A Light-Weight Multi-Agent System Manages 802.11 Mesh Networks

Ante Prodan & John Debenham

Abstract A light-weight multi-agent system is employed in a “self-organisation of multi-radio mesh networks” project to manage 802.11 mesh networks. As 802.11 mesh networks can be extremely large the two main challenges are the scalability and stability of the solution. The basic approach is that of a distributed, light-weight, co-operative multiagent system that guarantees scalability. As the solution is distributed it is unsuitable to achieve any global optimisation goal — in any case, we argue that *global optimisation* of mesh network performance in any significant sense is not feasible in real situations that are subjected to unanticipated perturbations and external intervention. Our overall goal is simply to reduce maintenance costs for such networks by removing the need for humans to tune the network settings. So stability of the algorithms is our main concern.

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 struc-

Ante Prodan
University of Technology, Sydney, Australia e-mail: aprodan@it.uts.edu.au

John Debenham
University of Technology, Sydney, Australia e-mail: debenham@it.uts.edu.au

ture, 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. We describe the role of the load balancing algorithm that our methods take as a given input. The measurement of interference cost is discussed in Section 4. Methods for the adjusting the channels in a multi-radio mesh networks for predictable load are described in Section 5, as well as a method for adjusting the links. Future plans are described in Section 6.

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.

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 $i = 1, \dots$, and links are denoted by Greek letters: α, β, γ,.... 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 α, S_α is the set of links that contain nodes n's interference range $\forall n \in \alpha$.

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

Γ_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 4.

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 5. 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".

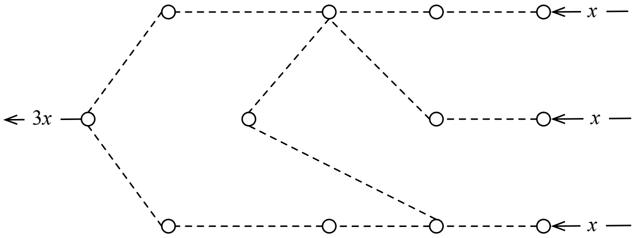
802.11 related terms: BSS — the basic service set. Portal — is the logical point at which MSDUs from an integrated non-IEEE 802.11 LAN enter the IEEE 802.11 DS (distribution system). WM — Wireless Medium. IBSS — Independent Basic Service Set. MSDU — MAC Service Data Unit.

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

Fig. 1 The load balancing algorithm determines the allocation of load.



as it has priority $q > p$. That is an interface can only overrule a request to organise if it has higher priority.

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.

- "**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 Measuring Interference Cost

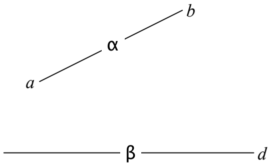
Suppose that during some time interval Δt two interfaces a and b are transmitting and receiving on channels Γ_a and Γ_b. During Δt, the *interference limit* that interface x imposes on interface y, $\tau_{y|x}$, is a ratio being the loss of traffic volume that interface y could receive if interface x 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 Γ_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 Γ_b as

[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 — see [6].

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



experienced by interface a listening on channel Γ_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 Γ_β, and we ignore the possibility of them both transmitting at the same time[4]. Further suppose that v_β is the proportion of Δt for which either interface c or interface d is transmitting. Then for some κ_β, $0 \leq \kappa_\beta \leq 1$: $v_c = \kappa_\beta \times v_\beta$, 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_α is the proportion of Δt for which either interface a or interface b is transmitting. Then for some κ_α, $0 \leq \kappa_\alpha \leq 1$: $v_a = \kappa_\alpha \times v_\alpha$, $v_b = (1 - \kappa_\alpha) \times v_\alpha$. Then as a will only receive interference when it is listening to b transmitting:

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

[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].

and so:

$$\begin{aligned} f(\alpha \mid \beta) = \\ & (1-\kappa_\alpha)\times\nu_\alpha\times\nu_\beta\times\big(\tau_{a|c}\times\kappa_\beta+\tau_{a|d}\times(1-\kappa_\beta)\big) \\ & +\kappa_\alpha\times\nu_\alpha\times\nu_\beta\times\big(\tau_{b|c}\times\kappa_\beta+\tau_{b|d}\times(1-\kappa_\beta)\big) \end{aligned} \tag{1}$$

Note that ν_α, ν_β, κ_α and κ_β are provided by the load model, and the $\tau_{x|y}$ are provided by the interference model.

5 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 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 α 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 $\Gamma_\alpha^t$;
    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 $\Gamma_\alpha = 1, \ldots, K, \Gamma_\alpha \neq \Gamma_\alpha^t$ do{
        if $\sum_{\beta \in S_\alpha} f(\alpha \mid \beta) + \sum_{\beta \in S_\alpha} f(\beta \mid \alpha) < B \times \varepsilon$ then{
            $\Gamma_\alpha^{t+1} \leftarrow \Gamma_\alpha$;
            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.

