# Multiagent System for Channel Selection and Topology Control on 802.11 based Mesh Networks

**Ante Prodan Vinod Mirchandani John Debenham**
**Faculty of IT, University of Technology, Sydney**
**NSW 2007, Australia**
{**aprodan, vinodm, debenham.** }**@it.uts.edu.au**

**Keywords:** self-organisation, multiagent systems, wireless networks

## Abstract

To address the problems of topology control and channel assignment on a multi-radio mesh networks a distributed, light-weight, co-operative multiagent system that guarantees scalability has been developed and validated by simulation. Our overall goal is twofold, to select channels so to reduce interference and improve connectivity by shortening paths between portal and clients nodes. As this system is to be deployed over large networks the scalability and stability of the solution are of the main concern. The algorithms developed have been implemented and evaluated with the help of NetLogo, a multiagent simulation tool. The attributes of the developed algorithms are demonstrated through the comprehensive simulation result analysis.

## 1. INTRODUCTION

The work discussed is based on previous work in the area of mesh networking and in particular in distributed algorithms at Columbia University, Microsoft Research, University of Maryland and Georgia Institute of Technology. In particular: [1], [2], [3] and [4].

Recent work on 802.11 Mesh Networks, such as [5], is predicated on a network whose prime purpose is to route traffic to and from nodes connected to the wired network — in which case there is assumed to be no traffic between end-user nodes. This introduces the conceptual simplification that mesh nodes can be seen as being grouped into clusters around a wired node where each cluster has a tree-like structure, rooted at a wired node, that supports the traffic. This is the prime purpose of 802.11 Mesh Networks in practice. In the work that follow we have, where possible, moved away from any assumptions concerning tree-like structures with the aim of designing algorithms for quite general mesh networks. Our methods have, where possible, been designed for the more general classes of "wireless ad-hoc networks" or "wireless mesh networks".

There are three principal inputs to this work that we assume are available to the proposed methods:

- A load model. Given any contiguous set of nodes in a mesh, the *load model* specifies the actual or desired level of traffic flowing into, or out of, nodes in that set.
- A load balancing algorithm. Given any contiguous set of nodes in a mesh and the load model for that set, the *load balancing algorithm* determines how the traffic is allocated to links in the mesh so as to reach its desired destination where it leaves the mesh.
- An interference model. Given any contiguous set of nodes in a mesh, the *interference model* stipulates the interference level that each node in the mesh gives to the other nodes in the mesh given a known level of background interference due to transmission devices that are external to the mesh.

The work described below makes no restrictions on these three inputs other than that they are available to every node in the mesh. The load model, and so too the load balancing algorithm, will only be of value to a method for self-organisation if together they enable future load to be predicted with some certainty. We assume that the load is predictable.

In Section 2. we introduce some terms, concepts and notation. Section 3. describes the illocutions that make up the communication language used by the light-weight co-operative multiagent system that achieves self-organisation. Section 4. describes the role of the load balancing algorithm that our methods take as a given input. The measurement of interference cost is discussed in Section 5.. Methods for the adjusting the channels in a multi-radio mesh networks for predictable load are described in Section 6., and for adjusting the links in Section 7.. Future plans are described in Section 8..

## 2. BASIC TERMS AND CONCEPTS

The discrete time intervals mentioned below, e.g. $t$, $t+1$, are sufficiently spaced to permit what has to be done to be done.

Available channels: $1,\ldots,K$.

A *node* is a set of radio interfaces (or "antennae") where each *interface* is associated with a particular *channel*, together with a controller that (intelligently we hope) assigns the channel on each interface. Interfaces that are part of the same node are assumed to be 'close' topologically, but this is

not important. We assume for simplicity that each interface has its own, independent MAC layer.

A *link* is a pair of interfaces where each interface is assigned the same channel. The idea is that two interfaces communicate through a shared link. That is, if an interface is part of a link its state will be "listening and transmitting", otherwise its state will be "listening only".

Notation: nodes are denoted by Latin letters: $a$, $b$, $c$,..., the interfaces for node $a$ are denoted by: $a[i]$ for $i = 1,\ldots$, and links are denoted by Greek letters: $\alpha$, $\beta$, $\gamma$,.... The interfaces communicate using an illocutionary communication language that is defined informally (for the time being) with illocutions being encapsulated in quotation marks: "·".

For any node $n$, $S_n$ is the set of nodes in node $n$'s interference range. Likewise, for any link $\alpha$, $S_\alpha$ is the set of links that contain nodes $n$'s interference range $\forall n \in \alpha$.

Given a node $a$, define $V_a = \cup_{n \in S_a} S_n$.

$\Gamma_x^t$ is channel used by $x$ to communicate at time $t$ where $x$ may be either an interface or a link.

$f(\cdot,\cdot)$ is an *interference cost function* that is defined between two interfaces or two links. It estimates the cost of interference to one interface caused by transmission from the other interface. This function relies on estimates of the interference level and the level of load (i.e.: traffic volume). So this function requires an *interference model* and a *load model*. This function is described in Section 5..

