PlayNCool: Opportunistic Network Coding for Local Optimization of Routing in Wireless Mesh Networks

Peyman Pahlevani, Daniel E. Lucani, Morten V. Pedersen, Frank H.P. Fitzek
Aalborg University, Department of Electronic Systems
Email: {pep|del|mvp|ff}@es.aau.dk

***Abstract*—This paper introduces PlayNCool, an opportunistic protocol with local optimization based on network coding to increase the throughput of a wireless mesh network (WMN). PlayNCool aims to enhance current routing protocols by (i) allowing random linear network coding transmissions end–to–end, (ii) recoding at intermediate nodes, and more importantly (iii) the identification and selection of helpers for each individual link. PlayNCool is easy to implement, compatible with existing routing protocols, and relies on simple intuition and closed–form, local optimization techniques. The intuition behind our protocol is that each helper to a link in a multi–hop path reinforces that link by listening to coded packets transmitted in the link and by judiciously choosing when to start transmitting to make the data exchange faster and more efficient. This paper pays special attention to techniques to determine how much a helper should wait before springing into action based on channel conditions for the optimization of a single link, i.e., the helper will *play it cool* by only speaking after it has heard enough to be truly useful. These techniques constitute a key feature of PlayNCool and are applicable in large scale mesh networks. We show that PlayNCool can provide gains of more than 3x in individual links, which translates into a large end–to–end throughput improvement, and that it provides higher gains when more nodes in the network contend for the channel at the MAC layer, making it particularly relevant for dense mesh networks.**

I. Introduction

Routing in wireless meshed networks (WMNs) has been subject to research in numerous publications. Multiple proposals have been presented and only a few made it to commercial exploitation. Traditional routing in wireless mesh networks is based on a calculating a single route for each flow [1], [2], [3]. The route is established by a node knowing to which neighbor a packet should be forwarded in order to reach a specific destination. Although some protocols utilize information from the physical and link layers to create the route, e.g., packet loss rate as in [3], they fail to exploit the broadcast nature of the wireless channel and its potential for allowing multiple nodes to receive a single node's transmissions. Thus, a transmission from a source S intended for destination D will use a single route as shown in Fig. 1(a), although it could potentially exploit additional nodes or routes. However, the broadcast nature of the wireless channel allows neighboring nodes to overhear transmissions of the main route. Considering that wireless channels suffer from significant packet losses, this overhearing opens interesting possibilities for exploiting neighboring nodes to forward a packet opportunistically in order to enhance overall performance. More recent work has provided opportunistic routing protocols that attempt to exploit the broadcast nature of wireless channel to increase network throughput. The ExOR protocol [4] constitutes an example of an opportunistic routing protocol for unicast flows, which uses a broadcast transmission at each relay instead of unicast as standard protocols would use, thus allowing all neighbors to receive packets. One disadvantage of the ExOR protocol is that it needs coordination among nodes in order to avoid duplicate packet transmissions, which inherently increases the network's signaling overhead. To address this problem, [5], [6] used random linear network coding to reduce coordination overhead and further increase throughput performance. MORE [5] is an opportunistic routing protocol, where the source sends random linear combinations of packets of a batch (typically called a generation in network coding literature). Each helper will take an incoming packet and recode, i.e., make a new linear combination using the incoming packet and the packets in the helper's buffer, finally transmitting the coded packet. Although MORE naturally exploits the benefits of multiple routes by providing new routing protocol for network coding, it is not compatible with existing routing protocols in mesh networks, which can become a barrier for its widespread deployment. Our work proposes a protocol, PlayNCool, that is compatible with standard routing protocols in wireless mesh networks, while still harnessing the potential of network coding over multiple routes. In particular, we propose the idea of selecting local helper nodes that can improve each link's individual performance and thus enhance the end–to–end service. Each link can select a subset of helper nodes to increase reliability of a link as shown in Fig. 1(b). PlayNCool protocol can be adapted with existing protocol in such a way that it can use the link quality information from routing protocol when the rout-

