An Experimental Study of Fractional Cooperation in Wireless Mesh Networks

Anthony Calce, Nariman Farsad, and Andrew W. Eckford
Dept. of Computer Science and Engineering, York University
4700 Keele Street, Toronto, Ontario, Canada M3J 1P3
E-mail: calce@cse.yorku.ca, nariman@cse.yorku.ca, aeckford@yorku.ca

***Abstract*—Fractional cooperation is a decentralized, low-complexity wireless networking protocol in which nodes have the ability to dynamically select a fraction of its resources to commit to forwarding, and where sources may use more than one relay to convey information to the destination. In this paper, an implementation and a series of experiments are presented to demonstrate the practical performance and effectiveness of fractional cooperation. A low-complexity MAC layer protocol is used, which employs fractional cooperation using LT codes in the absence of central coordination. Experimental results from real-world trials are given, which show that this protocol can maintain a reasonable throughput when nodes are abruptly entering and leaving, making it ideal for a dynamically changing system, such as an ad-hoc network. The redundancy of information seen in the network makes this scheme robust to unfavourable channel conditions.**

I. Introduction

With the prevalence of wireless networks and ever increasing applications, the scale of wireless networks grows year by year. Concurrently, the need for high-speed data transmission also grows every year as the Internet becomes an essential part of society. However, as the size of the wireless networks grow, maintaining higher data rates becomes ever more challenging. One solution to this problem is spreading the load across different nodes in the network.

The nodes of wireless networks can be distributed over wide areas, where multiple hops may be needed in order for information to reach a destination. Cooperation [1], [2] provides a possible solution to this problem by allowing each node to assist its neighbours in transmitting information to a data sink. The basic cooperative system consists of three nodes: a source, a relay, and a destination. The relay can employ various cooperative schemes such as decode-and-forward [3] and demodulate-and-forward [4], [5], to assist the source in transmitting its information bits to the destination. Furthermore, where links are affected by fading, cooperation can be used to increase the spatial diversity of the system.

In a large scale wireless network, there are multiple sources and multiple relays, where each source node is typically in radio range of more than one relay. Furthermore, in systems such as wireless mesh networks (WMN) [6], each node can be a source and a relay. When a traditional cooperative scheme is employed, each source has a dedicated number of relay(s), where each relay forwards all the source's transmission bits. However, relays might have their own information to transmit to the destination or might be unwilling to dedicate all their resources to relaying; furthermore, allocation of resources is a difficult problem. To overcome this issue, fractional cooperation [7], [8] can be employed, where each relay node can forward a *fraction* of the sources' transmission bits. A related fractional strategy in cellular networks was independently proposed in [10], where mobiles are permitted to use multiple base stations to increase throughput to cell-edge users for multiple-input multiple-output (MIMO) systems.

In this work, we develop a low-complexity medium access control (MAC) layer protocol, employing fractional cooperation, and test it on practical sensor networking hardware in real-world experiments. Our contributions in this work are as follows:

- We implement our MAC layer protocol on Imote2 sensor nodes, creating a WMN. To the best of our knowledge, this is the first work where fractional cooperation is implemented in a real wireless system. Compared to simulations, real-world experiments are uncommon in the wireless literature.
- Through experiments on our implemented WMN, we show that the proposed protocol can reliably transmit information over the network, even when each node in the network relays a fraction of the source's transmission, and there is no centralized coordination of the fractions.
- Using the proposed protocol, nodes can reliably enter and exit the network at any time. We also show that when our protocol is employed, the throughput decreases gracefully as relay nodes exit the network. This is in contrast to a traditional cooperative system, where an exodus of relay nodes can result in a sudden, precipitous drop in throughput. Based on the obtained results we conclude that fractional cooperation is suitable in large scale networks where nodes enter and exit the network constantly.

