Autonomic Partitioning for the Smart Control of Wireless Mesh Networks

Sergio Gramacho  
Dept. of Computer Science  
Emory University  
Atlanta, US  
sgramac@emory.edu

Felipe Gramacho  
College of Engineering  
Georgia Institute of Technology  
Atlanta, US  
fmmg3@gatech.edu

Avani Wildani  
Dept. of Computer Science  
Emory University  
Atlanta, US  
avani@mathcs.emory.edu

Abstract—Real-world deployments of low-cost, peer-to-peer Wireless Mesh Networks (WMNs) for communication in underserved settings are hampered by low throughput capacity and high complexity of network control. We present an autonomic smart-agent based WMN design that organizes a node placement (NP) into dynamic network partitions to promote the WMNs’ capacity through increased frequency diversity and advanced network control. Through a custom-built simulation framework that supports agent decision making under real concurrency settings, we show that our smart partitioning technique achieves fast convergence to stable partition sets, relying on local information, under extreme node churn conditions. The design is robust to the average WMN NP density and produces partitions isolated at the physical and link layers with the properties of bounded diameter and node degree, and elected partition control node.

Index Terms—Wireless Mesh Networks, Self-Organization, Topology Control

I. INTRODUCTION

A Wireless Mesh Network (WMN) is an appealing network architecture for low-cost and wide geographical coverage. However, theoretical studies predict insufficient capacity for WMNs [1]–[3], which limit their utility in applications such as enabling large scale peer-to-peer connectivity in regions of the world with limited communication services [4]. Promising analytical WMN capacity scaling results close to linearity have in common the logical or physical grouping of nodes [5], [6], but none of these models takes into account real-world limitations such as the complexity involved in realizing the analytical models as real applications in the distributed nature of WMNs.

We propose a new autonomous agent model to support increased frequency diversity and advanced network control in realistic, low-cost, peer-to-peer, and unconstrained wireless mesh networks.

First, a self-organizing [7] autonomic agent embedded into wireless mesh nodes dynamically evolves a large node placement (NP) into diameter and mesh node degree bounded WMN topology partitions isolated at the physical layer. In a future phase, a secondary self-healing [7] agent will interconnect partitions to recover broad WMN connectivity.

The physical layer partitioning will allow increased parallelism of communication flows on different partitions, which can use different frequencies and improve overall capacity. The autonomic properties guaranteed by the agent will bound the control workload despite the unconstrained settings (density, scale) assumed (details on Section III).

This work concentrates on the first and self-organized partitioning phase. We combine an experimental approach based on simulation with numerical evaluation. Although the node degree manipulation has been analytically studied before through transmission power control [8], our agent design, Smart, relies solely on the self-organization mechanism at the physical layer to achieve a practical degree manipulation, adding partitioning and diameter control. Moreover, we show that the distributed execution of Smart in the mesh nodes leads to stable partitioning solutions. Furthermore, this distributed agent design resembles a consensus mechanism in the primary-backup class [9], relying on leader election, and exploiting properties of the Degree/Diameter Graph problem [10] to achieve convergence. We employ an extensive set of experiments to validate the results mentioned above.

Our contributions are:

- A self-organizing agent design (the Smart agent) that creates WMN partitions with a bounded diameter and node degree and is robust to node density variation,
- An extensive convergence analysis of Smart given its choice of autonomic properties, showing agent optimizations for solving property violations and for providing fast convergence to valid and stable partition sets.
- An agent design robust to connectivity failures, given its reliance on local information.

II. BACKGROUND

For this work, we focus on mesh nodes with a minimal number of wireless interfaces while supporting frequency diversity on distinct partitions. Factors such as node density, link diversity, and cost have been shown to contribute to the throughput capacity of WMNs [5], [11]–[13].

A. Density and Growth

The WMN topology can be densifying or extended regarding its growth pattern [5], and show homogeneous or inhomogeneous density [3]. We are interested in Extended topologies: an approximately constant node density in the area, and area increase when the number of nodes scales up [5].
B. Autonomic Self-Organization

From its inception by IBM [14], [15], a common interpretation of autonomic computing emerged over time. A Self-Organizing (SO) system regards agents controlling (maintaining, improving, recovering) properties in the presence of an external process that adds or removes nodes; and Self-Healing (SH) consists in existing nodes adding or removing edges on the system to recover to a previous condition [16].

