Minimum Latency Broadcasting in Multi-Radio Multi-Channel Multi-Rate Wireless Mesh Networks

Junaid Qadir[1], Chun Tung Chou[1], Archan Misra[2]
[1]School of Computer Science and Engineering, University of New South Wales, Australia
[2]IBM T J Watson Research Center, Hawthorne, New York, USA
Email: {junaidq, ctchou}@cse.unsw.edu.au,archan@us.ibm.com

Technical Report UNSW-CSE-TR-0608

April 28, 2006

THE UNIVERSITY OF
NEW SOUTH WALES

SYDNEY • AUSTRALIA

***Abstract*—We address the problem of minimizing the worst-case broadcast delay in multi-radio multi-channel multi-rate (MR^2-MC) wireless mesh networks (WMN). The problem of '*efficient*' broadcast in such networks is especially challenging due to the numerous inter-related decisions that have to be made. The multi-rate transmission capability of WMN nodes, interference between wireless transmissions, and the hardness of optimal channel assignment adds complexity to our considered problem. We present four heuristic algorithms to solve the minimum latency broadcast problem for such settings and show that the '*best*' performing algorithms usually *adapt* themselves to the available radio interfaces and channels. We also study the effect of channel assignment on broadcast performance and show that channel assignment can affect the broadcast performance substantially. More importantly, we show that a channel assignment that performs well for unicast does not necessarily perform well for broadcast/multicast. To the best of our knowledge, this work constitutes the first contribution in the area of broadcast routing for MR^2-MC WMN.**

1 Introduction

Wireless mesh networks (WMN) [1], where potentially-mobile mesh clients connect over a relatively-static multi-hop wireless network of mesh routers are viewed as a promising broadband access infrastructure in both urban and rural environments [2]. However, the relatively low spatial reuse of a single radio channel in multi-hop wireless environments (due to wireless interference) remains an impediment to the wide-spread adoption of WMN as a viable access technology. It has been shown that network capacity drops off as the number of nodes is increased in single-channel wireless networks [3]. With recent advancements in wireless technology rendering the usage of multiple radios affordable, a popular current trend is to equip mesh nodes with multiple radios, each tuned to a distinct orthogonal channel. The usage of multiple radios can significantly improve the capacity of the network by employing concurrent transmissions in the network [4][5][6]. Another feature widely available in commodity wireless cards, which are envisioned to connect the wireless mesh nodes, is the ability to transmit at multiple transmission rates. WMN nodes can utilize the flexibility of multi-rate transmissions to make appropriate range and throughput/latency tradeoff choices across a wide range of channel conditions. While this flexibility has traditionally been used only for unicast, it has recently been proposed for use in broadcasting scenarios as well [7] [8]. In the near future, multi-radio multi-channel multi-rate (MR^2-MC) WMNs are expected to gain a niche in the wireless market due to adoption and support from leading industry vendors [9] and active research from the research community.

An important open question in MR^2-MC WMNs which we attempt to address in this paper is how to perform '*efficient*' broadcast in such networks. We gauge this efficiency in terms of *'broadcast latency'* which we define as the maximum delay between the transmission of a packet by the source node and its eventual reception by all receivers. The minimum latency broadcasting (*MLB*) problem is particularly challenging in MR^2-MC meshes due to a myriad of complex, inter-twined decisions that need to be made. The authors of [6] have hinted about some of the potential problems that can be faced for broadcast routing in multi-radio meshes (*vis-a-vis* channel assignment).

The MLB problem, apart from its theoretical significance, is an important practical problem in WMN. The presence of several multi-party applications—such as software updates to all devices, local content distribution (e.g., video feeds) in community networks and multimedia gaming—often impose stringent latency requirements on the underlying network and motivate the study of the MLB problem. The MLB problem has been studied for '*single-radio single-channel*' (SR-SC) wireless networks, both for the single-rate [10] and the multi-rate case [7] [8]. To the best of our knowledge, the MLB problem for MR^2-MC WMNs has not been addressed in literature and our work is the first contribution in this area. We shall show that the MLB problem for MR^2-MC meshes is a more complex problem than for SR-SC multi-rate meshes (single-radio meshes are a special case of multi-radio meshes). The differences between *single-rate* and *multi-rate* MLB problem, for the case of *SR-SC* meshes, are demonstrated in [7] [11] and the complexity of each problem is proven NP-hard in [7] [11] and [10] respectively.

2 Background and Related Work

Broadcasting in wireless networks is a fundamentally different problem to broadcasting in wired networks due to the '*wireless broadcast advantage*' (WBA) [12]. The WBA arises due to the broadcast nature of the wireless channel where—assuming omni-directional antennas are being used—a transmission by a node can be received by all neighboring nodes that lie within its communication range. This situation is quite different to that of wired networks where the cost to reach two neighbors is generally the sum of the costs to reach them individually. This is due to the shift in paradigm from the *'link-centric'* nature of wired networks to the *'node-centric'* nature of wireless communications. A lot of research has focussed on achieving '*efficient*' broadcast in multi-hop wireless networks and mobile ad hoc networks. The metrics typically used are energy consumption [12] [13], the number of transmissions [14] [15], or the overhead in route discovery and management [16]. The limited work done for the broadcast latency metric has focussed only on SR-SC networks [7] [10] [11].

Our current work builds upon our previous work on minimizing broadcast latency in a SR-SC *multi-rate* WMN [7] [8], where we introduced the new concept of link-layer *multi-rate* multicast, in which a node can adjust its link-layer multicast transmission rate to its neighbors. We showed that multicast in a multi-rate WMN has two features not found in a single-rate WMN. Firstly, if a node has to perform a link-layer multicast to reach a number of neighbors, then its transmission rate is limited by the smallest rate on each individual link, *e.g.,* if a node n is to multicast to two neighboring nodes m_1 and m_2, and if the maximum unicast rates from n to m_1 and m_2 are, respectively, r_1 and r_2, then the maximum rate n can use is the minimum of r_1 and r_2. Secondly, for a multi-rate WMN, the broadcast latency can be minimized by exploiting an extra degree-of-freedom where some nodes transmit the same packet more than once, but at a different rate to different subsets of neighbors (called as '*distinct-rate transmissions*'). Based on these insights, we presented the '*WCDS*' and '*BIB*' algorithms in [7] [11] as heuristic solutions for the MLB problem in SR-SC multi-rate mesh networks. Both these algorithms consider the WBA and the multi-rate capability of the network, and also incorporate the possibility of multiple *distinct-rate*

Transmission rate (Mbps)	Transmission range
1	483
2	370
5.5	351
11	283

Table 1: Relationship between transmission latency and range

transmissions by a single node. Details of these algorithms are available at [7] [11].

The assignment of channels in MR^2-MC wireless meshes plays a very important part in determining the actual performance of the network. Generally there are two conflicting objectives for any channel assignment protocol: while nodes will usually benefit from increased *'connectivity'* among themselves, the channel assignment protocol also tries to reduce some measure of '*interference*'. The different channel assignment strategies can be classified into *static*, *dynamic*, and *hybrid* categories [6]. For our current work, we do not consider dynamic and hybrid strategies, due to the non-negligible interface switching delay and synchronization requirements involved in such strategies.

Amongst the static channel assignment strategies, the simplest approach is the '*common channel approach*' (CCA) (*e.g.* [4]), in which all nodes are assigned a common set of channels. The benefit of this approach is its simplicity and that the connectivity of the network is a multiple of the connectivity of a single channel mesh. In an alternative approach called '*varying channel approach*' (VCA), interfaces of different nodes may be assigned to a different set of channels (*e.g.* [5]). With this approach, there is a possibility of a network partition, unless the interface assignment is done carefully. In yet another approach called '*interference survivable topology control*' (INSTC) [17], the channel assignment is made such that the induced network topology is interference-minimum among all k-connected topologies.

3 Network and Interference Model

We follow the notation introduced by [17] to represent our channel assignment; also we use a similar network model to that described by [5]. Each node in the network can transmit at multiple-rates. There are totally C non-overlapping orthogonal frequency channels in the system and each node is equipped with Q radio interfaces where $Q \leq C$. The Q radio interfaces have omni-directional antennas, and unit disk graph model is assumed. In order to efficiently utilize the network resources, two radio interfaces at the same node are not tuned to the same channel. Using the Qualnet simulator [18] as a reference, we obtain the transmission rate versus transmission range relationship in Table I, assuming a two-ray propagation model. Note also that the interference range in Qualnet is $520m$. The transmission range is a decreasing function of transmission rate as Table 1 illustrates.

We use an undirected graph $G_T = (V, E_T, L_T)$ to model the given mesh network topology *before channel assignment*, where V is the set of vertices, E_T is the set of edges and L_T is the set of weights of edges in E_T. The vertex v in V corresponds to a wireless node in the network with a known location. An undirected edge (u, v), corresponding to a wireless link between u and v, is in the set E_T *if and only if* $d(u, v) \leq r$ where $d(u, v)$ is the Euclidean distance between u and v and r is the range of the *lowest-rate* transmission. The latency of a link $l(u, v)$ is the latency of the '*fastest*' transmission rate that can be supported between nodes u and v. The set L_T contains the latencies of all links in E.

Channel assignment: A channel assignment $\mathcal{A}$ assigns each vertex v in V, Q different channels denoted by the set: $A(v) = \{a_1(v), a_2(v), \ldots, a_Q(v) : a_i(v) \neq a_j(v), \forall i \neq j;\ a_i(v) \in C, \forall i\}$ where $a_i(v)$ represents the channel assigned to i^{th} radio interface at node v.

The topology defined by $\mathcal{A}$ is represented by $G = (V, E, L, \Lambda)$ in the following natural way: There is an edge $e = (u, v, k)$ on channel $\lambda(e) = k$ between nodes u and v in G *if and only if* $d(u, v) \leq r$ (i.e. edge$(u, v) \in E_T$) and $\lambda(e) \in \mathcal{A}(u) \bigcap \mathcal{A}(v)$. The latency of the edge e is the latency of the fastest transmission rate supported on e. The set L contains the latency of each edge in E; similarly the set Λ contains the channel used on each edge in E. Note that G may be a *multi-graph*, with multiple edges between the same pair of nodes, when the node pair shares two or more channels. We use the same notation to refer to *vertices and nodes*, to *edges and links*, and to *weight of edges and latency of links* without confusion, the usage being clear from the context.

