On the Average-throughput Performance of Code-based Scheduling Protocols for Wireless Ad Hoc Networks

Carlos H. Rentel and Thomas Kunz
Dept. of Systems and Computer Engineering, Carleton University,
Ottawa, Canada. Emails: {crentel, tkunz@sce.carleton.ca}

MobiHoc 2005

Carleton UNIVERSITY
Canada's Capital University

Introduction

We investigate the use of codes for the scheduling of transmissions in a wireless Ad Hoc network. Codes have been traditionally used for error detection and correction when information is transmitted over noisy channels. However, scheduling is another lesser known application of codes.

The first relevant use of codes for scheduling purposes can be traced back to the work of G. Solomon [1], who proposed the use of Reed-Solomon codes for the scheduling of time-frequency slots in a Frequency-Hoping Multiple Access system with the objective of minimizing *hits* (packet collisions) among multiple terminals. Chlamtac and Farago [2] proposed the use of polynomial evaluation in a Galois field to schedule the transmissions in a multi-hop packet radio network; their approach is similar to the use of Reed-Solomon codes (i.e., Reed and Solomon proposed the construction of Reed-Solomon codes using polynomial evaluation), however they were the first to incorporate code-based scheduling in the context of Wireless Ad Hoc networks albeit with some restrictions.

We generalize the approach in [2] by noticing that *any* code (linear or non-linear) can in principle be used for scheduling purposes in a Wireless Ad Hoc network and, in this work, investigate the average throughput performance of Reed-Solomon codes using a coding theory approach. The authors in [3] proposed a generalization of the method in [2] in terms of Orthogonal Arrays (OAs), however, many codes produce OAs (including Reed-Solomon codes) and others do not. Furthermore, different code constructions can produce different performance results, therefore it is important to focus on more constructive ways to view the scheduling problem based on codes. Figure 1.a shows de concept of a code, and Figure 1.b shows the time-slotted structure used by a code-based scheduler.

Figure 1.a

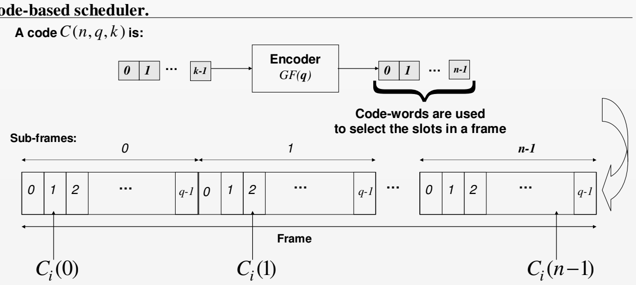

Figure 1.b

Code-based Scheduling vs. Contention-based scheduling

A lower-bound throughput of a code-based scheduling protocol can be written as follows,

$$G_{\min} = \frac{n-(n-d_{\min})I_{\max}}{nq} \qquad (1)$$

Where n is the length of the code, q is the dimension of the Galois Field over which the code is defined, $d_{\min}$ is the minimum distance of the code and $I_{\max}$ is the maximum number of interferers a node can have. Eq. (1) is the ratio of the maximum number of free-colliding slots to the frame size for a code-based scheduling protocol.

One of the most important advantages of a code-based scheduling protocol is its potential for a minimum throughput guarantee as given in (1). However, some constrains must be satisfied, in particular,

$$q^k \geq N$$

$$n \geq (n-d_{\min})I_{\max}+1 \Rightarrow n \geq \frac{1-d_{\min}I_{\max}}{1-I_{\max}} \qquad (2)$$

Regardless of the minimum performance guarantee, it is important to study the average throughput performance of code-based scheduling protocols and, in particular, compare that performance with the one obtained by a representative contention-based protocol.

<u>We have analytically compared the average throughput of a code-based protocol based on the columns of an Orthogonal Array and slotted-ALOHA</u>. Each node is assigned a unique column of an Orthogonal Array of strength two. Note that this comparison has a good degree of generality since many codes can produce the columns of such arrays.

An average throughput of code-based scheduling protocols based on OAs was derived in [3], and is given by,

$$\overline{G}_i^c = \sum_{w=0}^{n}(n-w)\frac{\binom{n}{w}C_i^w}{\binom{q^k-1}{i}}\frac{1}{nq} \qquad (3)$$

Where C_i^w is the number of different ways in which i code-words coincide in w specific positions with a given code-word, and nq is the frame size. The way to compute C_i^w is given in [3] and it is based on generating functions (that same method is used here to obtain our OA results).