Berns et al. [7] proposed a formal definition for forms of autonomic computing (self-* behaviors). They categorize internal attributes controlled by autonomic agents as safety and liveness properties. A safety property is a condition that should not be violated while a liveness property is a condition that should eventually be reached. Furthermore, agents rely on local information: the agent’s internal state and the state of its direct neighbors.

Using definitions in [7], a Self-Stabilizing (SS) system starts in an arbitrary configuration, and recovers to a legal state (safety properties in valid states). A Self-Organizing (SO) system maintains, improves, or restores safety properties in the face of actions related to nodes entering/leaving the system.

As our agent model converges to stable solutions, we can call it a form of SS. Also, our agent model is a representation of SO given its operation under the addition and removal of nodes. Finally, using Lemma 5 in [7], we claim that our agent model is also Self-Configuring.

C. Partitioning to Support Alternative Control Models

Li et al. [17] logically partition the network control in the Software Defined Network (SDN) paradigm using mesh-like topologies comparable to WMNs. The intention is to create logical network segments, each managed by a different SDN controller, to keep the controlling load within bounds. The work does not evaluate improvement on the network capacity by the partitioned control. Controllers are predefined and cannot be changed. The design relies on a separate out-of-band control network for controller interaction. A Master controller executes part of the segmentation algorithm and delegates the second part to the Zone controllers, characterizing a centralized network partitioning scheme. Enforcing frequency diversity in this single radio/channel setting was not feasible. The solution applies to small-medium networks, given the complexity of graph partitioning [18]. Araujo et al. [19] adds in-band control and dynamic definition of zone controllers. However, authors do not explicit their algorithms for partitioning.

Google applied the centralized control paradigm to the wide-area connectivity of their data-centers with the Internet. The flexible choices of routes rendered increased efficiency of WAN links [20]. It was critical to logically partition the control plane into zones to bound the controlling latency and workload. The partition of the network control plane is critical to our problem. However, we approach the problem under the added complexity of unconstrained network size and topology.

Özgür et al. [5] present a theoretical Multiple-input and multiple-output (MIMO) scheme, claiming nearly linear scaling on extended WMNs. Complexity is a limitation for practical utilization. It relies on a hierarchical org. for data flow and on Cooperative MIMO: clusters of nodes behave as MIMO arrays for tx and reception.

Liu et al. claimed a breakthrough [6] in a three-dimensional heterogeneous network setting with hierarchical routing. They apply partitioning based on three-dimensional clustering of nodes, achieving nearly linear scaling by enforcing spatial diversity through tx power control and time diversity through Time Division Multiple Access (TDMA) on the intra-group communication.

We build on the concept of topology manipulation into groupings of nodes as explored by [5], [6], [17], [20]; however, without expecting specific characteristics from the underlying structure of NPs. We guide our design of topology manipulation to support the application of the centralized network control in WMNs in general settings at large scales.

III. DESIGN OF SELF-ORGANIZING WMN NODES

We expect no preliminary network design or manual configuration, characterizing a self-configuring network solution [7]. Under the addition and removal of nodes, this design must guarantee a set of properties critical to the applicability of SDN into WMNs: a self-organizing design [7]. A critical question in dynamic and autonomic network formation is the convergence to stable configurations and how the parameters of the self-configuring, self-organizing agents affect such convergence likelihood.

The agents evolve a large NP (a set of geographical positions) of wireless mesh nodes into partitions holding the properties of bounded diameter and node degree. The network formation is the result of a distributed algorithm executed by the agents in the WMN nodes (software components) which independently make decisions that collectively lead to the final topology. Such decisions are based on the agent’s design that enforces its safety and liveness properties. The organization occurs both at the physical and link layers. The physical isolation allows for using orthogonal frequencies on different partitions. At the link-layer, the use of different network IDs (such as the BSSID - Basic Service Set ID) creates logical isolation to provide robustness to the case of equal frequency in neighboring partitions.

