Re-establishing Network Connectivity in Post-Disaster Scenarios through Mobile Cognitive Radio Networks[*]

Angelo Trotta[†1], Marco Di Felice[‡1], Luca Bedogni[§1] and Luciano Bononi[¶1]

[1]Department of Computer Science and Engineering, University of Bologna, Italy

Abstract

Network interoperability and self-organization constitute important communication requirements in disaster recovery scenarios. Here, the original communication infrastructure might be partially or completely damaged, and the whole network might be partitioned into segments (called islands in the following) that might operate on different frequencies/wireless technologies. In this paper, we investigate techniques to maximally re-establish the connectivity among heterogeneous islands through the utilization of specialized repairing units called Stem Nodes (SNs). A SN combines spectrum reconfigurability (offered by the Software Defined Radio technology) with self-positioning and dynamic routing functionalities, and thus it is able to replace damaged components of the original infrastructure. Moreover, sets of SNs can self-organize into multi-hop mesh structures connecting heterogeneous islands. We study the problem of determining the optimal deployment of SNs so that the number of connected devices of the original network is maximized. Given the NP-hardness of the problem, we propose approximated solutions with reduced computational complexity. We then compare the centralized solution with a distributed algorithm (based on virtual springs approach) that enables SNs to explore the environment in both space/frequency domains, and to self-organize into virtual mesh structures. Simulation results confirm the effectiveness of the distributed algorithm to maximally re-establish the network connectivity even on large-scale scenarios.

[*]DOI: 10.1109/MedHocNet.2013.6767405

[†]trotta@cs.unibo.it

[‡]difelice@cs.unibo.it

[§]lbedogni@cs.unibo.it

[¶]bononi@cs.unibo.it

1 Introduction

In these last years, Cognitive Radio (CR) technology [1] has emerged as one of the key solutions to deploy highly reconfigurable wireless systems which are able to adapt their transmitting configuration to the characteristics of the environment. Most of the CR-based applications proposed so far rely on the dynamic spectrum access capabilities of Software Defined Radio (SDR) platforms with the aim of enhancing the spectrum utilization, and of meeting the Quality of Service (QoS) requirements of wireless applications. However, the potential of CR and SDR technologies to support multiple air-interfaces and modulation formats, and thus to guarantee interoperability among heterogeneous wireless networks is far to be completely investigated. Beside the advantages in terms of pervasive wireless access for the end-users, network interoperability constitutes a fundamental communication requirement in special scenarios like the emergency ones [2][3]. Indeed, recent natural calamities (e.g. the earthquake of Japan in 2011) demonstrated worldwide the fragility of the cellular infrastructure, and the difficulties of first-responders to coordinate in presence of network partitions and heterogeneous wireless access technologies [2][4].
In this paper, we investigate the application of CR technology to address the interoperability issues, and to re-establish the network connectivity in large-scale wireless networks whose functionalities might be compromised by the occurrence of a natural calamity. Without loss of generality, we consider a pre-disaster scenario in which multiple wireless devices operating on different frequencies/technologies accesses (e.g. WiFi, WiMAX, etc) are interconnected together through dedicated infrastructure components (e.g. bridges, routers, etc). After the occurrence of a natural calamity, parts of the original infrastructure might be disrupted, and network partitions (called *islands* in the following) might be formed on different frequencies. In [5], we have investigated the utilization of dedicated, self-configuring repairing units, called Stem Nodes (SNs), which might dynamically replace the functionalities of a compromised node or chain of nodes. Each SN is an highly adaptive wireless device combining the spectrum reconfigurability offered by CR/SDR technologies, with self-positioning and dynamic routing capabilities. Through the SDR interfaces, each SN is able to sense the environment, detect the presence of islands operating on different frequencies and reconfigure its radios in order to serve as a bridge. Moreover, through wheels and controlled mobility algorithms, each SN can explore the environment, and adjust its own location. Finally, through cooperation techniques, sets of SNs can autonomously self-organize into mesh structures that guarantee multi-hop connectivity among the islands (see Figure 1 for an example). In this paper, we further investigate the utilization of SNs on emergency scenarios, and we provide two important extensions in terms of theoretical and evaluation results. First, we study the problem of determining the optimal deployment of SNs (in terms of placement and frequencies to use) so that the number of connected devices is maximized. We show that this issue is similar to the well-known problem of determining the Steiner Minimum Tree (SMT) [13][14][15][16] on a weighted graph in metric space. Given the

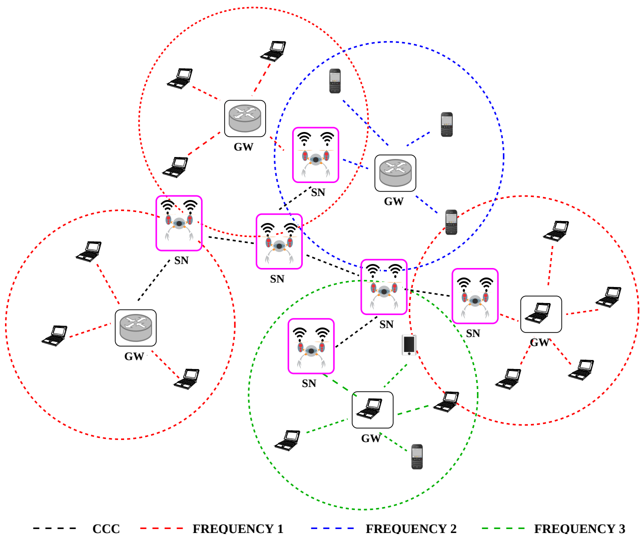


Figure 1: The post-disaster scenario with the SNs.

computational hardness of computing the SMT on large-scale graphs, we study centralized algorithms that provide an approximation ratio to the optimal solution while reducing the computation time on specific topologies. Second, we compare the centralized and distributed algorithms on realistic large-scale scenarios, and we demonstrate the performance improvement of our approach over existing chain-based solutions [9].
The rest of the paper is organized as follows. In Section 2, we review the state-of-art of wireless systems for disaster-recovery, focusing on CR and SDR applications. In Section 3 we discuss the system model and we formally introduce the Maximum Vertexes Clustering (MVC) problem. In Section 4 we provide optimal and approximated centralized solutions to the MVC problem, while in Section 5 we briefly describe the distributed protocol presented in [5]. Section 6 compares the centralized and distributed solutions through extensive Omnet++ simulations. Conclusions follow in Section 7.