An interface is either 'locked' or 'unlocked'. A locked interface is either locked because it has committed to lock itself for a period of time on request from another interface, or it is 'self-locked' because it has recently instigated one of the self-organisation procedures in Section 6.. A locked interface is only locked for a 'very short' period during the operation of each of those procedures. This is simply to ensure that no more than one alteration is made during any one period — this is necessary to ensure the stability of the procedures. We also say that a node is locked meaning that all the interfaces at that node are locked.

The abbreviation SNIR means "signal to noise plus interference ratio".

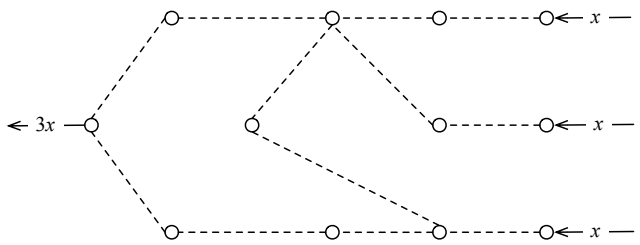
802.11 related terms: BSS — the basic service set. Portal — is the logical point at which MSDUs from an integrated non-IEEE 802.11 LAN enter the IEEE 802.11 DS (distribution system). WM — Wireless Medium. IBSS — Independent Basic Service Set. MSDU — MAC Service Data Unit.

## 3. THE COMMUNICATION LANGUAGE

Multiagent systems communicate in illocutionary languages. The simple language defined here will in practice be encoded as a small block in a packet's payload.

- "**propose organise**$[a,b,p]$" sent from interface $a$ to interface $b \in V_a$, where $V_a$ is as above. This message advises interface $b$ that interface $a$ intends to instigate the proactive logic with priority $p$.
- "**overrule organise**$[a,b,q]$" sent from interface $b$ to interface $a$. This message advises interface $a$ that interface $b$ intends to issue a **propose organise** statement as it has priority $q > p$. That is an interface can only overrule a request to organise if it has higher priority.

**Figure 1.** The load balancing algorithm determines the allocation of load.




The following three illocutions refer to interfaces being "locked" — this is simply a device to prevent interfaces from adjusting their settings when interference measurements are being made. See Section **??**.

- "**propose lock**$[a,b,s,t]$" sent from interface $a$ to interface $b$ requests that interface $b$ enter the locked state for the period of time $[s,t]$.
- "**accept lock**$[a,b,s,t]$" sent from interface $b$ to interface $a$ commits to interface $b$ entering the locked state for the period of time $[s,t]$.
- "**reject lock**$[a,b,s,t]$" sent from interface $b$ to interface $a$ informs interface $a$ that interface $b$ does not commit entering the locked state for the period of time $[s,t]$.

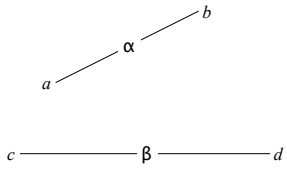
## 4. THE LOAD BALANCING ALGORITHM

We assume that if the external demands on a set of nodes $S$ are known and that there is a *load balancing algorithm* — that may or may not be intelligent — that determines how the load is routed through $S$. Figure 1 shows a set of twelve nodes connected by a mesh that is shown as dashed lines. The load on the mesh is shown by the four solid arrows. We assume that the load balancing algorithm will determine how the load is allocated to the links in the mesh.

## 5. MEASURING INTERFERENCE COST

Suppose that during some time interval $\Delta t$ two interfaces $a$ and $b$ are transmitting and receiving on channels $\Gamma_a$ and $\Gamma_b$. During $\Delta t$, the *interference limit* that interface $x$ imposes on interface $y$, $\tau_{y|x}$, is a ratio being the loss of traffic volume that

**Figure 2.** Definition of $f(\alpha \mid \beta)$.




interface $y$ could receive if interface $x$ were to transmit persistently divided by the volume of traffic that interface $y$ could receive if interface $x$ was silent:

$$\tau_{y|x} = \frac{(m_y \mid \text{ interface } x \text{ silent}) - (m_y \mid \text{ interface } x \text{ persistent})}{m_y \mid \text{ interface } x \text{ silent}}$$

where $m_y$ is the mean SNIR observed by interface $y$ whilst listening on channel $\Gamma_y$, where as many measurements are made as is expedient in the calculation of this mean[1]. The *interference load* of each interface, $v_a$ and $v_b$, is measured as a proportion, or percentage, of some time interval during which that interface is transmitting. Then the *observed interference* caused by interface $b$ transmitting on channel $\Gamma_b$ as experienced by interface $a$ listening on channel $\Gamma_a$ is: $\tau_{a|b} \times v_b$, and the *observed interference cost* to interface $a$ is[2]:

$$f(a \mid b) \triangleq \tau_{a|b} \times v_b \times (1 - v_a)$$

and so to interface $b$:

$$f(b \mid a) = \tau_{b|a} \times v_a \times (1 - v_b)$$

Now consider the interference between one interface $a$ and two other interfaces $c$ and $d$. Following the argument above, the *observed interference* caused by interfaces $c$ and $d$ as experienced by interface $a$ is[3]: $\tau_{a|c} \times v_c + \tau_{a|d} \times v_d - \tau_{a|\{c,d\}} \times v_c \times v_d$. The observed interference cost to interface $a$ is:

$$f(a \mid \{c,d\}) = (1 - v_a) \times \left(\tau_{a|c} \times v_c + \tau_{a|d} \times v_d - \tau_{a|\{c,d\}} \times v_c \times v_d\right)$$

