Efficient Multipath in Sensor Networks using Diffusion and Network Coding

Alberto Lopez Toledo[1] and Xiaodong Wang
Electrical Engineering Department, Columbia University, New York, NY 10027
Email: {alberto,wangx}@ee.columbia.edu

***Abstract*— In this paper we propose to employ the use of network coding to achieve an adaptive equivalent solution to the construction of disjoint multipath routes from a source to a destination. It exploits both the low cost mesh-topology construction, such as those obtained by diffusion algorithms, and the capacity achieving capability of linear network coding. We present a simple randomized network coding scheme that can be efficiently employed in sensor networks, solving two of its main problems: knowledge of the underlying network capacity and rate control. Our solution easily adapts to the changing conditions of the network and it can be used by the sinks to increase or decrease capacity and reliability on demand. Keywords: Network coding, sensor network, directed diffusion, multicast.**

I. Introduction

A typical approach to increase reliability in sensor networks is to exploit path diversity. The dense deployment of nodes in a sensor network makes multipath routing a suitable and cheap technique to cope with the frequent topological changes and consequently unreliable communication services. Multipath routing has been used to improve the robustness of data delivery, and it has been shown to be effective in energy saving, load balancing and increasing the network lifetime. More importantly, the extra path diversity reduces the end-to-end delay and the frequency of route discoveries and improves the network security.

The key issue in multipath routing is the proper multipath maintenance. The more constraints imposed on the structure of the paths (e.g., totally disjoint paths), the more complex and costly is to establish them and to maintain them. In most existing schemes the practical objective is to guarantee that a certain number of alternative routes to the destination exist, either to transmit copies of the same data to gain robustness or to transmit different data simultaneously to increase throughput. The max-flow min-cut theorem establishes that the maximum flow to the destination is equivalent to the number of disjoint paths. Hence, for the unicast case, an alternative to multipath construction is to use alternative methods that guarantee a certain maxflow to the receiver without relying heavily on specific path constructions.

In this paper we propose the use of a simple network coding approach to enable capacity achieving communications in a loosely controlled multipath topologies. We use simple braided-based mesh schemes (e.g., directed diffusion [1]) that are the easiest to construct, but at the same time are more likely to obtain less disjoint paths. While in most cases the maxflow of a mesh is difficult to calculate in a distributed manner, we show a simple linear code construction that integrates with the low cost mesh multipath construction and guarantees the maxflow of the topology. Moreover our proposed scheme permits the sources and the sink to easily determine the maxflow of the topology and hence to estimate the equivalent number of disjoint paths, allowing an adaptive adjustment of the mesh construction algorithm to reach the required number of disjoint paths.

[1]Alberto Lopez Toledo is supported by the Rafael del Pino Foundation.

II. Definitions

We consider the problem of communication in a network, where a set of source nodes S transmit data to a set of destinations T. The *communication network* consists of a number of links that interconnect some nodes, and it can be represented as a directed graph $G = (V, E)$ where E is the set of edges and V is the set of vertices or nodes. We represent an edge with the tuple $e = (u, v)$, and we say that the edge is directed, i.e., it transmits information from its *tail* u to its *head* v. The tail and head of an edge $e = (u, v)$ are denoted by $tail(e) = u$ and $head(e) = v$ respectively. Without loss of generality we assume each edge $e \in E$ has unit capacity, i.e., node u can send information through the edge (u, v) at a rate of at most 1 bits per time unit. We can represent links with different capacities as multiple edges connecting two nodes. The *capacity* $C_G(s, t)$ of the network G from node s to node t, as the maximum rate that information can be transmitted from s to t.

We define the *input-set* $\Gamma_I(u)$ and the *output-set* $\Gamma_O(u)$ of a node u as the set of edges that terminates and originates from that node, respectively, i.e. $\Gamma_I(u) = \{e \in E \mid head(e) = u\}$ and $\Gamma_O(u) = \{e \in E \mid tail(e) = u\}$. We also define *in-degree* $\delta_I(u)$ and *out-degree* $\delta_O(u)$ as the cardinalities of the input and output sets, respectively, i.e., $\delta_I(u) = |\Gamma_I(u)|$ and $\delta_O(u) = |\Gamma_O(u)|$.