2 Related Works

In these last years, lots of attentions have been dedicated to wireless networks for emergency communication. Starting from the assumption that a centralized infrastructure might not be available in a post-disaster scenario, most of these studies rely on ad hoc networking principles, and propose solutions for dynamic route configuration or first responders' coordination. To this purpose, adaptive routing protocols [17] have been extensively studied over dynamic topology scenarios like the emergency ones. Similarly, controlled mobility techniques have been investigated to enable a swarm of robots to explore the scenario and to self-organize into virtual backbones that connect the survivors. Among the others, we cite the well-known virtual force algorithm (VFA) [6], the virtual spring mesh algorithm (VSM) [7] and the enhancements proposed in [8], where the authors consider the problem of enabling a swarm of robots to explore an unknown area while keeping the mesh connectivity among robots. Cognitive Radio (CR) applications have also been proposed for emergency communication. In [10], the functional requirements and system requirements of mobile ad hoc networks built on top of CR systems are reviewed, and a novel MAC protocol is proposed for emergency environments. In [3], five spectrum sharing model are discussed to support the bandwidth requirements of public safety applications. In [2], the authors propose to utilize TV-white space for deploying Portable Cognitive Emergency Networks (PCEN) among first responders, and analyze the interference problem between neighbour PCENs. In [11], the DSA capabilities of CR devices are leveraged to access lower frequencies that provide larger transmission range for emergency-related data. While all these works study the application of CR principles to increase the available bandwidth for safety applications, less research is focused on CR systems for interoperability issues. In [12], the authors discuss how communication with survivors can be accomplished in post-disaster scenarios, and the advantages of CR systems for geo-localization. In our approach, network interoperability is provided by specialized recovering units (the SNs) that combine the three layers of reconfigurability (i.e. spectrum, positioning and routing) into a single device architecture. Compared to our previous publication [5], we address here the problem of connectivity recovering from an algorithmic point of view, we determine optimal and approximated centralized solutions, and we compare them with the distributed algorithm of [5] over simulated large-scale network scenarios.

3 Network Model for Post-Disaster Recovery

3.1 System Model

We consider a post-disaster scenario in which the original wireless infrastructure has been partitioned into several network segments, called *islands* in the following. Each island i is composed by a set of homogeneous n_i devices, each working on the same frequency f_i and utilizing the same access technology (e.g. WiFi, Zigbee, etc). Without loss of generality, we assume in this work a maximum number of different frequencies equal to N_{freq}. Moreover, we assume that on each island i, there exists a gateway G_i to which all the other devices are attached. The gateways perform intra-island routing, and might be part of the original infrastructure (still working after the disaster), or they might be dynamically chosen among the survived devices, in order to cover the routing task. To this purpose, any clustering scheme with gateway election metrics like [18] can be used. In order to favour the recovery operations, each gateway G_i periodically broadcasts `HELLO` messages on frequency f_i, including the n_i value (i.e. the number of devices on the current island). Figure 1 shows the post-disaster scenario with heterogeneous islands operating on different frequencies/access technologies.

3.2 Mobile CR devices for Post-Disaster Operations

The scope of this paper is to investigate solutions to maximally re-establish the connectivity among the islands through the utilization of specialized recovering units, called Stem-Nodes (SN) to emphasize their ability to accomplish multiple tasks based on the system needs. More specifically, each SN offers three layers of self-reconfiguration:

- *Spectrum Reconfigurability.* Each SN is equipped with $N_{sdr} > 3$ radio interfaces provided with Software Defined Radio functionalities. One radio is used to communicate with the other SNs on a Common Control Channel (whose deployment is out of the scope of this paper, interested readers can refer to [1]), while the remaining ones can be used to transmit data on any of the N_{freq} frequencies, opportunely configuring the network stack based on the wireless technology in use on frequency f_i.
- *Location Reconfigurability.* Each SN is provided with wheels and motion control, and thus can self-adjust its own location in the current scenario.
- *Route Reconfigurability.* Each SN runs a link state protocol (like OLSR), through which it can dynamically re-configure the routing process based on the known network destinations.

During the emergency, SNs are injected into the scenario by first responders or safety agencies. While moving, a SN senses the spectrum frequencies. Once two or more gateways are detected, a SN can reconfigure its radio settings and routing functionalities in order to work as a bridge among them. Moreover, multiple SNs can self-organize into mesh structures that can potentially interconnect all the devices of the scenario, as depicted in Figure 1.

3.3 Problem Formulation

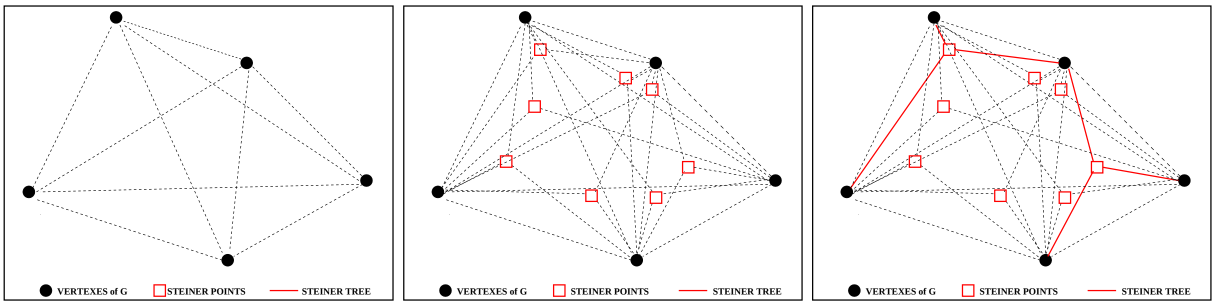
Like in [18], we say that two islands i and j are interconnected (and we denote this fact with the notation: $i \leftrightarrow j$) whether there exists a path between G_i and G_j passing through SNs. Let S be the sets of islands interconnected through the SNs such that:

$$i \leftrightarrow j \ \forall i \in S, \forall j \in S, i \neq j \tag{1}$$

Moreover, we define the cardinality of S, i.e. the total number of devices connected through the SNs:

$$|S| = \sum_{i \in S} n_i \tag{2}$$

as a metric to reflect the ability of SNs to repair the original network infrastructure. In this paper, we address the problem of determining the optimal configuration of SNs so that the value of $|S|$ maximized. The configuration of