II. Communication Protocol

A. Fractional Cooperation

In fractional cooperation, each node selects a fraction $\alpha \in [0, 1]$, representing the fraction of any neighbour's transmission that it will relay; for example, if $\alpha = 0.5$ and a neighbour transmits a packet of 100 bits, then the relay will choose 50 of those 100 bits for retransmission. The manner in which these

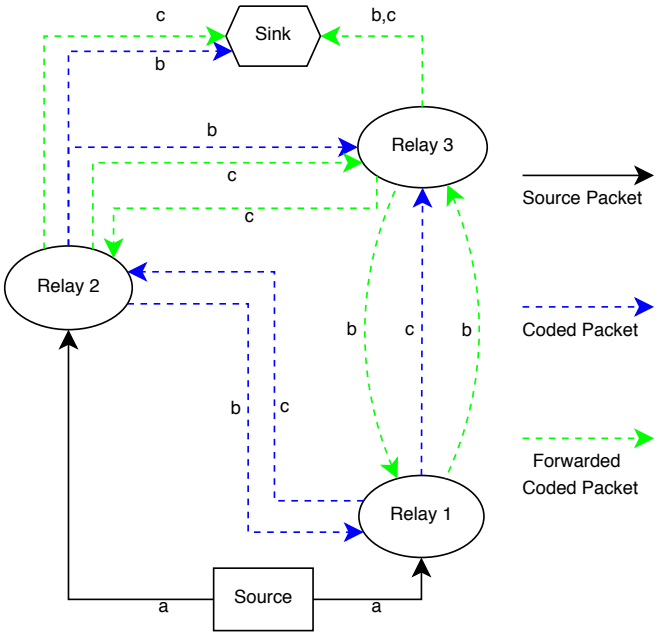

Fig. 1: A multiple relay system

bits are selected is explained below. Both the choice of α and the identities of the bits to be relayed are made by the relay, and not coordinated with other nodes (however, it is assumed that the destination knows which symbols are selected). See also [7], [8] for more details.

Since the relays are not coordinating the selection of fractions, it is impossible to ensure that every source symbol is ultimately selected for relaying. As a result, it is essential to use an erasure-correcting code to "fill in the blanks" and recover any symbols which were missed during the selection process. The LT code [11] is used, as it has excellent decoding performance and low encoding complexity (which is essential for our hardware).

B. Relay Model

The main goal of this protocol is to achieve a low complexity communication scheme which incorporates fractional coded cooperation. In order to fulfill this requirement, all nodes in the model communicate by broadcasting. A few nodes are selected as sources, but all nodes (except the sink) have the ability to behave as relays. A source cannot be its own relay. Nodes in this system follow two rules:

- Source nodes transmit *uncoded* information packets.
- If a node receives an uncoded packet, it will select a fraction α of the bits in the packet, use those bits to create a coded packet (using an LT code), and broadcast the coded packet to all its neighbours.
- If a node receives a coded packet, it will rebroadcast the packet in its entirety without decoding or re-encoding.

A small example of this relay model is shown in Figure 1. A source node broadcasts packet a which is seen by two relays. Coded packets b and c are generated by these relays and are broadcast. At this point, any relay which receives a coded packet will forward it without any processing. Since this is broadcast scheme, nodes may receive duplicate packets throughout this process, for example, relay 1 forwarding packet b back to relay 2. These transmission links have been omitted from the figure for ease of viewing.

Since the nodes are unaware of the success of transmission, or the location of the sink, we use a packet time to live (TTL) (measured as a maximum number of hops) as a simple means of preventing flooding. Furthermore, we assume that the destination has knowledge of, but no control over, the identities of source symbols that the nodes are selecting for relay. As we will discuss below, this is equivalent to knowledge of the LT code employed by each relay node; in practice, this is implemented by making the destination aware of each relay's random number seed, which can be expressed compactly.

Due to the nature of the protocol, every source packet sent will trigger the generation of a certain number of coded and forwarded packets. A source packet consists of k symbols, with l bits per symbol. Traffic can be expressed as a sum of the number of source, coded, and forwarded packets generated per unit time. Each one of the s sources will generate a packet with size kl. For every source packet in the system, r coded packets are generated with size αkl. For every coded packet in the system, $r-1$ forwarded packets are generated with size αkl. We define the quantity Φ, expressed in bits per second (bps), to be the upper bound of traffic seen in the channel provided all nodes are within range of each other, and there is no interference between communication links. Given s sources, r relays, and a source packet rate λ expressed in packets per second, let n_{source}, n_{coded}, and $n_{\text{forwarded}}$ represent the number of source bits, coded bits, and forwarded bits generated per packet. Then Φ is given as follows:

$$\begin{aligned}\Phi &= (n_{\text{source}} + n_{\text{coded}} + n_{\text{forwarded}})s\lambda \\ &= \big((kl) + r(\alpha kl) + r(\alpha kl)(r-1)\big)s\lambda \\ &= kl(1+\alpha r^2)s\lambda\end{aligned} \tag{1}$$

When source nodes are also allowed to relay, the above equation becomes:

$$\Phi = skl\big(1+\alpha(r+s-1)^2\big)\lambda \tag{2}$$

To further reduce unnecessary forwarding, each node is able to cross-check packet id numbers against a buffer in order to determine duplicates. This guarantees us that Φ remains constant for systems which require a large TTL.