A *path* from node u to node v is a sequence of edges (u, w_1),(w_1, w_2) ... (w_n, v) in G. A *cycle* is a path from one node to itself. A network is called *cyclic* if it contains a cycle, and *acyclic* if it does not contain any cycle. We say two paths from u to v are *disjoint* if the paths do not share edges.

III. Sensor Reachback Problem

The reachback problem is fairly different from its multicast counterpart. Assume we have an acyclic graph $G = (V, E)$

where a set of senders $S \subseteq V$ wants to transmit *different* information to a single receiver $t \in V$. Note also that it cannot be seen as the inverse of the multicast problem because in the multicast problem the sender transmit the *same* information to all the receivers.

It has been shown that the reachback problem can be solved by routing, i.e., by finding a disjoint path from each s_i to t. However, this is often difficult, particularly if the topology is changing and the links are error-prone such as in sensor networks. Moreover, the network topology is often unknown for individual sensors. For this reason, sensors have to find its route to the sink. As sensors are resource limited common routing solutions such as distance-vector solutions are difficult to apply. Instead, more data-centric approaches such as directed diffusion [1] (described below) are used.

A. Directed Diffusion

Directed diffusion was introduced in [1] as a scalable routing protocol for sensor networks. A subscriber, or data sink, identifies data by a set of attributes. This *interest* is periodically propagated hop by hop in the network. Every node that receives an interest establish a *gradient*, a state information pointing to the next-hop direction of other nodes interested in the data. When a source node receives an interest message, it starts forwarding the data through all the neighbors from which it has received an interest. This initial flooding is called 'exploratory' and it is done at a very low rate. Its purpose is to find a suitable route to the sink. When this 'exploratory' data reaches the sink, the sink reinforces its preferred neighbor according to certain metric such as latency, hop count, throughput, etc., and 'reinforces' that neighbor in the path to the source. The reinforcement of the neighbors propagates hop by hop back to the source or sources, resulting in a chain of reinforced gradients from all sources to all sinks. The rest of the data is sent at its nominal rate through the reinforced path. Nodes can also send *negative reinforcements* when receiving data not relevant to them. Gradients are managed as soft-state, thus both interests and exploratory data are refreshed periodically.

An interesting feature of the diffusion protocol is that sinks (and other nodes) can reinforce more than one path. Denote h' as the *diffusion parameter*, i.e., the number of neighbors to which each node propagates the reinforcement. So, with minimal changes in the operation of the protocol, braided-like topologies can be constructed.

While this kind of mesh multipath construction can result in robust multipath delivery, diffusion mechanisms do not have control over the capacity of the underlying network, so they usually exploit the multipath delivery by sending same information per path. This means that without further knowledge, sensors can only use the braided topologies constructed by diffusion algorithms to increase reliability, but not to exploit the increased throughput of the inherent multipath.

IV. Network Coding

In their seminal work, Ahlswede *et. al.* showed in [2] that the multicast cut-bound could not be achieved for the general case by routing only. They showed, however, that the cut bound could be achieved by encoding the data received in the intermediate nodes. In such network coding approach, nodes are allowed to perform coding on the received data instead of the simple store-and-forward operation in the standard routing schemes.

Theorem (network coding) *Let $G = (V, E)$ be a directed graph, and let s and $T \subseteq E$ be vertices in G. Then $C_G(s, T)$ can be achieved by performing coding operations in the nodes.*

Li *et. al.* [3] showed that the multicast capacity can be achieved by performing only linear combination to the input symbols (linear network coding). Koetter and Medard [4] gave an algebraic interpretation of the linear network coding and showed that multicast capacity can be obtained by time-invariant schemes.