We call this design the Smart agent. The WMN partition diameter property bounds the communication latency, a critical requirement for intra-group SDN-based network control planes. The autonomic aspect is fundamental to this solution, considering that the graph partitioning problem is NP-Hard [18] or NP-Complete [21].

The second safety property of Smart is the control of the mesh node connectivity degree. In the SDN paradigm, this design bounds the number of events per new data flow handled by an SDN controller of partitions. Transmissions of a node $n_i$ in a WMN are received by all its neighbors given an inherently broadcast nature of wireless communication. In a path of distance $h$ for an average WMN node degree $dg$, $ev = dg \cdot h$ events arrive at the controller for every new flow initiated. Therefore, the bounded node degree limits the per-flow workload regarding network control events.
Moreover, the combined bounding on diameter and node degree limits the number of nodes per partition (the Degree/Diameter Graph problem [10]) regardless of any underlying NP density to support the precise workload control in WMN SDN controllers. Finally, our self-organization design also solves the question of electing a mesh node to act as an SDN controller. Similarly to the leader election concept in distributed consensus protocols [22], partitions evolve from a unique origin node, a clear candidate to act as the partition controller.

The general design of the agents involves cycles of operation with the following four phases:

1) Read the environment: internal states, neighbors’ states.
2) Evaluate states: on autonomic functions, deciding to maintain, improve, recover safety and liveness properties.
3) Act: send commands that enforce decisions.
4) Idle: wait until the next cycle (nodes use the network).

The second phase is the main differentiating element (properties and behavior), which is specific to each agent as described in the subsection III-A1. The commands available to use in the third phase are: node creates a mesh partition, node joins a partition, node leaves a partition. Also, a possible decision is maintain membership. We describe the phase 1 of the agent operating cycle in the subsection III-B.

![Generic Agent Cycle](image)

Fig. 1. Generic Agent Cycle. Specialized by different behaviors in the evaluate + act phase.

A. Autonomic Behavior of Agents

1) Smart node agent design: The Smart agent joins the largest nearby partition for which it finds the shortest path of hop-distance at most $h$ to the partition origin. It is trivial to verify that this behavior induces a diameter to partitions, at most, $d = 2 \times h$: the origin node is a member of any shortest path from border to the origin, and the diameter unites two shortest paths from origin to the border.

The is origin attribute is true for an instance of Smart which creates a partition, and false otherwise.

Optionally, Smart nodes assume a node degree $dg$ upper bound constraint to decide on their partition membership. If no membership option is valid, Smart nodes create a new partition. They continually review their membership decision following the generic agent cycle described in Section III.

Smart attempts to change its partition in every new cycle to the largest valid neighboring partition which is, at least, $sp$ percent larger (a threshold) than the agent’s current partition.

A valid neighboring partition has i) at least one node at comm. reach of the deciding node, ii) all nodes not violating safety properties, iii) all nodes will not violate safety properties after the addition of the deciding node.

More formally, let $m$ be a Smart agent node reviewing its properties. Let $P$ be the set of all partitions while $P_m \subseteq P$ is the set of all nearby partitions to $m$; let $k \in P_m$ be a nearby partition, $o_k$ is the origin node of partition $k$. Let $k_c$ be the current partition of $m$ if it is already connected, and $k_c \in P_m$.

Let $F_{distance \_sp}(i, j)$ be a function that returns the shortest-path distance between two nodes. Let $F_{DT} \rightarrow P_m$ be a function that eliminates nearby partitions $k$ for which $F_{distance \_sp}(m, o_k) > h, \forall k \in P_m$.

Let $F_{degree}(i)$ be a function that returns the degree of a node $i$; let $F_{DG} \rightarrow P_m$ be a function that removes partitions $k \in P_m$ if there exists any node $i \in k$ such that $i$ will be/is a neighbor to $m$ and $F_{degree}(i) \geq dg, \forall i \in k, \forall k \in P_m$.