If interfaces $c$ and $d$ are linked, as shown in Figure 2, then they will transmit on the same channel $\Gamma_\beta$, and we ignore the possibility of them both transmitting at the same time[4]. Further suppose that $v_\beta$ is the proportion of $\Delta t$ for which either

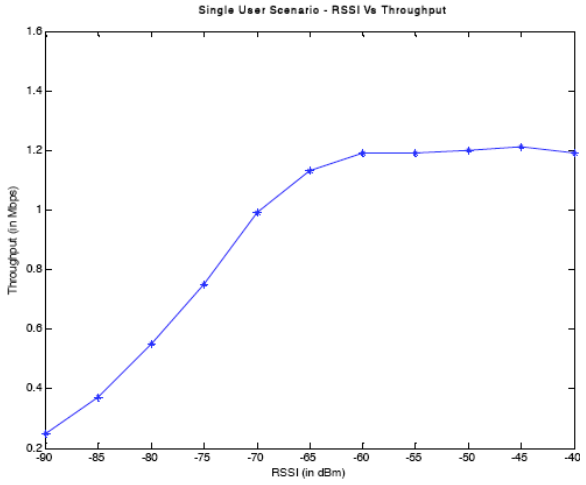
[1]For $\tau_{y|x}$ to have the desired meaning, $m_y$ should be a measurement of *link throughput*. However, link throughput and SNIR are approximately proportional as shown in Figure 3 — see [6].

[2]We assume here that whether or not interfaces $a$ and $b$ are transmitting are independent random events [7]. Then the probability that $a$ is transmitting at any moment is $v_a$, and the probability that $b$ is transmitting and $a$ is listening at any moment is: $(1 - v_a) \times v_b$.

[3]That is, the interference caused by either interface $c$ or interface $d$.

[4]The probability of two linked interfaces transmitting at the some time on an 802.11 mesh network can be as high as 7% — see [8], [9].

**Figure 3.** Relationship between SNIR and throughput.




interface $c$ or interface $d$ is transmitting. Then for some $\kappa_\beta$, $0 \leq \kappa_\beta \leq 1$: $v_c = \kappa_\beta \times v_\beta$, and $v_d = (1 - \kappa_\beta) \times v_\beta$. Thus:

$$f(a \mid \beta) = (1 - v_a) \times v_\beta \times \left(\tau_{a|c} \times \kappa_\beta + \tau_{a|d} \times (1 - \kappa_\beta)\right)$$

Now suppose that interfaces $a$ and $b$ are linked, and that $v_\alpha$ is the proportion of $\Delta t$ for which either interface $a$ or interface $b$ is transmitting. Then for some $\kappa_\alpha$, $0 \leq \kappa_\alpha \leq 1$: $v_a = \kappa_\alpha \times v_\alpha$, $v_b = (1 - \kappa_\alpha) \times v_\alpha$. Then as $a$ will only receive interference when it is listening to $b$ transmitting:

$$f(a \mid \beta) = v_b \times v_\beta \times \left(\tau_{a|c} \times \kappa_\beta + \tau_{a|d} \times (1 - \kappa_\beta)\right)$$

and so:

$$\begin{aligned} f(\alpha \mid \beta) = & \\ & (1 - \kappa_\alpha) \times v_\alpha \times v_\beta \times \left(\tau_{a|c} \times \kappa_\beta + \tau_{a|d} \times (1 - \kappa_\beta)\right) \\ & + \kappa_\alpha \times v_\alpha \times v_\beta \times \left(\tau_{b|c} \times \kappa_\beta + \tau_{b|d} \times (1 - \kappa_\beta)\right) \end{aligned} \quad (1)$$

Note that $v_\alpha$, $v_\beta$, $\kappa_\alpha$ and $\kappa_\beta$ are provided by the load model, and the $\tau_{x|y}$ are provided by the interference model.

## 6. ADJUSTING THE CHANNELS

Our solution is based on the distinction in multiagent systems between proactive and reactive reasoning. Proactive reasoning is concerned with planning to reach some goal. Reactive reasoning is concerned with dealing with unexpected changes in the agent's environment. So in the context of self-organising networks we distinguish between:

- a *reactive logic* that deals with problems as they occur. The aim of our reactive module is simply to restore communication to a workable level that may be substantially sub-optimal.

