Wireless Broadcast with Network Coding: Dynamic Rate Selection

Song Yean Cho and Cédric Adjih

Abstract Network coding is a novel method for transmitting data, which has been recently proposed. In this article, we study using network coding for one specific case of multicast, broadcasting. Precisely, we focus on (energy-) efficient broadcasting in a multi-hop wireless networks: transmitting data from one source to all nodes with a small number of retransmissions. It is known that the efficiency of network coding is essentially determined by the selected rates of each node. Our contribution is to propose a simple and efficient method for determining a rate selection. Our method adapts dynamically and uses only local dynamic information of neighbors: *Dynamic Rate Adaptation from Gap with Other Nodes (D.R.A.G.O.N.)*. The rationale of this rate selection method is detailed from some logical arguments. Experimental results illustrate the behavior of the method, and its excellent performance.

1 Introduction

Seminal work from Ahlswede, Cai, Li and Yeung [1] has shown that *network coding*, where intermediate nodes mix information from different flows, can achieve higher throughput for multicasting than classical routing. Since then, network coding has been shown to be specially useful in the context of wireless communications and in the context of multicast.

In this article, we will focus on using network coding as an *efficient* method to broadcast data to the entire wireless network, where the cost is the total number of transmissions. For single source broadcast, results from information theory indicate that optimality may be achieved by a simple coding

S. Y. Cho
Hipercom Team, LIX, École Polytechnique, France, e-mail: `Cho@lix.polytechnique.fr`

C. Adjih
Hipercom Team, INRIA Paris-Rocquencourt, France, e-mail: `Cedric.Adjih@inria.fr`

method, *random linear coding* [2]. Then the key parameter for a given instance of a network is essentially the average retransmission rate of the nodes — established on the entire duration of the stream (even for packet networks, see [3] for a recent synthesis of such results): the broadcast efficiency in terms of the number of transmissions can be evaluated with the average rates.

As a result, finding an optimal solution to minimize the cost consists of finding the optimal average rate of every node [6]: a *rate selection*. The problem is then reformulated:

• Problem statement: how to find a rate selection with good performance?

In fact, [4–6] have shown that optimal rate selection may be effectively obtained by solving linear programs – thus in polynomial time; and possibly in a distributed fashion.

However, a different, even simpler, approach is possible: previous work [7] has explored a simple strategy that essentially sets the same rate on every node, and had shown that it is asymptotically optimal (and at least 64 % more efficient than any method not using network coding). However this is an asymptotic result for networks with infinitely increasing density and area.

For actual networks, this rate selection may be adjusted with simple heuristics using local topology: such heuristics have been explored in [8]. They performed well; however in special cases such as sparse networks, it was found that the performance decreased, and local topology information does not seem sufficient to detect such cases.

In this article, we start from the same general idea of finding not necessarily an optimal rate selection, but a simple and efficient rate selection. We also start from the logic used in methods in [8], but with the novel approach of using simple dynamic information to adapt the rates, rather than static topology information: the rate of each node is set from the current state of the network. There are several advantages in doing so; first, the problematic cases which cannot be detected from static local topology information, may be discovered from dynamic information. Second, in a real network, the network itself evolves dynamically (for instance packet loss rate or topology may change), and hence the rate selection can no longer be a fixed constant rate. These dynamic features require dynamic adaption in any case.

Our key contributions are the proposal of a new heuristic for rate selections, its analysis, and an experimental investigation of the performance with simulations.

The rest of the paper is organized as follows: section 2 details the general framework, section 3 provides some background about network coding, section 4 details the new heuristic, `DRAGON`, section 5 analyzes performance with experimental results and section 6 concludes.

2 Framework

As indicated in the section 1, this article focuses on an efficient network coding method to perform broadcast.

We assume that a fixed number of nodes are present in a multi-hop wireless network. Inside the network, there is one source which will transmit a finite number of packets (referred to as the *generation size*). Other nodes are retransmitting the packets with network coding (as described in section 3).

The objective is broadcasting: eventually every node will have obtained a copy of the packets originated from the source.

The metric for efficiency is defined as the total number of transmitted packets per source packet. In general terms, as further described in section 5.1, the performance of a network coding method is derived from the average rates of the nodes. We focus on methods which explicitly select rates of the node: rate selection (see section 3.3).