(a) (b) (c)

Figure 2: The input graph G is shown in Figure 2(a). The graph after Step 1 and Step2 of the `3-MVC` algorithm is shown in Figures 2(b) and 2(c), respectively.

each SN is defined in terms of: (i) location where to move and (ii) frequency to utilize on each of the available N_{sdr} interfaces. We refer to this problem as the Maximum Vertexes Clustering (MVC) in the following.

4 The MVC Problem: Theoretical Approach

In this Section, we provide optimal and approximated solutions to the MVC problem stated above. To ease of modeling, we assume that N_{sdr}=4, i.e. each SN is equipped with four radios, one tuned on the CCC, and the other three used to connect to gateways. The case with $N_{sdr} \neq 4$ will be considered as future work. We model the post-disaster scenario of Figure 1 through a graph G=$\{V, E, l, w, c\}$, where:

- V is the set of vertexes in a metric space. In our case, V represents the set of islands/gateways to connect.
- E is the set of edges: $u(v_i, v_j)$, with $v_i \in V$, $v_j \in V$. The graph G is fully connected, i.e. there exists an edge $u(v_i, v_j) \in E$, $\forall$ $v_i \in V$, $v_j \in V$, $i \neq j$.
- $l : V \rightarrow N^+$ is a function assigning a label (an integer) to each vertex. In our case, the label of vertex v_i is the frequency f_i used by the corresponding gateway.
- $w : V \rightarrow N^+$ is a function assigning a weight to each vertex. In our case, $w(v)$=n_v, i.e. the weight of vertex v is represented by the cardinality of the island n_v. We define the weight of the graph G, i.e. $W(G)$, as follows:

$$W(G) = \sum_{v \in V} w(v) \tag{3}$$

- $c : E \rightarrow N^+$ is a function assigning a cost to each vertex. In our case, $c(u(v_i, v_j))$ represents the average number of SNs needed to connect gateways v_i and v_j. Let $d(v_i, v_j)$ be the Euclidean distance between v_i and v_j, and R the transmitting range of a SN, then $c(u(v_i, v_j))$ can be approximated as follows:

$$c(u(v_i, v_j)) = \left\lceil \frac{d(v_i, v_j)}{R} \right\rceil \tag{4}$$

In case $\frac{d(v_i,v_j)}{R}$ is equal to zero (i.e. two islands are overlapped), but $l(v_i) \neq l(v_j)$ (i.e. different frequencies are used), we set $c(u(v_i, v_j))$ equal to 1, to indicate the need of a SN working as a bridge. Finally, we define the cost of the graph G, i.e. $c(G)$, as the sum of the edge costs:

$$C(G) = \sum_{u \in E} c(u(v_i, v_j)) \tag{5}$$

Figure 2(a) shows the graph G corresponding to the scenario of Figure 1. Given G=$\{V, E, l, w, c\}$ and M (i.e. the number of available SNs), the MVC can be formulated as the problem of determining the optimal tree $T_{MVC}(P, D)$, where P=$T \cup S$, with the terminal points $T \subseteq V$ and S a set of new points (with $w(s) = 0 \ \forall \ s \in S$), such that the tree weight $W(T_{MVC})$ is maximized (i.e. the maximum number of devices are connected by the SNs), while the tree cost $C(T_{MVC}) \leq M$ (i.e. the number of SNs used to connect the islands does not exceed the current availability).

The MVC is similar to the well-known problem of determining the *Steiner Minimum Tree* (SMT) on a weighted graph in metric space. Given a graph G_s=$\{V_s, E_s, l_s, w_s, c_s\}$, the $SMT(G_s)$ is defined as the tree $T_s(P_s, D_s)$, with $V_s \subseteq P_s$, such that the tree cost $C(T_{MVC})$ is minimized [13][14][15][16]. Differently from the classical spanning tree problem, the SMT includes both vertexes of the graph (called *terminal points*) and new vertexes p (with $p \in P_s$ and $p \notin V_s$), called *Steiner points*. The optimal solution to the MVC problem can thus be determined in a straightforward way by calculating the $SMT(G_i)$ for each possible sub-graph G_i composed by a subset of vertexes of G, filtering the trees with $C(SMT(G_i)) \leq M$, and then returning the one having the maximum weight $W(SMT(G_i))$. The pseudo-code of the `OPTIMAL-MVC` algorithm is shown by Algorithm 1. Once the SMT has been computed, deciding the configuration of the SNs in terms of positions and frequencies is straightforward. For the first issue (i.e. positioning), SNs will be placed at each Steiner point $z \in P_s \setminus V_s$, and on each edge $u(v_i, v_j) \in D_s$ at distance R one from each other, such that multi-hop connectivity is provided among the vertexes v_i and v_j. For the frequency allocation on each SN, we observe that by construction the maximum degree of a Steiner point is three [13][14], which thus justifies our assumption of N_{sdr}=4. Given the fact that one radio is used to communicate with other SNs on the CCC, the remaining SDR interfaces will be tuned to the corresponding frequencies used by the gateways to which the SN is (eventually) attached.