It is assumed that the *channel assignment* is done independently from our broadcasting framework. This design decision reflects the practical reality that the channel assignment strategy will likely be dictated by other factors, including the presence of *unicast* traffic on the WMN. We have used the following three *static* channel assignment strategies in our current work: CCA, VCA and INSTC. For CCA, dedicated interfaces are allocated for the *same* Q channels at every node, therefore only Q channels are used in the network when using CCA. In VCA, an interface at all nodes is allocated the same channel to ensure a connected network; for the remaining $Q - 1$ interfaces, channels are chosen randomly from the remaining $C - 1$ channels. The last channel assignment scheme used is INSTC, which we use to construct *at least* a 1-connected topology (*i.e.* a connected topology).

Interference Model: We use a generalized conflict graph *based on transmissions* to model the effects of wireless interference between different multicast transmissions in MR^2-MC meshes. The conflict graph indicates which transmissions mutually interfere and hence cannot be active simultaneously.

A transmission b_i interferes with a transmission b_j, if both transmissions b_i and b_j are taking place on the *same* channel, *and* the receivers of the transmission b_i are within the interference range of the transmitting node of b_j or *vice-versa*. The transmissions b_i and b_j do not interfere otherwise.

4 Problem statement

Objective: The MLB problem is formally defined as follows. Let the connectivity graph $G = (V, E, L, \Lambda)$, number of radio interfaces (Q), number of channels (C), channel assignment $\mathcal{A}$ to all Q interfaces at each node $v \in V$, and the broadcast source node s be given. The objective is to construct a spanning tree $T_{MLB} = (V, E', L', \Lambda')$ (where $E' \subseteq E, L' \subseteq L, \Lambda' \subseteq \Lambda$) that minimizes the broadcast delay, found after interference-free transmission scheduling, to reach all broadcast nodes. For an arbitrary tree T, let the time that $v \in V$ receives the broadcast from the root node (the source) be $\delta_T(v)$. If there are J possible spanning trees of G, then T_{MLB} is the tree that reaches all broadcast nodes with least delay, amongst all J spanning trees of G as shown below:

$$T_{MLB} = \arg \min_{T_j, \forall j=[1,J]} (\max(\delta_{T_j}(v_i), \forall v_i \in V))$$

4.1 Hardness results for the problem

The NP-hardness of the MLB problem for MR^2-MC meshes can be established by following from the fact that the MLB problem is NP-hard for specific instances of MR^2-MC scenario i.e. for the SR-SC single-rate case [10] and for the SR-SC multi-rate case [7]. Thus the MLB problem for MR^2-MC WMNs is at least NP-hard.

5 Heuristic to construct MLB tree in MR^2-MC multirate mesh

In this section, we present heuristic algorithms to create efficient delivery trees for broadcasting in a MR^2-MC WMNs. Since the channel assignment is performed independently of our framework, the topology defined by the channel assignment process $\mathcal{A}$ is an input to our framework. Broadly speaking, any heuristic algorithm designed to solve the MLB tree in MR^2-MC meshes must make three important decisions at each node. Firstly, it has to decide whether a node should transmit (i.e., be a non-leaf node in the broadcast tree) or not, and if so, whether the transmission should occur over all or some of its radio interfaces. Secondly, the number of transmissions the node will actually make *must be determined according to the number of radio interfaces and channels available*, alongside the nodes covered in each of these transmissions. Lastly, the transmissions at each node must be scheduled to minimize the broadcast delay after due consideration of radio interference and the number of interfaces available.

MLB is a combination of many closely inter-related hard problems e.g. minimum latency tree construction, interference free transmission scheduling and the choice of rate and interface to use for transmissions are all intertwined sub-problems of the overall MLB problem. With the complexity of the problem in mind, we have decomposed our solution into three logically independent steps:

1) *Topology Construction*: The aim of this step is to compute a broadcast tree (or a spanning tree) T of the given topology that exploits the WBA, the multi-rate transmission capability *and the plurality of radio interfaces and channels available*. The transmitting nodes, their interfaces used for transmissions and the children/parent relationships between different nodes are all decided in this stage. However, the decision on the number of distinct-rate transmissions (either one or more) on any particular chosen interface is deferred to the next step.

2) *Downstream Multicast Grouping*: The aim of the 'multicast grouping' algorithm is to take the spanning tree constructed during the 'topology construction' stage and *determine both the rates and number of distinct-rate transmissions that each interface should perform.* Intuitively, the rationale behind multiple transmissions is to allow faster transmission to the more *critical* child nodes—i.e., those nodes that have leaf nodes with larger delivery latencies—at the expense of larger transmission latency to the other child nodes.

3) *Transmission Scheduling*: While the number of transmissions at each non-leaf node of the tree is determined after 'topology construction' and 'multicast grouping', the exact timing of the various transmissions especially relative to different branches of the tree still needs to be determined. The final step schedules all transmissions while taking into account that a node can only transmit after it has received the packet and interfering transmissions cannot occur concurrently. We are conceptually assuming a centralized scheduler in our current work, and plan to investigate the use of decentralized MAC schedulers in future work.

This decomposition of the overall optimization problem is clearly not optimal. For example, we obtain the multicast transmission sets and the transmission rate associated with each link layer multicast only after the 'multicast grouping'. However, a joint optimization is computationally infeasible except for trivially small mesh topologies.

We present four heuristic algorithms for the 'topology construction'; the first (Section 7.1) does not exploit the WBA, the second (Section 7.2) exploits WBA but not the availability of multiple interfaces on the same node, while the other two (Sections 7.3 and 7.4) differ in how they exploit both WBA and the interface diversity on individual nodes. We follow the 'topology construction' stage with a broad algorithmic approach for the 'multicast grouping' in Section 8. We then present the 'transmission scheduling' heuristic in Section 9.

6 Example topology

To explain the intuition of our algorithms, we will recourse to a simple example MR^2-MC wireless mesh network of 10 nodes in an area of $800 \times 800\ m^2$ throughout this work. We assume that Q (the number of interfaces) is equal to 2 and C (the number of channels) is equal to 4. The positioning of the nodes are as shown in the Figure 1. As mentioned earlier, the input topology to our algorithms depends on the channel-assignment scheme. We present the CCA, INSTC and VCA channel-assignment schemes for our example MR^2-MC WMN in Figures 1, 2 and 3. The source-node of the broadcast is represented by a *green square* marker, and the receiver nodes are represented by *blue circular* markers. We denote the channels assigned to the interfaces at a node (recall Q=2) in square brackets above the node marker. The node ID (or number) is represented below the node marker. As shown in Figures 1, 2 and 3, CCA scheme allocates the same set of channels at each node, whereas VCA scheme allocates *one* common-channel to all nodes, with the remaining channels allocated randomly. The INSTC scheme performs channel assignment—without enforcing a common-channel to be used amongst all nodes in the network—to minimize the interference while maintaining a connected network.

7 Topology Construction

The common input to each of our 'topology construction' heuristic algorithm is the channel assignment defined input topology $G = (V, E, L, \Lambda)$, broadcast source s in V, the set $\mathcal{L} = \{l_1, l_2, ..., l_k\}$ denoting set of latencies of all possible k transmission rates, and the channel assignments to all interfaces at each node $\mathcal{A}$.

7.1 Multi-Radio, Multi-Channel, Shortest-Path Tree (MSPT)

The MSPT algorithm (see Algorithm 1) is used to construct the SPT for MR^2-MC wireless meshes. The MSPT algorithm is very similar to the greedy Dijkstra algorithm, and works on the principle of edge relaxation. The MSPT algorithm differs from the general Dijkstra's algorithm, in that it also has to choose appropriate channels for each link it chooses for the MSPT (since a node pair can have multiple links on distinct channels). The 'transmission scheduling' step (discussed next in Section 9) greatly depends on the channel selections made during the 'topology construction'. Channel