978-1-4799-2851-4/13/$31.00 ©2013IEEE

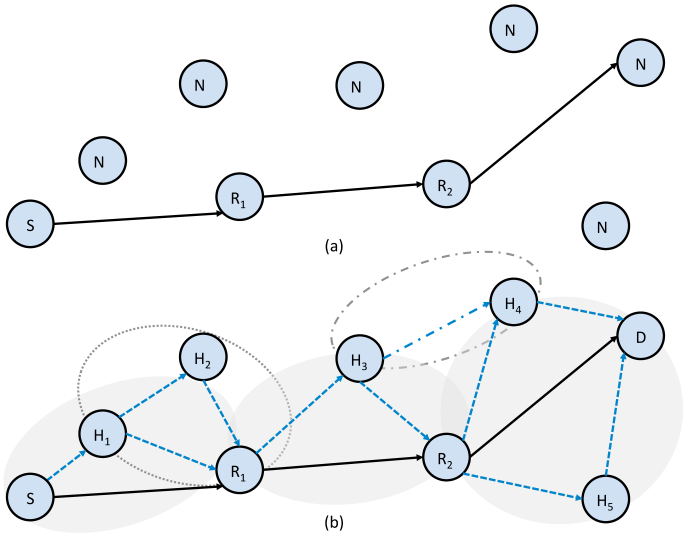


Fig. 1. (a) Traditional routing. (b) PlayNCool helper approach. Grey areas indicate local optimization with one or several helpers per link. H_2 acts as a helper of the link between another helper H_1 and R_1 illustrating the iterative potential of PlayNCool. The connection between H_3 and H_4 shows the potential, opportunistic overhearing between helpers to enhance performance.

ing protocol such as B.A.T.M.A.N provides the link quality information. In the case the routing protocol does not provide link quality information, PlayNCool enables the link quality discovery functionality. This functionality discovers the link quality information locally because each node only needs to know the link quality of the neighbors. In our approach, a relay node chooses the next hop using information derived from routing protocol (like AODV, DSDV, or B.A.T.M.A.N. [2], [7], [3]) and sends a coded packet to the next hop. However, the key to guaranteeing good performance is for the selected helper to *play it cool* and not transmit immediately, but rather wait to gather some coded packets before its starts to transmit. Since both a relay and an intermediate node will be competing for the wireless media, it is preferable for the helper to wait for some time to guarantee with high probability that the coded packets from the helper will be linearly independent from those at the receiver. How much to wait before activating the helper is key to our investigation, but we show that even simple heuristics will yield significant gains in performance. PlayNCool calculates the time to wait depending on channel characteristics, but also considering competition over the wireless channels by neighboring nodes. Note that when the competition between nodes is high, the helper and the sender will transmit together to increase the priority of the link. A final advantage of PlayNCool is that network coding is transparent to the upper layers and also compatible with existing routing layers. Thus, the routing layer does not know that coding takes place for each packet.