C. Encoding and Decoding Process

This section describes the encoding/decoding process from the generation of a source packet to its reconstruction at the sink; for ease of implementation, the process is slightly different from the LT codes found in the literature, in that operations are performed symbol-wise.

The process begins when a source node broadcasts an uncoded information packet in the network. Each relay node, upon receiving an uncoded packet, will begin its encoding step.

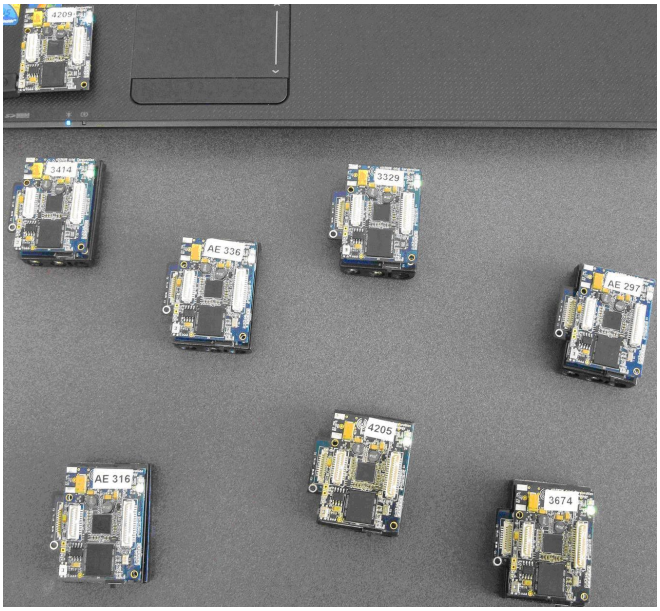

Fig. 2: Experiment setup

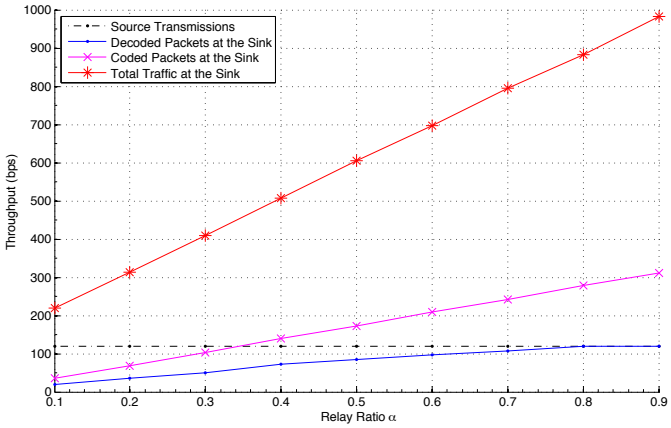

Fig. 3: Plot illustrating the decoding efficiency for varying α values

Given k information symbols received, a relay will generate $\lfloor \alpha k \rfloor$ LT coded symbols which are then encapsulated into a coded packet and broadcast. In [11], an LT coded symbol is generated as follows:

- A degree $1 \leq d \leq k$ is chosen for the encoding symbol with some probability given by the robust solition distribution [11].
- d symbols are chosen uniformly at random from the set of k information symbols. This subset of symbols is referred to as the neighbour set of the encoding symbol.
- The value of the encoding symbol is determined by performing the *symbol-wise* exclusive-or (XOR) operation between the value of each neighbour symbol.

Given distribution constants $c = 0.1$, $\delta = 0.01$, and $k = 30$, the robust solition distribution is approximated as: $P_r(d = 1, 2, 3, 4, 5, 6) = (0.13, 0.26, 0.10, 0.05, 0.04, 0.42)$

The decoder needs to be aware of both the degree of a coded symbol and its set of neighbour symbols. As noted in the previous section, this is accomplished by using a random seed in the encoding process which is utilized by a pseudorandom generator at the decoder to reproduce the exact distributions used in the encoding process.