In this article, we introduce a heuristic for selecting rates: `DRAGON`. It does not assume a specific type of network topology. However, it is suitable for networks where one transmission reaches several neighbors at the same time, that is, for wireless networks.

3 Network Coding Background

3.1 Random Linear Coding

Network coding consists in performing *coding* inside the network. One notable method of coding is *linear coding* [10] (see also [11]).

Starting with the assumption that all packets have identical size, with linear coding, the packets can viewed as vectors of coefficients of a fixed Galois field $\mathbb{F}_q^n$. Then this makes possible to compute linear combinations of them: this is the coding operation in linear coding. Since all packets originate from the source, at any point of time a node v will possess a set coded packets, every of which is a linear combination of the original source packets:

$$\text{i}^{th}\text{coded packet at node } v : p_i^{(v)} = \sum_{j=1}^{j=k} a_{i,j} P_j$$

where the $(P_j)_{j=1,...,k}$ are k packets generated from the source. The sequence of coefficients, $[a_{i,1}, a_{i,2}, \ldots, a_{i,n}]$ is the *coding vector* of coded packet $p_i^{(v)}$.

With linear coding, an issue is selecting coefficients for the previous linear combinations. Whereas centralized deterministic methods exist, [2] presented a coding method, which does not require coordination of the nodes, *random linear coding*: when a node wishes to transmit a packet, computes a linear combination of all the coded packets that it possess, with randomly selected coefficients (α_i), and sends the coded packet: coded_packet $= \sum_i \alpha_i p_i^{(v)}$ This approach is made practical, with the proposition in [12], to add a special header containing coding vector of the transmitted packet.

3.2 Decoding, Vector Space, and Rank

The node will recover the source packets $\{P_j\}$ from the packets $\{p_i^{(v)}\}$, considering the matrix of coefficients $\{a_{i,j}\}$ from section 3.1. Decoding amounts to inverting this matrix, for instance with Gaussian elimination.

Thinking in terms of coding vectors, at any point of time, it is possible to associate with one node v, the *vector space*, Π_v spawned by the coding vectors, and which is identified with the matrix. The dimension of that vector space, denoted D_v, $D_v \triangleq \dim \Pi_v$, is also the *rank* of the matrix. In the rest of this article, by abuse of language, we will call *rank of a node*, that rank and dimension. Ultimately a node can decode all source packets when the rank is equal to the the total number of source packets (*generation size*). See also [13]. It is a direct metric for the amount of useful received packets, and a received packet is called *innovative* when it increases the rank of the receiving node.

3.3 Rate Selection

In random linear coding, the remaining decision is *when* to send packets. This could be done by deterministic algorithms; for instance, [13] proposes algorithms which take a decision of sending or not another packet upon reception.

In this article, we consider "rate selections": at every point of time, an algorithm is deciding the rate of every node. We denote $\mathcal{V}$ the set of nodes, and $C_v(\tau)$ the rate of the node $v \in \mathcal{V}$ at time τ. Then, random linear coding operates as indicated on algorithm 1. With this scheduling, the parameter

---

Algorithm 1: Random Linear Coding with Rate Selection

1.1 Source scheduling: the source transmits sequentially the D vectors (packets) of a generation with rate C_s.

1.2 Nodes' start and stop conditions: The nodes start transmitting when they receive the first vector but they continue transmitting until themselves **and their neighbors** have enough vectors to recover the D source packets.

1.3 Nodes' scheduling: every node v retransmits linear combinations of the vectors it has, and waits for a delay computed from the rate distribution.

---

which varies, is the delay, and we choose to compute it as an approximation from the rate $C_v(t)$ as: delay $\approx 1/C_v(t)$.

Notice that the performance, the number of transmissions per source packet, is identical after scaling the rates of all nodes (source and intermediate nodes), so we will assume that, in practice, such an adjustment is globally done so that the network operates below maximum channel capacity and with low loss rate. Furthermore, for convenience in the presentation, we assume that the unit of the rates $\{C_v\}$ is arbitrarily set so that source rate is $C_s = M$, where M = average number of neighbors.