II. PlayNCool Description

PlayNCool is a protocol meant to operate in connection with standard routing protocols and carries out a local optimization for each link in order to increase that links' throughput by reducing the time to complete the transmission of a batch of packets. The routing protocol could provide the link quality information for PlayNCool. For example, the B.A.T.M.A.N routing protocol, which is implemented in layer two, could deliver the link quality information to the PlayNCool. On the other hand, if the routing protocol does not provide the link quality information, PlayNCool enables the link quality discovery functionality, which essentially can estimate the link quality by determining the number of coded packets sent from the source and helper and the received ones at the helper and relay for the current and previous generations of data packets. A sender generates random linear network coded (RLNC) packets from a batch of g original data packets, where each batch of packets is called a generation. Intermediate nodes or relays will take coded packets of a generation, store them, recode using RLNC by creating linear combinations of the packets in their buffer, and finally transmit these combinations. PlayNCool uses a link–by–link acknowledgment mechanism for each generation to allow the sender or an intermediate node in the route to start transmitting the next generation when the next intermediate node (or the destination) have received all packets of the current generation. When the link between two nodes in the route is weak, i.e., it has a high loss probability, the link will identify one or multiple helpers locally that can support the communication process. Each helper overhears transmissions and uses RLNC to code received packets. Although using RLNC to code the received packets reduces the probability of transmitting a linearly dependent coded packet from the perspective of the receiver, this probability can be quite large at the beginning of the transmission of a generation, when only a few packets are known at both the helper and the next relay. For this reason, a key feature of PlayNCool is that the helper does not start to transmit after receiving the first coded packet, but rather it will *play it cool* and start sending at a later time to increase the chances of transmitting a useful (linearly independent) coded packet. An added advantage is that the helper will not compete for channel access in the beginning, allowing the sender to transmit more often at first and thus allowing the helper and the next relay to gather more knowledge at that stage. PlayNCool proposes a variety of options for improving link performance. Let us explain these features through some motivating examples. In Fig. 1(b), R_1 and R_2 have a weak link quality. H_3 has good link with R_2, therefore, R_1 selects H_3 as a helper. When R_1 starts transmitting to R_2, H_3 receives packets as well but it waits for some time before starting to transmit. If H_3 has received enough packets, e.g., H_3 and R_2 have enough

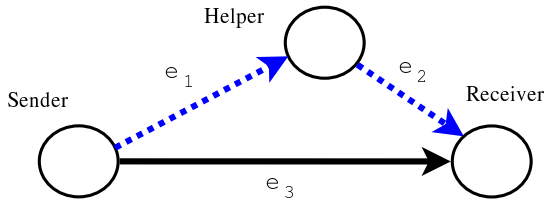


Fig. 2. The basic helper topology for local optimization. Sender and receiver are commonly relays (intermediate nodes) within the network.

coded packets amongst them to decode the generation, it starts recoding packets and transmitting to R_2. Since H_3's channel to R_2 is good, the time to complete the transmission will be reduced if compared to the time R_1 alone would need to complete. For links with higher losses, using only one helper may not be enough to increase the performance. Hence, the node selects more than one helper. In Fig. 1(b), R_2 selects H_4 and H_5 to increase the reliability. Using multiple helpers will increase the complexity and helper should use a policy to send packets. For example, let us consider the case of H_4 being near to the sender and H_5 being near to the receiver. Since R_2's link to H_4 is good, H_4 can quickly accumulate packets and start sending earlier. H_4 will later stop and allow for H_5 to send packets when H_5 and D have enough knowledge between them to decode the generation, because H_5 has the best link compared to the other nodes. A reason for using H_4 at the beginning could be to increase that links' priority by accessing the channel more often, because both R_2 and H_4 are transmitting for the benefit of that flow. If a helper does not have a good link to the receiver, it could select another (second degree) helper to increase the reliability of the helper link, as shown in Fig. 1(b). In this case, H_1 selects H_2 as its own helper. Having a helper for helper may be of practical use for large generation sizes, because to be useful it will need to gather enough coded packets. This may not be possible for small generation sizes. Finally, PlayNCool can allow opportunistic overhearing between helpers. For example, H_4 overhears packets from H_3 (Fig. 1(b)), which allows H_4 to start transmitting sooner, thus reducing the overall completion time. However, overhearing between helpers might increase the complexity in terms of the implementation perspective. Since the case of a single helper per link (Fig. 2) is the most common case, we proceed to analyze PlayNCool gains for it.