Unfortunately, the SMT computation is known to be an NP-hard problem, and in the `OPTIMAL-MVC` algorithm we have to calculate it for any possible subset of vertexes of G, i.e. $2^{|V|}$ times. Given the unaffordable computational cost of

Algorithm 1 The `OPTIMAL-MVC` algorithm

```
Initialize the optimal tree $T_{MVC}$={}.
Initialize the optimal weight $w_{opt}$=0.
for $G_i \in G$ do
  Determine Steiner-tree $SMT(G_i)$
  if $(C(SMT(G_i)) \leq M)$ and $(W(SMT(G_i)) \geq w_{opt})$ then
    Set $w_{opt}=W(SMT(G_i))$.
    Set $T_{MVC} = SMT(G_i)$.
  end if
end for
Return $T_{MVC}$
```

the `OPTIMAL-MVC` algorithm, we investigate approximated solutions to the MVC problem that however might provide an upper bound to the approximation ratio from the optimal. To this aim, we introduce the notion of full components (or, for short, components) of a MST, defined as a maximal subtree whose terminals coincide with its leave. A k-restricted MST is then defined as a Steiner tree whose components contain no more than k terminals (k-component). For $k \geq 4$, computing the k-MST(G) is a NP-hard problem too. However, for $k = 3$, the problem admits polynomial solution through the Melzak algorithm [14], and it is possible to prove that the approximation ratio ρ provided by the k-MST solution is at most equal to $\frac{5}{3}$ of the optimum [13], i.e.:

$$\rho = \frac{C(k\text{-MST(G)})}{C(\text{MST(G)})} \leq \frac{5}{3} \tag{6}$$

Based on this result, we propose the `3-MVC` algorithm to provide an approximated optimal solution to the MVC problem. The `3-MVC` works in two Steps as shown by the pseudo-code of Algorithm 2. The first Step manipulates the original graph G by computing the Steiner point z for each triple of vertexes $v_i \in V$, $v_j \in V$, $v_k \in V$ of G, adding z to set of vertexes[1], and then adding the corresponding edges $u(v_i, z)$, $u(v_j, z)$, and $u(v_k, z)$ to the set of edges. As a result, a new graph $G'(V', E')$ is built at the end of Step 1, as shown in Figure 2(b). Then the MVC is computed through the `MVCSearch` algorithm (detailed below) on the new graph G'. Briefly, the `MVCSearch` algorithm computes for each vertex $v' \in V'$ the best tree rooted in v' having a total cost less or equal to M. Among the candidate solutions, the tree with maximum weight is selected and returned by the `MVCSearch` algorithm. Figure 2(c) shows the tree structure built by the algorithm. The correctness of the `MVCSearch` algorithm is proved later in this Section.

The pseudo-code of the `MVCSearch` procedure is shown in Algorithm 3. Here, a T_{MVC} global structure is used to keep track of the best solution found so far, and then returned at the end of the algorithm (line 6). The `MVCSearch` works by invoking the `FindBestTree` procedure on each vertex of G, which explores all possible subtrees T_v rooted at v with $C(T_v) \leq M$, and saves the one with

[1]We assume that $z \neq v_i$, $z \neq v_j$ and $z \neq v_z$, otherwise no adding is done to V' and E'.

Algorithm 2 The 3-MVC algorithm

```
Given a graph $G$={$V, E, l, w, c$} and $M$:
Initialize: $V'$=$V$ and $E'$=$E$.
**Step1**: Build G':
**for** each ($v_i$, $v_j$, $v_k$) triples, with $v_i \in V$, $v_j \in V$, $v_k \in V$ **do**
    Compute Steiner point $z$ through the Melzak algorithm [14].
    Set $V'$=$V' \cup \{z\}$.
    Set $E'$=$E' \cup \{u(z, v_i)\} \cup \{u(z, v_j)\} \cup \{u(z, v_z)\}$.
**end for**
**Step2**: Compute the approximated MVC:
    $T_{3-MVC}$= MVCSearch(G,M).
    Return $T_{3-MVC}$.
```

maximum weight into T_{MVC}. Each iteration of `FindBestTree` keeps track of the current vertex v_{cur} visited by the algorithm, of the previous node v_{prev}, of the *frontier* F, defined as the set of vertexes not yet visited by the algorithm, and of the current tree T_v. At each iteration, the frontier F is increased by including the neighbours of v (denoted as $Neigh_v$ at line 13), the cost $C(T_v)$ is decreased of $c(u(v_{prev}, v_{cur}))$ (i.e. of the number of SNs needed to connect the two gateways) at line 15, and then the algorithm is recursively invoked on the new frontier and on the graph $G \setminus \{v_{cur}\}$ at line 16. When $C(T_v) > M$, i.e. no more gateways can be connected by SNs, the optimality the tree is checked at line 21 by comparing its weight $W(T_v)$ with the best value found so far (i.e. $W(T_{MVC})$). In case $W(T_{MVC})$==$W(T_v)$, the solution with lower cost (i.e. requiring less SNs) is preferred.

In the following, we provide a bound to the approximation ratio provided by the `3-MVC` algorithm.

Theorem 1. *Correctness of `MVCSearch`. Given a graph G={V, E, l, w, c}, and $G' = \{V', E', l, w, c\}$ the corresponding graph generated from G after Step1 of the `3-MVC` algorithm. Let $T(V_t, E_t)$ be the set of trees with $V_t \subseteq V$ and $E_t \subseteq E$, and T_{3-MVC} the tree returned by the `MVCSearch` algorithm. Then $C(T_{3-MVC}) \leq M$ and $W(T_{3-MVC})$=$max\{W(T'), \forall T' \in T(V_t, E_t)\}$.*

Proof. By construction the graph built on Step 1 of the Algorithm 2 contains all the possible *3 full components* in the graph. Since the algorithm checks all the possible path starting from each vertex in G having cost less then or equal to M, then it will also encounter the tree T' with $max\{W(T')\}$, and thus it will set T_{3-MVC}=T' (at line 22). □

Theorem 2. *Optimality of `3-MVC`. Given a graph G={V, E, l, w, c} and a number of available SNs M. Let $T_{MVC}(P_{opt}, E_{opt})$ be the tree returned by the `OPTIMAL-MVC` (with terminal nodes $V_{opt} \subseteq P_{opt}$, $V_{opt} \subseteq V$) and T_{3-MVC} the tree produced by our `3-MVC` algorithm. If $C(T_{MVC}) \leq \frac{3}{5} \cdot M$, then $C(T_{3-MVC}) \leq M$ and $W(T_{3-MVC}) = W(T_{MVC})$.*