Let $F_{neighbors}(k, i)$ be a function that returns the number of future neighbors of a node $i$ if it joins a neighboring partition $k \in P_m$; let $F_{NEIGH} \rightarrow P_m$ be a function that removes partitions $k \in P_m$ if $F_{neighbors}(k, m) > dg, \forall k \in P_m$.

Let $F_{size}(k)$ be a function that returns the size in number of nodes of a partition $k$, let $s_m$ be the size of the current partition $k_c$ of the agent $m$, let $F_{ES} \rightarrow P_m$ be a function that i) eliminates partitions $k \in P_m$ for which $F_{size}(k) < (1 + sp) \times s_m$, and ii) sorts $P_m$ by $F_{size}(k), \forall k \in P_m$ into the ordered list $P'_m$.

If $P'_m \neq \emptyset$, the first item $P'_m[0] = k_b$ is the best partition membership option regarding size. If $P'_m = \emptyset$ and $k_c \in P_m$ (current partition $k_c$, checked as valid regarding safety properties), we add back $k_c$ into $P'_m$ as an option (staying on the agent’s current partition).

On the second phase of the generic agent cycle, Smart executes: $F_{DT} \rightarrow P_m, F_{DG} \rightarrow P_m, F_{NEIGH} \rightarrow P_m$, and finally $P'_m = F_{ES} \rightarrow P_m$.

Let $A_S$ be the decision of Smart as a set of actions:

$$A_S = \begin{cases} \{create\_partition\} & \text{if } P'_m = \emptyset \\
\emptyset & \text{if } P'_m = \{k_c\} \\
\{leave\_partition(k_c), join\_partition(k_b)\} & \text{otherwise} \end{cases}$$

We evaluated the operation of versions of Smart holding different values for the safety property maximum node degree: 5, 10, and no degree constraint.

B. Agent information

Agents base their decisions on local information (internal and neighbors), independently of network connectivity. They read the environment (wireless scan) at the link-layer, which also include physical layer attributes (e.g., operating frequency). If connected to a network, nodes can obtain attributes from their routing layer (e.g., partition size in nodes derived from the node’s routing table, or an internal state of an SDN controller propagated to its managed SDN nodes). Nodes do not rely on synchronous communication to each other at the time of decision making (e.g., a network layer send/receive operation) neither information forwarding (such as in a distributed protocol). Such a design is robust to connectivity failures, supporting self-organization in the absence of any
initial input. It resembles a form of autonomic bootstrapping similar to the concept of self-stabilization [7].

Nodes share (broadcast) their local properties at frequent intervals as part of the wireless link-layer control protocol; nodes also read the environment (receive broadcasts) frequently. The broadcast interval $b$ is much smaller than the maximum observation interval: $b \ll o$, and $o = b \times k$ ($k$ an integer constant that supports observing the number of frequencies $c$ in use: $k \geq c$). Therefore, an observation action captures complete shared information from neighbors with high probability. In the WiFi standard, the default broadcast period $b$ is 0.1 sec for the link-layer control protocol (beacon interval) [23]. Our experiments have epochs (the generic agent cycle) of $e = 90$ sec to accommodate $a$) observation and decision making of agents integrated to the b) synchronous network communication that consumes the network. Precisely, $a$) takes place in the initial 10% of the epoch while in the remaining 90% the agent is idle, moment reserved for $b$).

The Information Elements field on a set of management packets of the IEEE 802.11 standards [23] exemplify a data structure for attribute sharing in a real execution setting.

In this work, the autonomic agents rely on a simulated agent platform, the ASim (Section IV-A), to obtain their information. Section III-C presents a detailed analysis of convergence of Smart under concurrent operation supported by ASim.

C. Convergence of the Smart agent

In this section, we analyze conditions for the consensus to a stable partition membership solution given the autonomic properties of the Smart agent. Convergence is an essential outcome of the network organization; a non-converging solution stresses the network control plane on routing information update in the distributed routing schemes or on managing a large number of network control events on an SDN controller.

1) Triggers for slow convergence: We recall that liveness properties motivate a node to review its current settings in favor of improved ones. Safety properties are hard limits not to be violated, restricting the possible partition membership options.

Each autonomic property will lead to violations in different conditions regarding joining or leaving a partition. The versions of Smart comprise different combinations of safety and liveness properties, thus leading to different expectations regarding their impact to a stable partitioning.