A. System model for local optimization

To decide about when a helper should start transmitting, the helper should have information about the link quality of sender, helper and the receiver as it is shown in Fig. 2. For our analysis, we consider that each node accesses the channel using Dynamic Allocation TDMA scheme where all active nodes have a slot. We define e_1, e_2, e_3 is the erasure probability between the sender and the helper, helper and receiver, and sender and receiver, respectively. We use RLNC to generate coded packets in generations of size g. We assume a large field size for our analysis and also assume that the transmissions from either the sender and helper are innovative, i.e., they provide a new DOF to the receiver. This is a simplifying assumption, but it is shown to provide surprisingly accurate results and a enabling a simple analysis to determine our policies. The key is that we exploit the structure of the problem in terms of the channel conditions to understand how much the helper should wait before transmitting. If the helper waits for enough time, it will accumulate enough coded packets to have at least some different DOFs with respect to the receiver. Thus, RLNC transmissions from the helper will indeed be innovative with high probability. We define r the number of transmission from the sender before the helper starts transmitting. The total number of transmissions from helper is defined as k . Moreover, to compare different approaches, the expected completion time for a policy $\mathcal{P}$ is defined as $T_{\mathcal{P}}(x-1)$ for $x-1$ neighbors where each neighbor consumes one slot in the TDMA scheme. Finally, we define a degree of freedom (DOF) as a linearly independent combination of the original packets. As discussed before, the use of a helper should be judicious. If a helper starts transmission too early, it may transmit useless packets, i.e., linearly dependent packets at the receiver. This has the added disadvantage of consuming channel resources, which could have been used by the sender. However, if the helper starts transmitting too late, the receiver may have already received most of the packets from sender directly limiting the benefits and the need for a helper. Thus, there is an inherent trade–off. The helper should transmit when those transmissions provide the most impact at the receiver. Another key trade–off is whether and when the sender should stop transmitting. Note that the sender can completely delegate the transmission of a generation to the helper, which has a better channel. However, in the presence of a large number of nodes competing for the wireless medium, keeping transmissions from the sender could still improve overall performance. To better understand these trade–offs, we analyze three policies. The first two are simple heuristics, while the later is a more advanced mechanism at the core of PlayNCool.

B. PlayNCool variations

We describe the different PlayNCool variations.

WFF: Wait for all degrees of freedom: In this approach, the helper starts transmitting when it has gathered all DOF. Both the helper and the sender will be actively transmitting. The expected number of transmissions until the helper receives all DOF is given as $r = g/(1-e_1)$. Since we assume TDMA and all nodes have the same priority for channel access, the sender

and the helper share the channel equally. Hence, the number of transmission for each node after activating the helper is equal to k. The total number of received packet from sender and the helper is equal to g, where $g = r \cdot (1 - e_3) + k \cdot (1 - e_2) + k \cdot (1 - e_3)$ which incorporates the two transmission phases. The expected number of transmissions from helper is thus $k = (g - r \cdot (1 - e_3))/((2 - e_3 - e_2))$. If we consider the effect of $x-1$ active neighbor nodes with an equal share of the resources. Then, before the helper starts to send the packets, there are x active nodes, meaning that the sender transmits once every x time slots. Therefore $r \cdot x$ is the expected completion time to perform r transmissions. When the helper starts transmitting, the sender is also active. Therefore, each one transmits once every $x + 1$ slots, meaning that every $\frac{x+1}{2}$ slots either the sender or the helper transmit to the receiver. The estimate of the expected completion time is given as $T_{WFF}(x - 1) = r \cdot x + k \cdot (x + 1)$.

WFH: Wait for a fraction of the degrees of freedom: The sender sends coded packets until the DOF at the helper and the receiver are enough to decode. At this point, the sender stops sending packets and the helper starts transmitting. This state is reached in the expectation when both receiver and helper have jointly full degree of freedom. Thus, the number of transmissions before the helper starts helping is given by $r = g/(1 - (e_1 \cdot e_3))$, while the number of transmissions from helper is:

$$k = \frac{r \cdot (1 - e_1) \cdot e_3}{1 - e_2} = \frac{g \cdot (1 - e_1) \cdot e_3}{(1 - e_1 \cdot e_3) \cdot (1 - e_2)}. \quad (1)$$

If we consider that $x-1$ neighbors are actively transmitting as for WFF, then the completion time is given as $T_{WFH}(x-1) = r \cdot x + k \cdot x$. Because at each phase either the sender or the helper transmit.