- a *proactive logic* that, when sections of the network are temporarily stable, attempts to adjust the settings on the network to improve performance.

The reactive logic provides an "immediate fix" to serious problems. The proactive logic, that involves deliberation and co-operation of nearby nodes, is a much slower process.

A node (i.e.: router) with omnidirectional interfaces has three parameters to set for each interface: [1] The channel that is assigned to that interface; [2] The interfaces that that interface is linked to, and [3] The power level of the interface's transmission. Methods are describe for these parameters in the following sections. The following section describes how these three methods used combined in the proactive logic algorithm. The following methods all assume that there is a load balancing algorithm (see Section 4.) and that it is common knowledge. The following methods are independent of the operation of the load balancing algorithm.

Informally the proactive logic uses the following procedure:

- *Elect* a node $a$ that will manage the process
- *Choose* a link $\alpha$ from $a$ to another node — precisely a trigger criterion (see below) permits node $a$ to attempt to improve the performance of one of its links $\alpha \ni a$ with a certain priority level.
- *Measure* the interference
- *Change* the channel setting if appropriate

The following is a development of the ideas in [1].

**choose** node $a$ **at** time $t-2$;
**set** $V_a = \cup_{n \in S_a} S_n$;
$\forall x \in V_a$ **transmit** "**propose organise**$[a,x,p]$";
**unless** $\exists x \in V_a$ **receive** "**overrule organise**$[a,x,q]$" in
$[t-2,t-1]$ where $q > p$ **do** {
  $\forall x \in V_a$ **transmit** "**propose lock**$[a,x,t,t+1]$";
  **if** $\forall x \in V_a$ **receive** "**accept lock**$[a,x,t,t+1]$" in $[t-1,t]$
  **then** {
    **unless** $\exists x \in V_a$ **receive** "**reject lock**$[a,x,t,t+1]$"
    **do** {improve $a$;}
  }
}
where: improve $a = \{$
  **choose** link $\alpha \ni a$ on channel $\Gamma^t_\alpha$;
  **set** $B \leftarrow \sum_{\beta \in S_\alpha} f(\alpha \mid \beta) + \sum_{\beta \in S_\alpha} f(\beta \mid \alpha)$;
  **if** (feasible) **re-route** $\alpha$'s traffic;
  **for** $\Gamma_\alpha = 1, \ldots, K, \Gamma_\alpha \neq \Gamma^t_\alpha$ **do**{
    **if** $\sum_{\beta \in S_\alpha} f(\alpha \mid \beta) + \sum_{\beta \in S_\alpha} f(\beta \mid \alpha) < B \times \varepsilon$ **then**{
      $\Gamma^{t+1}_\alpha \leftarrow \Gamma_\alpha$;
      **selflock** node $a$ **in** $[t+1, t+k]$;
      **break**;
    };
  };
  $\forall x \in V_a$ **transmit** "$\alpha$'s interference test signals";
  **apply** load balancing algorithm to $S_a$;
}

The statement **selflock** is to prevent $a$ from having to activate the method too frequently. The constant $\varepsilon < 1$ requires that the improvement be 'significant' both for node $a$ and for the set of nodes $S_a$. The stability of this procedure follows from the fact that it produces a net improvement of the interference cost within $S_a$. If a change of channel is effected then there will be no resulting change in interference outside $S_a$.

The above method reduces the net observed inference cost in the region $V_a$. It does so using values for the variables that appear on the right-hand side of Equation 1. If those values are fixed then the method will converge. The method above suggests the possibility that traffic is re-routed during the reassignment calculation — this is not essential.

**Interference model.** We assume that each node, $a$, knows the channel of every node in $V_a$. We assume that each node is capable of measuring the strength of signals from every node in $V_a$. So if each node had access to all of this information from the point of view of every node in $V_a$, and, perhaps the level of background noise around $V_a$ then $a$ can derive estimates for the $\tau_{x|y}$ factors for all $x$ and $y$ in $V_a$. In particular, $a$ will be able to estimate all these factors to evaluate Equation 1 as required by the above algorithm. *In addition*, the procedure above suggests that if node $a$ is involved in changing its channel then at the end of this process — time permitting — it should transmit a 'beep-silence-beep-silence' message to enable every other node in $V_a$ to observe the actual $\tau$ values. *Further*, it is reasonable to suggest that this transmission of test signals could be carried out periodically in any case when network load permits.

The initialisation process that we had used in 4.1, for our self-organisation algorithm to obtain its performance evaluation involved the construction of a spanning tree. The spanning tree was constructed from a root interface (mesh portal) that spans a designated area of the mesh network. The spanning tree's nodes are called seed nodes. The seed node in turn then builds a cluster of connected nodes around itself. Each seed node was sequentially selected along the spanning tree to cover most of the area in the wireless mesh region.