Let $\boldsymbol{g}_e = [g_{e,1}, ..., g_{e,h}]$ denote the *global encoding coefficient* for edge e. The following theorem gives the necessary conditions for the existence of a linear network code [5].

Theorem 1 (Conditions for network codes) *Let $G = (V, E)$ be an acyclic directed graph in which a sender s transmits a multicast session to a set of receivers $T \subseteq V$ such that $C_G(s, T) = h$. A capacity-achieving linear network coding assignment exists for G if and only if there is a set of global encoding vectors $\boldsymbol{g}_e$ that satisfy:*

$$\boldsymbol{g}_e \in \{span(\{\boldsymbol{g}_{e'}, : head(e') = tail(e)\}), \quad \forall e \in E, \qquad (1)$$

$$rank(\{\boldsymbol{g}_e, : tail(e) = t\}) = h, \quad \forall t \in T. \qquad (2)$$

Network coding is not only useful for increasing the capacity of the multicast sessions. Robustness to network failures and error correction are also important areas in which network coding has been recently explored. Link or node failures that change the network topology can cause loss of large volumes of transmitted data, making efficient recovery schemes essential. However, while network coding has some great advantages, it is not a substitute of routing. Plain network coding cannot be used, for example, in the case of a dense sensor field because that would imply that all the nodes in the field would perform the encoding operations. Hence we need to limit the scope of the encoding operations via routing. Moreover, we need to adapt the network coding capabilities to the specific network requirements in terms of capacity, robustness and reliability.

V. Network Coding in Sensor Networks

Network coding can be used to achieve the capacity of the network if the topology and other network parameters such as number of senders, receivers and the min-cut from each sender to each receiver are known. In [6], [7], polynomial time algorithms are developed for the construction of network codes, but they are centralized algorithms and require knowledge about the entire topology of the network. In [8], a decentralized algorithm is proposed that requires minimal knowledge about the topology. However, similar to other algorithms, it requires the previous acquisition of h disjoint paths from senders to receivers, or, at least, the previous knowledge of the min-cut

from the sources to each of the receivers. This information is now known in a sensor network.

Ho *et.al.* introduced in [9] a randomized approach to network coding, in which nodes independently and randomly select linear mappings from its input edges to its output edges over some field. They showed that the probability of decoding can be made arbitrarily small by increasing the size of the field over which the code is constructed. The algorithm is simple, each node will forward in its output edges a random linear combination of the symbols received on the input edges. When the alphabet size is large enough the probability of picking the same random combination is small and the probability of successful decoding high.

Randomized network coding is regarded as a suitable implementation in practical networks because it can be completely decentralized, it does not depend on the topology and furthermore it is very simple to implement. However, the application of randomized network coding to a sensor network has three main problems: first, if the maximum rate between the sender and the receiver $C_G(s,t)$ is not known, the sender may send at a rate above the capacity, in which case the receiver may not be able to decode all the information sent by the source; or the source may send at a rate below capacity, wasting resources. Second, same as the existing algorithms in the literature, because the route to the destination is not known, existing randomized network coding forwards the random combination of the input symbols through *all* the output links of a given node. In a dense sensor network this implies using many nodes that are not interested in the information, with the consequent power waste. Finally, because the size of the alphabet depends on the number of edges that are assigned random codes, if this number is not known, the sources have to be conservative when selecting the alphabet size, hence wasting resources.

A. Combining Network Coding with Directed Diffusion

Let first consider a simple situation with one sink t and only one source node s, with $C_G(s,t) = h$. When h is unknown, there are some facts that are in order.

Theorem 2 (Acyclicness of directed diffusion) *The topology $G = (V, E)$ resulting from applying the directed diffusion protocol to a sensor field is acyclic.*

Because directed diffusion constructs acyclic networks, we can construct a network code over a directed diffusion topology. However, because the resulting topology is not known, the exact value of $C_G(s,t)$ is also unknown, so randomized network coding cannot be used. The reason is that the sender has no way to know the number of distinct symbols to send through the outgoing edges.