PlayNCool policy: In this approach, the helper must receive reasonable number of the packets before start sending (but not all DOF). When the helper starts sending, it will also overhear transmissions from the sender. Therefore, the number of DOF in the helper can be increased after the helper is active. The reasonable number of the DOF in the helper depends on the link quality between sender, helper, and receiver. For this reason, we define two operating regions. First, we consider the case of $(1 - e_1) \cdot e_3 > 1 - e_2$, i.e., when the helper is close to the sender and far from receiver. In this case, the number of incoming innovative packets to the helper is higher than the number of outgoing innovative packets from the helper. Thus, if the helper receives the first innovative packet from the sender, the helper should start transmitting. The number of transmission from the sender before the helper starts helping is given by $r = \frac{1}{(1-e_1) \cdot e_3}$. The second region matches the case when the helper is closer to the receiver, more specifically, when $(1 - e_1) \cdot e_3 \leq 1 - e_2$. In this case, the helper should receive enough innovative packets before start sending. The number of received DOF must be large enough that the receiver does not receive any non–innovative packets from the helper. Given that the desired outcome is for the number of received coded packets to be equal to g, then $g = r \cdot (1 - e_3) + k \cdot (1 - e_2) + k \cdot (1 - e_3)$, and thus,

$$k = (g - r \cdot (1 - e_3))/(2 - e_3 - e_2). \quad (2)$$

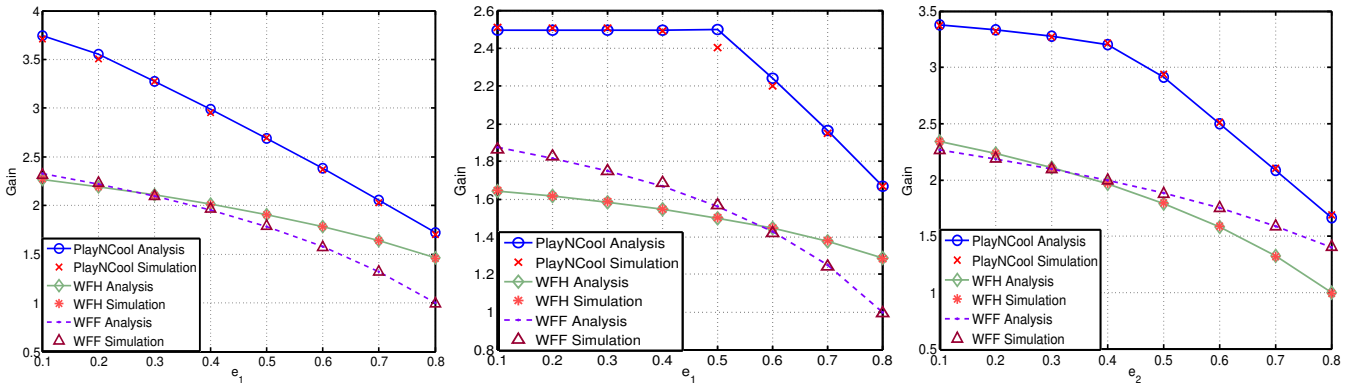
The number of received DOF in the helper should be at least equal to the number of transmitted DOF from helper. Therefore, the subtraction of received DOF and transmitted DOF is considered to be zero:

$$r \cdot (1 - e_1) \cdot e_3 + k \cdot (1 - e_1) \cdot e_3 - k \cdot (1 - e_2) = 0. \quad (3)$$