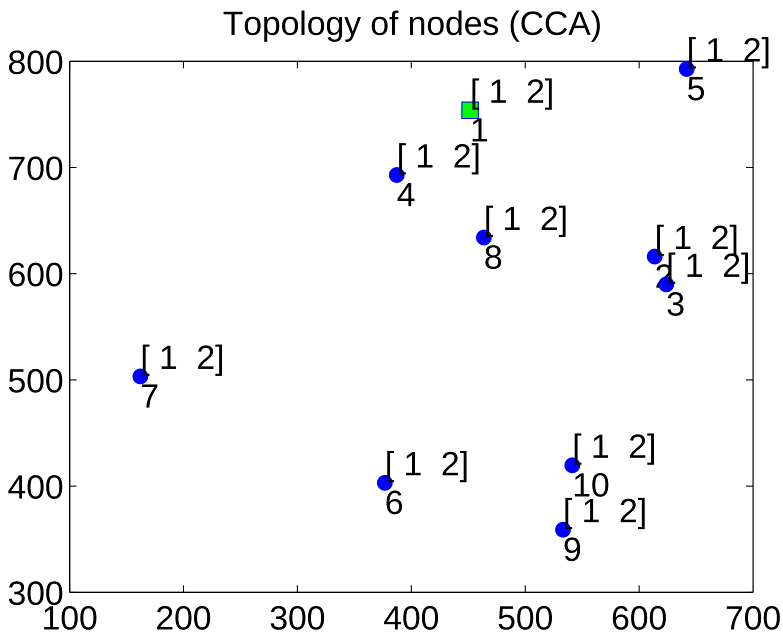


Figure 1: The topology defined by the channel assignment scheme CCA

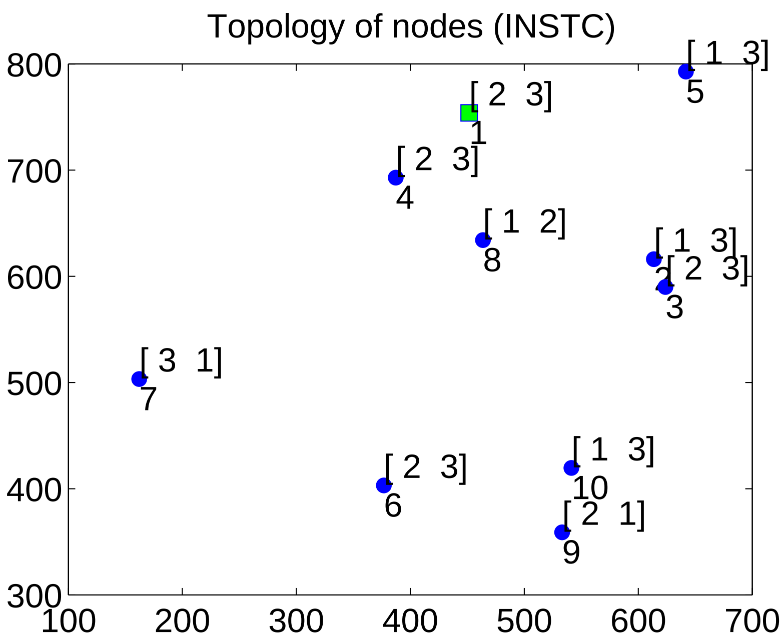


Figure 2: The topology defined by the channel assignment scheme INSTC

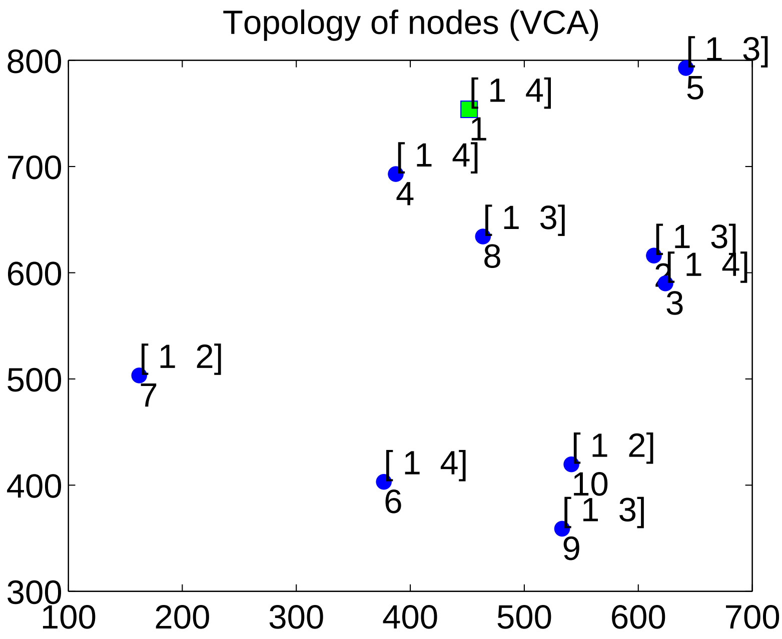


Figure 3: The topology defined by the channel assignment scheme VCA

selection, if done poorly without due consideration of radio interference, can dramatically degrade performance even for the same spanning tree.

7.1.1 Algorithm

The MSPT algorithm starts by initializing the '*labels*' of all nodes to ∞. The label of any node represents the '*cost*' of its *current* shortest path to the source s; with a label of ∞ indicating the absence of a path. The set $\mathcal{R}$ (representing the nodes, whose shortest paths to s have not been finalized yet) is initialized to contain all nodes in V. The algorithm starts by putting d (the node relaxed at the next iteration) equal to s for the initial round. The basic operation of MSPT algorithm is edge relaxation: if there is an edge from u to v, then the shortest known path from s to u (having cost $label(u)$) can be extended to a path from s to v by adding edge (u, v) at the end. This path will have length $label(u) + l(u, v)$ where $l(u, v)$ is the latency of link between vertices u and v. If this is less than the current $label(v)$, we can replace the current value of $label(v)$ with the new value. After edge relaxation in each round, the set of nodes whose labels are reduced from their former values are referred to as I.

Amongst the nodes in I, those connecting to d on the same latency transmission $l_u \in \mathcal{L}$ are denoted by I_{lu}, and are assigned a single channel if sharing a common channel. The channel chosen is the *'least-used'* in the

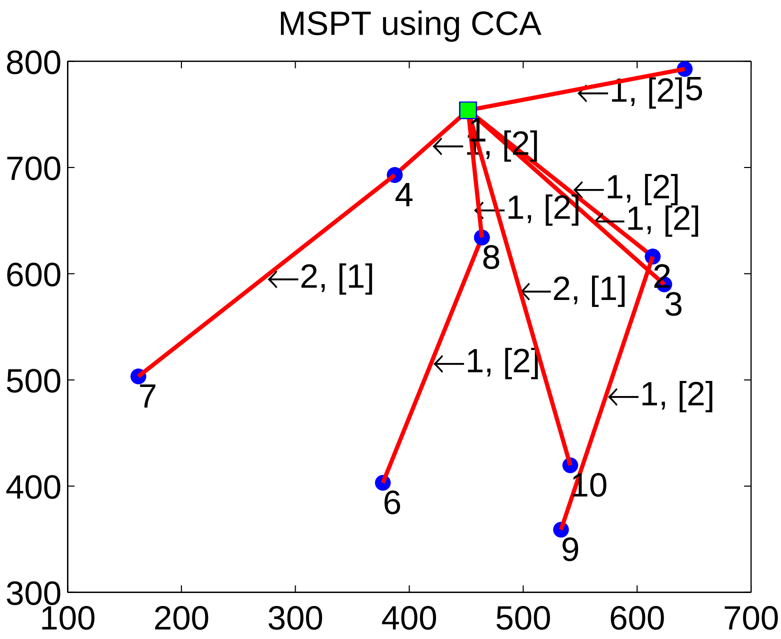


Figure 4: The MSPT for the channel-assignment scheme CCA

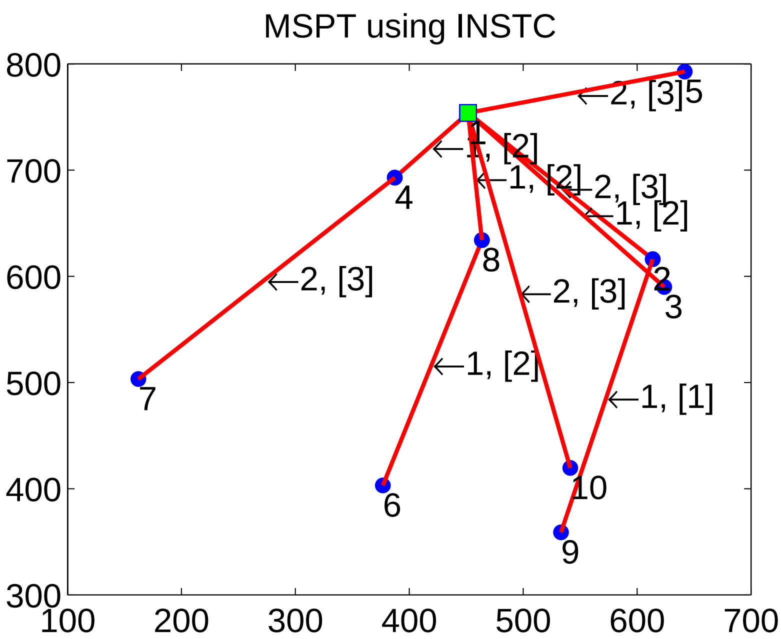


Figure 5: The MSPT for the channel-assignment scheme INSTC

Algorithm 1 MSPT construction

Input: $[s, G = (V, E, L, \lambda), \mathcal{L} = \{l_1 \cdots l_k\}]$
Initialize $\text{label}(v_i) = \infty, \forall v_i \in V$;
$\mathcal{R} = [1 \cdots |V|]$; $d = s$; $R = R \backslash \{s\}$;
while $(V \backslash \mathcal{R} \neq \emptyset)$ **do**
  $N =$ connecting nodes of d;
  $label_{new} = label$;
  $label_{new}(N) =$
  $\min((label(d) + cost(d, N)), (label(N)))$;
  I $\leftarrow$ nodes s.t. $label_{new}(nodes) < label(nodes)$
  $P_{MSPT}(I) = d$;
  $E_{MSPT} = E_{MSPT} \cup edge(d, I)$
  $L_{MSPT}(edge(d, I)) = l(d, I)$;
  for $u = 1$ to |unique-latency-transmissions| at d **do**
   find all nodes I_{lu} **s.t.** $I_{lu} \in I$ and $l(d, I_{lu}) = l_u \in \mathcal{L}$
   $\Lambda_{MSPT}(I_{lu})) =$ least-used channel in the conflict graph of the transmission I_{lu}
  end for
  $label = label_{new}$; $d = \arg\min(label(\mathcal{R}))$;
  $R = R \backslash \{d\}$
end while
Output: $[P_{MSPT}, L_{MSPT}, \Lambda_{MSPT}, label]$

conflict graph of this transmission. *Thus, MSPT is based on the Dijsktra algorithm and does not explicitly consider the WBA; it only considers the use of a less contended channel among available channels between a candidate node pair.* Edge relaxation is applied until all values $label(v)$ represent the cost of the shortest path from s to v. MSPT is mathematically described in Algorithm 1. After $|V| - 1$ rounds, the shortest path from each vertex $v \in V$ to s is determined.

7.1.2 MSPT for our example network (as shown in Section 6)

We now refer back to our example network, shown in Section 6, to explain the working of MSPT. The MSPT algorithm constructed trees are depicted in Figures 4, 5 and 6 for CCA, INSTC and VCA channel-assignment schemes, respectively. The tree construction is done similar to Dijkstra's algorithm. We will focus more on how appropriate channels are chosen for MSPT tree links. Recall that during the tree-construction, the channel used for a transmission is the least-used channel in the *conflict-graph of that particular transmission.*

Initially, for all the channel-assignment schemes we consider, the source node 1 has transmissions at latency 1 and 2; since both these transmissions interfere with each other, they are assigned different channels. The nodes 2, 4 and 8 also transmit at latencies 1, 2 and 1, respectively. It is

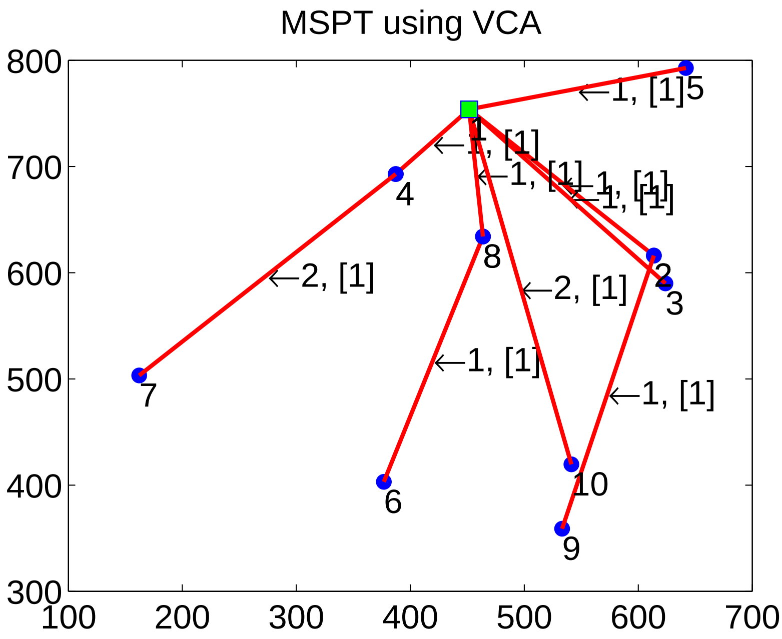


Figure 6: The MSPT for the channel-assignment scheme VCA

preferred that different channels be chosen for these three transmissions, as all of them interfere with each other. Since, CCA only utilizes $Q \leq C$ number of channels, in our example, we can only use channels 1 and 2, as Q=2. Although INSTC and VCA generally use more channels than CCA, their connectivity and WBA exploitation generally reduces due to their greater channel-diversity—as the probability of two nodes sharing a common channel is minimized with increasing channel-diversity. The path from the source-node to each node has the lowest possible cost in the MSPT (i.e., without considering interference, MSPT is the best tree). It shall be seen in Table 2 that MSPT, despite being the shortest-path-tree, is not necessarily the '*best*' tree with respect to broadcast latency after accouting for wireless interference. The performance of MSPT with CCA, INSTC and VCA channel-assignment schemes, for our particular example, is 4, 4 and 7, respectively.