Corollary 1. *Let $G = (V, E)$ be an acyclic directed graph in which a sender s transmits to a receiver t. Assume $C_G(s,t) = h$. Then given a capacity achieving assignment of global coding vectors, we have $C_G(s,t) \leq \Gamma_O(s)$.*

Proof: It follows from $rank(\{\boldsymbol{g}_e : tail(e) = t\}) \leq \Gamma_O(s)$. □

Corollary 2. *Let $G = (V, E)$ be an acyclic directed in which a sender s transmits to a receiver t. Let assume $C_G(s,t) = h$. Then a valid assignment of the global coding vectors is $\boldsymbol{g}_{e',i} \in \{\boldsymbol{\epsilon_j}\} \quad \forall e' \ni head(e) = s$.*

Proof: We know that a network coding solution exist in which all the edges contain linear independent global coding vectors. In particular the unit vectors are linearly independent. □

The above results show that the sender may start assigning the unit vectors on its outgoing edges and that if a capacity achieving network code is used, $C_G(s,t)$ cannot be greater than the number of incoming edges at the sink.

Theorem 3 (Rank at the sink) *Let $G = (V, E)$ be an acyclic directed graph in which a sender s transmits to a receiver $t \in V$. Let the network use the directed diffusion mechanism to establish the paths from s to t, i.e., t broadcasts interest through all its outgoing edges and s reinforces all the available paths to the sink. If we use a randomized network code with sufficiently large alphabet size, with the initial assignment being as indicated in Corollary 2, then:*

$$rank(\{\boldsymbol{g}_e, : tail(e) = t\}) = C_G(s,t), \quad \forall s \in E. \tag{3}$$

Proof: It follows Theorem 2 and Corollaries 1 and 2. □

The above results indicate that the global coding vector assignment of a randomized network coding algorithm serves indeed for the purpose of calculating the $C_G(s,t)$ *at the receiver* if every node always assigns unused linearly independent vectors to its outgoing edges. Even if the source is sending at a rate *above capacity*, the sink will receive information at the capacity rate. The receiver could then feedback the information to the sender that would use the appropriate transmission rate. Better, we can use the directed diffusion algorithm to take advantage of the interest propagation phase as follows:

Algorithm 1 Interest propagation phase

1: In the interest propagation phase the sink t assign $\boldsymbol{g}_{e',i} = \epsilon_i, \forall e' \ni tail(e) = t$, where ϵ_i is the i–th unit vector in $\mathcal{I}^{\delta_I(t)}$.
2: For the rest of the nodes: forward a random linear combination of the incoming vectors through the links from where vectors where not received.

This forwarding of vectors is concurrent to the *interest propagation phase*, so it does not require extra transmissions. Algorithm 1 implements the randomized network coding algorithm in the reverse direction, i.e., from the sink to the sources. Each node in the network will receive a message from a link with a vector from $GF(q)^{\delta_I(t)}$, where $GF(q)$ is the field used by the sink for the network code. From Theorem 3 we know that the rank of those vectors is $C_G(t,s)$.

Theorem 4 (Rank at the sender) *Let $G = (V, E)$ be an acyclic directed graph in which a sender s transmits to a receiver $t \in V$. Let the network use the directed diffusion mechanism to establish the paths from s to t. Then, if the sink*

uses Algorithm 1 in the interest propagation phase, then, the vectors $\boldsymbol{g}_e$ assigned by Algorithm 1 are such that:

$$rank(\{\boldsymbol{g}_e, : head(e) = s\}) = C_G(s,t), \quad \forall s \in E \qquad (4)$$

Proof: By construction of the directed diffusion network, the sink starts forwarding interest messages through all its outgoing edges. Let G' be the network resulting from the interest propagation phase (i.e., from the sink to the sources). By Theorem 3 we have that the set of vectors $\boldsymbol{g}_e$ received at the senders have rank $C_{G'}(t,s)$. Consider the network G resulting from the senders reinforcing all the paths from where an interest message is received. By symmetry we have $C_G(s,t) = C_{G'}(t,s)$, which is the rank that the sender receives in the initial interest propagation. □

