Improving Network Performance in MESH Topology of Wireless Network

D.Vikkiramapandian M.E
1 (Assistant Professor /Department of Information Technology ,
K.L.N. College of Information Technology, Sivagangai Dt, Tamilnadu, India.)
vikramkln@yahoo.com

Abstract -Link failures often occurs in Wireless Mesh Networks (WMNs) due to various reasons. To recover from these link failures, expensive manual cooperation is needed or else it will result in poor degradation of network performance. For that we present this paper and here we introduce a new system called Automatic network Settings Reconfiguration (ANSR). By using this system, we can be able to recover from those link failures locally. The system reconfigures its network settings cooperatively among local mesh routers. By doing so, the network performance is preserved. Evaluation results shows that it provides better results than existing recovery schemes from failures.

I.INTRODUCTION

Wireless Mesh Network (WMN) is a communications network made up of radio nodes organized in a mesh topology. It is being developed in a variety of applications, such as public safety, environment monitoring, and city-wide wireless internet services. Even though enormous solutions for recovering WMNs from link failures, they still have limitations as follows. They often require Global configuration changes, which is not suitable for frequent link failures. In fault tolerant routing protocols, more network resource is required than link-level network reconfiguration.

To overcome the above limitations, we introduce a new system Automatic network Settings Reconfiguration (ANSR). It autonomously reconfigure its local network settings for real-time recovery from link failures. It helps to minimize the changes of healthy network settings.

Next, ANSR also includes a monitoring protocol that enables a WMN to perform real-time failure recovery in conjunc- tion with the planning algorithm. The accurate link-quality information from the monitoring protocol is used to identify network changes that satisfy applications' new QoS demands or that avoid propagation of QoS failures to neighboring links (or 'ripple effects'). Running in every mesh node the monitoring protocol periodically measures wireless link-conditions via a hybrid link-quality measurement technique, as we will explain in Section IV.

Based on the measurement information, ANSR detects link failures and/or generates QoS-aware network reconfiguration plans upon detection of a link failure.

ANSR has been implemented and evaluated extensively via experimentation on our multi-radio WMN test-bed. ANSR's local reconfiguration improves network throughput and channel-efficiency by more than 26% and 92%, respectively, over the local re-routing scheme.

II.MOTIVATION

Motivation behind this work is as follows Necessity of Self-Reconfigurability Maintaining the performance of WMNs in

the face of dynamic link failures remains a challenging problem. However, such failures can be withstood (hence maintaining the required performance) by enabling mr-WMNs to autonomously reconfigure channels and radio assignments, as in the following examples.

Recovering from link-quality degradation: The quality of wireless links in WMNs can degrade due to severe interference from other co-located wire- less networks operating on the same or adjacent channels cause significant and varying degrees of losses or collisions in packet transmissig the tuned channel of a link to other interference-free channels, local links can recover from such a link failure.

Satisfying dynamic QoS demands: Links in some areas may not be able to accommodate increasing QoS demands from end users (*QoS failures*) depending on spatial or temporal locality. Re-association is donehere to avoid communication failures.

Coping with heterogeneous channel availability: Links in some areas are being used for emergency response. Such links can seek and identify alternative channels available in the same area.

Motivated by these three and other possible benefits of using reconfigurable mr-WMNs, in the remainder of this paper, we would like to develop a system that allows mr-WMNs to autonomously change channel and radio assignments (i.e., *self-reconfigurable*) to recover from the channel-related link failures mentioned above.

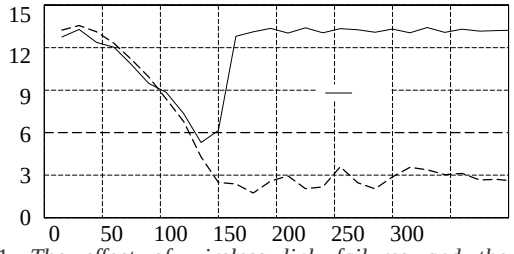


Fig. 1. *The effect of wireless link failures and the need for network reconfiguration:*

A wireless link in WMNs often experiences link-quality degradation due to interference from co-existing other networks. To recover from such a link failure, the network needs (at 130 s) to switch to non- interfering channel in real time. Link-quality measurements are made using IEEE 802.11a NIC in our testbed.

Network Model and Assumptions

Multi-radio WMN: A network is assumed to consist of mesh nodes, IEEE 802.11-based wireless links, and one control gateway. Each mesh node is equipped with n radios, and each radio's channel and link assignments are initially made. The gateway is connected to the Internet via wire-line links as well as to other mesh routers via wireless links.