5.1 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 τ 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.

5.1.1 Expected SNIR.

The complete SNIR at the receiver based on a set of interfering links in the carrier sensing range is given by[5]

$$\begin{gathered} \text{SNIR} = \frac{P_r}{N + \sum_{k=1}^{n} I_k} \\ N = K \times W \times T \end{gathered} \tag{2}$$

where: P_r = Received power of the frame, $\sum_k I_k$ = Received powers of the set of n interfering nodes (interfaces), N = Thermal noise, k = Boltzmann constant, W = Spectral bandwidth of the carrier (For example the channel bandwidth is 22 MHz in 802.11b), and T = Absolute temperature at the receiver.

Let us assume that the node (interface) j wants to trigger the proactive logic i.e. possibly change channel with node (interface) i. Then Equation 2 gives the sum of the interferences from the neighbouring links[6] in the carrier sensing range:

$$\sum_{k=1}^{n} I_k = \sum_{<k,l>\in R} \frac{P_{kl} \times G_{jk} \times G_{kj}}{\text{PL}_{kj}}$$

where: R = Set of all links that interfere with link α between i and j node (interfaces), P_{kl} = Power transmitted by the node (interface) k to node (interface) l, G_{jk} = Gain of Antenna of node (interface) j towards node (interface) k, G_{kj} = Gain of Antenna of node (interface) k towards node (interface) j, and PL_{kj} = Path loss suffered by the signal while traversing from the node (interface) k to the node (interface) j.

The values for 802.11 interfaces transmit power, Antenna gains are generally specified by the vendor in the data sheets of the equipment.

A general formula for calculating path loss (PL) in the Friis free space i.e. Line of Sight (LOS) link between the transmitter and receiver is given by[7]

[5] Analyses of Measurements and Simulations in Multi-hop Ad-hoc Environment, Report IST-2001-37385 6HOP D2.3

[6] see "Topology Planning for Long Distance Wireless Mesh Networks", Indian Institute of technology, Kanpur.

[7] Simon Haykin, Communication Systems, 4th edition, John Wiley & Sons Inc, 2001.

$$\mathrm{PL} = -10 \times \log_{10}(G_t G_r) + 10 \times \log_{10} \left(\frac{4 \times \pi \times d}{\lambda} \right)^2$$

where: G_t = Gain of the transmitting antenna, G_r = Gain of the receiving antenna, d = Distance between the transmitting and receiving antennas, and λ= Transmission Wavelength. In our Wireless Mesh Network the GPS in the nodes can measure d.

However, in most of the scenarios for urban areas the link between the transmitter and receiver will generally be Non LOS (NLOS). In these cases we can determine the path loss PL_{kj} by using the Co-operation in the field of Scientific and Technical research project 231 (COST231) adopted propagation model called as the *Walfisch-Ikegami model*[8].

Therefore the formula for the expected SNIR is given by:

$$\mathbb{E}(\mathrm{SNIR}) = \frac{P_{ij} \times G_{ij} \times G_{ji}}{N + \sum_{<k,l>\in R} \mathrm{PL}_{kj}}$$

5.1.2 Expected BER and FER.

The BER is based on the type of modulation scheme that is used by the PHY layer of the radio to transmit the data. For example 802.11b uses different modulation schemes for different data rates such as: Differential Binary Phase Shift Keying (DBPSK) for 1 Mbps, Differential Quadrature Phase Shift Keying (DQPSK) for 2 Mbps and Complimentary Code Keying (CCK) for 5.5 and 11 Mbps [9].

Each of the modulation schemes has a different formula for calculating the BER, which can be referred to in[10].

For example the BER in an Additive White Gaussian Noise (AWGN) channel for DBPSK is given by[11]:

$$\mathrm{BER} = \frac{1}{2} \times \exp(-\mathrm{SNIR})$$

Assuming that each bit error is an independent event, then a simple relationship between BER and FER is given by[12]:

$$\mathrm{FER} = 1 - (1 - \mathrm{BER})^n$$

where: n = Number of bits in the frame.

[8] J.S. Lee and L.E. Miller, CDMA Systems Engineering Handbook, Artech House, 1998.

[9] Ji Zhang, "Cross-Layer Analysis and Improvement for Mobility Performance in IP-based Wireless Networks", Ph.D. Thesis, Sept. 2005.

[10] A. Ranjan, "MAC Issues in 4G", IEEE ICPWC, pp. 487-490, 2005.

[11] Simon Haykin, Communication Systems, 4th edition, John Wiley & Sons Inc, 2001.

[12] Analyses of Measurements and Simulations in Multi-hop Ad-hoc Environment, Report IST-2001-37385 6HOP D2.

6 Conclusion and future work

In our previous work we have proposed an intelligent multiagent 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 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