and continue to build on our analysis.

We would like to vary the threshold for the number of retries per hop, in order to discover the optimal value, to evaluate energy efficiency through data packet scheduling, as detailed in Section III-C, and to evaluate and implement the extensions to DSR to find routes in disconnected environments, as described in Section III-A.

## ACKNOWLEDGMENTS

We would like to gratefully acknowledge Dr. Bartosz Wietrzyk for providing the code from his Energy Efficient

Route Discovery development, and the data from his cattle monitoring work [18].

## REFERENCES

- [1] J. Ott, D. Kutscher, and C. Dwertmann, "Integrating DTN and MANET routing," in *Proceedings of the 2006 SIGCOMM workshop on Challenged networks*. ACM New York, NY, USA, 2006, pp. 221–228.
- [2] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing," in *Second IEEE Workshop on Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA'99, 1999*, pp. 90–100.
- [3] D. Johnson, D. Maltz, J. Broch *et al.*, "DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks," *Ad hoc networking*, vol. 5, pp. 139–172, 2001.
- [4] K. Fall and S. Farrell, "DTN: an architectural retrospective," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 5, pp. 828–836, 2008.
- [5] K. Scott and S. Burleigh, "Bundle Protocol Specification. draft-irtf-dtnrg-bundle-spec-02, Work in progress," September 2005.
- [6] Z. Zhang, "Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: Overview and challenges," *IEEE Communications Surveys & Tutorials*, vol. 8, no. 1, pp. 24–37, 2006.
- [7] A. Pentland, R. Fletcher, and A. Hasson, "Daknet: Rethinking connectivity in developing nations," *Computer*, vol. 37, no. 1, pp. 78–83, 2004.
- [8] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebraNet," *COMPUTER ARCHITECTURE NEWS*, vol. 30, no. 5, pp. 96–107, 2002.
- [9] J. Burgess, B. Gallagher, D. Jensen, and B. Levine, "Maxprop: Routing for vehicle-based disruption-tolerant networks," in *Proc. IEEE Infocom, 2006*.
- [10] A. Balasubramanian, B. Levine, and A. Venkataramani, "Dtn routing as a resource allocation problem," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM New York, NY, USA, 2007, pp. 373–384.
- [11] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," *Lecture Notes in Computer Science*, pp. 239–254, 2004.
- [12] A. Doria, M. Uden, and D. Pandey, "Providing connectivity to the saami nomadic community," *generations*, vol. 1, no. 2, p. 3.
- [13] M. Demmer and K. Fall, "DTLSR: delay tolerant routing for developing regions," in *Proceedings of the 2007 workshop on Networked systems for developing regions*. ACM New York, NY, USA, 2007.
- [14] P. Hui, A. Chaintreau, R. Gass, J. Scott, J. Crowcroft, and C. Diot, "Pocket Switched Networking: Challenges, feasibility and implementation issues," *Lecture Notes in Computer Science*, vol. 3854, p. 1, 2006.
- [15] J. Scott, P. Hui, J. Crowcroft, and C. Diot, "Haggle: A networking architecture designed around mobile users," *IFIP WONS*, vol. 2006, 2006.
- [16] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*. ACM New York, NY, USA, 2005, pp. 244–251.
- [17] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay tolerant networks," in *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*. ACM New York, NY, USA, 2008, pp. 241–250.
- [18] B. Wietrzyk, M. Radenkovic, and I. Kostadinov, "Practical MANETs for Pervasive Cattle Monitoring," in *Networking, 2008. ICN 2008. Seventh International Conference on*, 2008, pp. 14–23.
- [19] V. Dyo and C. Mascolo, "Efficient Node Discovery in Mobile Wireless Sensor Networks," *Lecture Notes in Computer Science*, vol. 5067, pp. 478–485, 2008.
- [20] M. Gosnell, R. Albarelli, M. Cheng, and B. McMillin, "Energy balanced broadcasting through delayed intelligence," in *International Conference on Information Technology (ITCC)*. IEEE Computer Society, 2005.
- [21] A. Lindgren, C. Mascolo, M. Lonegan, and B. McConnell, "Seal2seal: A delay-tolerant protocol for contact logging in wildlife monitoring sensor networks," in *Proceedings of IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS08)*, 2008.
- [22] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and Y. Haobo, "Advances in network simulation," *Computer*, vol. 33, no. 5, pp. 59–67, 2000.