7.2 Multi-Radio, Multi-Channel, Weighted Connected Dominating Set Tree (MWT)

The MWT algorithm (see Algorithm 2) is an extension to the WCDS algorithm, which is designed for the MLB problem for SR-SC multi-rate networks [7] [11]. In SR-SC multi-rate WMNs, WCDS performs creditably against other low-latency broadcast heuristics, because WCDS considers both: the multi-rate nature of the network and the WBA of the underlying wireless medium. The MWT, like WCDS, is a greedy heuristic algorithm that decides the '*best*' transmission in each round, from a set of eligible transmissions. However, as we shall see, *MWT does not consider the availability of multiple interfaces on each node, and thus fails to exploit the potential advantage of parallel transmissions at any intermediate node.* The objective of the MWT algorithm is explained below.

Let us assume that the input topology, as shown in the beginning of this section (Sec. 7), is provided as input to our algorithm. MWT attempts to find a set of transmitting nodes, represented as $Y = \{y_1, y_2, ...\} \subset V$. Let us denote the set of distinct latency transmissions at an arbitrary node $y_i \in Y$ as $\hat{\mathcal{L}}_{yi} \subseteq \mathcal{L}$. Further assume that these transmissions take place on the channels $\hat{\mathcal{C}}_{yi}$. The set Y is constructed such that:

1) Every element of $V \backslash Y$ (all leaf nodes) are covered by a transmission (y_i, l_i, c_i) **s.t.** $y_i \in Y$, $l_i \in \hat{\mathcal{L}}_{yi}$ and $c_i \in \hat{\mathcal{C}}_{yi}$. The term (y_i, l_i, c_i) denotes a transmission by node y_i, with transmission latency l_i, on channel c_i. The leaf node to be 'covered' by (y_i, l_i, c_i) must have an available interface assigned to c_i.

2) The weighted sum $\sum_{(\forall y, \forall i)} \hat{\mathcal{L}}_{yi}$ is minimum.

3) The set Y is connected.

Algorithm 2 MWT construction

Input: $[s, \mathcal{A}, C, G = (V, E, L, \Lambda), \mathcal{L} = \{l_1 \cdots l_k\}]$
$\mathcal{R} \leftarrow \{s\}$
while $(V \backslash \mathcal{R} \neq \emptyset)$ **do**
$(\hat{n}, \hat{l}, \hat{c}) = \arg\max_{n \in \mathcal{R}, l \in \mathcal{L}, c \in \mathcal{A}(n)} f(n, l, c)$
(**where** $f(n, l, c) = (|N(n, l, c) \backslash \mathcal{R}| \div l)$)
{if multiple $(\hat{n}, \hat{l}, \hat{c})$ with max f, choose whose
$\hat{c}$ is least used in the conflict graph of $(\hat{n}, \hat{l}, \hat{c})$ }
$A \leftarrow N(\hat{n}, \hat{l}, \hat{c}) \backslash \mathcal{R}$;
$P_{MWT}(A) = \hat{n}$; $L_{MWT}(A) = \hat{l}$; $\Lambda_{MWT}(A) = \hat{c}$;
$\mathcal{R} \leftarrow \mathcal{R} \cup A$
end while
Output: $[P_{MWT}, L_{MWT}, \Lambda_{MWT}]$

7.2.1 Algorithm

The algorithm starts by making the source node s eligible to transmit. This is done by moving s to the set $\mathcal{R}$ which keeps track of the eligible-nodes (nodes that have received the transmission already and are eligible to transmit). We say that a node is covered and is eligible for transmission if it is in the set $\mathcal{R}$. We refer to (n, l, c) as a '*combination*' or as a '*transmission combination*', and define it as the transmission by an eligible node $n \in \mathcal{R}$, with latency $l \in \mathcal{L}$, on channel $c \in \mathcal{A}(n)$. We use the term $N(n, l, c)$ to

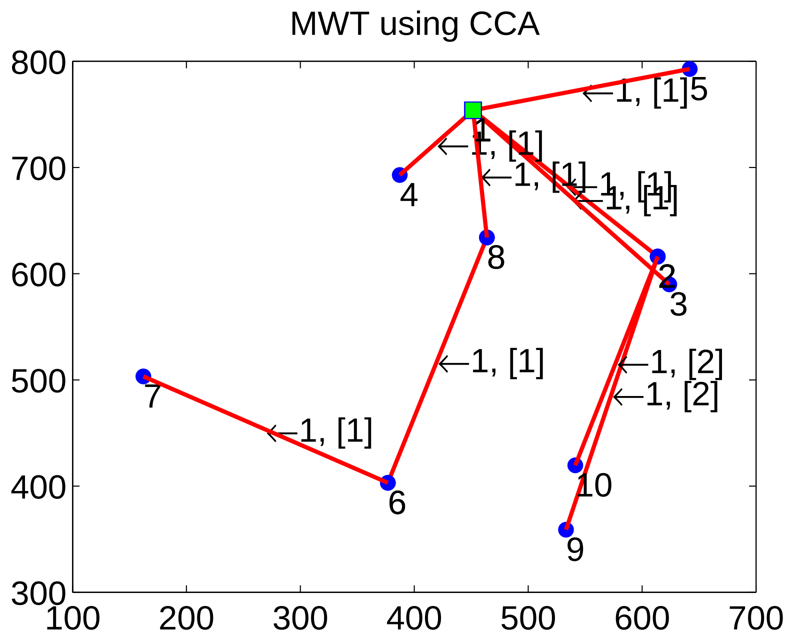


Figure 7: The MWT for the channel-assignment scheme CCA

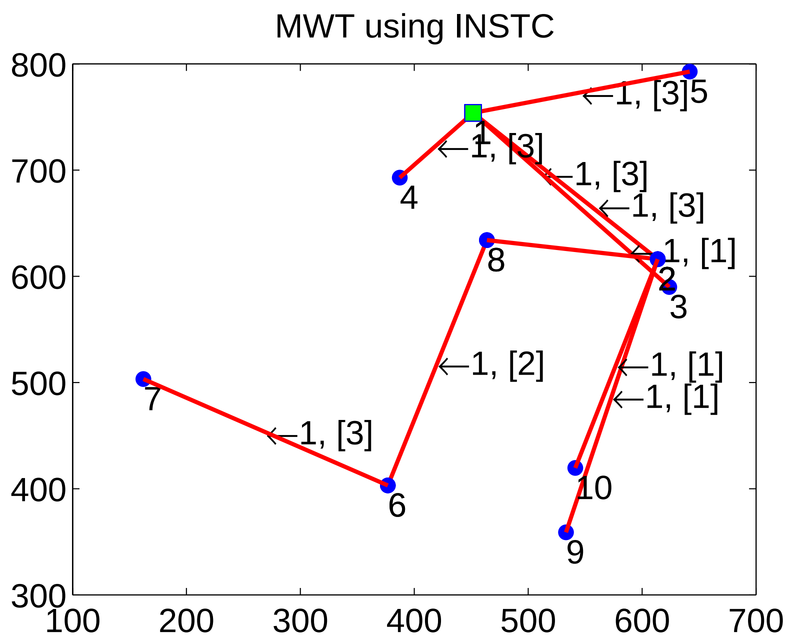


Figure 8: The MWT for the channel-assignment scheme INSTC

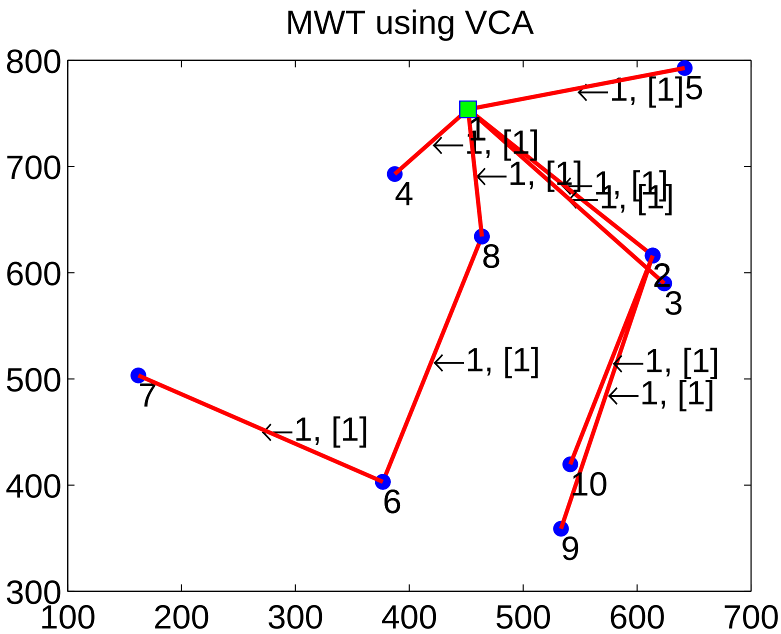


Figure 9: The MWT for the channel-assignment scheme VCA

refer to all neighbors of the n which are reachable by the transmission combination (n, l, c). For any transmission combination (n, l, c)—the quantity $|N(n, l, c)\backslash\mathcal{R}|$ (also represented as A in Algorithm 2) is the number of "not-yet-covered nodes" reachable by this transmission combination.

