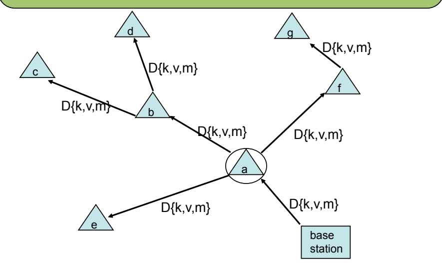
The Disseminated ACKnowledgment (DACK) Protocol for Data Collection in Wireless Sensor Networks

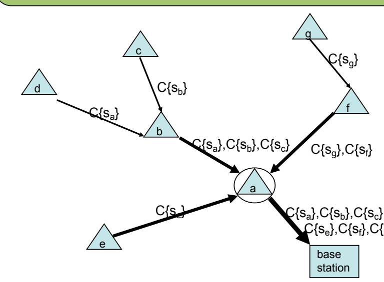
John-Paul Arp and Bradford G. Nickerson
University of New Brunswick Faculty of Computer Science

Dissemination



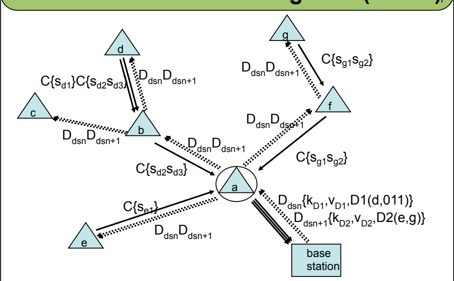
Dissemination algorithms can reliably propagate a message to all nodes in a large network within seconds. In the example above the **base station** sends a message **m**, with a key **k**, and a version number **v**. Each node must have a common key tied to a variable, or array where the message is stored. When motes detect beacons from other motes advertising newer version number **v** for **k**, they request the newer version.

Collection



Data Collection algorithms make a best effort to deliver data from nodes in a WSN to a common sink, or base station. In the above example nodes a through g each send one collection packet. Modern WSN collection algorithms can achieve over 90% reliability in many real world deployments.

Disseminated ACKnowledgment (DACK)



The DACK protocol disseminates acknowledgments of data collected via a collection algorithm. In the above example, solid lines represent the path of a collection packet from the node to the base station. Dashed lines represent a dissemination packets being disseminated from the base station to each node in the network. Node **a** is a **gateway** node connected to a **base station**. In this snapshot, collection packet **1** from node **d** is lost between node **c** and node **b** on it's path to the gateway. Once the dissemination packet $\mathbf{D_{DSN}}$ is received at mote **d**, mote d realizes it will need to resend the data that was contained in packet **1** in its next report.

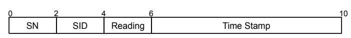
Motivation

- Improve reliability of data collection in wireless sensor networks (WSNs) using dissemination to achieve end-to-end acknowledgement of data samples

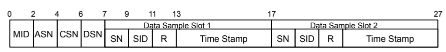
Objectives

- Minimize Energy Consumption
- Test the protocol via simulation, and experiment
- Integration with the Sensor Web Language (SWL) WSN development platform

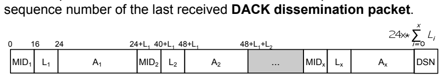
Packet Structures



Structure of a **Data Sample** with **Reading**, a sequence number **SN** and a **Time Stamp**. **SID** is a sensor identifier which maps the sample to the sensor channel number, type, and data acquisition parameters.

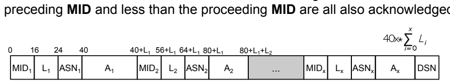


Structure of a **DACK collection packet** with two **data samples**, each placed in a "slot". **MID** is the mote identifier. **ASN** is the **SN** of the last acked data sample. **CSN** is the **SN** of the last sent data sample. **DSN** is the sequence number of the last received **DACK dissemination packet**.