Upon receiving a coded packet, the sink will recreate the degree and neighbour set of each coded symbol (as seen in columns 3 and 4 of Table I). The sink handles the decoding of packets individually such that all coded symbols are analyzed before moving on to the next coded packet.

A symbol is considered decoded when its degree is equal to one (since there is only one neighbour symbol $x \leq k$, the value of the code is equal to the x^{th} information symbol). Once all degree one symbols are handled, the algorithm tries to reduce the degree of every other symbol to one. This is done by comparing already decoded information symbols to the neighbour set and observing if that symbol is an element of it. If so, a XOR operation is performed between the value of the coded symbol and the decoded information symbol, the element is removed from the neighbour set, and the degree is reduced by one. This process is repeated until the source message is fully reconstructed.

TABLE I: Sink processing of coded packets generated in a one-source, two-relay system

Information Symbol

23	11	230	92

Coded Symbol x

id_1	seed	92	28	11

Coded Symbol y

id_2	seed	186	166	177

Code Symbol Processing

	Symbol	Degree	Neighbour Set	Value	Decoded Information Symbol
id_1	1	1	{4}	92	(, , ,92)
	2	2	{1,2}	28	(, , ,92)
	3	1	{2}	11	(,11, ,92) (23,11, ,92)
id_2	1	2	{2,4}	87	(23,11, ,92)
	2	4	{1,2,3,4}	166	(23,11,230,92)
	3	3	{2,3,4}	177	unused

A short example of this is seen in Table I; this example assumes $\alpha = 0.75$ and $k = 4$. The source analyzes symbols $id_1(1, 2, 3)$ and determines that symbols 1 and 3 can be automatically decoded. It then cross-references the second decoded information symbol with coded symbol 2 to retrieve information symbol 1. It then continues to process symbols $id_2(1, 2, 3)$ and following the same procedure, manages to decode the information packet. Since the packet was decoded after analyzing symbol 2, all subsequent symbols and coded packets received are discarded (this includes packets received by the forwarding process).

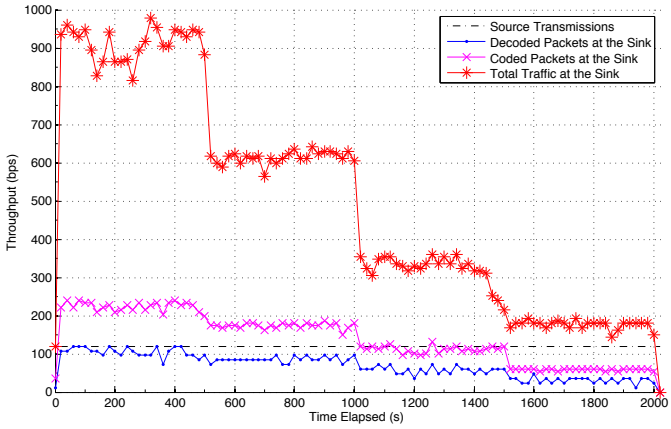

Fig. 4: One source transmitting to four initial relays, one relay being shutoff every 500s.

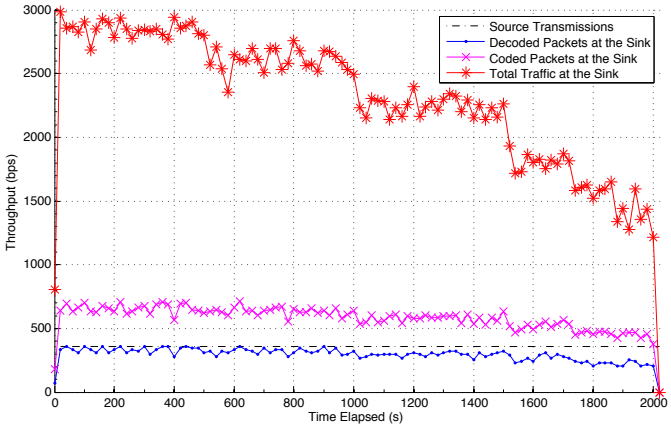

Fig. 5: Three sources transmitting to six initial relays, one relay being shutoff every 500s

D. Limitations

The sink node experiences a performance penalty if the number of information symbols k is large. The main contributing factor to this problem is that the decoding algorithm in this model was designed in a recursive fashion. It was shown in [11] that the decoding step runs in linearithmic time. Due to the simplicity of the communication protocol, large traffic is seen at the sink if there are too many nodes in close proximity. Under these conditions, this model suffers from channel saturation.