Solving a violation in the agent’s current partition membership requires leaving the current partition in favor of another (perceivably valid) option. In attempting to maintain its liveness by always choosing its best option, a node can change its partition membership, also implying a sequence of leave, join partitions.

When leaving a partition, a node can lead other nodes to violations. The lemma that follows describes these violations.

**Lemma 1. The action of leaving partitions can cause safety violations of the diameter bound property.**

*Proof.* Assume a partition $P$ with its origin node $o_P \in P$. We define disconnected nodes to a partition $P$ as nodes in any segment $S_i$ of $P$ so that $o_P \notin S_i$ (not in the segment of the origin node). A node $n_j \in P$ leaving its partition can create segments in $P$ thus disconnecting nodes $n_j \in P$ if $n_j \in S_i$ and $o_P \notin S_i$. A disconnected node is in a safety violation condition given its infinite distance to the origin node.

Moreover, $n_i$ can induce increased distance to the origin node $o_P \notin P$ due to the elimination of the shortest path of a node $n_j \in P$ to the origin. If the distance of $n_j$ to the origin goes above the limit $h$, the node $n_j$ is in safety violation. □

When joining partitions, the concurrent execution of the agents’ read + evaluate + act (REA) phases could lead to decisions based on outdated information which could lead to safety property violations. We analyze Smart’s two safety properties, partition diameter bound, and node degree bound regarding the join event in the two lemmas that follow.

**Lemma 2. The concurrent addition of nodes will not induce violations of the diameter bound property.**

*Proof.* Assume the concurrent addition of two new nodes $n_{1}, n_{2}$ to a partition $P$. The concurrent REA does not change the condition that $n_{1}, n_{2}$ used, say, the existence of a node $n_{0} \in P$ with distance to the partition origin at most $h - 1$. Therefore, although both $n_{1}$ and $n_{2}$ were not aware of each other as future neighbors when they made their join decision, they end up with a valid distance of $h$ based on their connectivity to $n_{0}$.

**Lemma 3. The concurrent addition of nodes can induce violations of the maximum degree bound property.**

*Proof.* Assume a partition $P$ and the existence of a node $n_{0} \in P$ with degree $d_{g} - 1$ ($d_{g}$: the maximum node degree bound). If the nodes $n_{1} \notin P, n_{2} \notin P$ concurrently join the partition $P$ using an outdated perception of $n_{0}$’s degree below the limit, they unintentionally induce a degree $d_{g} + 1$ onto $n_{0}$, causing a safety violation. □

In summary, the version Smart-any of the agent: 1) only induces a violation of the maximum distance to origin property when leaving a partition to maintain its liveness or to solve a violation induced to it by other nodes that left its partition.

The version Smart-dg can induce violations of the maximum distance to origin property 1) when leaving a partition 1a) to maintain its liveness or 1b) to solve a safety violation. Also, Smart-dg can induce violations of the maximum node degree property 2) when joining a partition due to outdated information given concurrent REA.

Finally, the following theorem illustrates a divergence scenario by Smart-dg.

**Theorem 1.** Smart-dg can diverge to obtain a stable partitioning.

*Proof.* Assume two nearby nodes $n_{1}, n_{2}$ that both can join partitions $P_{1}, P_{2}$. Also, assume the interval $l$ comprising the REA phases of the generic agent cycle. If, say, $n_{2}$ starts reading the environment after $n_{1}$, there exists the chance that $n_{2}$’s evaluation will not reflect the effects of $n_{1}$’s action into the environment. As a consequence, $n_{1}$ and $n_{2}$ could decide...
to join the same partition, say, $P_i$, and cause a violation of the maximum node degree safety property. This is the violation described in the Lemma 3. If we assume no change to the timing of the agents’ cycles, the same concurrent REA effect could occur later and lead both to decide on joining the other partition, say, $P_j$. Again, it could lead to a safety property violation of the Lemma 3. Therefore, it is possible that $n_1, n_2$ enter a continued change of partition memberships, characterizing a divergence to a stable partitioning.