3.4 Performance of Wireless Network Coding

One notable result relates to the performance of random linear coding, and is one of a series of information-theoretic results about network coding, starting with [1]. It is the following [3]: with random linear coding, when the rate of the other nodes are set, the source may transmit at a rate arbitrarily close to some fixed rate, the *maximum broadcast rate* and at the end of the broadcast, all the destinations can decode with a probability p_e. The error probability p_e may be made arbitrarily small, by increasing the generation size, (and independently of field size). As a result, in practice, an additional termination protocol may be used to ensure that all nodes can decode, but it has asymptotically negligible cost, and the performance is determined by the maximum broadcast rate.

The maximum broadcast rate of the source may be computed as the *min-cut of an hypergraph* using the average rates of other nodes [1, 6, 14].

Precisely, it is known that the source s can transmit individually to any node in the network, with a rate, which may be computed as the capacity of the min-cut of the network (viewed as an hypergraph). Denote $C_{\min}(s, t)$ the capacity of the min-cut between s and t. This is the *min-cut max-flow theorem* [16] and does not require network coding.

The groundbreaking result from [1] and several following generalizations (for instance [3, 14]) is that the source may transmit to several nodes at the same time (multicast), using network coding, and at the rate of the bottleneck destination. Thus, denoting $C_{\min}^{\mathrm{all}}(s)$ the maximum broadcast rate of the source s for broadcast, we have $C_{\min}^{\mathrm{all}}(s) \triangleq \min_{t \in \mathcal{V} \setminus \{s\}} C_{\min}(s, t)$.

The dynamic behavior of the rank of one node is also known: assume that the source is transmitting with a rate C_s lower or equal to the maximum broadcast rate . Then the rank of one node t, will grow linearly with time τ as: $D_t(\tau) \approx C_s \times \tau$. If the source is transmitting with a rate too large, then it will be bounded by the min-cut to the destination t: $D_t(\tau) \approx C_{\min}(s, t) \times \tau$.
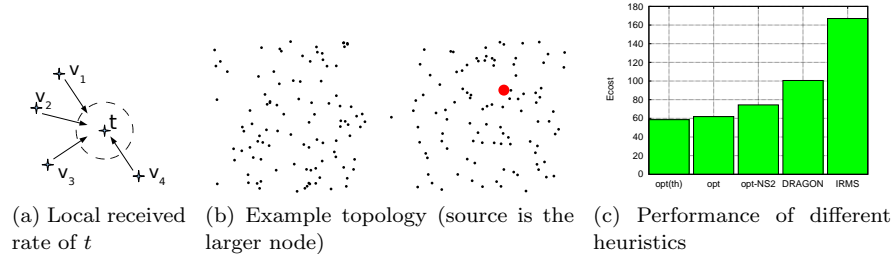
4 Heuristics for Rate Selection

In this section, we describe related work, prior heuristics for rate selection in section 4.2, and the related proposed dynamic heuristic, DRAGON in section 4.3. Before, we introduce the general concept of *local received rate*, in section 4.1, from which all these heuristics are actually derived; they are also connected to the maximum broadcast rate obtained as in section 3.4.

4.1 Local Received Rate and Optimality

We define the *local received rate* as: the total rate received by one node from its neighbors. An example is represented on Fig. 1a: the local received rate

of t is the total rate from its neighbors v_1, v_2, v_3, v_4. It is not difficult to see that the broadcast rate of the source cannot be larger than the local received rate of any destination node (see formal details in [9]).

(a) Local received rate of t

(b) Example topology (source is the larger node)

(c) Performance of different heuristics

Fig. 1

The maximum broadcast rate of the source is lower than any local received rate, but one result links closely the two in [7]: a specific rate selection was introduced (*"IREN/IRON": Increased Rate for Exceptional Nodes, Identical Rate for Other Nodes*), where all nodes have the same rate, except from the source and the nodes near the edge of the network. One result is that the maximum broadcast rate is asymptotically the average local received rate[1]. From another viewpoint, the viewpoint of one node transmitting one packet, one metric is to count the number of its receiving neighbors for which the packet is useful (precisely, innovative, section 3.2): when it is useful for several nodes, it is a *multi-benefit* transmission ; when it is useful for all receivers, it is an *maximal-benefit* transmission. From that viewpoint, "IREN/IRON" has the same asymptotic cost as a method where all transmissions are maximal-benefit transmissions. As a result, it is a method which is asymptotically optimal – for the metric of the total number of transmissions.