The slotted-ALOHA protocol is relatively simple to implement if compared to a code-based scheduling protocol. Assuming that we have knowledge of the number of neighbors i of a given node x, then the probability of successful transmission of x in slotted-ALOHA is given in (4)

$$\overline{G}_i^s = p(1-p)^i \qquad (4)$$

where p is the transmission probability of a node. The optimum value of p is,

$$\frac{\partial \overline{G}_i^s}{\partial p} = (1-p)^i - i(1-p)^{i-1}p = 0 \Rightarrow p^* = \frac{1}{1+i} \qquad (5)$$

Substituting p^* into (4) we obtain the average throughput of a node using slotted-ALOHA and when it is surrounded by i interferers,

$$\overline{G}_i^s = \frac{1}{i+1}\left(1-\frac{1}{i+1}\right)^i \qquad (6)$$

Figure 2 shows $\overline{G}_i^c$ and $\overline{G}_i^s$ for a number of i neighbors. The curves shown are for the different sub-frame sizes (q) between 3 and 27 (see Figure 1.b) , and for OAs of strength two with $n = q$. The OA curves match the ones in [3]. The OA curves with smaller values of q decay more rapidly as the number of neighbors of the given node increases. As can be observed, the expected throughput of slotted-ALOHA is always larger than the expected throughput of the OAs considered. OAs of strength three are not much different in terms of average throughput when compared to strength two as observed in the results shown in [3]. <u>*The latter fact strongly suggests the need for average performance improvements of code-based scheduling*</u>. Note however, that the metric in (3) represents an ensemble average of a given node's throughput. That is, (3) combines all the possible code-words available in a code even when the number of nodes is less than the total number of available code-words. However, it is often the case that only a sub-set of the code-words will be used in a *particular* network. The next section presents an algorithm that selects N code-words of a code in order to achieve more average throughput when N (the number of nodes in the network) is smaller than the total number of code-words available (q^k) in a given code $C(n,q,k)$.

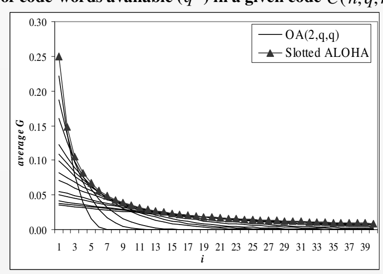

Figure 2. Average throughput of OAs (strength two) and slotted-ALOHA

Code-selection

The code-selection procedure chooses code-words that have mutual maximum-average Hamming distance. We tested the performance of this algorithm by computing the sample-average throughput of a given node using the selected code-words of the singly-extended RS codes that maximize Eq. (1) subject to (2). The code-selection algorithm is formally described as follows,

Assigned code-words are denoted as w_i where $i = \{0,1,\ldots,N-1\}$ and N is the number of nodes in the network. Unassigned code-words are denoted as W_j where $j = \{0,1,\ldots,q^k-1\}$.
The code-words are selected as follows

1. Set $i = 0$ and start by picking a code-word W_j randomly for the given node (node 0) out of the q^k code-words. That is, $w_{i=0} = W_x$, where x is a uniformly distributed and discrete random variable that takes values from the set $= \{0,1,\ldots,q^k-1\}$
2. Set , $i = i+1$
3. Pick a code-word W_j for the next node (node i), such that its Hamming distances d_{W_j,w_i} with respect to the already picked code-words satisfies,

$$\max_{W_j \neq w_i}\left\{\frac{1}{i}\sum_{i=0}^{i-1} d_{W_j,w_i}\right\}, \quad (7)$$

4. Set, $i = i+1$ and repeat step 3 until $i > N-1$.

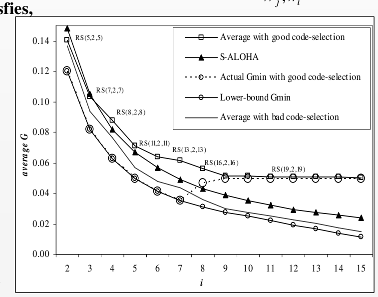

Figure 3. Average throughput of RS codes with the code-selection algorithm and slotted-ALOHA

Conclusion

The need for improved average performance of code-based scheduling is highlighted and a method for code-word selection is proposed. Initial results show potential for the improvement of the performance of code-based scheduling protocols through the use of code-word selection. Code-based scheduling protocols have the potential to out-perform contention-based scheduling protocols in average and minimum guarantee performance metrics.

References:

[1] G. Solomon, "Optimal frequency hopping sequences for multiple-access," Proceedings of the Symposium of Spread Spectrum Communications, vol. 1, AD-915 852, 1973, pp. 33-35.
[2] I. Chlamtac and A. Farago, "Making transmission schedules immune to topology changes in multi-hop packet radio networks," IEEE/ACM Trans. Networking, vol. 2, No. 1, Feb. 1994, pp. 23-29.
[3] V.R. Syrotiuk, C. J. Colbourn and A. C. H. Ling, "Topology-transparent scheduling for MANETs using orthogonal arrays," International Conference on Mobile Computing and Networking, San Diego, CA, 2003, pp. 43-49.