The previous theorem is an important result towards a distributed operation of the protocol. In the absence of any knowledge about the capacity of the graph, as it is in the case of a sensor network in which a variable diffusion algorithm as Algorithm 1 is used, the sink can start assigning to its outgoing edges all the unit vectors $\boldsymbol{\epsilon_i}$ of dimension h', where h' is the diffusion parameter $h' \in \{1, ..., \Gamma_O(t)\}$. After the initial interest propagation, all the senders s_i will have the information $C_G(s_i,t)$ simply by observing the rank of the vectors received. In fact, this is true for all the intermediate nodes. A summary of the resulting algorithm is below.

Algorithm 2 Directed diffusion and network coding

1: Sink estimates an initial h'.
2: **loop**
3: Sink refreshes interests as in Algorithm 1, reinforcing h' paths.
4: Sources calculate $h = C_G(s,t)$ from the rank of the coding vectors received from the sink.
5: Sources start their transmission using a randomized network code through all the paths from where they received a reinforcing message at a rate of h symbols per time unit.
6: Sink monitors the network conditions and recalculates h' depending on the network requirements.
7: **end loop**

Consider for example the scenario in Figure 1 in which Algorithm 1 is used in the interest propagation phase (on the left). Initially the sink assigns to its outgoing edges different vectors from $\mathbb{R}^{\delta_I(t)}$, in this case $\mathbb{R}^2$. For simplicity we use vectors from the field of the real numbers. As the interest is propagated, each intermediate node will forward through the edges from which no interest is received a random linear combination of the vectors received from the sink. For example, node a receives the vector (0,1) and randomly generates (0,7), forwarding it to nodes s_1 and s_3. Node c, on the other hand, receives the vectors {(0,3),(2,0)} and generates the vector (7,4) that is forwarded to node s_2. Now observe the rank of the vectors received in every node corresponds to the min-cut to the sink: $C_G(s_1,t) = rank(\{(0,7),(0,3)\}) = 1$, $C_G(s_2,t) = rank(\{(0,9),(7,4),(2,3)\}) = 2$ and $C_G(s_3,t) = rank(\{(0,7)\}) = 1$. Note that the three senders receive a set of vectors whose rank correspond to the min-cut of the resulting network after all the paths are reinforced, knowing the number of symbols they can transmit per time slot.

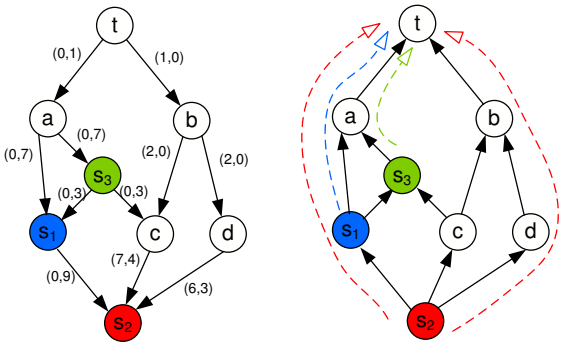
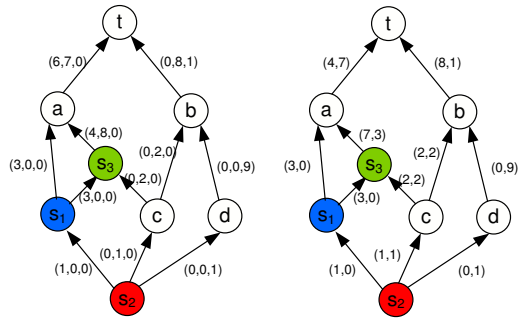


Figure 1: Distribution of codes in the interest propagation phase.



(a) When $C_G(s_2,t)$ is not known.

(b) When $C_G(s_2,t)$ is known.

Figure 2: Randomized network coding in the reinforcement phase.