4.2 Prior Static Heuristics using Cut At Destination

From section 4.1, we saw that the maximum broadcast rate is related to the local received rate in some cases of homogeneous networks. However, when the nodes have different numbers of neighbors, a first step is to adjust it, in order that the local received rate would be at least the broadcast rate of the source, $C_s \triangleq M$, for every destination: indeed, every node in the network should receive at least a total rate M from its neighbors.

A heuristic was explored in [8], inspired from [13]. A simple way to ensure that the local received rate of every node is at least equal to M, is the following: when a node has h neighbors, every of its neighbors would have a rate at least equal to $\frac{M}{h}$. Then the local received rate is always at least the sum of h such rates; indeed greater or equal to M. This yields the following rate selection:

---

[1] for lattice networks where the size of the network area grows towards infinity or for random unit-disk networks where both density and network area grow towards infinity.

- IRMS (Increased Rate for the Most Starving node) [8]: the rate of a node v is set to a constant value C_v, with:

$$C_v = k \max_{u \in H_v} \frac{M}{|H_u|}$$

where H_w is the set of neighbors of w, $|H_w|$ is its size, and $k = 1$ is a global adjustment factor.

In [13], and in a slightly different context (not explicitly a rate selection, broadcast all-to-all), theoretical arguments were given for ability to decode in the case of general networks when $k \geq 3$; but again, [7] is showing that $k = 1$ is sufficient, asymptotically.

In [8], IRMS ($k = 1$) was explored experimentally, and although overall good performance was observed, in the case of sparser networks, phenomena occurred where only a few nodes would connect one part of the network to another, in a similar fashion to the center node in Fig. 1b. In networks similar to Fig. 1b, the rate of the nodes linking two parts of the network, would be dependent on how many of them are present: such information is not available from local topology information.

4.3 New Dynamic Heuristic, `DRAGON`

The previous heuristics for rate selection were static, using simple local topology information, and the rates would be constant as long as the topology would remain identical.

In this article, a different approach is chosen. The starting point is the observation that with fixed rates, the rank of a node v, will grow linearly with time, as $D_v(\tau) \approx C_{\min}(s, v) \times \tau$, i.e. proportionally to the capacity of the min-cut from the source s to the node (see section 3.4). With a correct rate selection, one would expect this min-cut to be close to the source rate $C_{\min}(s, v) \approx M$ in every node, and hence would expect that all the ranks of all nodes grow at the same pace. Failure to do so is a symptom that the rate selection requires adjustment.

From $D_v(\tau)$, the rank of a node v at time τ, let us define $g_v(\tau)$, the maximum gap of rank with its neighbors, normalized by the number of neighbors, that is:

$$g_v(\tau) \triangleq \max_{u \in H_v} \frac{D_v(\tau) - D_u(\tau)}{|H_u|}$$

We propose the following rate selection, `DRAGON` *Dynamic Rate Adaptation from Gap with Other Nodes*, which adjusts the rates dynamically, based on that gap of rank between one node and its neighbors, as follows:

- `DRAGON`: the rate of node v is set to $C_v(\tau)$ at time τ as:
  - if $g_v(\tau) > 0$ then: $C_v(\tau) = \alpha g_v(\tau)$ where α is some constant
  - Otherwise, the node stops sending encoded packets until $g_v(\tau)$ becomes larger than 0

This heuristic has some strong similarities, with the reasonings presented in section 4.2, and with IRMS. Consider the local received rate of node v: DRAGON ensures that every node will receive a total rate LocalReceivedRate(v) at least equal to the average gap of one node and its neighbors scaled by α, that is, formally, that the local received rate at time τ verifies:

$$\texttt{LocalReceivedRate}(v) \geq \alpha \left(\frac{1}{|H_v|} \sum_{u \in H_v} (D_u(\tau) - D_v(\tau)) \right)$$

This would ensure that the gap would be closed in time $\approx \leq \frac{1}{\alpha}$, if the neighbors did not receive new innovative packets.