Limitations of Existing Approaches

Given the above system models, we now discuss the pros and cons of using existing approaches for self-reconfigurable WMNs.

Localized reconfiguration:

Existing channel assignment and scheduling algorithms do not consider the degree of configuration changes from previous network settings, and hence they often require *global* network changes to meet all the constraints. They are unsuitable for dynamic network reconfiguration that has to deal with frequent local link failures.

Next, the *greedy* channel-assignment algorithm suffers from the ripple effect, in which one local change triggers the change of additional network settings at neighboring nodes due to association dependency among neighboring radios. Finally, interference-aware channel-assignment algorithm could further improve its flexibility by considering both radio diversity (i.e., link association) and local traffic information.

QoS-awareness: Reconfiguration has to satisfy QoS constraints on each link as much as possible. First, given each link's bandwidth constraints, existing channel-assignment and scheduling algorithms can provide approximately optimal network configurations. However, as pointed out earlier, these algorithms may require global network configuration changes from changing local QoS demands, thus causing network disruptions. We need instead a reconfiguration algorithm that incurs only local changes while maximizing the chance of meeting the QoS demands.

Next, the greedy algorithm might be able to satisfy particular links' QoS demands may fail to meet QoS demands if the links in the new channel experience interference from other co-existing networks that operate in the same channel.

III.THE ANSR ARCHITECTURE

Overview

ANSR is a distributed system that is easily deployable in IEEE802.11-based mr-WMNs. Running in every mesh node, ANSR supports self-reconfigurability via the following distinct features:

• *Localized reconfiguration*: ANSR generates reconfiguration plans that allow for changes of network configurations only in the vicinity where link failures occurred, while retaining configurations in areas remote from failure locations.

• *QoS-aware planning*: ANSR effectively identifies QoS-satisfiable reconfiguration plans by (i) estimating the QoS-satisfiability of generated reconfiguration plans and (ii) deriving their expected benefits in channel utilization.

• *Autonomous reconfiguration via link-quality monitoring*: ANSR accurately monitors the quality of links. Based on the measurements and given links' QoS constraints, ANSR detects local link failures and autonomously initiates network reconfiguration.

• *Cross-layer interaction*: ANSR enables re-routing for reconfiguration planning in addition to link-layer reconfiguration.

Algorithm 1 ANSR Operation at mesh node i

(1) Monitoring period (tm)
1: **for every** link j **do**
2: measure link-quality (lq) using passive monitoring;
en**d for**
4: send monitoring results to a gateway g;
(2) Failure detection and group formation period (tf)
5: **if** link l violates link requirements r **then**
6: request a group formation on channel c of link l;
7: **end if**
8: participate in a leader election if a request is received;
(3) Planning period (M , tp)
9: **if** node i is elected as a leader **then**
10: send a planning request message (c, M) to a gateway;
11: **else if** node i is a gateway **then**
12: synchronize requests from reconfiguration groups Mn
13: generate a reconfiguration plan (p) for Mi ;
14: send a reconfiguration plan p to a leader of Mi ;
15: **end if**
(4) Reconfiguration period (p, tr)
16: **if** p includes changes of node i **then**
17: apply the changes to links at t;
18: **end if**

19: relay p to neighboring members, if any

Algorithm 1 describes the operation of ANSR. First, ANSR in every mesh node monitors the quality of its outgoing wireless links at every tm (e.g., 10 sec) and reports the results to a gateway via a management message. Second, once it detects a link failure(s), ANSR in the detector node(s) triggers the formation of a group among local mesh routers that use a faulty channel, and one of the group members is elected as a leader using the well-known bully algorithm, for coordinating the reconfiguration. Third, the leader node sends a planning-request message to a gateway. Then, the gateway synchronizes the planning requests—if there are multiples requests—and generates a reconfiguration plan for the request. Fourth, the gateway sends a reconfiguration plan to the leader node and the group members. Finally, all nodes in the group execute the corresponding configuration changes, if any, and resolve the group. We assume that during the formation and reconfiguration, all messages are reliably delivered via a routing protocol and per-hop retransmission timer.

Planning for Localized Network Reconfiguration

The core function of ANSR is to *systematically* generate localized reconfiguration plans. A *reconfiguration plan* is defined as a set of links' configuration changes (e.g., channel switch, link association) necessary for a network to recover from a link(s) failure on a channel. ANSR systematically generates reconfiguration plans that localize network changes by dividing the reconfiguration planning into three processes—feasibility, QoS-satisfiability, and optimality—and applying different levels of constraints.