Algorithm 3 The `MVCSearch` algorithm

```
1: Initialize: $T_{MVC}$=$\{\}$.
2: for $v \in V$ do
3:     Initialize $T_v$=$\{\}$.
4:     Invoke FindBestTree($\{v\}$,$G$,$M$,$T_v$,NULL).
5: end for
6: Return the tree $T_{MVC}$.
7:
8: procedure FindBestTree($F$,$G$,$b$,$T_v$,$v_{prev}$)
9: Initialize $checkTree$=true.
10: for $v_{cur} \in$ F do
11:     if $(c(u(v_{cur}, v_{prev})) \leq b)$ and $(u(v_{cur}, v_{prev})) \in E'$ then
12:         Set $checkTree$ = false.
13:         Update the frontier $F$=$(F \cup Neigh_v)\setminus \{v_{cur}\}$.
14:         Add vertex $v$ and edge $u(v_{cur}, v_{prev})$ to $T_v$.
15:         Update current cost $b'$=$b - c(u(v_{cur}, v_{prev}))$.
16:         FindBestTree($F$,$G \setminus \{v_{cur}\}$,$b'$,$T_v$, $v_{cur}$).
17:         Remove vertex $v$ and edge $u(v_{cur}, v_{prev})$ to $T_v$.
18:     end if
19: end for
20: if $checkTree$ == true then
21:     if $(W(T_{MVC}) < W(T_v))$ or $((W(T_{MVC}) == W(T_v))$ and $(C(T_{MVC}) >$
        $C(T_v)))$ then
22:         Set the new optimal solution $T_{MVC} = T_v$.
23:     end if
24: end if
```

Proof. Let $T^*_{3-MVC}(P^*, D^*)$ (with the terminal nodes $V^* \subseteq P^*$) be the $3-MST$ tree having $V^* = V_{opt}$ (i.e. having the same set of vertexes of T_{MVC}). By Equation 6 we have that $C(T^*_{3-MVC}) \leq \frac{5}{3} \cdot C(T_{MVC})$ and thus $C(T^*_{3-MVC}) \leq M$. Theorem 1 enunciates that the `MVCsearch` algorithm determines the T_{3-MVC} tree that maximizes $W(T_{3-MVC})$ with $C(T_{3-MVC}) \leq M$. Since T^*_{3-MVC} is a subset of the graph built at Step 1 of the algorithm and that there exists no tree having a weight greater then $W(T^*_{MVC})$ with cost less then or equal to M, then the tree T_{3-MVC} found by the algorithm is exactly T^*_{3-MVC}. Given that $V^* = V_{opt}$, then $W(T_{3-MVC}) = W(T_{MVC})$. □

We now analyze the Computational Complexity (CC) of the `3-MVC` algorithm. To this aim, we introduce a parameter η as the mean of the cost values $c(u(v_i, v_j))$ of the graph G, and we set $n = |V|$. It is easy to see that Step 1 requires to compute the Steiner point for each triples of vertexes of G, which can be computed in $O(n^3)$. For Step 2, the `MVCSearch` algorithm (abbreviated as f in the following) is invoked recursively on each node of the frontier, which has cardinality $|V| = n$, i.e.:

$$CC(f(M, n)) = O(n) \cdot CC(f(M - \eta, n - 1)) \tag{7}$$

In the previous Equation, the number of factors is equal to $\frac{M}{\eta}$, i.e. the algorithm ends when no more SNs can be utilized to reach the vertexes of G. Thus, we have that:

$$CC(f(M, n)) = \prod_{i=0}^{\frac{M}{\eta}} (n - i) \sim O(n^{\frac{M}{\eta}}) \tag{8}$$

Combining Step 1 and Step 2, we can conclude that the total complexity of the `3-MVC` algorithm is in the order of $O(n^3) + O(n^{\frac{M}{\eta}})$. For scenarios with low values of SNs (i.e. M) or acceptable values of the ratio $\frac{M}{\eta}$ (e.g. large-scale scenarios), the `3-MVC` algorithm requires affordable resources, and produces an effective improvement over the `OPTIMAL-MVC` algorithm in terms of computational costs.

5 Distributed Algorithm

In [5], we described a distributed algorithm through which SNs can self-organize into a distributed mesh structure (called STEM-Mesh) that connect isolated gateway. The algorithm does not require a centralized controller, and relies only on local interactions among the SNs. To this aim, we assume that each SN (e.g. SN_i) will periodically broadcast a `BEACON` message on the CCC containing information about the current position and the radio configuration (e.g. the last channel sensed). Figure 3 shows a screenshot of the SN mesh in an Omnet++ simulation. The algorithm that enables the mesh formation and maintenance is based on the Virtual Spring Mesh scheme defined in [7][8]. Here, virtual spring forces act among mobile robots according to a natural length l_0 and a stiffness constant k. In our study, we extended [8] by considering three types of virtual spring forces (defined as ST, spring type) that can act on each SN_i:

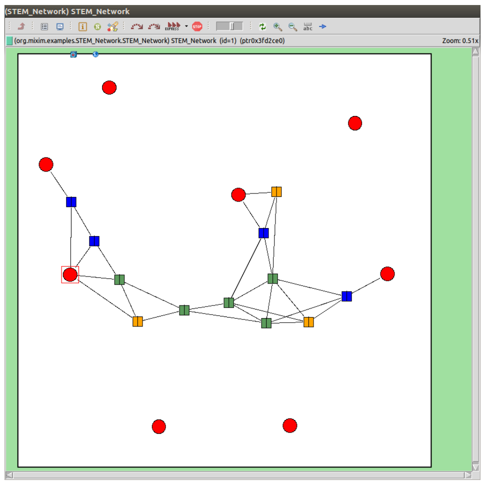

(STEM_Network) STEM_Network
(org.mixim.examples.STEM_Network.STEM_Network) STEM_Network (id=1) (ptr0x3fd2ce0)
Zoom: 0.51x

Figure 3: The screen-shot of the STEM-Mesh structure: gateways are represented through circles, SNs through rectangles. Scout SNs are in orange.