## 6.1. Results and Discussion

The interference cost reduction for a link discussed herein is measured as the difference between absolute interference ($AI$) values obtained before the channel assignment process and after the channel assignment process. For example, if $AI_{before} = 5$ and $AI_{after} = 4$ the absolute difference is $AD = 1$

which is 20% decrease in the absolute interference. Consequently, the performance is always expressed as a percentage of the decrease. Our simulation studies consider realistic scenarios of different node densities and topologies in a typical wireless mesh network hence are more reflective of evaluating the true performance of the algorithm. In these studies the mean of interference cost (*IC*) reduction across all topologies and network (node) densities obtained is 36.7.

**Impact of network (node) density on the performance.** As the density of network increases (i.e. an increase in the number of routers located within the same area) the *IC* reduction relatively decreases. This trend is shown across all the topologies.

We attribute this result to the limited number of non-overlapping channels available in IEEE 802.11b/g standard that in tight proximities of the nodes (i.e. increase in node densities) shows more effects of a higher absolute interference and thus a relatively lower interference cost (*IC*) reduction [10]. Furthermore, the impact of node density on the algorithm is relatively consistent for all topologies at the same router densities. From Figure 4 it can also be observed that the range of the interference reduction across the topologies at router densities of 35 routers and 100 routers is 1.55 and 1.58, respectively.

**Impact of typical topologies on the interference cost**. Figure 5 shows the variation in the interference cost reduction as a function of network topology across different node densities. It can be deduced that the impact of the topologies on the performance of the algorithm (i.e. in terms of interference cost reduction) is insignificant. The mean of *IC* reduction calculated from the data obtained shows that the topology with the smallest average *IC* reduction is the completely random with a mean of 36.02 and topology with the most *IC* reduction is the random grid with a mean of 37.12. The difference in performance between best and worst case is just 1.1 which confirms that the performance of the algorithm is almost completely independent of the type of topology.

**Performance bounds**. In addition to previously discussed results for the algorithm, we have calculated the 98% confidence bounds per link for absolute interference values across all topologies and different network densities. On comparison of the respective interference values the 98% confidence interval per link interference cost is smaller and tighter after selforganisation is invoked in contrast to before its invocation.

**Performance Comparison across the Network**. In this study, we obtained interference cost (*IC*) in different regions of the MR-WMN for the same set of links before and after the self-organisation algorithm is invoked. Comparison of the results obtained is shown in Figure 6 where the Interference cost is on the *X*-axis. From Figure 6 we can see that there were no nodes (red dots) that caused more interference after the self-organisation than it had caused before (blue dots) the

**Figure 4.** Interference cost reduction as a function of node density.

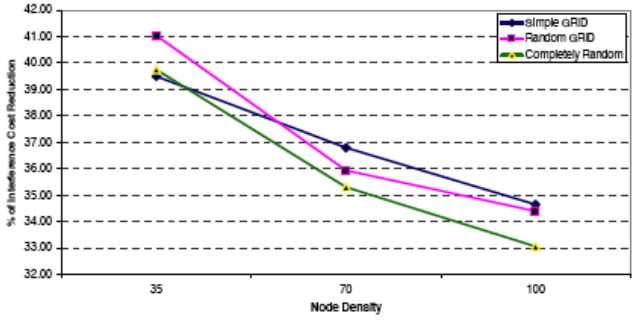
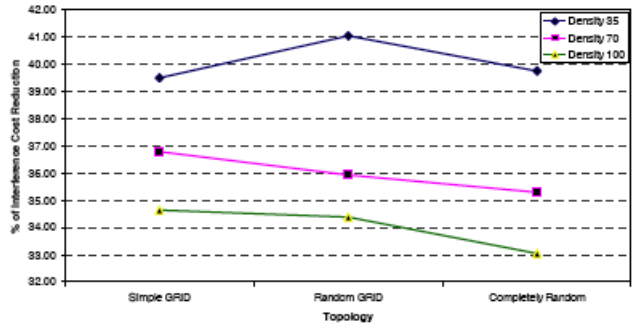



**Figure 5.** Interference cost reduction as a function of topologies.




self-organisation was invoked.

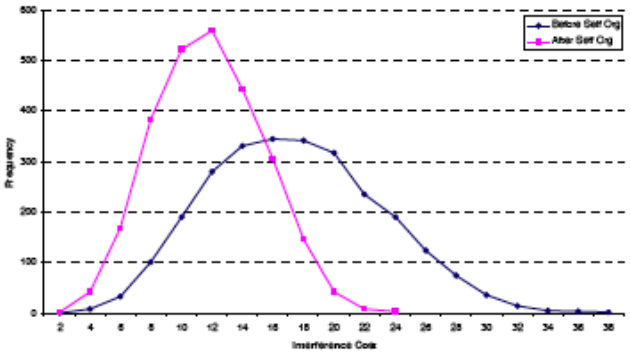
## 7. REDUCING THE PATH LENGTH

The algorithm for the path reduction is precisely the same as the algorithm in Section 6. but with the following 'improve' methods.