Thus, we can observe that DRAGON is constructed as a feedback control: from the expected dynamic behavior of network coding (namely a linear increasing with time of the information received, see section 3.4), it will detect inadequacies in the rate selection, and will adjust it accordingly. As further detailed in [9], DRAGON is decreasing the amount of transmissions from nodes with lower rank to nodes with higher rank, as such transmissions might be non-innovative (whereas, in the opposite direction, they always are). DRAGON can be understood as an algorithm decreases the rates, hence cost, of nodes whose transmissions are not absolutely guaranteed to be beneficial, and adjust them back only when problems are evidenced by a growing gap in rank.

We provide further insights on the dynamic behavior of DRAGON in [9] by considering approximations on linear networks: differential equations show that the network would converge to steady state exponentially with a time on the order of magnitude of $\frac{1}{\alpha}$, and that in steady state, the gap between neighbors, from the source to the edge, would be a constant $\frac{2M}{\alpha}$. Notice that insights from linear networks apply to general networks, as a lower bound, by considering in isolation one path from the source to one destination.

5 Experimental Results

5.1 Model, Metrics, Environment and Scenarios

In order to evaluate the performance of DRAGON, we performed extensive simulations, which are detailed in this section. The focus of the DRAGON algorithm is on wireless ad hoc networks, and simulations were performed either for random uniform graphs (inside a square) or with the reference network of Fig. 1b, in which one node connects two parts. This last network is of special interest because it exemplifies features found in sparse networks, where static heuristics fail.

The default simulation parameters are the following: number of nodes = 200; range is defined by M, expectation of number of nodes in one neighborhood, $M = 8$ by default; position of the nodes: random uniform i.i.d -ornet. on Fig. 1b; generation size = 500; field $\mathbb{F}_p$ with $p = 1078071557$; $\alpha = 1$ (for DRAGON) The simulator used was self-developed, with an ideal wireless

model with no contention, no collision and instant transmission/reception. For comparison purposes, NS-2 (version 2.31) was also used with its default parameters.

The metric for (energy-)efficiency that is used in simulations is: the total number of transmitted packets per source packet, and is denoted as E_{cost}. $E_{\text{cost}} \triangleq \frac{\text{Total number of transmitted packets}}{\text{Generation size}}$. As mentioned in section 1, the optimal rate selection may be computed from a linear program [6]: it is the rate selection which minimizes E_{cost}. In some scenarios, we computed numerically this minimum E_{cost} (by a linear program solver), to obtain a reference point.

We also implemented a termination protocol which uses the state of neighbors piggybacked on coded packets to decide to stop transmission ; the generation size was choosen to be sufficiently large to offset its cost.
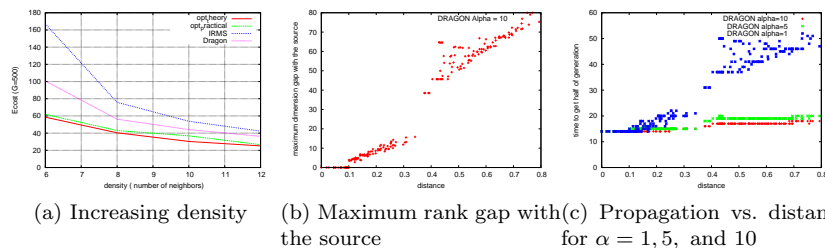

(a) Increasing density   (b) Maximum rank gap with (c) Propagation vs. distance
                              the source   for $\alpha = 1, 5,$ and 10

Fig. 2

5.2 Efficiency of DRAGON

In this section, we start the analysis of the performance of various heuristics, by considering their efficiency from E_{cost}. Simulations were performed on several graphs with default parameters but $M = 6$ – relatively sparse networks – and with three rate selections: optimal rate selection, IRMS, and DRAGON. The Fig. 1c represents the results (for an average of 6 random graphs).

5.2.1 Theory and Practice

The first 3 bars, are unrelated with DRAGON, and in essence, attempt to capture the gap between theory and practice.