In the next phase of the directed diffusion algorithm, the sources will reinforce the paths and use the regular randomized network coding algorithm, correctly using vectors from $GF(q)^{C_G(s,t)}$. In the absence of knowledge about $C_G(s,t)$, if the source wants to achieve the full network capacity, it has to use assign random code from $GF(q)^{\Gamma_O(s)}$, i.e., it will send $\Gamma_O(s)$ distinct symbols through its outgoing edges. However, as Theorem 3 shows, the sink will receive vectors with rank $C_G(s,t)$. If $C_G(s,t) < \Gamma_O(s)$ the sender would be sending at a rate greater than the capacity. On the other hand, if the source starts assigning vectors from $GF(q)^{C_G(s,t)}$, assuming q is large enough, the sink would be able to decode all the symbols, and, as we will see in Section VI-A, even when some of the links fail.

To highlight the importance of the knowledge of $C_G(s,t)$ when using random network coding consider the same scenario as Figure 1 and assume source s_2 does not know $C_G(s_2,t)$ (Figure 2(a)). In that case, to fully utilize the network capacity, s_2 sends three symbols through its outgoing edges x_1, x_2, x_3. The sink would not be able to decode the symbols without other information. Figure 2(b) shows the case in which s_2 knows $C_G(s_2,t) = 2$, and assigns to

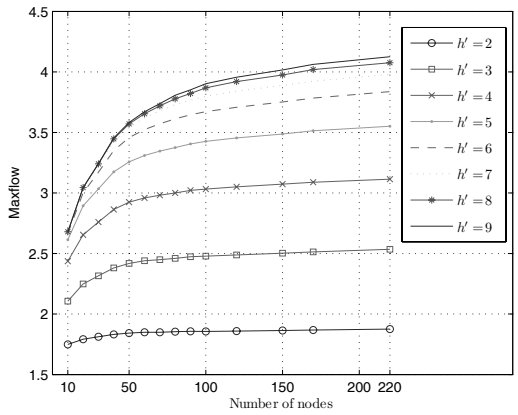


Figure 3: Average maxflow as h' changes.

its outgoing edges linearly independent vectors in $GF(q)^2$, sending the following information through its outgoing edges: $x_1, x_1 + x_2, x_2$.

B. Rate Adaptation

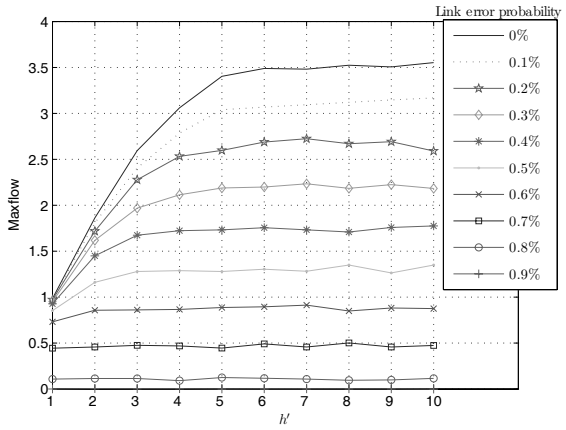
Our algorithm provides a way of probing the network by reinforcing h' paths in an adaptive manner. Upon reception of the interest vectors, the sender have the exact knowledge of the min-cut to the sink, and may decide to reinforce more or less paths, knowing at each moment how many symbols it can transmit at a given time by inspecting the rank of the received coding vectors.

We simulated the operation of the directed diffusion algorithm for different values of h' using an ad-hoc event-based simulator developed in MATLAB. Sensors are uniformly distributed in the field and neighbors are distance based. The results are averaged over 1000 runs of the experiment. Fig. 3 shows the maxflow of the network for different h' values. Note that as the network grows the path diversity allows the algorithm to get closer and closer to h'. Also, as h' grows the maxflow of the network approaches the real capacity of the whole sensor field (an average of 4.2 in our experiments when all the nodes participate). Note that the maxflow follows h', so the sink has deterministic control over the capacity, i.e., can change h' according to the importance of the data and the communication requirements with minimum changes to the existing diffusion protocols. Such adaptive change in capacity allows the sink to control the sensor network delivery as in the case of event-driven data-centric protocols, where sinks determine the communication requirement.