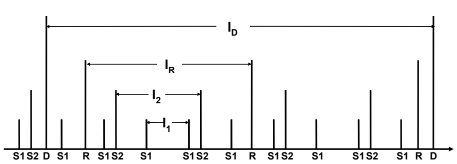
Bit structure of a **D1 type DACK dissemination packet** containing **x** acknowledgments consisting of a sequence of **(MID, L, A)** triplets. **A** is a vector of **'1's** and **'0's**, in which **'1'** indicates an acked data sample and **'0'** indicates an unacked data sample. **L** is the bit length of **A**. The **n**-th bit of **A** references the **SN = ASN+n**.

Bit structure of a **D2 type DACK dissemination packet** containing **x** acknowledgments consisting of a sorted sequence of **MIDs** and **'-'** characters. The **'-'** characters indicate that all **MIDs** greater than the preceding **MID** and less than the proceeding **MID** are all also acknowledged.



Bit structure of a **D3 type DACK dissemination packet** containing **x** acknowledgments consisting of a sequence of **(MID, ASN , L, A)** 4-tuples.

Timing Diagram



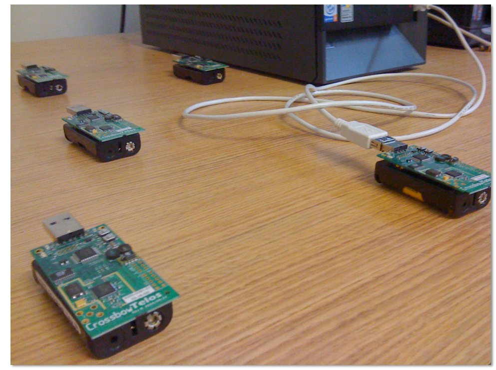
An example of the timing intervals required on a mote using the DACK protocol.

I_1 and I_2 = two (changeable) sampling time intervals tied to two separate sensors on a single mote,

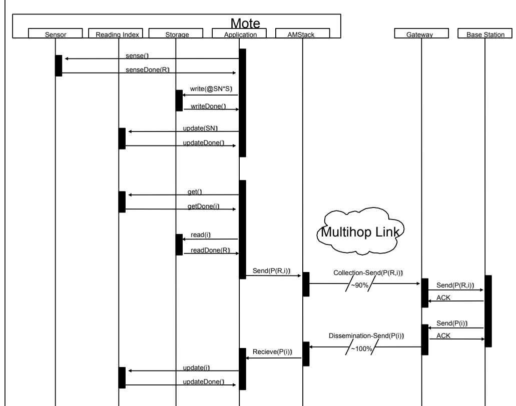
I_R = mote's report interval, and

I_D = interval at which disseminated data should be received on the mote from the base station.

A WSN with TelosB Motes



Sequence Diagram



Sequence diagram of a node and base station using the DACK Protocol. AMStack indicates the TinyOS active message radio stack.

Early Simulation Results

Early Simulations show that the DACK protocol can can recover more than 50% of lost samples in small WSNs in a noisy environment. DACK was implemented in TinyOS using DIP[1] for disseminationand CTP[2] for collection. In the 16 node simultion two nodes eventually became disconnected entirely, so the results are not exact.

	Noise	Samples Generated	Lost Samples	Recovered Samples	Dissemination Cost
16 node grid simulation*	-40.0 to -90.0 dB	~4800	1202	771	~600

[1] Kaisen Lin and Philip Levis, Data discovery and dissemination with DIP, IPSN '08: Proceedings of the 2008 International Conference on Information Processing in Sensor Networks (ipsn 2008) (Washington, DC, USA), IEEE Computer Society, 2008, pp. 433–444.

[2] Kyle Jamieson, Omprakash Gnawali, Rodrigo Fonseca and Philip Levis, CTP: Robust and efficient collection through control and data plane integration, Stanford Information Networks Groups. 2008. Tech Report. SING-08-02. Available at: http://sing.stanford.edu/pubs/sing-08-02.pdf