Feasible plan generation:

• *Avoiding a faulty channel*: To fix a faulty link(s), ANSR can use (i) a channel-switch (S) where both end-radios of link AB can simultaneously change their tuned channel, (ii) a radio- switch (R) where one radio in node A can switch its channel and associate with another radio in node B, and (iii) a route- switch (D) where all traffic over the faulty link can use a detour path, instead of the faulty link.

• *Maintaining network connectivity and utilization*: ANSR maximizes the usage of network resources by making each radio of a mesh node associate itself with at least one link and by avoiding the use of same (*redundant*) channel among radios in one node.

• *Controlling the scope of reconfiguration changes*: It finds a locally optimal solution by considering more network changes or scope.

Choosing the best plan:

• *Breaking a tie among multiple plans*: Multiple reconfiguration plans can have the same benefit, and ANSR needs to break a tie among them. ANSR uses the number of link changes that each plan requires to break a tie. Although link configuration changes incur a small amount of flow disruption (e.g., in the order of 10 ms), the less changes in link configuration, the less network disruption.

Complexity of ANSR

The network monitoring part in the reconfiguration protocols is made highly efficient by exploiting existing data traffic and consumes less than 12 Kbps probing bandwidth. In addition, the group formation requires only O(n) message overhead (in forming a spanning tree), where n is the number of nodes in the group.

IV. SYSTEM IMPLEMENTATION A ND EXPERIMENTATION

We have implemented ANSR in a Linux OS and evaluated it in our testbed. We first describe the implementation details, and then present important experimental results on ANSR.

A. Implementation Details

We discuss the software architecture of ANSR. First, ANSR in the network layer is implemented using netfilter , which provides ANSR with a hook to capture and send ANSR-related packets such as group-formation messages. In addition, this module includes several important algorithms and protocols of ANSR: (i) *network planner*, which generates reconfiguration plans only in a gateway node; (ii) *group organizer*, which forms a local group among mesh routers; (iii) *failure detector*, which periodically interacts with a network monitor in the device driver and maintains an up-to-date link-state table; and (iv) *routing table manager*, through which ANSR obtains or updates states of a system routing table.

Next, ANSR components in the device driver are implemented in an open source MADWiFi device driver. This driver is designed for Atheros chipset-based 802.11 NICs and allows for accessing various control and management registers (e.g., `longretry`, `txrate`) in the MAC layer, making net- work monitoring accurate. The module in this driver includes (i) *network monitor*, which efficiently monitors link-quality and is extensible to support as many multiple radios as possible; and (ii) *NIC manager*, which effectively reconfigures NIC's settings based on a reconfiguration plan from the group organizer.

B. Experimental Setup

To evaluate our implementation, we constructed a multi- hop wireless mesh network testbed on the fourth floor of the Computer Science and Engineering (CSE) building at the University of Michigan. The testbed consists of 17 mesh nodes and has multiple (up to 5) links. Each node is delib- erately placed on either ceiling panels or high-level shelves to send/receive strong signals to/from neighboring nodes. On the other hand, each node will experience enough multi-path fading effects from obstacles and interference from co-existing public wireless networks.

ANSR HARDWARE PROTOTYPE

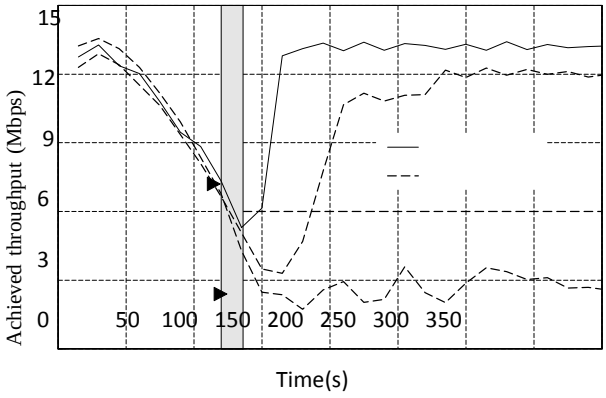
As shown in Figure, each mesh node is a small-size wireless router—Soekris board 4826-50 (Pentium-III 266Mhz CPU, 128 MB memory). This router is equipped with two EMP IEEE 802.11 a/b/g miniPCI cards and 5-dBi gain indoor omni-directional antennae. Each card operates at IEEE 802.11a frequency with a pseudo ad-hoc mode, and is set to use fixed data rate and transmission power. Next, all nodes run the Linux OS (kernel-2.6), a MADWiFi device driver (version 0.9.2) for wireless interfaces, and the ANSR implementation. In addition, ETX and WCETT routing metrics are imple- mented for routing protocols. Finally, the Iperf measurement tool is used for measuring end-to-end throughput, and the numbers are derived by averaging the experimental results of 10 runs, unless otherwise specified.