2) Optimizations to improve convergence: The key challenge to reduce convergence time is letting the agents achieve an ordering of execution of their REA such that it minimizes the impact of concurrency on corrupting the environment information.

A first solution for achieving such a minimal corruption ordering is the randomization of the agent execution on the interval $l$ of the agent cycle after detecting a safety property violation. We named it RAND. Once detecting itself in an invalid state, the agent introduces a random delay on the start of its next REA, attempting to induce a separation to the REA of other nearby agents. Other problems related to concurrent operation apply randomized delays such as accelerating the convergence of the leader election on the Raft consensus protocol [22] and the channel access mechanism of the DCF (distributed coordination function) used in WiFi [23].

A second optimization consists in defining an arbitrary ordering for solving violations amongst nodes of a partition. However, distributed agents agreeing on a unique sequence is a complex problem [9], [22], [24]. We apply a relaxed ordering based on the distance of nodes from the partition origin: nodes closer to the origin wait for their neighbors which are away from the origin and also in violation. This simple design creates a partial ordering that resembles prioritizing nodes closer to the border of partitions for violation solution. This optimization is named PORD.

3) Modeling divergence: Here we model a divergence indication for the case of agents applying no optimizations for violation solution. We split the membership divergence scenario of Smart-dg in three conditions: i) the proximity of nodes letting them to interfere on each other’s environment, ii) the probability of some concurrency level on nodes’s REA phases ($P_C$), iii) the probability of impact given the concurrency level ($P_I$). While $i)$ is a function of node density and probability distribution for nodes’ positions; the other two conditions are functions of the timing parameters of the generic agent cycle.

As described in Section III-B, let $b$ be the agent’s broadcast interval, $c$ the number of frequencies to observe, $k \geq c$ a constant that allows reading all frequencies, and the combined REA interval $l = b \times (k + 1)$, assuming the duration of evaluate + act as $b$ for simplicity. Finally, $e$ is the duration of the total agent cycle $Y = \{b_1, b_2, ..., b_d\}$, $e \geq l$, and $d = e/b = |Y|$ the number of broadcast intervals $d$ in the epoch $e$.

We recall that a divergence scenario requires at least one node continually changing decisions over time. For that, a node $n_i$ needs to be affected by, at least, concurrent decisions of a neighboring node, or by concurrent decisions of neighbors of its neighbors. The former directly affects attributes of the node $n_i$ while the latter can induce changes to neighbors (neigh($n_i$)) that are observed by $n_i$. Therefore, nodes in an area $A = f(2r)$ of a reference node $n_i$ can directly induce safety property violations. In the interest of space, we omit the deduction of Eq. 2.

$$P_D = \left( \frac{b}{c} \right) \cdot \frac{k + 2}{2} \cdot (4\pi r^2 \cdot N_D - 1)$$ (2)

From Eq. 2, reducing the indication $P_D$ implies $a)$ decreasing $b, k; b)$ decreasing the number of nodes involved in $E_i$; or $c)$ increasing the epoch $e$. Any of these alternatives create undesired collateral effects; however, the optimizations described in sub-section III-C2 achieves $b)$ given its ordering approach without enforcing restrictions in the communication range $r$ or node density $N_D$. Furthermore, the increase of $e$ has diminishing returns, given its inverse relation to $P_D$. 

D. Visual outcome of agents’ behavior

![Fig. 2. Underlying maximum possible connectivity of 1000 nodes on a ≈ 1.44 Km² region in Chatham county, GA (nearby Savannah, GA). Shows the maximum set of neighboring options (or a single partition solution). Wireless standard IEEE 802.11a. Density of 1/1000 node/m².](image)

Fig. 2 represents the maximum underlying connectivity if all nodes operate on the same frequency in the physical layer, the same logical network (SSID, BSSID) at the link-layer, using their nominal transmit power. Therefore, no additional connectivity can exist. Node placement based on the U.S. buildings data set from [25].