Combining Eq. 2 and Eq. 3, r is equal to: $r = -g \cdot C(e_1, e_2, e_3)/(D(e_1, e_2, e_3) - (1 - e_3) \cdot C(e_1, e_2, e_3))$, where $C(e_1, e_2, e_3) = (-1 + e_2 + e_3 - e_1 \cdot e_3)$ and $D(e_1, e_2, e_3) = (2 - e_3 - e_2) \cdot (e_3 - e_1 \cdot e_3)$. Using similar arguments as in the previous two approaches, the completion time in the presence of $x - 1$ neighbors is given by $T_{PlayNCool}(x - 1) = r \cdot x + k \cdot ((x + 1)/2)$.

III. Performance Comparison and Simulation Results

In this section, we compare the different helper approaches discussed in the previous section. These approaches are compared in terms of the gain under different system and channel conditions. Gain for policy $\mathcal{P}$ in the presence of $x - 1$ neighbors, $G_{\mathcal{P}}(x - 1)$, is defined as the completion time without a helper divided by the completion time of a helper approach with policy $\mathcal{P}$: $G_{\mathcal{P}}(x-1) = g \cdot x/((1-e_3)T_{\mathcal{P}}(x-1))$. We start by using simulations to compare and verify the results provided for the various approaches proposed in Section III. The simulation was carried out using the C++ KODO library [8]. Since the library enables researchers and industry to implement and test new network coding algorithms and protocols as well as to perform simulations and implementation on commercial devices, we have chosen it in order to translate our results to real–life implementation in the near future. For simplicity, the topology of the simulation includes sender, helper and receiver. The sender intends to send a generation of g packets to the receiver. The helper overhears packets from sender and it helps the communication depending on the policy implemented. The medium access model is based on the TDMA assumption made so far. We assumed that the sender and helper have information about the link quality between nodes in the local cluster. Fig. 3 shows our simulation and analytical results obtained from Section II for cases where e_1 (or e_2) is varied while e_3 and e_2 (or e_1) are kept fixed. The generation size is set to 100 and the number of neighbors which are competing to access the channel is equal to five. The loss probability between sender and the receiver is 80% unless stated otherwise. In this configuration, the PlayNCool



(a) e_2=0.3, e_3=0.8 (b) e_2=0.6, e_3=0.8 (c) e_1=0.3, e_3=0.8

Fig. 3. Simulation and analytical comparison for different approaches. g=100, x=5.

approach has a gain of 3.5, i.e., it is 3.5 times faster than the case with no helper, and this gain is much higher than the other approaches. PlayNCool has higher priority to access the channel because both the sender and helper compete for the channel. We also observe that the match between the simulations and our simple analysis matches very well for WFF and WFH, and for a wide region in PlayNCool. For the later, there is some difference to the predicted gain, which is due in part to our simplifying assumptions. However, our simple analysis still provides a good indication of the scheme's expected performance.

A. The Number of Neighbors

The competition between nodes to access the channel has a significant effect in the gain. When the number of neighboring nodes which are transmitting packets in the same frequency is small, the most of nodes have a high probability to transmit the packets. However, by increasing the number of neighbors, the competition between nodes to access the channel will be increased. The effect of the number of neighbors is considered in Fig. 4. In the case of WFH, the source is not transmitting when the helper is active. Therefore, the number of active nodes does not make any advantage in comparison with no–helper approach. As shown in Fig. 4, the WFH gain is independent on the number of neighboring nodes, i. e it has a fix gain as we change x while maintaining other parameters fix. When we have no neighboring nodes, WFH outperforms both WFF and PlayNCool in Fig. 4. The reason behind this is that the sender stops transmitting when the helper and receiver have accumulated enough DOF. This allows helper to transmit without competing with sender and thus take advantage of its better link quality to the receiver. However, when the number of active nodes is increased, the WFF and the PlayNCool perform better. That is because the fact that the helper and the sender both transmit, meaning they both have equal share with the others $x-1$ active neighbors, inherently gives the higher priority to the flow from the sender to the receiver. In Fig. 4, we also observe that although WFF and PlayNCool main-