All eligible combinations $(\forall n \in \mathcal{R}, \forall l \in \mathcal{L}, \forall c \in \mathcal{A}(n))$ are given a 'priority' measure defined as the product of "not-yet-covered nodes" and the rate of transmission i.e. $\frac{1}{l}$, or as $|N(n, l, c)\backslash\mathcal{R}| \div l$. The priority is defined such to reflect the desire to both include as many nodes as possible in a single transmission, yet keep the transmission rate high (even though a higher transmission rate implies a smaller range, and thus, a smaller set of covered nodes).

In each round of the algorithm, the node with maximum 'priority' is selected. In case of multiple combinations (n, l, c) having the same priority, the combination transmitting on the channel $\hat{c}$, which is the least-loaded channel within the conflict graph of the transmission as explained in Section 7.1, is chosen. The algorithm completes its execution when all the nodes have been covered, i.e. when $V\backslash\mathcal{R} = \emptyset$. The algorithm returns the sets P_{MWT}, L_{MWT} and Λ_{MWT}, where $P_{MWT}(v_i)$ is the parent node of v_i, $L_{MWT}(v_i)$ is the latency of the link connecting v_i and $P_{MWT}(v_i)$, and $\lambda_{MWT}(v_i)$ is the channel used on the link connecting v_i and $P_{MWT}(v_i)$, $\forall v_i \in V$. The MWT is now readily constructed using these sets.

7.2.2 MWT for our example network (as shown in Section 6)

We refer to the example network in Section 6 to illustrate the working of the MWT algorithm. The trees constructed by the MWT algorithm for the channel-assignment schemes of CCA, INSTC and VCA are depicted in Figures 7, 8, and 9, respectively.

Referring to the case of MWT using CCA (Figure 7), the choice of the (n, l, c) combination at the end of each successive round is (1,1,1), (2,1,2), (8,1,1), and (6,1,1), respectively. These combinations (n,l,c) are drafted to the tree because their metric $f(n, l, c)$—i.e., 5, 2, 1 and 1 respectively—is the maximum during their respective rounds. The MWT for INSTC and VCA channel-assignment schemes is constructed similarly by adding the highest-priority transmission to the tree at the completion of each round. After the 'transmission scheduling' stage, discussed in Section 9.1, the results obtained for MWT using CCA, INSTC and VCA are 3, 4, and 4, respectively, as shown in Table 2.

7.3 Locally Parallelized, Multi-Radio, Multi-Channel, WCDS Tree (LMT)

The development of LMT algorithm, which we discuss in this section, is motivated by the observation that MWT–while taking into account the WBA and multi-rate nature of the underlying medium–does not '*as readily*' exploit the interface diversity on individual nodes. This observation can be explained more intuitively by noting that MWT is inherently biased, by its priority metric, to include transmissions that cover greater number of uncovered nodes. This metric tends to work well when the number of radio interfaces/channels are small. However, it fails to exploit the increased opportunities for parallel '*faster*' transmissions (on different orthogonal channels) when the number of interfaces are higher.

Accordingly, the LMT algorithm is based on the observation that a node m covered by a transmission combination (n, l, c) may also be covered by combination $(n, \hat{l}, \hat{c})$ where $l > \hat{l}$ and $c \neq \hat{c}$. Thus we may be able to cover node m for free on an orthogonal channel $\hat{c}$ without paying penalty on delay. This is done by considering node m as a covered node of $(n, \hat{l}, \hat{c})$ but not (n, l, c).

Consider as an example, the case of a MR^2-MC *clique* topology where source s can reach all receivers in a single-hop. Let us assume that receivers connect to s on different latencies. The nodes closer to s can be reached with lower latency, whereas nodes further away can only be reached with higher latency. The MWT may prefer a single transmission at s to cover all receivers at the slowest rate (by dint of covering more nodes). This does not exploit the fact that s could have, in *parallel* and on different channels, sent the packet faster to some subset of the nodes, for potentially better latency performance.

Algorithm 3 LMT construction

Input: $[s, \mathcal{A}, C, G = (V, E, L, \Lambda), \mathcal{L} = \{l_1 \cdots l_k\}]$
$\mathcal{R} = \{s\}$
while $(V \backslash \mathcal{R} \neq \emptyset)$ **do**
  $(\hat{n}, \hat{l}, \hat{c}) = \arg\max_{n \in \mathcal{R}, l \in \mathcal{L}, c \in C} f(n, l, c)$
  $\{$**where** $f(n, l, c) = (|N(n, l, c) \backslash \{\mathcal{R} \bigcup RN_{(n,l,c)}\}| \div l)$
  and $RN_{(n,l,c)} = \cup_{\forall (l_i \in \mathcal{L}) < l, \forall (c_i \in (\mathcal{A}(n) \backslash \{c\}))} N(n, l_i, c_i)\}$
  {if multiple $(\hat{n}, \hat{l}, \hat{c})$ with max f, choose whose
  $\hat{c}$ is least used in conflict graph of $(\hat{n}, \hat{l}, \hat{c})\}$
  $N_{covered} = N(\hat{n}, \hat{l}, \hat{c}) \backslash \{\mathcal{R} \cup RN_{(\hat{n}, \hat{l}, \hat{c})}\}$
  $A \leftarrow N_{covered}$;
  $\mathcal{R} \leftarrow \mathcal{R} \cup A$
  $P_{LMT}(A) = \hat{n}$; $L_{LMT}(A) = \hat{l}$; $\Lambda_{LMT}(A) = \hat{c}$
end while
Output: $[P_{LMT}, L_{LMT}, \Lambda_{LMT}]$

7.3.1 Algorithm

The LMT algorithm is identical to MWT, except in the calculation of the priorities of eligible transmissions at each round. In MWT, the '*best*' transmission in any particular round is the transmission (n, l, c) with maximum $f(n, l, c) = (|\text{neigh covered}| \div l)$ where 'neigh covered' is $(N(n, l, c) \backslash \mathcal{R})$. In LMT, the term 'neigh covered' is redefined to be $N(n, l, c) \backslash \{\mathcal{R} \cup RN_{(n,l,c)}\}$ where the set $RN_{(n,l,c)}$ contains all nodes that n can cover in *parallel*, at a lower latency than l, on a channel different than c of the (n, l, c) combination.

The nodes covered in each round are added to $\mathcal{R}$, which contains nodes eligible to transmit during the next round. Unlike MWT, where all non-covered neighboring nodes $N(\hat{n}, \hat{l}, \hat{c}) \backslash \mathcal{R}$ of the chosen transmission $(\hat{n}, \hat{l}, \hat{c})$ are added to $\mathcal{R}$; in LMT, only the nodes in $N(\hat{n}, \hat{l}, \hat{c}) \backslash \{\mathcal{R} \cup RN_{(\hat{n}, \hat{l}, \hat{c})}\}$ are added.

The algorithm completes its execution when all the nodes have been covered, i.e. when $V \backslash \mathcal{R} = \emptyset$. The algorithm returns the sets P_{LMT}, L_{LMT} and Λ_{LMT}, where $P_{LMT}(v_i)$ is the parent node of v_i, $L_{LMT}(v_i)$ is the latency of the link connecting v_i and $P_{LMT}(v_i)$, and $\Lambda_{LMT}(v_i)$ is the channel used on the link connecting v_i and $P_{LMT}(v_i)$, $\forall v_i \in V$. LMT can now be readily constructed from these sets.

7.3.2 LMT for our example network (as shown in Section 6)

We refer to the example network in Section 6 to illustrate the working of the LMT algorithm. The trees constructed by the LMT algorithm for the channel-assignment schemes of CCA, INSTC and VCA are depicted in Figures 10, 11, and 12, respectively.