IV. RESULTS

This section presents the agent and the wireless network simulation environment used in this work, the experimentation settings, and results of agents convergence.
### A. Experimentation Platform

We built an experimentation platform to support agent decision-making, named Agent-based Networks Simulation (ANS), which is composed of the following modules:

- **Agent Simulator - ASim**: a discrete event simulator written in Python to allow for fast prototyping of agent models and independence of specific network simulators. It implements the high-level decision making of agents in the nodes.

- **Network Simulator - NetSim**: a layer built on top of the well known and validated ns-3 network simulator [26] to dynamically command the configuration of network nodes and their network stack. NetSim is built in C++ using the programming guidelines of ns-3.

### B. Experimentation settings

The following list describes NPs and settings of the wireless subsystem.

1. Nodes have randomized positions controlled by an NP random variable (uniformly random). We used a common set of 30 different seeds to all agent models, creating 30 different NPs.
2. We control the average NP density over the experimentation area as described in Section IV-C.
3. Nodes use the IEEE 802.11a wireless standard.
4. The mesh nodes’ IEEE 802.11a physical layer (5 GHz band) has a 20 MHz bandwidth. The tx power is 16 dBm, the gain of the tx/rx antennas is 1 dBi (defaults).
5. The link-layer of the mesh nodes uses the IBSS (Independent Basic Service Set) mode, creating WMNs through multi-point association.

### C. Experiments evaluating convergence

![Fig. 4. Convergence for all optimizations for small epochs (e = 9s): the slowest conv. of up to 16 epochs, and 30 for outlier cases (diverged).](image)

We simulated the convergence of Smart enabling REA concurrency, a real scenario in which agents can decide based on outdated information.

For the results in this section, assume $b = 0.1, k = 8, r = 100$ (IEEE 802.11a). The plots show results for 12 agent-density pairs of Smart which are a combination of one degree constraint in $dq = \{\text{any}, 10, 5\}$ with one node density inverse in $\{1600, 1200, 800, 400\}$ $m^2/nodes$. Figs. 4, 5 aggregate results for all optimization sets: **NONE, RAND, RAND+PORD**. Each agent-density result combines experiments of 30 different NPs, and each NP with a total of 400 nodes. Each experiment had a total time of 30 epochs. At least 90% of nodes are spawned in the first epoch to represent the extreme case of 90% node churn on the return of a power outage. The $sp$ parameter of the partition size liveness property is 0.2 for the origin node and 0.1 for any other node. The evaluation of $sp$ will be the subject of future work.

Using Fig. 4 left ($Opt=NONE$), in which agents apply no optimizations for solving safety violations, we confirm assumptions of the analytical model for divergence of Section III-C: a) the Smart-any agent has much faster convergence; b) increased density of nodes increases the time to convergence. We add that a more constrained degree (e.g., $dq = 5$) increases convergence time. Although the $PD$ expression does not capture the aspect of degree constraint as a parameter, the intuition of its impact is that a more constrained degree (smaller $dg$ value) will increase the likelihood of more nodes in violation.

When comparing the slow convergence shown on Fig. 4 left to the faster convergence on Fig. 4 center ($Opt=RAND$) and right ($Opt=RAND+PORD$), we verify the importance of optimizations to cope with small epochs and highly constrained maximum degrees (e.g., $dg = 5$). Small epochs allow for faster absolute convergence while the highly constrained degree improves capacity and reduces the number of control events in an SDN setting (see discussions on Section III and V). Moreover, Fig. 4 left presents cases of divergence: the outlier points with convergence time of 30 epochs did not converge until the end of the experiment.
Finally, we confirm the positive impact on convergence of increasing the epoch time. Fig. 5 presents faster convergence in the number of epochs. However, in terms of absolute time they represent a slower total convergence. For epochs $e = 9$ sec, assuming an upper bound of $U_b = 9$ epochs (Fig. 4 right), the convergence time is $C \approx 9 \times 9 = 81$ sec. For epochs $e = 90$ sec, an upper bound of $U_b = 7$ epochs (Fig. 5 right) renders a convergence time of $C \approx 90 \times 7 = 630$ sec.