- *Mesh-to-Mesh* Spring (ST_0). The end-points of the spring are SN_i and any another neighbour SN_j, so that connectivity is preserved among the devices, while a minimum link quality is guaranteed for each link, through the Link-Budget (LB) metric introduced below.
- *Mesh-to-Island* Spring (ST_1). The end-points of the spring are SN_i and a gateway of island j (e.g. G_j) sensed in its range, so that the island is connected to the mesh.
- *Mesh-to-Frontier* Spring (ST_2). The end-points of the spring are SN_i and a point of the mesh frontier. This force is introduced to enable space exploration by special SNs (called *scout* SNs).

Each virtual spring force $\vec{F_i^j}$ applying on SN_i is modelled according to the well-known Hooke's law:

$$\vec{F_i^j} = -k_j^{ST} \cdot \delta \cdot \vec{u} \tag{9}$$

Here, $\vec{u}$ is the unit vector between SN_i and the other end-point of the spring, δ is the displacement, i.e. the difference between the actual spring length and the natural spring length l_0, and k^{ST} the stiffness of the spring, based on its type. We consider fixed values of k^{ST} for Mesh-To-Mesh (k^{ST_0}) and Mesh-to-Frontier (k^{ST_2}) spring types. In the case of Mesh-to-Island Springs, we make k^{ST_1} parametric and proportional to the cardinality of the island j, i.e. n_j. As a result of this choice, the SNs will be more attracted by gateways of islands composed by a larger number of nodes, in accordance with the goal function defined by Equation 2. Instead of measuring the displacement in terms of spatial distance among the end-points (as in [7][8]), in this paper we propose a formulation of δ

based on the link quality of the communication between SN_i and SN_j or G_j. To this aim, we assume that, once receiving a `BEACON` message from SN_j or an `HELLO` message from G_j, SN_i will compute the Link Budget of the link $i \leftrightarrow j$ (i.e. $LB(i,j)$) as follows:

$$LB(i,j) = Pr^i - RS^i_{thr} \tag{10}$$

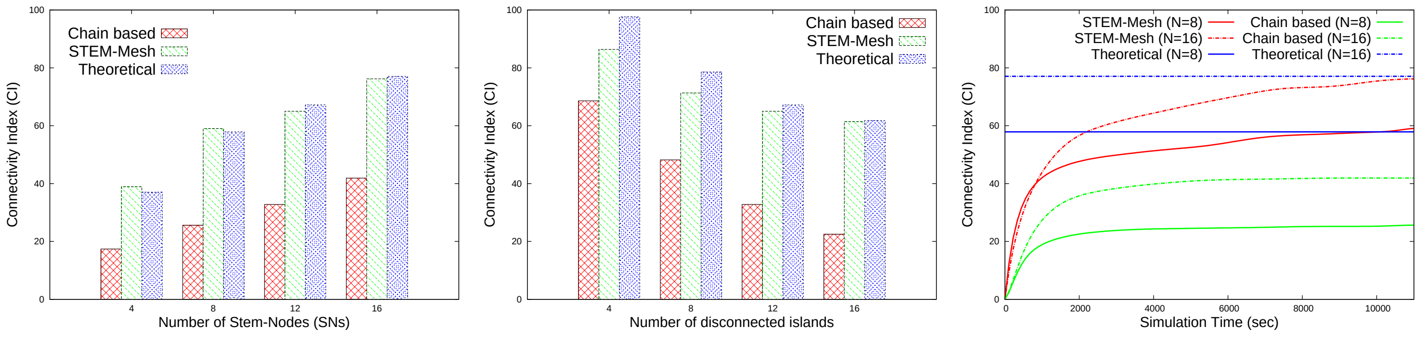
where Pr^i is the receiving power at SN_i, and RS^i_{thr} its receiving sensitivity. In telecommunications, the LB metric reflects the stability of a link in terms of fading margin, based on the propagation conditions of the environment [19]. In this paper, we are interested in creating the STEM-Mesh so that a minimum link-quality (which directly translates into on achievable data-rate, as discussed in [18]) is guaranteed among each couple of SNs. To this purpose, we introduce a requested Link-Budget value (LB_{req}) and we formulate the displacement δ in terms of distance between the actual value of $LB(i,j)$ and LB_{req}, as follows:

$$\delta = \sqrt[\alpha]{\frac{max(LB(i,j), LB_{req})}{min(LB(i,j), LB_{req})}} - 1 \tag{11}$$

Here, α is the propagation decay exponent (equal to 2 in our scenario). At periodic time intervals, each SN_i computes the virtual spring forces $\vec{F_i^0}$, $\vec{F_i^1}$, ... $\vec{F_i^n}$ acting on it, and determines the resultant force $\vec{R_i} = \sum_{k=0}^{n} \vec{F_i^k}$. Then, it moves in the direction given by $\vec{R_i}$ with constant acceleration till maximum speed v_{MAX} is reached (equal to 3 m/s in our experiments). In [5] we also described additional mechanisms to favour convergence of the algorithm and to reduce the risks of ping-pong effects caused by nodes that continuously change their locations. For space reasons, we omit further details on the mesh creation algorithm.

In order to detect the presence of islands, SNs must also explore the scenario in both spatial and frequency domain. The first task (i.e. spatial exploration) is delegated to special SNs, called scouts, probabilistically chosen among the nodes of the mesh frontier. To this aim, we defined the visibility zone of SN_i with respect to vector direction $\vec{d}$ as the cone centered at SN_i with a sweep angle of θ. Scout SNs are probabilistically chosen among the SNs that do not have any other node in their visibility range. On each scout node, an additional spring force is applied with direction $\vec{d}$, and constant values of δ. For what regards frequency exploration, each SN periodically senses the spectrum in order to detect an `HELLO` message from islands' gateways. More specifically, each SN_i keeps statistics on the number of sensing actions performed on each frequency h, considering also neighbours' sensing actions (this information is communicated into the `BEACON` message). Let $SE^i(h)$ be the current statistic for frequency h and SN_i. At each T_B interval, SN_i senses the frequency h', which is the least sensed (and thus mostly unexplored) channel based on its history, i.e.:

$$h' = argmin_{\forall h \in N_{freq}} SE^i(h) \tag{12}$$



(a) (b) (c)

Figure 4: The CI for the three schemes as a function of the number of SNs (4(a)), of the number of the islands (4(b)) and of the simulated time (4(c)).