III. Experiments

This section describes the implementation of our protocol using Imote2 [12] sensor nodes which are factory configured to run Microsoft .NET Micro Framework 2.0 and are programmed with the C# language. Figure 2 displays a setup used during experimentation.

A. Encoding and Decoding Simulation

A simulation of the encoding/decoding process was performed and it was shown that over 5000 trials, given an information packet consisting of k symbols and constants c and δ, a reception overhead of about 60% was observed. That is to say, the decoder would need to see $1.6k$ coded symbols on average to successfully complete a single decode. Given $\alpha = 0.5$, that would equate to 3.2 coded packets.

B. Results

In this context, the primary features of fractional cooperation are robustness, fairness, an increase in transmission range, and the absence of central coordination. Four experiments were designed to test these aspects of our protocol.

A sensitivity analysis was first done on α to determine what effect it would have to total throughput seen at the sink. Figure 3 describes these results. Each data point was generated using a single source transmitting to three relays for 200s. Increasing α results in a trade-off between higher decoding efficiency and extra traffic.

TABLE II: Parameters used in experiments

Parameters	
Information symbols, k	30
Symbol length, l	8 bits
Relay ratio, α	0.5
Source send rate, λ	0.5 packets per second

The next experiment is set up using five nodes; one acting as the sink, one as the source, and four as dedicated relays. Nodes are positioned such that they are all within range of each other. During the course of the experiment, one relay node is shutoff every 500s. This experiment is designed to show that the protocol is robust to relays leaving the system. For simplicity, we chose α to be the same for all relays. The results are divided into 20s time buckets, and are observed over 2000s in Figure 4.

As relays are dropped from the system, the sink node experiences a decrease in the amount of coded packets it receives, which results in a reduction of decoding efficiency. The traffic seen at the sink is consistent with Equation 2. For every source packet sent, there are r coded packets generated (assuming all relays are within range of the source). The simulation in Section III-A tells us that under the system parameters, an average of 3.2 coded packets are required to successfully decode. Since there are four relays present at the start of the experiment, we can expect to reach close to 100% decoding efficiency during this period.

The third experiment mimics the previous in all aspects, except there are three sources broadcasting instead of one. As previously mentioned in Section II-B, all nodes are able to relay. As such, for every s sources in the network we have an extra $s-1$ relays. This test is designed to show the robustness of the protocol when there are a large number of relays.

As Figure 5 illustrates, the presence of two extra source nodes greatly increases sink traffic in our model; so much so that it reaches channel saturation and traffic no longer conforms to Equation 2. As we can see, the sink is decoding

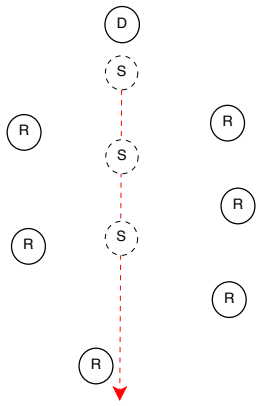

Fig. 6: Six relays positioned over a field size of 180cm×300cm. A source node is moving farther away from the sink destination.

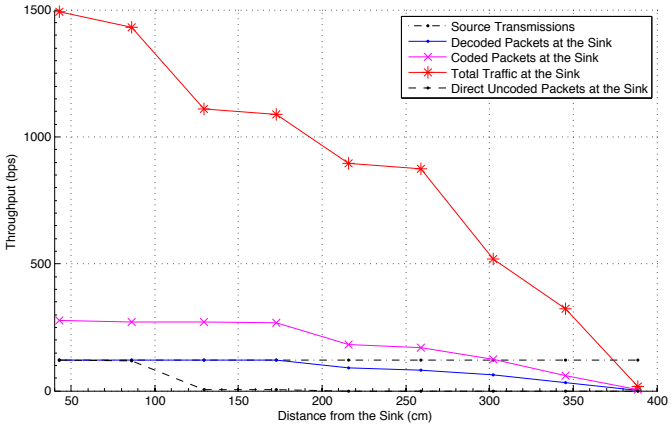

Fig. 7: Plot illustrating the benefits of fractional cooperation over direct non-routed paths

at almost the same rate as the source is transmitting, even as relays are being dropped. This shows that provided there are enough relays in the system, an exodus of a few nodes will not cause significant performance degradation.