C.Experimental Results:

We evaluated the improvements achieved by ANSR, includ- ing throughput and channel efficiency, QoS satisfiability, and reduction of ripple effects.

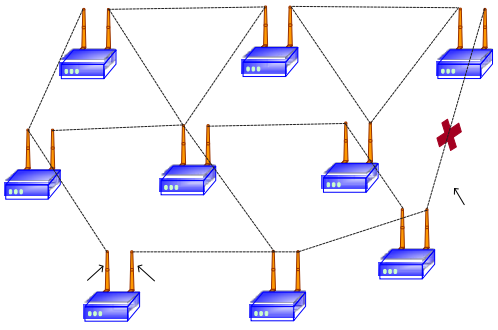
1) Throughput and channel-efficiency gains: We first study throughput and channel-efficiency gains via ANSR's real-time reconfiguration. We run one UDP flow at a maximum rate over a randomly-chosen link in our testbed, while increasing the level of interference every 10 seconds. We also set the QoS requirement of every link to 6 Mbps, and measure the flow's throughput progression every 10 seconds during a 400-second run. For the purpose of comparison, we also ran the same scenario under the local re-routing with a WCETT (Weighted Cumulative Expected Transmission Time) metric, and static channel-assignment algorithms. Note that we do not intentionally run a greedy algorithm in this single-hop scenario, because its effect is subsumed by ANSR.

Figure compares the progression of throughput achieved by the above three methods. ANSR effectively re- configures the network on detection of a failure, achieving 450% and 25.6% more bandwidth than static-assignment and local re-routing, respectively. ANSR accurately detects a link's QoS-failure using link-quality monitoring information, and completes network reconfiguration (i.e.,channel switching) within 15 seconds on average, while the static-assignment experiences severe throughput degradation. Note that the 15- second delay is due mainly to link-quality information update and communication delay with a gateway, and the delay can be adjusted. Further, within the delay, actual channel switch delay is less than 3 ms, which causes negligible flow disruption. On the other hand, the local re-routing improves the throughput



by using a detour path, but still suffers from throughput degradation because of an increased loss rate along its detour path.

2) QoS-satisfaction gain: ANSR enhances varying QoS demands. To show this gain, we first assign links and channels in our testbed as shown in following figure. Here, nodes G, A, and C are a gateway, a mesh router in a conference room, and a mesh router in an office, respectively. We assume that mobile clients in the conference room request video streams through the router A during a meeting, and after the meeting, they return to the office room and connect to the router C. While increasing the number of video streams, we measure the total number of admitted streams after network reconfiguration for each place. We use static, WCETT routing metric that finds a path with diverse channels and ANSR for reconfiguration. QoS-aware reconfiguration planning in ANSR improves the chance for a WMN to meet the varying QoS demands, on average, by 200%. A static channel- assignment algorithm cannot support



more bandwidth than the initial assignment (e.g., 9.2 Mbps from G to C). More- over, using the WCETT metric helps find a path that has channel diversity (e.g.,

RR_1 favors the path G→H→I→F→C, and RR_2 favors the path G→H→E→F→C), but consumes more channel resource, due to the use of longer paths. On the other hand, ANSR effectively discovers and uses idle channels through reconfiguration, thus satisfying QoS demands by up to three times more than static-assignment algorithms

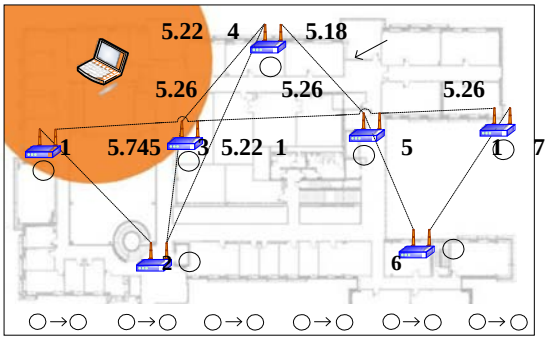
.