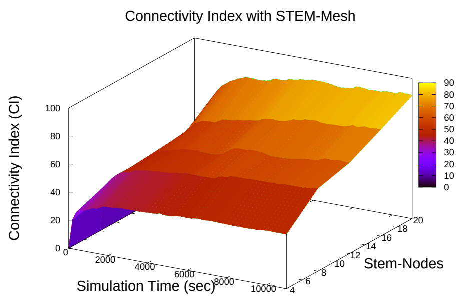
Through Equation 12, we ensure that sensing activity is fairly shared among all the SNs, and at the same time we guarantee that all frequencies are explored equally.

6 Performance Evaluation

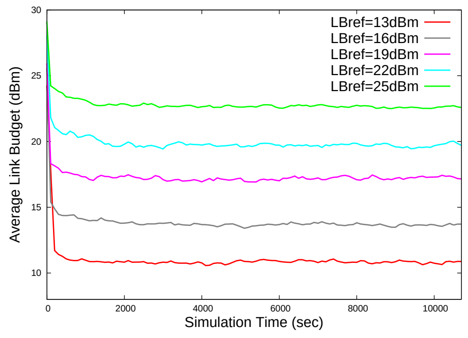
In this Section, we compare the performance of the distributed STEM-Mesh scheme with those of the centralized `3-MVC` algorithm, which provides an approximated optimal solution as discussed in Section 4. Moreover, we also evaluate the chain-based scheme proposed in [9], through which mobile robots self-organize into backbone structures, like the STEM-Mesh. In our implementation, the algorithm presented in [9] has been slightly adapted to our environment, i.e. we replace the mobile robots with the SNs, and we create dynamic chain structures whose head and tail are the gateways to be connected. Frequency allocation is performed as in the STEM-Mesh scheme. Using Omnet++ as simulation tool, we model an area of 1.5x1.5Km^2, with $N = 100$ end-user devices uniformly distributed over a varying number of islands, operating over four frequencies (i.e. N_{freq}=4). Gateway positions, frequency used and number of nodes in each island are decided randomly. In Section 6.1, we investigate the ability of the three schemes to repair the original network infrastructure, and to connect the fragmented islands. In Section 6.2, we further study the convergence of the STEM-Mesh scheme over time.

6.1 Connectivity Analysis

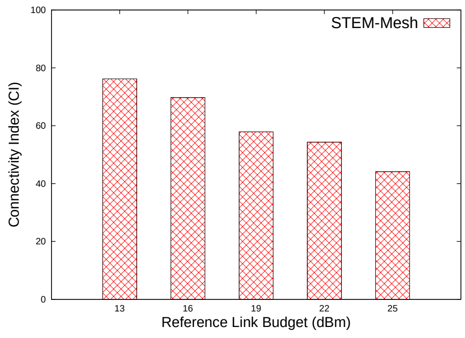
As main performance metric, we consider the Connectivity Index (CI), expressed as the ratio of the end-user devices connected through SNs (given by $|S|$ in Equation 2), over the total number of devices N. It is easy to see that the CI metric depends on the policy used to deploy the SNs mesh in terms of loca-



(a)



(b)



(c)

Figure 5: The CI metric of STEM-Mesh as a function of the number of SNs and of the simulated time is shown in Figure 5(a), of the requested LB_{req} value in Figure 5(c)). The average LB value over simulation time is shown in Figure 5(b).

tion/frequencies. Figure 4(a) shows the CI for a network scenario partitioned into 12 islands, as a function of the number of available SNs (i.e. M). The STEM-Mesh scheme and the `3-MVC` algorithm provide higher performance than the chain-based approach, mainly due to the ability of providing better coverage of the environment. Moreover, Figure 4(a) reveals that the performance of STEM-Mesh are comparable, and in some configurations slightly better than the `3-MVC` algorithm, which means that in some cases the distributed algorithm provides better approximation of the optimal SMT than Equation 6. Figure 4(b) confirms this analysis, by showing the CI as a function of the number of islands available in the scenario. We consider in this case a configuration with M=16 SNs. For all the evaluated schemes, the CI decreases with the number of islands. However, the STEM-Mesh scheme overcomes the chain-based approach for all the M values, and provide similar performance than the centralized `3-MVC` algorithm for $M \geq 12$. For $M < 12$, the STEM-Mesh scheme is below the approximated optimal solution. We found that this effect is caused by the exploration phase, which in STEM-Mesh is performed by scouts SNs, chosen with a fixed probability ϵ (equal to to 0.1 in our experiments). However, for configurations with few SNs, it is likely that there might be explored areas due to the fact that all the available SNs are already placed, and no more scouting activity is performed. To solve this issue, adaptive exploration strategies will be considered as future work. Finally, in Figure 4(c) we show the CI value over simulation time. We consider a scenario with 12 islands, and two configurations of M (i.e. 8 and 16 SNs), while the horizontal lines denote the CI values returned by `3-MVC` algorithm. This result confirms the self-organization capabilities of the STEM-Mesh algorithm, reflected by the fact that the CI value dynamically increases during the exploration phase, till the approximated optimal value is reached after 8000 (for M=8) and 10000 seconds (for M=16).