The final experiment is set up as seen in Figure 6. Six relay nodes were positioned over an area of about five square meters. A source node is set up to broadcast for 200s at fixed locations away from the sink. This test is intended to show to that our protocol achieves a significant boost in range over direct transmission.

As depicted in Figure 7, throughput of decoded packets at the sink tends to gracefully falloff with distance. Our protocol achieves over a three fold increase in transmission range over a direct non-routed path. The extra redundancy of duplicated packets allows us to employ this protocol in unfavourable channel conditions and expect reasonable throughput.

IV. Conclusions

This paper has presented a low-complexity MAC layer protocol which employs fractional cooperation using LT codes in the absence of central coordination. A node has the ability to dynamically select a relay ratio, which controls how much, if any, of its resources it is willing to commit to forwarding. As we have shown, a system with many relays is robust to noisy channel conditions because there is a large amount of duplicate information in the network. Due to the broadcast nature of this protocol, packets are able to avoid forwarding loops by cross-checking packet ids, thus making this scheme practical for larger systems which have a high packet TTL.

Our approach is particularly useful for hardware with complexity constraints, such as the iMote hardware often found in sensor networks. Further, our results prove that this protocol can maintain decent throughput when nodes abruptly enter or exit the system, making this ideal for a dynamically changing system such as an ad-hoc network.

For future work, we plan to perform further experiments on a larger testbed once the decoding algorithm is restructured to run more efficiently. This would allow us to increase the number of information symbols sent by a source, resulting in greater system throughput.

References

[1] A. Sendonaris, E. Erkip, and B. Aazhang, "User cooperation diversity-part I: system description," *IEEE Trans. Commun.*, vol. 51, pp. 1927–1938, Nov. 2003.

[2] A. Sendonaris, E. Erkip, and B. Aazhang, "User cooperation diversity, Part II: Implementation aspects and performance analysis," *IEEE Trans. Commun.*, vol. 51, pp. 1939–1948, Nov. 2003.

[3] A. Nosratinia, T. Hunter, and A. Hedayat, "Cooperative communication in wireless networks," *IEEE Commun. Mag.*, vol. 42, no. 10, pp. 68–73, October 2004.

[4] D. Chen and J. N. Laneman, "Modulation and demodulation for cooperative diversity in wireless systems," *IEEE Trans. on Wireless Commun.*, vol. 5, no. 7, pp. 1785–1794, Jul. 2006.

[5] J. P. K. Chu and R. S. Adve, "Implementation of co-operative diversity using message-passing in wireless sensor networks," in *Proc. IEEE Globecom*, St. Louis, MO, pp. 1167–1171, Dec. 2005.

[6] Y. Yan *et al.*, "Performance Analysis of IEEE802.11 Wireless Mesh Networks," *IEEE Int. Conf. Commun.,* pp. 2547-2551, May 19-23 2008.

[7] A.W. Eckford *et al.*, "Low Complexity and Fractional Coded Cooperation for Wireless Networks," *IEEE Trans. Wireless Commun.,* vol. 7, pp. 1917-1929, May 2008.

[8] A.W. Eckford *et al.*, "Low-complexity strategies for cooperative communications," *in Cooperative Wireless Commun.,* ed. Y. Zhang, H.-H. Chen, and M. Guizani, CRC Press: Boca Raton, FL, pp. 53-72, 2009.

[9] N. Farsad, A. W. Eckford , "Resource Allocation via Linear Programming for Multi-Source, Multi-Relay Wireless Networks," *in Proc. IEEE International Conference on Communications (ICC)*, pp.1–5, May 2010.

[10] N. Kusashima *et al.*, "Fractional Base Station Cooperation Cellular Network," *7th Int. Conf. on Inform., Commun. and Signal Processing,* pp. 1-5, 8-10, Dec. 2009.

[11] M. Luby, "LT Codes," *in Proc. 43rd Symp. on Found. of Comp. Sci.,* pp. 271-280, Nov. 16-19 2002.

[12] Imote2: http://memsic.com/support/documentation/wireless-sensor-networks/category/7-datasheets.html?download=139