The first bar (label: $opt(th)$), is the optimal E_{cost} as obtained directly from the linear program solution, without simulation. The second bar (label: opt), is the actual measured E_{cost} in simulations in the ideal model wireless model, with optimal rate selection. The third bar (label: $opt - NS2$) is the actual measured E_{cost} in simulations with the simulator $NS - 2$ (packet size 512, coding vector headers included).

As one might see, with the default parameters, the measured efficiency when performing actual coding with optimal rates (and generation size = 500), gives a result rather close to numerical value of the optimal: within a few percents. Another result is that the impact of the physical and MAC

layer, as simulated by NS-2 (with 802.11, two ray ground propagation, omni-antenna), is limited as one can see: $\approx 20\%$ (and the channel occupation rate was approximately of $1/3$).

The results are close. Therefore, because the purpose of our algorithm is not to perform congestion control (and because the parameters chosen would made some simulations operate above the channel capacity), in the remaining results, we will present results with the ideal wireless model.

5.2.2 Efficiency of different heuristics

The two last bars of Fig. 1c represent the efficiency of `DRAGON` and IRMS respectively. As one may see in this scenario, the ratio between the optimal rate selection, and `DRAGON` is around 1.6, but without reaching this absolute optimum, `DRAGON` still offers significantly superior performance to IRMS.

The gain in performance comes from the fact that the rate selection IRMS, has lower maximum broadcast rate (in some parts of the network), than the actual targeted one, and hence than the actual source rate. As a result, in the parts with lower min-cut, the rate of the nodes is too high compared to the innovation rate. Whereas with `DRAGON` such phenomena should not occur for prolonged durations: this is one reason for its greater performance.

5.2.3 Impact of Density

On Fig. 2a, simulations were performed on random graphs (avg. of 6), with default parameters and with increasing radio range. The modified parameter is M, the average number of nodes in one disk representing radio range. As one may see, `DRAGON` performs well, comparatively to IRMS in sparser networks which are the most problematic cases, for reasons explained previously. For denser networks, the gap between IRMS, `DRAGON` and the optimal rate selection closes.

5.3 Closer Analysis of the Behavior of `DRAGON`

5.4 Impact of α

In `DRAGON`, one parameter of the adaptation is α, and is connected to the speed at which the rates adapt. The table `I` indicates the total number of transmissions made, for the reference graph Fig. 1b, and for `DRAGON` with different values of α; and also for IRMS.

Value of α	$\alpha = 1$	$\alpha = 5$	$\alpha = 10$	IRMS
Total cost	16083	16272	17734	64411

Table 1

As one might see, first, the efficiency of IRMS on this network is rather low (1/4 of DRAGON): indeed, the topology exemplifies properties found in the cases of networks where IRMS was found to be less efficient: two parts connected by one unique node. For various choices of α, it appears that the performance of DRAGON decreases when α increases. This evidences the usual tradeoff between speed of adaptation and performance. However the decrease in performance is rather limited: we ascribe this fact to two factors. The first one is, again, that for identical average rates, different rate selections will asymptotically have identical performance, whether the rates are oscillating dramatically or not. The second one, is that in DRAGON, the nodes stop sending encoded packets, whenever they are unsure if its transmissions are beneficial to their neighbors.

5.4.1 Comparison with Model

On Fig. 2b, some results are represented, when running DRAGON, with $\alpha = 10$ on the reference network in Fig. 1b. Consider one node. At every time, it has a rank, and this rank can be compared with the number of packets already sent by the source: the difference between the two should be ideally 0, and a larger value is an indication of the delay in in propagation of the source packets. Hence that difference can be taken as a metric. We make the following statistics: for each node, we identify the maximum value of that difference over the entire simulation. We then plot one point on the graph: the x coordinate is the distance of the node to the source, whereas the y is this difference.

Therefore the graph indicates how the "gap of rank with the source" evolves with distance. First, we see that there is a large step near the middle of the graph: this is the effect of the center node, which is the bottleneck and which obviously induces further delay. Second, two linear parts are present on each side of the step: this confirms the intuitions given by the models in [9] about a linear decrease of the rank of the node from the source to the edge of the network.