6.2 Convergence Analysis

In this Section, we analyze the converge of the STEM-Mesh algorithm. More specifically, in Figure 5(a) we depict the CI value (on the z-axis), as a function of the simulation time (on the x axis) and of the number of SNs (on the y axis). We consider a configuration with 12 islands. The 3D curve reveals two properties of STEM-Mesh: (i) the CI metric increases over time till a converge point is reached, which thus reflects the fact that SNs will dynamically explore the environment and fix their locations once a local maximum is reached, and (ii) the CI metric increases with the number of SNs, thus demonstrating the ability of STEM-Mesh to maximally exploit the available resources. Figure 5(a) and 5(b) investigate the average link quality of the connections created among the SNs. As explained in Section 5, in STEM-Mesh the SNs adjust their relative locations in order to guarantee a requested Link Budget threshold (LB_{req}) on each link. In Figure 5(a), we depict the average LB value as a function of the simulation time, for M=16, and different LB_{req} values. After an initial setup phase (the SNs are injected in the scenario from the same starting point), the SNs dynamically explore the environment and enlarge the mesh, causing a brief decrease of the average LB value. Once the requested LB_{req} value is reached, the SNs do not modify their relative positions, as shown by Figure 5(a). This result demonstrates the ability of STEM-Mesh to meet the QoS requirements, expressed by the LB_{req} value. Finally, Figure 5(c) shows the CI metric as a function of the LB_{req} value, over the same configuration of Figure 5(b). Intuitively, increasing the requested LB_{req} translates into a reduction of the relative distance among the SNs, and thus in smaller coverage of the STEM-Mesh. This produces a decreasing trend of the CI metric, as correctly shown by Figure 5(c).

7 Conclusions and Future Works

In this paper, we have addressed the issue of re-establishing the connectivity among heterogeneous network partitions in post-disaster scenarios. To this aim, we have proposed the utilization of specialized repairing units, called Stem Nodes (SNs), provided with self-positioning, cognitive radio and routing functionalities. We have formulated the problem of determining the optimal deployment of the SNs in terms of location/frequencies, so that the number of end-user devices interconnected through the SNs mesh is maximized. We have determined the optimal SNs deployment through a graph-theory formulation, provided an approximated solution with theoretical upper-bounds, and compared with a distributed algorithm (called STEM-Mesh) through which the SNs self-organize into mesh structures according to a virtual-springs based approach. Future works include: the extension of the theoretical analysis for $N_{freq} > 4$, and experimental validation of STEM-Mesh on small-scale network deployments.

Acknowledgment

This work is supported by Italian MIUR funds under the project PRIN-2009 STEM-Net: "STEM" devices for self-organizing wireless NETworks.

References

[1] I. F. Akyildiz, W. Y. Lee, M. C. Vuran, and S. Mohanty. NeXt Generation/Dynamic Spectrum Access/Cognitive Radio Wireless Networks: A Survey. *Computer Networks Journal*, 50(1), pp. 2127- 2159, 2006.

[2] G. P. Villardi, G. De Abreu and H. Harada. TV white space technology. *IEEE Vehicular Technology Magazine*: 7(2), pp. 47-53, 2012.

[3] R. Ferrus, O. Sallent, G. Baldini and L. Goratti. Public safety communications. *IEEE Vehicular Technology Magazine*: 7(2), pp. 54-61, 2012.

[4] Y. Takeuchi. Radio Policy in Japan. Ministry of Internal Affairs and Communications, Tokyo, Japan, IEEE 802.15-11-0674-00-0000, 2011.

[5] M. Di Felice, A. Trotta, L. Bedogni, L. Bononi, F. Panzieri, G. Ruggeri, V. Loscrı', P. Pace, STEM-Mesh: Self-Organizing Mobile Cognitive Radio Network for Disaster Recovery Operation. accepted for publication in *Proc. of IEEE IWCMC*, Cagliari, Italy 2013.
Tech Report: `http://www.cs.unibo.it/~difelice/iwcmc13.pdf`

[6] Y. Zou and K. Chakrabarty. Sensor deployment and target localization in distributed sensor networks. *ACM Transactions on Embedded Computer Systems*: 3(1), 61-91, 2004.

[7] B. Shucker and J. Bennet. Target tracking with distributed robotic macrosensors. *Proc. of IEEE MILCOM*, Atlantic City, USA, 2005.

[8] K. Derr and M. Manic. Extended virtual spring mesh (EVSM): The distributed self-organizing mobile ad hoc network for area exploration. *IEEE Transactions on Industrial Electronics*: 58(12), pp. 5424-5437, 2011.

[9] S. Nouyan and M. Dorigo. Chain Based Path Formation in Swarms of Robots. *Springer Lecture Notes in Computer Science Volume*: 4150(1), pp. 120-131, 2006.

[10] P. Pawelczak, R. V. Prasad, L. Xia and I. G. M. M. Niemegeers. Cognitive Radio Emergency Networks - Requirements and Design. *Proc. of IEEE DYSPAN 2005*, Baltimore, USA, 2005.

[11] W. Wang, W. Gao, X. Bai, T. Peng, G. Chuai and W. Wang. A framework of wireless emergency communications based on relaying and cognitive radio. *Proc. of IEEE PIMRC*, Athens, Greece, 2007.

[12] A. Gorcin and H. Arslan. Public safety and emergency case communications: opportunities from the aspects of cognitive radio. *Proc. of IEEE DYSPAN*, Chicago, USA, 2008.

[13] A. Borchers and D.-Z. Du. The k-Steiner Ratio in Graphs. *Proc. of ACM STOC*, Las Vegas, USA, 1995.

[14] Z. A. Melzak. On the problem of Steiner. *Canadian Mathematical Bullettin*, 1961, pp 143-148.

[15] J. Byrka, F. Grandoni, T. Rothvoß and L. Sanità. An Improved LP-Based Approximation for Steiner-Tree. *Proc. of ACM STOC*, Cambridge, USA, 2010.

[16] G. Robins and A. Zelikovsky. Tighter bounds for graph Steiner tree approximation. *SIAM Journal on Discrete Mathematics*: 19(1), pp. 122-134, 2005.

[17] T. A. Ramrekha and C. Politis. A Hybrid Adaptive Routing Protocol for Extreme Emergency Ad Hoc Communication. *Proc. of IEEE ICCCN*, Zurich, Switzerland, 2010.

[18] M. Di Felice, L. Bedogni and L. Bononi. Dynamic Backbone for Fast Information Delivery in Vehicular Ad Hoc Networks: An Evaluation Study. *Proc. of ACM PEWASUN*, Miami Beach, USA, 2011.

[19] T. S. Rappaport. Wireless communications: Principles and Practice. Prentice Hall Press, 2002.