Finally, the PORD optimization provides additional benefits not presented here in the interest of space: a significant reduction on the number of node partition changes (moves) and safety violation events. Node moves are at most 2000, 1500, 1000 and safety violations are at most 1200, 700, 500 respectively for the optimization sets NONE, RAND, RAND+PORD. This reduction relieves the adaptation stress of network layers 2-3 to the topological changes induced by the mesh agents.

Figs. 6, 7 provide insights on the evolution of safety violations. The initially erratic control of the safety properties later converged to the defined objectives.

D. Resulting WMN partitioning structure

This section analyzes the resulting partitioned WMN topology regarding the size of the partitions, recalling that the combined node degree and partition diameter constraints result in a bounded size for the partitions: the Degree/Diameter Graph problem [10].

The resulting partitioning structure is a relevant outcome regarding recovering the global connectivity of the WMN (by Self-Healing agents). A highly fragmented partition set with mostly small partitions (below the maximum size possible given the degree/diameter constraint) implies the need of a larger number of nodes with a second wireless interface. Moreover, a large number of partitions when compared to the available set of frequencies might render frequency diversity inefficient due to the reuse of frequencies by nearby WMN partitions. Therefore, the ideal outcome is a partition set with the largest possible size to minimize the necessary number of mesh nodes with a 2nd wireless interface for inter-partition connectivity and efficient frequency diversity.

Fig. 8 confirms the bounded sizes as outcomes of the partitioning determined by Smart-dg. For Smart-any, we verify that the undesirable outcome of a single partition containing all 400 nodes is possible for high node densities. Moreover, smaller degrees induce smaller maximum partition sizes. Furthermore, the different optimizations applied did not change the outcome regarding partition sizes (omitted in the interest of space).

Fig. 9 presents the frequency of partition sizes over all 30 experimented NPs. It shows results of two versions of Smart-dg (dg05, dg10) and Smart-any. Results confirm that Smart-any cannot enforce a bounded partition size. We find non-partitioned topology outcomes (node density of 1/400 nodes/m², size 400), as previously shown in Fig. 8.

Comparing the versions of Smart-dg (dg05, dg10), we realize that a more relaxed degree constraint (e.g., $dg = 10$) resulted in fewer small partitions. We hypothesize that this result will facilitate the global WMN connectivity by the self-healing function, given the existence of fewer partitions.

Therefore, there is a trade-off regarding the selection of max. degree for Smart-dg. A more constrained max. degree (e.g., $dg = 5$) induces a lower control workload in the SDN paradigm (less nodes, reduced number of control events per flow). However, the topology structure with a larger number of small partitions requires a higher number of nodes with dual wireless interfaces for recovering global connectivity.

V. CONCLUSION AND FUTURE WORK

This work proposes and evaluates the autonomic WMN topology manipulation by self-organizing agents, presenting the Smart agent which is robust to a range of node densities, and converges to stable topologies.

The partitioned topology produced by Smart holds the properties of bounded partition diameter and node degree allowing for the general application of the SDN paradigm to
the local routing regime of large scale WMNs. Partitions with bounded diameter induce bounded latency on intra-partition (local) communication, critical for the implementation of the SDN control network. The bounded node degree limits the number of control events in an SDN setting given that transmissions of a WMN node are received by all its neighboring nodes as the result of an inherently broadcast nature of wireless communication, multiplying the control events per new flow by the typical node degree. Moreover, the combined degree/diameter enforcement bounds the number of nodes in a partition [10] for a precise per partition SDN control plane workload management. Finally, the partition formation approach of the Smart agent around an origin node elects this node a definite candidate for SDN control functions.

Furthermore, the convergence analysis and results provide confidence in achieving stable partitioning even in extreme node churn conditions of the majority of nodes activating nearly at the same time (e.g., power outage return).

As future work, we will address WMN global connectivity. Newly introduced Self-Healing agents will react to adversary actions that partition the WMN after an organizing action, characterizing a competitive relationship between the organizing and healing agents. The healing agents will leverage frequency diversity provided by a per-node average number of wireless interfaces in the range [1, 2] which enforces the directives of a low-cost, low-complexity solution on the design of our autonomic mesh nodes.

ACKNOWLEDGMENT

This research was partially supported by CAPES Brazil and the Laney Graduate School from Emory University.

REFERENCES