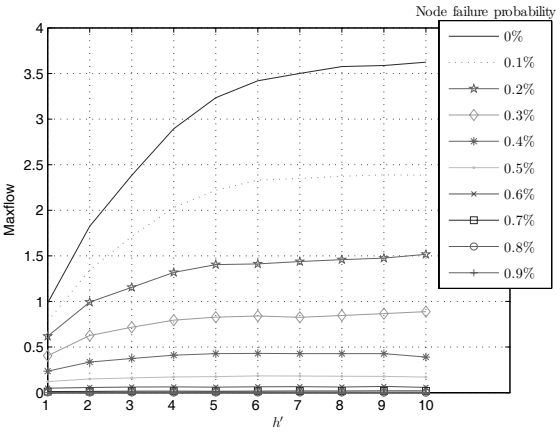
VI. Simulation Results

A. Robustness and Reliability

The capability of constructing adaptively capacity-achieving connections between a source and a destination allows not only to meet the throughput requirements, but also, as a benefit to the routing counterpart, to enhance the robustness of the network. We performed our simulations using an ad-hoc event-based simulator developed in MATLAB. Sensors are uniformly distributed in the field. We constructed the basic topology based on distance, i.e., each sensor decides that a



(a) When link failure occur.



(b) When node failure occur.

Figure 4: Average maxflow of a 50 nodes sensor network as h' increases.

sensor is its neighbor if it falls within a certain radius. We typically used a radius of approximately 2% of the length of the field. For each simulation, we randomly generated 1000 instances of the sensor field, and then ran our algorithm for each one of those 1000 instances and averaged the results. We used our algorithm in a unicast communication between a source and a destination. We do not consider packet loss due to wireless errors, but a more general failure of an edge or a node. Note that while a routing-only scheme will attempt to create h' disjoint-paths, the network coding approach can make use of extra edges for the combinations, hence increasing the probability of success.

Figs. 4(a) and 4(b) present the scenario of link and node failure respectively from the sink point of view, i.e., the maxflow obtained under certain conditions for a given diffusion parameter h'. This results can be used as a guideline for the selection of the diffusion parameter h' to obtain the desired network maxflow. For example, considering only link failures, a sink may decide not to increase the value of h' as it will not have any impact on maxflow but it will use more links and hence will result in larger power consumption. Note that the real benefit of the network coding approach in this scenario is

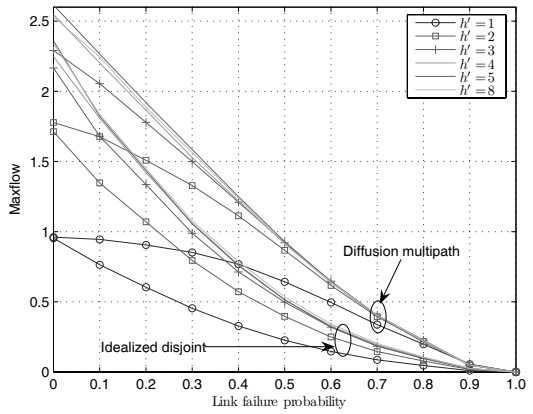


Figure 5: Maxflow of diffusion and network coding vs. idealized disjoint multipath for a sensor network of 20 nodes.