Finally, one may find simulations results for $\alpha = 1, \alpha = 5$, and $\alpha = 10$ on Fig. 2c. As previously, one dot represents a node in the network, and the x coordinate is the distance of the node to the source ; but this time the y-coordinate is the time at which the node has received exactly half of the generation size. This yields further indication on the propagation of the coded packets from the source. Indeed if we compare the difference of time between nearest node from the source, and furthest node, we get a propagation time. It is around 35 for $\alpha = 1$, 6 for $\alpha = 5$ and 4 for $\alpha = 10$: roughly, it is inversely proportional to α, as expected. In addition, one sees that with higher values of α (greater reaction to gaps), the impact of the bottleneck in center node, is dramatically reduced.

6 Conclusion

We have introduced a simple heuristic for performing network coding in wireless multi-hop networks: `DRAGON`. It is based on the idea of selecting rates of each node, and this selection is dynamic. It operates as a feedback control, whose target is to equalize the amount of information in neighbor nodes (the rank), and hence indirectly in the network. The properties of efficiency of `DRAGON` are inherited from static algorithms, which are constructed with a similar logic. Experimental results have shown the excellent performance of the heuristics. Further work includes addition of congestion control methods.

References

1. R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung, *"Network Information Flow"*, IEEE Trans. on Information Theory, vol. 46, no.4, Jul. 2000
2. T. Ho, R. Koetter, M. Médard, D. Karger and M. Effros, *"The Benefits of Coding over Routing in a Randomized Setting"*, International Symposium on Information Theory (ISIT 2003), Jun. 2003
3. D. S. Lun, M. Médard, R. Koetter, and M. Effros, *"On coding for reliable communication over packet networks"*, Technical Report #2741, MIT LIDS, Jan. 2007
4. Z. Li, B. Li, D. Jiang, L. C. Lau, *"On Achieving Optimal Throughput with Network Coding"* Proc. INFOCOM 2005.
5. Y. Wu, P. A. Chou, and S.-Y. Kung, *"Minimum-Energy Multicast in Mobile Ad Hoc Networks using Network Coding"*, IEEE Trans. Commun., vol. 53, no. 11, pp. 1906-1918, Nov. 2005
6. D. S. Lun, N. Ratnakar, M. Médard, R. Koetter, D. R. Karger, T. Ho, E. Ahmed, and F. Zhao, *"Minimum-Cost Multicast over Coded Packet Networks"*, IEEE/ACM Trans. Netw., vol. 52, no. 6, Jun. 2006
7. C. Adjih, S. Y. Cho and P. Jacquet, *"Near Optimal Broadcast with Network Coding in Large Sensor Networks"*, 1st Workshop Information Theory for Sensor Networks, Sante Fe, Jun. 2007 (WITS'07).
8. S. Y. Cho, C. Adjih and P. Jacquet , *"Heuristics for Network Coding in Wireless Networks"*, Proc. International Wireless Internet Conference (WICON 2007), Texas, USA, October, 2007, Accepted.
9. S. Y. Cho and C. Adjih *"Network Coding for Wireless Broadcast: Rate Selection with Dynamic Heuristics"*, INRIA RR-6349, Nov 2007.
10. S.-Y. R. Li, R. W. Yeung, and N. Cai. "Linear network coding". IEEE Transactions on Information Theory , Februray, 2003
11. R. Koetter, M. Medard, *"An algebraic approach to network coding"*, IEEE/ACM Transactions on Networking, Volume 11, Issue 5, Oct. 2003
12. P. A. Chou, Y. W, and K. Jain, *"Practical Network Coding"*, Forty-third Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, October 2003
13. C. Fragouli, J. Widmer, and J.-Y. L. Boudec, *"A Network Coding Approach to Energy Efficient Broadcasting"*, INFOCOM 2006
14. A. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros, *"Capacity of Wireless Erasure Networks"*, IEEE Trans. on Information Theory, vol. 52, no.3, pp. 789-804, Mar. 2006
15. D. S. Lun, M. Médard, R. Koetter, and M. Effros, *"Further Results on Coding for Reliable Communication over Packet Networks"* International Symposium on Information Theory (ISIT 2005), Sept. 2005
16. Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin, *"Network Flows: Theory, Algorithms and Applications"*, Prentice Hall, 1993.