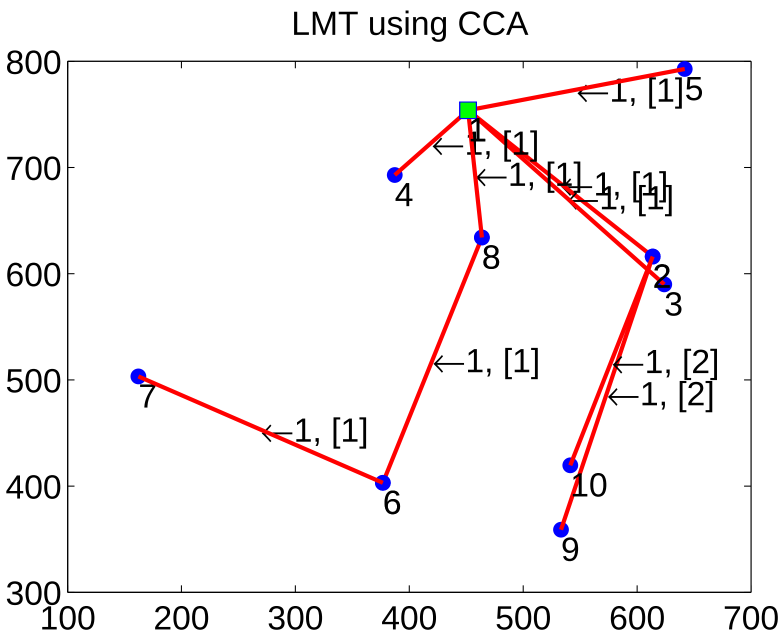


Figure 10: The LMT for the channel-assignment scheme CCA

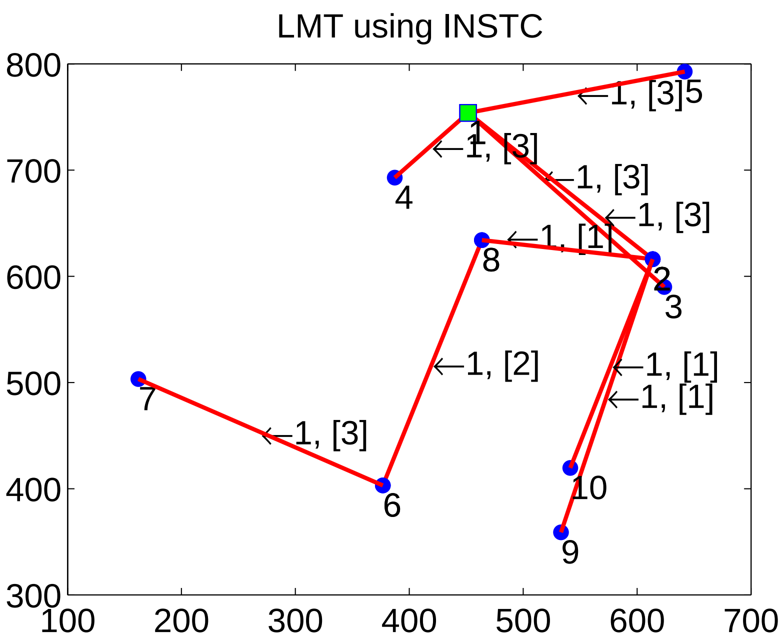


Figure 11: The LMT for the channel-assignment scheme INSTC

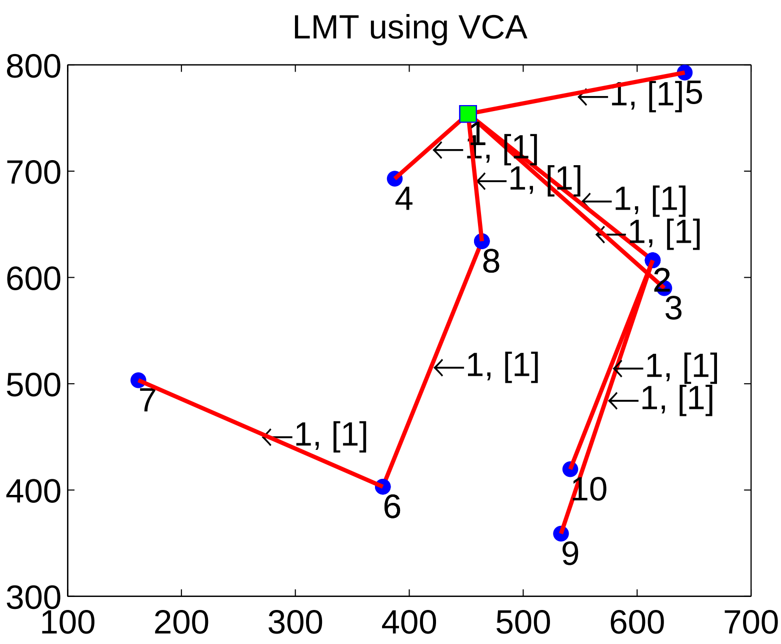


Figure 12: The LMT for the channel-assignment scheme VCA

For ease of exposition, we have intentionally chosen a very small network i.e., a network of only 10 nodes. In a network of this size, the opportunities to *parallelize* transmissions are limited. The trees constructed using LMT algorithm for CCA, INSTC and VCA channel-assignment schemes are identical to those constructed using the MWT algorithm for these schemes; as in our example scenario, the transmissions (included in the MWT) are already at the 'quickest' rates, and *parallelizing* to '*quicker*' rates on alternative channel is not possible. After the 'transmission scheduling', discussed in Section 9.1, the results obtained for LMT using CCA, INSTC and VCA are 3, 4, and 4, respectively, as shown in Table 2.

7.4 Parallelized, Approximate-Shortest, Multi-Radio, Multi-Channel, WCDS Tree (PAMT)

The PAMT algorithm, like the LMT algorithm, is adapted from the MWT algorithm, and is designed to be *adaptive* to number of radio interfaces and channels available. The PAMT algorithm is intended as an improvement over the LMT algorithm. The LMT algorithm, during any particular round, might decide to cover some nodes with a transmission that has a longer latency path to s (the source node) compared to other eligible transmissions (*by currently unused interfaces on other intermediate nodes*) that can possibly take place on an alternative, non-interfering channel in '*parallel*'. Such

Algorithm 4 PAMT construction

Input: $[s, \mathcal{A}, C, G = (V, E, L, \Lambda), \mathcal{L} = \{l_1 \cdots l_k\}]$
$\mathcal{R} = \{s\}$; $label(s) = 0$
while $(V \backslash \mathcal{R} \neq \emptyset)$ **do**
  $(\hat{n}, \hat{l}, \hat{c}) = \arg\max_{n \in \mathcal{R}, l \in \mathcal{L}, c \in \mathcal{A}(n)} f(n, l, c)$
  {if multiple $(\hat{n}, \hat{l}, \hat{c})$ with max f, choose whose
  $\hat{c}$ is least used in conflict graph of $(\hat{n}, \hat{l}, \hat{c})$**}**

---

  where $f(n, l, c)$ **is calculated as:**
  $X = Y_{(n,l,c)} = N(n, l, c) \backslash \mathcal{R}$
  $label_{trans} = label(n) + l$;
  if $X \neq \emptyset$ **then**
   $nodes_{tmp} = \cup_{(\forall c_{tmp} \in \mathcal{A}(n) \backslash \{c\}, \forall l \in \mathcal{L})} N(n, c_{tmp}, l)$
   $nodes_p = nodes_{tmp} \cap \mathcal{R}$
   for $x = 1$ to $|X|$ **do**
    for $y = 1$ to $|nodes_p|$ **do**
     $latency_{node}(y) = l(nodes_p(y), X(x))$
     $label_{node}(y) = label(nodes_p(y))$
     $label_{round}(y) = latency_{node}(y) + label_{node}(y)$
     if $label_{round}(y) < label_{trans}$ **then**
      $Y_{(n,l,c)} = Y_{(n,l,c)} \backslash \{X(x)\}$; **break**
     end if
    end for
   end for
  end if
  $X = Y_{(n,l,c)}$
  $f(n, l, c) = |X| \div l$

  $A \leftarrow Y_{(\hat{n}, \hat{l}, \hat{c})}$
  $\mathcal{R} \leftarrow \mathcal{R} \cup A$
  $label(A) = label(\hat{n}) + \hat{l}$
  $P_{PAMT}(A) = \hat{n}$; $L_{PAMT}(A) = \hat{l}$; $\Lambda_{PAMT}(A) = \hat{c}$
end while
Output: $[P_{PAMT}, L_{PAMT}, \Lambda_{PAMT}]$

a decision is possible despite the fact that in LMT, nodes always attempt to use '*fastest*' possible transmitting rates to connect to its neighbors. The following simple example illustrates this idea.

First of all, let us define as the total cost (latency) of the path from a node n to source s as *label* of n. Let us assume that node n can reach a set of nodes Y by transmitting on channel c with latency l_1. The *labels* of all nodes in Y would then be $label(n) + l_1$. Let us assume further that $Y' \subset Y$ can also be covered by a transmission of some other node n' (assume $label(n')$ $< label(n)$) on channel c', with same latency l_1. If covered by transmission of n', nodes in $Y' \subset Y$ have a label of $label(n') + l_1$. Since $Y' \subset Y$, LMT would prefer the transmission of n to that of n' (as it covers more nodes)

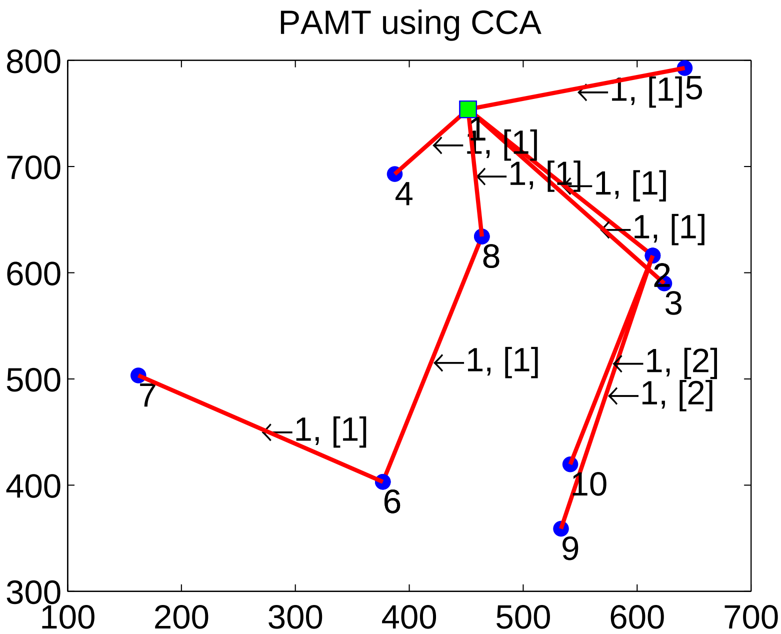


Figure 13: The PAMT for the channel-assignment scheme CCA

and therefore would cover all the nodes in Y with n's transmission; this is despite the fact that nodes in $Y' \subset Y$ can be covered with a smaller path cost to s, if n' transmits in parallel on an alternative channel c'.

7.4.1 Algorithm

The PAMT algorithm is also adapted from the MWT algorithm, like the LMT algorithm. PAMT works in a greedy manner, similar to the method of MWT and LMT, to choose the 'best' transmission in each round. The priority metric $f(n, l, c)$ for each transmission (n, l, c), however is calculated differently for PAMT. The PAMT algorithm maintains an extra parameter called *label* for each node, denoting the cost of its path to s (source node). The algorithm begins by adding node s to $\mathcal{R}$, which is the set of nodes that are eligible to transmit during the next-round. The *label* of s is set to 0, and the *label* for all other nodes is set to ∞. During the execution of each round, PAMT tries to find out which transmission (or edge(s)) should be added to the tree. The set $Y_{(n,l,c)} = N(n, l, c) \backslash \mathcal{R}$ contains all hitherto 'uncovered nodes' that can be covered by this transmission (n, l, c). The label of this transmission denoted by $label_{trans}$ is equal to $label(n) + l$.