**Link adjustment with known traffic load.** Suppose that node $a$ has interference range $S_a$. Let $M_a$ be the set of nodes in $S_a$ excluding node $a$. Then use the method in Section 6. with the following 'improve' method:

improve $a = \{$
  **for** link $\alpha \ni a$, where $\alpha = [a,b]$
  suppose $\alpha$ is on channel $\Gamma^t_\alpha$;
  **set** $B \leftarrow \sum_{\beta \in S_\alpha} f(\alpha \mid \beta) + \sum_{\beta \in S_\alpha} f(\beta \mid \alpha)$;
  **if** (feasible) **re-route** $\alpha$'s traffic;
  **set** $\gamma \leftarrow \alpha$;
  **for** $y \in M_a$ **do** $\{$
    **for** $\Gamma_{[a,y]} = 1, \ldots, K$, **do** $\{$
      **if** $\sum_{\beta \in S_a} f([a,y] \mid \beta) + \sum_{\beta \in S_a} f(\beta \mid [a,y]) < B \times \varepsilon$
      **then** $\{$
        **set** $\gamma \leftarrow [a,y]$;

**Figure 6.** Comparison of *IC* across the network before (blue) and after (red) selforganisation.




**selflock** node $a$ **in** $[t+1, t+k]$;
**break**;
};
};
};
$\forall x \in V_a$ **transmit** "$\gamma$'s interference test signals";
**apply** load balancing algorithm to $S_a$;
}

**Trigger for attempting to adjust a link with known traffic load.** Consider a mesh with known traffic load such as that illustrated in Figure 1. Suppose that the load balancing algorithm has allocated load to links on the mesh, and let link $(a,b) = \arg\max_{x \in N_a^t} \rho(x)$. If replacing $(a,b)$ with $(a,x)$ would mean that there exists a cut through the mesh that traverses $(a,x)$ and that all other links on that cut have a load $< \rho(a,b)$ then let node $a$ initiate the link adjusting procedure. Likewise if replacing $(a,b)$ with $(y,b)$. **This is provisional. Have to double check. There could be a smarter way.**

## 7.1. Reactive Logic

The relationship between the reactive and proactive logics is determined by:

**if event** [link $\alpha$ is broken] **then** {
**activate** [activate the *Reactive Method* for link $\alpha$];
$\forall x \in \alpha$ **if state** [node $x$ locked by "**accept lock**$[a,x,s,t]$"
**then** {**transmit** "**reject lock**$[a,x,s,t]$";}
}

where the *Reactive Method* is as follows; it simply fixes disasters as they occur possibly with a configuration that is less satisfactory than the prior. It has no implications for neighbouring interfaces, and so it presents no instability issues.

**Reactive Method.** Important assumption for the functioning of the reactive logic discussed here is that all interfaces capable of reactive reconfiguration use omnidirectional antennas. The benefits and shortcomings of the usage of different antennas are discussed in details in our previous report. Two interfaces connected through directional antenna behave similarly to a wired point to point link because they cannot connect to any other interface to which their antennas are not aligned. This does not represent an impediment for the proposed architecture since majority of nodes will be equipped with omnidirectional antenna.

For the implementation of reactive logic we propose usage of simple mechanisms that are derived from routing protocols recently developed for stationary multi-radio mesh networks [4]. In conjunction with an appropriate routing protocol these mechanisms should ensure high reactivity in minimising effect of link interruptions caused by various factors.

**Link adjustment with unknown traffic load.** Suppose that node $a$ has interference range $S_a$. Let $M_a$ be the set of nodes in $S_a$ excluding node $a$. For nodes $x, y \in S_a$, let $c(x,y)$ denote the cost[5] of the least cost path that connects $x$ and $y$. We assume that: $(\forall x,y) c(x,y) = c(y,x)$, and that if the least cost path between nodes $u$ and $v$ is a subset of the least cost path between $x$ and $y$ then $c(u,v) \leq c(x,y)$. Let $N_a^t$ be the set of links in $S_a$ at time $t$, and $N_a^t(\ominus[a,x], \oplus[a,y])$ denotes the network configuration with link $[a,x]$ replaced by $[a,y]$. Let $C(N_a^t)$ denote the cost of the path of greatest cost in $S_a$: $C(N_a^t) \triangleq \max_{x,y \in S_a} c(x,y)$. Choose the pair of nodes $b$ and $c$ by:

$$(b,c) = \arg \min_{(x,y)|[a,x] \in N_a^t, y \in M_a} C(N_a^t(\ominus[a,x], \oplus[a,y]))$$

and swap link $[a,b]$ for link $[a,c]$ if:

$$C(N_a^t(\ominus[a,b], \oplus[a,c])) < C(N_a^t) \times \varepsilon$$

where $\varepsilon < 1$ is a threshold constant [11].

## 7.2. Results and Discussion

This part of study firstly proposes the method for the link substitution that results with the reduction of the path length. Secondly, to provide the insight in algorithms effectiveness we produce over 3000 simulations. The simulation results are statistically processed and the outcomes for 3 different densities (35,70 and 100) are obtained.