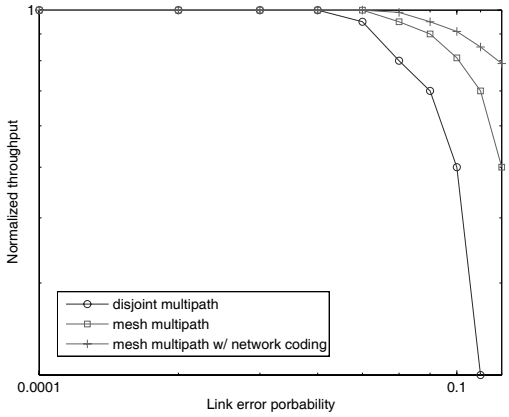


Figure 6: Bytes received at the sink vs. number of sources.

that this modification comes almost at no cost, as the interest propagation for the directed diffusion approach and the global network coding vector propagation are performed together, so the code is automatically established as the diffusion protocol updates the routing information. This cost is very low in comparison to the establishment and maintenance of disjoint-paths (an NP problem). Moreover the simple network coding mixing guarantees that the underlying topology obtains the network capacity without requiring complex routing schemes to obtain the disjoint paths.

B. Comparison with Ideal Disjoint Multipath

We consider the performance of our proposed scheme with an idealized and perfect disjoint multipath protocol that we call *idealized disjoint*. Fig. 5 shows the maxflow for different values of h' and link failure, and it is compared with the idealized disjoint protocol that would obtain the same maxflow as our protocol for that h'. As expected the mesh topology is able to absorb the failed links, while the network coding approach guarantees the maxflow of the resulting topology.

C. Comparison with Directed Diffusion

In this simulation we compare the operation of our algorithm with two versions of multipath multicast described in [10], namely a meshed multipath (M-MPR) constructed using directed diffusion in which each node sends different data to each neighbor (selective forwarding), and a disjoint multipath (D-MPR), also based on directed diffusion but in which disjoint paths are constructed. We performed the simulations in the ns-2 simulator version 2.27 [11]. We modified the *diffusion3* implementation to include the possibility of multiple path selection and we developed the simple network coding application of code assignment. Our simulations used 500 uniformly randomly distributed sensors over a 500×500 meters field. Antennas are omnidirectional with a coverage of 40 m. Each source sends UDP data with 50 bytes packets during 200 seconds.

Fig. 6 shows the normalized throughput received at the sink averaged in 1000 simulations when the link failure probability increases. As we can see, the throughput obtained by the network coding mechanism is significantly higher than that of the multipath directed diffusion. Note that the network coding approach has exactly the same resource utilization as the (M-MSR) approach, and as shown in [10], is more energy efficient than its disjoint counterpart, as it requires less retransmissions.

VII. Conclusion

We have presented a practical scheme that achieves an adaptive equivalent solution to the construction of disjoint multipath routes from a source to a destination by the combination of low cost mesh-topology construction, such as those obtained by diffusion algorithms, and the capacity-achieving linear network coding. We have implemented our proposed algorithm in MATLAB and in the ns-2 simulator and have shown that it outperforms existing methods.

References

[1] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Trans. Networking*, vol. 11, no. 1, pp. 2–16, 2003.

[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.

[3] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inform. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[4] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[5] Y. Wu, K. Jain, and S.-Y. Kung, "A unification of edmonds' graph theorem and ahlswede et al's network coding theorem," in *Proc. 42nf Annual Allerton Conf. Communication Control and Computing*, Oct. 2004.

[6] P. Sanders, S. Egner, and L. Tolhuizen, "Polynomial time algorithms for network information flow," in *Proc. 15th ACM Symp. Parallel Algorithms Architectures*, 2003.

[7] S. Jaggi, P. S. Chou, and K. Jain, "Low complexity algebraic multicast network codes," in *Proc. IEEE Intl. Symp. Inform. Theory*, July 2003.

[8] C. Fragouli and E. Soljanin, "Network coding and error correction," in *Proc. IEEE Inform. Theory Workshop*, Oct. 2004.

[9] T. Ho, M. Mdard, J. Shi, M. Effros, and D. R. Karger, "On randomized network coding," in *Proc. 41st Annual Allerton Conf. Communication Control and Computing*, Monticello, IL, Oct. 2003.

[10] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, no. 4, pp. 11–25, 2001.

[11] Network simulator 2. http://www.isi.edu/nsnam/ns.