During the calculation of priority for each transmission (n, l, c), X contains the neighboring nodes $Y_{(n,l,c)}$ of the transmission (n, l, c). For each node in X, neighboring nodes are searched ($nodes_p$ in Algorithm 4) to find

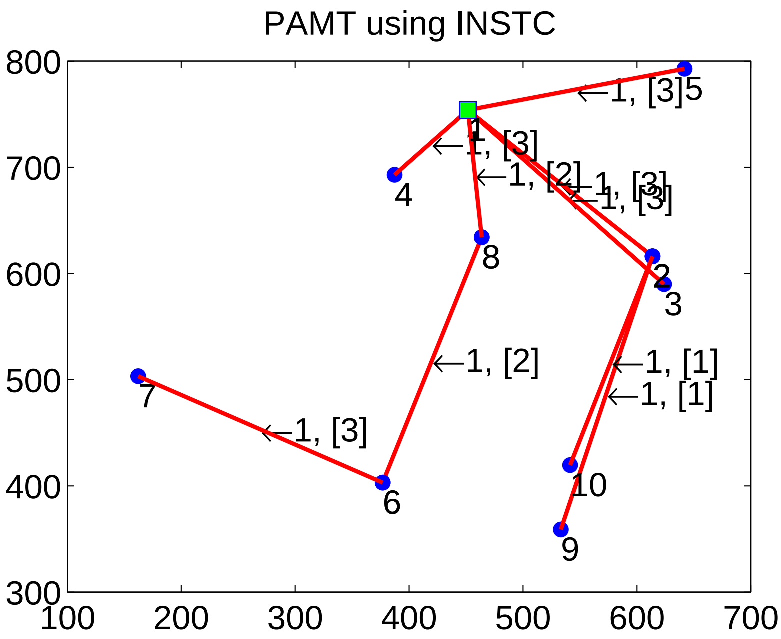


Figure 14: The PAMT for the channel-assignment scheme INSTC

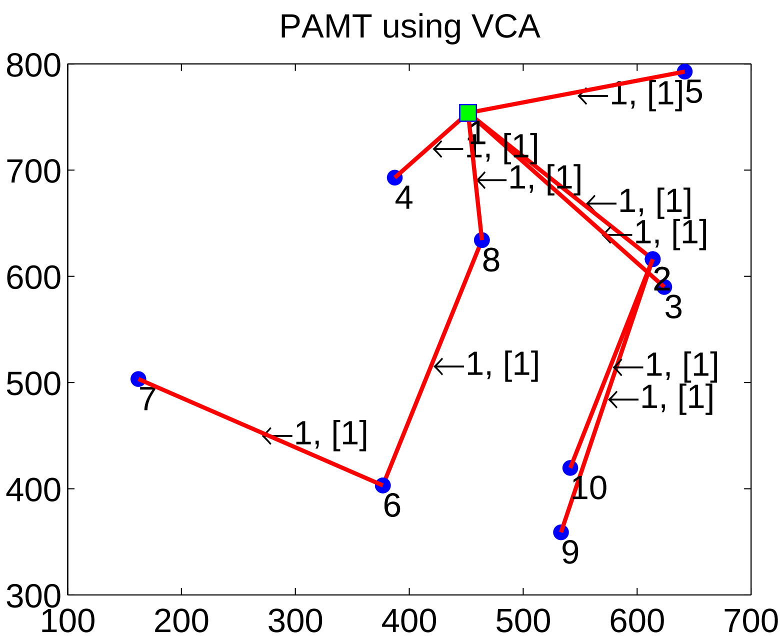


Figure 15: The PAMT for the channel-assignment scheme VCA

out if they can offer a lower-cost path to s, on an *alternative* channel to c. If such a path is found, then this node *should* not be covered in the transmission (n, l, c). This node, therefore, is not considered a covered-node of (n, l, c) and is deleted from $Y_{(n,l,c)}$. After all nodes in X are checked in a similar manner, $Y_{(n,l,c)}$ contains the actual number of nodes that will be covered by the transmission (n, l, c). The priority of the transmission (n, l, c) is then calculated by dividing $Y_{(n,l,c)}$ by l.

In case of multiple transmissions having the same priority, the transmission whose channel $\hat{c}$ is least-used in the conflict graph of that transmission, is chosen. After completion of each round, covered-nodes are added to $\mathcal{R}$. The algorithm completes its execution when all the nodes have been covered, i.e. when $V \backslash \mathcal{R} = \emptyset$. The algorithm returns the sets P_{PAMT}, L_{PAMT} and Λ_{PAMT}, where $P_{PAMT}(v_i)$ is the parent node of v_i, $L_{PAMT}(v_i)$ is the latency of the link connecting v_i and $P_{PAMT}(v_i)$, and $\Lambda_{PAMT}(v_i)$ is the channel used on the link connecting v_i and $P_{PAMT}(v_i)$, $\forall v_i \in V$. The PAMT is constructed from these sets.

7.4.2 Remarks

It can be shown that the method of LMT of not considering a node as a covered-node of combination (n, l, c), if a higher-rate transmission (n, l', c') of n with $l' < l$ and $c' \in \mathcal{A}(n) \backslash \{c\}$ can cover it, is a special case of PAMT. In PAMT, a node is not considered a covered-node of combination (n, l, c), if there exists an *eligible* transmission (n', l', c') on an *alternative* channel c' with latency l', using which would result in a shorter *label* for the covered node. Due to the fact that higher-rate transmissions of the same node have lower-latency, another transmission on a higher-rate on an alternative channel would always result in a lower-label. Therefore, PAMT is more general than LMT.

7.4.3 PAMT for our example network (as shown in Section 6)

We refer to the example network in Section 6 to illustrate the working of the PAMT algorithm. The trees constructed by the PAMT algorithm for the channel-assignment schemes of CCA, INSTC and VCA are depicted in Figures 13, 14, and 15, respectively.

The PAMT for CCA scheme is identical to MWT for CCA scheme, as the chosen transmissions in the tree already are the '*quickest*' and the reached nodes have the shortest paths to the source-node. However, the PAMT for INSTC scheme is different to the MWT and LMT for INSTC scheme; this is because node 8 can be reached by a transmission by the source-node on an alternate-channel (i.e., on channel 2, the channel used earlier by the source-node was 3). The choice of the (n, l, c) combination at the end of each successive round, for PAMT using INSTC, is (1,1,3), (2,1,1), (1,1,2), (8,1,2) and (6,1,3) in the order of their addition. These combinations (n, l, c) are drafted to the tree because the metric $f(n, l, c)$ for these combinations (i.e. 4, 2, 1, 1 and 1 respectively) is the maximum at the end of their respective round. This improves the broadcast latency performance of PAMT for INSTC—from 4 to 3, as shown in Table 2. The results for PAMT using CCA and VCA (identical to MWT and LMT results) are 3, and 4, respectively, as shown in Table 2.

8 Multicast grouping

The output of the 'topology construction' stage is a directed broadcast tree. The non-leaf nodes of the tree represent the transmitting nodes. Any non-leaf node can have possibly multiple outgoing edges with different weights. This translates in a 'physical' sense into multiple link layer multicasts. These link-layer multicasts with *different* transmission rates can take place simultaneously, *only and only if*, these transmissions take place on orthogonal channels (of-course at any node, multiple outgoing edges having same latency weight corresponds to a single transmission due to WBA). *In the case where different-latency transmissions are on the same channel*, a decision has to be made to either retain or discard the lower latency transmission(s). The function of the 'multicast grouping' stage is to make this very decision.

The decision is made, while keeping in mind that a '*slower*' transmission has a wider '*reach*' and vice versa. This implies that the '*slowest*' transmission can cover all neighboring nodes, albeit at the cost of increased latency. This trade-off has earlier been studied for the case of SR-SC multi-rate WMNs in our earlier work [11]. In the case of SR-SC multi-rate WMNs, all transmissions take place *on the same channel*, requiring grouping decisions whenever there are multiple *different-latency* transmissions on a node. For the case of MR^2-MC WMN, a grouping decision needs to be made when the different-latency transmissions are on *the same channel*; transmissions on different channels can take place simultaneously. The multicast grouping algorithm, that we use for MR^2-MC is identical to the grouping algorithm described in [11], for the case of SR-SC multi-rate WMN, which we include here for completeness. The only difference is that, in MR^2-MC WMN, the grouping algorithm is only invoked for the case of different-latency transmissions *on the same channel* at a node (and *not on a different orthogonal channel*).

In order to find the topology which minimizes the broadcast latency, we must make a number of decisions, including which node is to multicast, and if so, how many times it is to multicast, whom the recipients are and its timing. As stated earlier, the result of the topology construction is a broadcast tree which specifies that the non-leaf nodes of the broadcast tree will multicast to its child nodes, in possibly multiple transmissions. However, the number of times a transmitting node (i.e. non-leaf node of the broadcast tree) will multicast and the recipients of each multicast still have not been decided. In case where a node multicasts only once, then the recipients will be all its child nodes. For the case where a node is to multicast more than once, a different subset of child nodes will be reached in each multicast such that these subsets together form a partition of the set of child nodes. The aim of the multicast grouping stage is to determine the number of multicasts to be made and their recipients.

We begin by defining the concept of valid transmission sequence at a transmitting (i.e. non-leaf) node of the broadcast tree. Consider for example a transmitting node n which has two child nodes c_1 and c_2, which can be reached using a minimum latency of $d_1(=1)$ and $d_2(=2)$ time units respectively. Node n can reach these nodes in a number of valid transmission sequences. For example, it can first multicast to c_1 (with latency 1) followed by another multicast to c_2 (with latency 2). We will denote this valid transmission sequence as (d_1, d_2). An alternative valid transmission sequence for node n is (d_2) which reaches both nodes in one multicast. These two are the only two valid transmission sequences for this example. The sequence (d_1) is invalid because it does not reach all the child nodes. In addition, (d_2, d_1) is invalid because the second transmission is unnecessary since both nodes are already reached by transmission d_2 whose coverage area is greater. In general, consider a transmitting node n which has m child nodes $c_1, \cdots, c_m$ that are reachable using minimum latency of $d_1, \cdots, d_m$ respectively. Let k denote the number of distinct latencies in $d_1, \cdots, d_m$ and let us denote these distinct latencies as $\mathcal{L} = l_1, \cdots, l_k$. Without loss of generality, we assume that $l_k \geq \cdots \geq l_1$. A valid transmission sequence is a r-tuple $(1 \leq r \leq k)$ whose entries are drawn from $\mathcal{L}$ such that

1. Each latency in $\mathcal{L}$ appears in the r-tuple at most once.
2. The latencies in the r-tuple appear in a strictly increasing order.
3. The last entry of the r-tuple must be l_k.

Let $T_V(n)$ be the set of all valid transmission sequences for node n. Since node n uses k distinct rates to reach its child nodes, T_V contains 2^{k-1} valid transmission sequences.

Since our goal is to minimize the broadcast latency, we are interested to find the valid transmission sequence at all the transmitting nodes such that they together will minimize the broadcast latency. For ease of reference, we will refer to the optimal valid transmission sequence at a transmitting node as the '*Cardinal Sequence*' (CS). Also, if a transmitting node n and all its descendants use their cardinal sequences for transmission, the delay it takes a packet to reach all ns descendants will be called node ns '*Cardinal Value*'

(CV). The aim of the multicast grouping stage is to find the CS and CV at each transmitting node of the network,

Since the choice of CS and CV at a transmitting node n depends on the CSs and CVs of all the transmitting nodes who are descendants of n, the grouping algorithm should proceed from the leaf nodes of the broadcast tree back to the root. For the rest of the description, we will show how the CS and CV of an arbitrary transmitting node n can be determined. We assume that the CSs and CVs of all the transmitting nodes who are descendants of n are already known. Also, for initialization, we define the CV of all leaf nodes to be zero.