**Simulation parameters**. We have used a Java based framework to carry out the simulations for the results shown and discussed in this section. The key attributes of the simulation were:

[5]The precise meaning of this cost function does not matter. It could be simply the number of hops, or some more complex measure involving load and/or interference.

- Number of interfaces per router was randomly selected from 3 to 5.
- Default signal strength was 100 mW (20 dBm — Signal strength for each interface was randomly generated with +/- 25% variation.
- Network size had an area of $750m \times 500m$

The simulations were carried for realistic node densities and topologies as specified in:

| Node Densities | Topology |
|---|---|
| 35 | Grid +-5% variation, Grid +-50% variation. Random topology |
| 70 | Grid +-50% variation, Grid +-5% variation. Random topology |
| 100 | Grid +-50% variation, Grid +-5% variation. Random topology |

In addition to the simulation parameters described above we limited the number of links to $n-1$; where $n$ is number of router (density) in a network. Consequently, the number of links created was 34,69 and 99 for the corresponding network densities. In addition to these link numbers we tested the effectiveness of the link substitution algorithm by creating additional 10 links when link substitution reached efficiency threshold. The number of the substituted link was limited in all simulation to (10, 20, 30, 40 and 50) and separate results are shown.

Firstly we will show the effectives of link substitution algorithm through all link densities, various numbers of substituted links as well as with and without 10 additional links. From data depicted in Figure 7 we can conclude that as number of link substituted is increased the percentage of the path length reduction is increased. However, depending on network density and the number of substituted links this process reaches a threshold. This is in particular obvious in case on 35 router density with 40 and 50 substituted links and 10 additional links (on the left side of Figure 7) where the threshold of around 32% of path reduction is evident. Furthermore, the creation of additional links is further contributing towards reduction in path length. This can be best observed in the middle of figure 8 where results for 30 substituted links and 30 substituted + 10 additional links are depicted.

We continue our result analysis with the comparison of path lengths with and without link substitution. From the Figure 8 we can observer that our method significantly reduces path length by eliminating longer paths (maximum path length is 8 with the link substitution and 14 without it). This method also increases the number of shortest path (in particular paths 2 and 3 hops long).

We confirmed the effectives of the algorithm by providing the mean path length (in hops) for various network densities with (blue) and without (red) the link substation. From Figure 9 it can be deduced that in all cases examined the path

**Figure 7.** Percentage of path length reduction vs. number of substituted (*SL*) and additional links (*AL*) for different network densities (when not specified the number of additional link is 0).

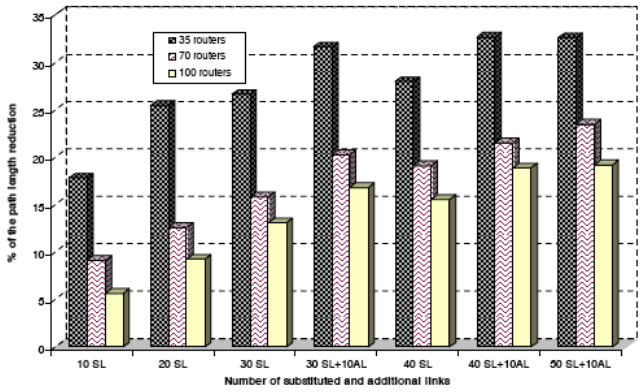
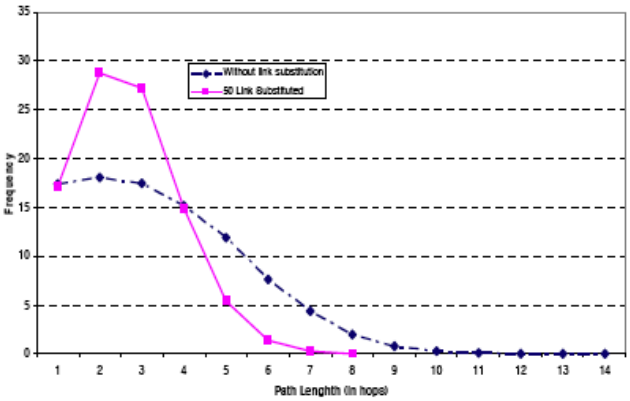



**Figure 8.** Frequency distribution of the path length (in hops) without and with link substitution algorithm at 100 node network density and 10 additional links.




length mean is reduced with the link substitution. In addition, we can observer that the variance is also reduced through the link substitution for approximately 0.5. The means provided are depicted with 98% confidence interval.

We conclude this analysis with the comparison of *IC* (interference cost) with and without link substitution. As we can see from Figure 10 the link substitution process has no affect on *IC* or *IC* reduction algorithm.

## 8. CONCLUSION & FUTURE WORK

In our previous work we have proposed an intelligent multi-agent system based self-organising algorithm for multi-radio wireless mesh networks (MR-WMN) that can operate on any radio technology. The algorithm ensures scalability by progressively assigning the channels to nodes in clusters during

**Figure 9.** Mean Path Length (in hops) for different network densities without and with link substitution (50 link substitutions).