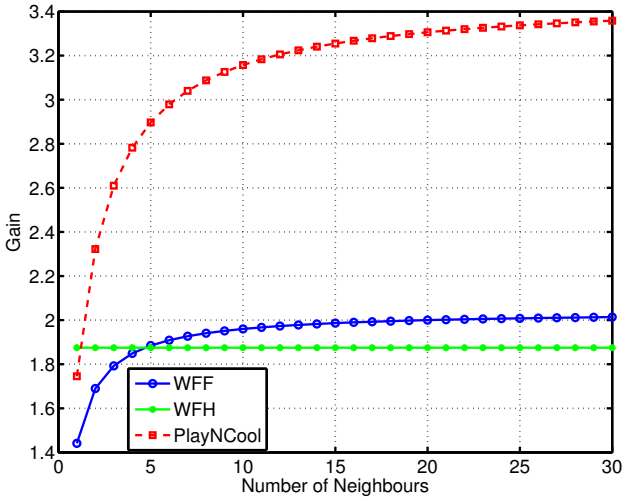
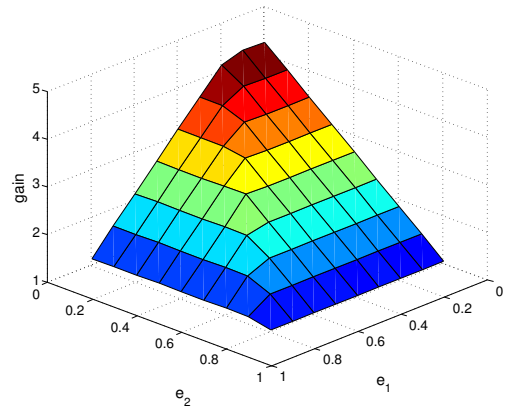


Fig. 4. The effect of the number of active nodes on the gain for different approaches. e_1=0.3, e_2=0.5, e_3=0.8, g=100.

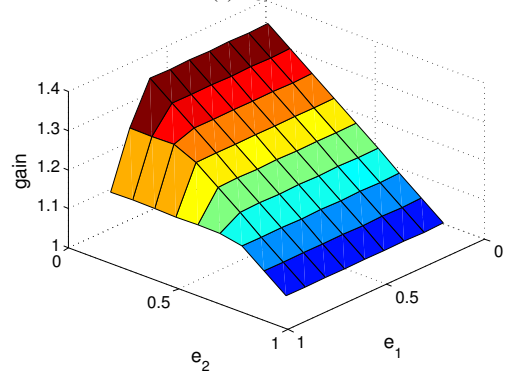
taining the source active when the helper starts transmitting, PlayNCool outperforms WFF by 60% to 80%. This is because the helper does not start transmitting until it receives half of degree of freedom but the PlayNCool approach activates the helper as soon as it receives enough degree of freedom and the helper is activated earlier. Therefore, the gain of the PlayNCool is much higher than the two other approaches.

B. The gain of PlayNCool

Using PlayNCool is helpful when the link quality between the sender and the receiver is weak. In Fig. 5(a), the link loss probability between sender and receiver is 80%. By using PlayNCool, the gain increases more than four fold. Fig. 5(b) illustrate that when the loss probability between sender and receiver is mild, in this case 30%, the gain is not much high but it is still near to 1.4 and in the wide range of link error probabilities it is near to 1.3. Fig. 5(b) also shows that the gain is mostly determined by e_2 when the loss probability between sender and receiver (e_3) is mild. On the other hand, when the e_3 is high the gain is highly dependent on both e_1



(a) e_3=0.8



(b) e_3=0.3

Fig. 5. Gain of PlayNCool for different link quality. e_3=0.8, g=100, x=5.