3) Avoidance of ripple effects: We also studied ANSR's effectiveness in avoiding the ripple effects of network reconfig- uration. Following figure shows initial channel and flow assignments in a part of our testbed. In this topology, we run 6 UDP flows ($f_1, \cdots, f_6$) each at 4Mbps, and measure each flow's through- put while injecting interference into a target channel. We run same scenarios with 2 different interference frequencies (5.28 and 5.2 Ghz) to induce failures on different links. Also, we use three failure-recovery methods (i.e., local re-routing, greedy, and ANSR) for comparison.

Since ANSR considers the effects of local changes on neigh- boring nodes via *aBAR*, it effectively identifies reconfiguration plans that avoid the ripple effects. It shows the average throughput improvement of the flows after network reconfig- urations, with each of the three recovery schemes. First, with interference on 5.28 Ghz, nodes 1, 3, and 5 experience link- quality degradation, degrading 5 Mbps throughput among the 6 flows. Under ANSR, the network performs reconfiguration and recovers an average 98% of the degraded throughput (4.8 Mbps). On the other hand, the flows via local re-routing achieves 82% of the throughput (3.2 Mbps), because of the use of detour paths (f_5 :5→4→3, f_6 :1→2→3). On the other hand, while partially recovering from the original link failures, the greedy approach causes throughput degradation of neighboring links. This is because one local greedy channel-switching (from 5.26 to 5.32 Ghz) requires the neighboring links' channel (e.g., between nodes 5 and 7) to change, creating interference to other neighboring nodes' link (e.g., between nodes 6 and 7) that use adjacent channels (e.g., 5.3 Ghz).

Next, in the second interference case ,ANSR identifies QoS-satisfying reconfiguration plans, achieving 97% throughput improvement over the degraded throughput. On the detection of the interference, ANSR switches channel of all the fault-related links, based on its planning algorithm to other channel. Naturally, this result (configuration and throughput) is the same as that achieved by the greedy method. On the other hand, the local re-routing causes heavy channel contention for detour paths, degrading neighboring flows' performance (i.e., f_5) as well as others'

Interfering node



V.CONCLUSION

We first make concluding remarks and then discuss some of future work.

A. *Concluding Remarks*

This paper presented an Autonomous Network Settings Reconfiguration (ANSR) that enables a multi-radio WMN to autonomously recover from wireless link failures. ANSR generates an effective reconfiguration plan that requires only local network configuration changes by exploiting channel, radio and path diversity. Furthermore, ANSR effectively identifies re- configuration plans that satisfy applications' QoS constraints, admitting up to two times more flows than static assignment, through QoS-ware planning. Next, ANSR's on-line reconfig- urability allows for real-time failure detection and network reconfiguration, thus improving channel-efficiency by 92%. Our experimental evaluation on a Linux-based implementation have emonstrated the effectiveness of ANSR in recovering from local link-failures and in satisfying applications' diverse QoS demands.

B. *Future work*

Joint optimization with flow assignment and routing: ANSR decouples network reconfiguration from flow assignment and routing. Reconfiguration might be able to achieve better per- formance if two problems are jointly considered. Even though there have been a couple of proposals to solve this problem, they only provide theoretical bounds without considering practical system issues. Even though its design goal is to recover from network failures as a best-effort service, ANSR is the first step to solve this optimization problem, which we will address in a forthcoming paper.

Use of ANSR in IEEE 802.11b/g WMNs: ANSR is mainly evaluated in IEEE 802.11a networks, where 13 orthogonal channels are available. However, ANSR can also be effective in a network with a small number of orthogonal channels (e.g., 3 in IEEE 802.11b/g). Because

ANSR includes a link- association primitive, it can learn available channel capacity by associating with idle interfaces of neighboring nodes, and it further limits the range of a reconfiguration group (e.g., nodes within 4 hops).

REFERENCES

[1].Kyu-Han Kim and Kang G.Shin, "Self-Reconfigurable Wireless Mesh Network", Networks in proceedings of IEEE/ACM , April 2011

[2].K. Ramanchandran, E. Belding-Royer, and M. Buddhikot, "Interference- aware channel assignment in multi-radio wireless mesh networks," in *Proceedings of IEEE InfoCom*, Barcelona, Spain, Apr. 2006.

[3].I.Akyildiz, X.Wang, and W. Wang, "Wireless mesh networks:A survey," Computer Networks, no. 47, pp. 445–487, 2005.

[4].MIT,Roofnet,http://www.pdos.lcs.mit.edu/roofnet.