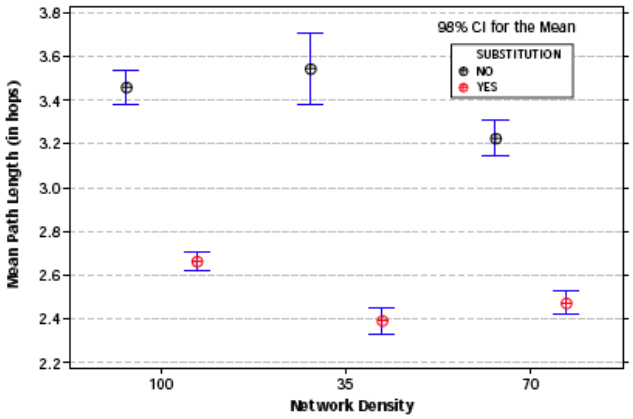
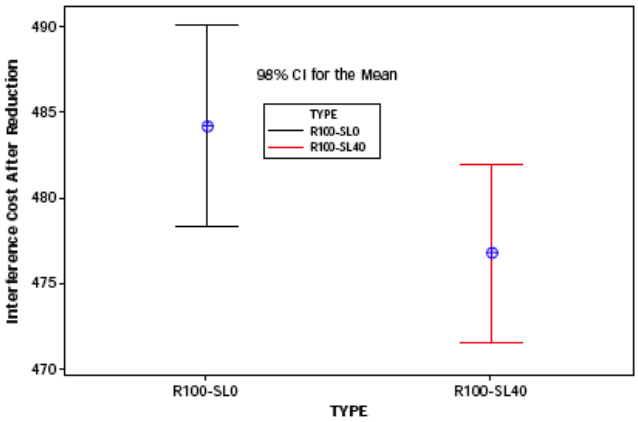



**Figure 10.** The mean of the *IC* after reduction with 40 link substitution (on left) and without link substitution (on right) for the network of 100 nodes.




the WMN system start up phase. The stability is offered by means of the proactive and reactive logic of the algorithm. These attributes were validated through analysis and simulation.

Through the work described in this report we have examined motivation and developed an algorithm for the topological control of MR-WMN. The goal of this algorithm is to increase the number of shortest paths to the portal nodes without adversely effecting interference cost. In addition to interference cost reduction implementation of this algorithm on MR-WMN further improve the system capacity.

Our future work will be focused on the development of our Java framework that is multi threaded so each node is represented as an independent thread. We believe that this will enable us to develop algorithms for tuning the capacity of the network links according to fluctuations in demand by mobile users.

## REFERENCES


[1] Ko, B.J., Misra, V., Padhye, J., Rubenstein, D.: Distributed Channel Assignment in Multi-Radio 802.11 Mesh Networks. Technical report, Columbia University (2006)

[2] Mishra, A., Rozner, E., Banerjee, S., Arbaugh, W.: Exploiting partially overlapping channels in wireless networks: Turning a peril into an advantage. In: ACM/USENIX Internet Measurement Conference. (2005)

[3] Mishra, A., Shrivastava, V., Banerjee, S.: Partially Overlapped Channels Not Considered Harmful. In: SIGMetrics/Performance. (2006)

[4] Akyildiz, I.F., Wang, X., Wang, W.: Wireless mesh netowrks: a survey. Computer Networks (2005) 445–487

[5] Raniwala, A., Chiueh, T.c.: Architecture and Algorithms for an IEEE 802.11-based Multi-channel Wireless Mesh Network. In: Proceedings IEEE Infocom '05, IEEE Computer Society (2005)

[6] Vasudevan, S.: A Simulator for analyzing the throughput of IEEE 802.11b Wireless LAN Systems. Master's thesis, Virginia Polytechnic Institute and State University (2005)

[7] Leith, D., Clifford, P.: A self-managed distributed channel selection algorithm for wlans. In: Proceedings of RAWNET, Boston, MA, USA (2006) 1–9

[8] Duffy, K., Malone, D., Leith, D.: Modeling the 802.11 Distributed Coordination Function in Non-saturated Conditions. IEEE Communication Letters **9** (2005) 715 – 717

[9] Tourrilhes, J.: Robust Broadcast: Improving the reliability of broadcast transmissions on CSMA/CA. In: Proceedings of PIMRC 1998. (1998) 1111 – 1115

[10] Mishra, A., Shrivastava, V., Agarwal, D., Banerjee, S., Ganguly, S.: Distributed channel management in uncoordinated wireless environments. In: Proceedings of the 12th annual international conference on Mobile computing and networking, Los Angeles, SA, USA (2006) 170–181

[11] Ramachandran, K., Belding, E., Almeroth, K., Buddhikot, M.: Interference-aware channel assignment in multi-radio wireless mesh networks. In: Proceedings of Infocom 2006, Barcelona, Spain (2006) 1–12