and e_2 as shown in Fig. 5(a). In ETX based routing protocol such as B.A.T.M.A.N, when the link quality of the helper is better than the link quality of the main link, routing protocol chooses the helper as a next hop. Therefore, the main link must be better than the helper with the condition of $1-(1-e_1)\cdot(1-e_2) > e_3$. However, when the link quality of helper is not better than the main link, PlayNCool gain is high. For example, in the Fig. 5(a), when $e_1 = 0.6$ and $e_2 = 0.6$, the gain of PlayNCool is close to three fold. From a practical perspective, each sender could have look–up tables with the information presented in Fig. 5 for different values of e_3. If the routing protocol provides information related to the loss rate, e.g., B.A.T.M.A.N., or if this information is computed by PlayNCool at a higher layer,then the sender could explicitly send a request to the most efficient node to become a helper or it could broadcast the information about the link quality to its neighbors and from them to the and each helper could then automatically determine whether or not it should help.

IV. Conclusion

In this paper, we presented a local optimization protocol using network coding being compatible to standard routing protocols for WMNs and yet provide additional gains in terms of throughput and delay performance. Our protocol, called PlayNCool, leverages the path from source to destination created by the routing protocol to then identify helper nodes for individual links. These helpers' goal is to increase throughput and reliability of each link and thus increase the end–to–end performance of the system. The advantage of such a local enhancing method is that it can exploit local information already available to the sender and destination in a specific link in the overall path. In particular, we have identified that the link quality from sender to receiver and helper to receive are key to determine the potential gains provided by the system. A key feature of PlayNCool is not just to decide which helper to use, but also when and how much this helper should contribute. We show that the effect of neighboring nodes is shown to increase the potential improvement of our scheme. Our results show that PlayNCool can increase the gain more than 4x in some scenarios. Our future work will focus on more practical aspects for the successful deployment of PlayNCool. A particular focus will be on the simulation of the protocol under other medium access channels, e.g., CSMA/CA, and more realistic conditions as well as understanding the end–to–end gains provided by the local optimization. Another key aspect to study is the effect of helpers over other flows and particularly the overall performance of a network using helpers extensively for its flows. We expect that the gains will be maintained because (i) the helpers only participate during a small fraction of the time, and (ii) each link will require less transmissions overall, which may reduce interference over other flows. Thus, the overall effect is expected to be positive even if we involve additional nodes. Finally, we will follow with an implementation on commercial nodes.

V. Acknowledgment

This work was partially financed by the Green Mobile Cloud project granted by the Danish Council for Independent Research (Grant No. 10-081621).

References

[1] D. B. Johnson, D. A. Maltz, and J. Broch, "Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks," in *Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5*. Addison-Wesley, 2001, pp. 139–172.

[2] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *proceedings of the 2nd ieee workshop on mobile computing systems and applications*, 1997, pp. 90–100.

[3] A. Neumann, C. Aichele, M. Lindner, and S. Wunderlich, "Better approach to mobile ad-hoc networking," in *IETF Internet Draft*.

[4] S. Biswas and R. Morris, "Exor: Opportunistic multi-hop routing for wireless networks," in *SIGCOMM*, 2005, pp. 133–144.

[5] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "More: A network coding approach to opportunistic routing," in *SIGCOMM*, 2007, pp. 169–180.

[6] D. Koutsonikolas, C.-C. Wang, and Y. Hu, "Ccack: Efficient network coding based opportunistic routing through cumulative coded acknowledgments," in *INFOCOM, 2010 Proceedings IEEE*, 2010, pp. 1–9.

[7] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers," 1994, pp. 234–244.

[8] M. V. Pedersen, J. Heide, and F. H. Fitzek, "Kodo: An open and research oriented network coding library," in *Workshop on Network Coding App. and Pro. (NC-Pro 2011)*, Valencia, Spain, May 2011.