Let us assume that node n uses k distinct transmission rates to reach its child nodes, then the set of all valid transmission sequences at node n, denoted by $T_V(n)$, has 2^{k-1} valid transmission sequences $S_q(1 \leq q \leq 2^{k-1})$. The CS at node n is determined by comparing the broadcast latency achieved by all possible $S_q \in T_V$ and then choosing the S_q with the least broadcast latency as the CS. The CV of the node is then the latency associated with the chosen CS. If node n uses the transmission sequence S_q, let $D(n)_{Sq}$ denote the resulting latency required to reach all the descendants of n, we can formally define CS and CV of node n as

$$CS(n) = \arg \min_{Sq \in T_V} (D(n)_{Sq}) \tag{1}$$

$$CV(n) = \min_{Sq \in T_V} (D(n)_{Sq}) \tag{2}$$

We will now detail how $D(n)_{Sq}$ can be computed. Let S_q be the r-tuple $(S_{q,1}, \cdots, S_{q,x}, \cdots, S_{q,r})$. Since the coverage area of a higher latency transmission is larger, thus with the transmission sequence S_q, some of the child nodes of n will receive the same packet multiple times. In particular, let $N(n)_{Sq,x}$ denote the child nodes of n that are reachable by a multicast of latency $S_{q,x}$ but are not reachable by $S_{q,x-1}$. In other words, the nodes in $N(n)_{Sq,x}$ receives their packets from n for the first time via a multicast of latency $S_{q,x}$ and will receive the same packet a total of $(r - x + 1)$ times. Note also that the sets $N(n)_{Sq,x}(x = 1, \cdots, r)$ effectively partition the child nodes of n into r disjoint subsets. Let $D(n)_{Sq,x}$ denote the delay it takes n to reach all the nodes in the set $N(n)_{Sq,x}$ and their descendants. Assuming that the transmission of the descendants of $N(n)_{Sq,x}$ do not interfere with each other, we have

$$D(n)_{Sq} = \max_{1 \leq x \leq r} D(n)_{Sq,x} \tag{3}$$

As mentioned a number of times before, the decisions we need to make are highly coupled. Thus, by ignoring the inter-branch interference, we obtain an approximation which makes the problem tractable. The inter-branch interference will be taken into account in the scheduling stage in Section 9.

We propose to compute $D(n)_{Sq,x}$ using the following formula:

$$D(n)_{Sq,x} = \sum_{i=1}^{x} S_{q,i} + \max_{i \in N(n)_{Sq,x}} CV(i) + \sum_{i=1}^{x-1} SCDelay(S_{q,i}) \tag{4}$$

This equation is obtained by assuming the following *modus operandi*: Node n first transmits at latency $S_{q,1}$ reaching the nodes in $N(n)_{Sq,1}$. If some of the nodes in $N(n)_{Sq,1}$ are transmitting nodes, they will then begin their transmission to their respective downstream neighbors in parallel. (Note that we are again ignoring inter-branch interference). Note that node n does not begin transmitting at latency $S_{q,2}$ immediately after finishing transmitting at $S_{q,1}$. We assume that node n waits until all the transmissions from $N(n)_{Sq,1}$ and their descendants have proceeded sufficiently so that the $S_{q,2}$-transmission of node n does not interfere with those of $N(n)_{Sq,1}$ and their descendants. This operation then repeats itself until all transmissions in S_q have been made.

With this *modus operandi* in mind, we can now explain how Equation 4 comes about. We begin with the case for $x = 1$ where we have $D(n)_{Sq,1} = S_{q,1} + max_{i \in N(n)_{Sq,1}} CV(i)$. Recall that $D(n)_{Sq,1}$ is the delay it takes to reach all the nodes in $N(n)_{Sq,1}$ and their descendants. The first term $S_{q,1}$ is simply the time it takes to reach the nodes in $N(n)_{Sq,1}$. After the packets have been received by the nodes in $N(n)_{Sq,1}$, we assume that the transmissions by the nodes in $N(n)_{Sq,1}$ will proceed in parallel, so the maximum time it takes all these transmissions to reach the end of their branches is given by the second term. Note that this follows from our definition of CV.

We now explain the derivation of Equation 4 for $x > 1$. The first two terms of the equation bear similar meaning to what is explained in the last paragraph, so we will focus on the third term only. Recall from our description of the *modus operandi* that the $S_{q,x}$-transmission of node n will only begin after the downstream transmissions caused by the $S_{q,x-1}$-transmission have proceeded sufficiently. The time gap between these two transmissions by node n is $SCDelay(S_{q,x-1})$. Here the prefix SC stands for single-channel as this delay is caused by the fact that all these transmissions are taking place on a *single-channel*.

Recall that the time separation $SCDelay(S_{q,x-1})$ is needed so that the $S_{q,x}$-transmission is not interfered by the transmissions by the nodes in $N(n)_{Sq,x-1}$ and their descendants. In order to compute $SCDelay(S_{q,x-1})$, we will first need to identify those transmissions which may interfere with the reception of the nodes in $N(n)_{Sq,x}$. Let $\mathcal{T}_{Sq,x-1}$ be all transmitting node in $N(n)_{Sq,x-1}$. Let $\hat{t} \in \mathcal{T}_{Sq,x-1}$, the set $N(\hat{t})$ consists of all nodes $\hat{n}$ with the following properties (1) $\hat{n}$ is a descendant of $\hat{t}$. (2) The transmission of the parent of $\hat{n}$ interferes with the reception of nodes in $S_{q,x}$. (3) Either $\hat{n}$ is a leaf node or the transmission of $\hat{n}$ and its descendants do not interfere with the reception of nodes $S_{q,x}$. In other words, the transmissions in $N(\hat{t})$ are the first ones that do not interfere with the $S_{q,x}$-reception. Thus, we have

$$SCDelay(S_{q,x-1}) = \max_{\hat{t} \in \mathcal{T}_{Sq,x-1}} (CV(\hat{t} - \min_{\hat{n} \in \mathcal{N}(\hat{t})} CV(\hat{n}))) \quad (5)$$

The term in parenthesis in the above equation essentially estimates the time it takes the transmissions due to $\hat{t}$ and its descendants to clear the interference range of the nodes in $N(n)_{Sq,x}$.

Having looked at how $D(n)_{Sq,x}$ (Equation 4) was obtained, we can see how Equations 1, 2, 3, and 4 can be used together to obtain the CV at a transmitting node. This process can be performed recursively starting from the leaf nodes back to the root of the broadcast tree.

In addition to deciding on the transmission sequence at each transmitting node, the results of the above computation will also be helpful in deciding the timing of the transmissions in the scheduling stage. Recall that the CV of a transmitting node n can be interpreted as the time required to reach all the descendants of node n. Thus, when it comes to scheduling all the multicast transmissions that are to be made, we can use the analogous concept of the CV of a transmission as a measure of the urgency of the transmission. If S_{qx} is a transmission within the CS of node n (i.e. S_{qx} is a chosen transmission), then the CV of S_{qx} is in fact given by Equation 4.

9 Transmission scheduling

The '*transmission scheduling*' algorithm tries to schedule the transmissions to minimize the broadcast delay whilst ensuring that interfering transmissions are not scheduled together for simultaneous transmission. Our 'transmission scheduling' algorithm is very similar to the scheduling algorithm for the case of SR-SC multi-rate WMNs presented in [7] [11]. We modify the algorithm presented in [7] [11] according to the interference model—described in Section 3—used in our current work. These modifications are required to ensure that transmissions on orthogonal channels can be scheduled together.

The broadcast tree generated after the multicast grouping phase can be modeled by a directed tree $T = (V, E, L, \Lambda)$. The transmitting nodes are represented by branching vertices (i.e. non-leaf nodes) in the tree T. Let us denote the number of transmitting nodes in the network by k, the set of transmitting nodes denoted by $V_b = \{b_1, b_2 \cdots b_k\}$. Let us denote the set of w different-latency transmissions at any arbitrary node b_i by $\hat{\mathcal{L}}_{bi} = \{b_{i1} \cdots b_{iw}\}$. The set B contains all the transmissions in the network, $B = \{b_{ij}\}, \forall b_i \in V_b, \forall j \in \hat{\mathcal{L}}_{b_i}$. We model the interference between transmissions in an MR^2-MC WMN by using a conflict graph for each channel. The conflict graph $G_{ci} = (B, E_{ci})$ models the interference, *on channel* $i \in C$, between the set of transmissions B. The set of conflicting edges E_{ci} contains an edge (b_{ij}, b_{kl}) *only and only if* both transmissions are on the same channel i, *and* the

Algorithm 5 Transmission scheduling with multiple radios and channels

Input: $\forall b_{ij} \in B$ **(all** j **trans. at transmitting node** $b_i(1 \leq i \leq k)$
Input: $\lambda(b_{ij})$ **(channel** b_{ij} **transmits at** ch_i**)**
Input: $l(b_{ij})$
Set $time = 0$
Initialize $E \leftarrow \cup_{\forall j}\{b_{sj}\}$
Initialize $T = \emptyset$
while $(E \neq \emptyset$ or $T \neq \emptyset)$ **do**
  while $E \neq \emptyset$ **do**
    $b_{ij} = \arg\max b_E.CV \ (\forall b_E \in E)$
    $E = \{E \backslash b_{ij}\}$
    $p = |$transmissions of $b_i \in T|$
    if $(b_{ij}, b_T) \notin E_c(\lambda(b_{ij}))$ in $G_c(\lambda(b_{ij}) \ \forall b_T \in T$ **then**
      if $p <$ Radios **then**
        $T \leftarrow \{T \cup b_{ij}\}$;
        Set $\tau(b_{ij}) = time$
        Set $\delta((b_{ij}) = time + l(b_{ij})$
      else
        $E_{Next} \leftarrow b_{ij}$
      end if
    end if
  end while
  $NextStop$= min $\delta(b_T) \ \forall b_T \in T$
  $NextTrans$= $\{b_N\} : (\forall b_N \ \delta(b_N) = NextStop)$
  $E \leftarrow E \cup b_{children}$ of $NextTrans$
  $T = T - NextTrans$
  $E = E - NextTrans$
  $E = E \cup E_{Next}$
  $time \leftarrow NextStop$
end while
Output: $\tau(b), \delta(b) \ \forall_{(1 \leq b \leq |b|)}$

transmitter of b_{ij} interferes with the receivers of the transmission b_{kl} or vice versa.

Formally, the transmission schedule is the mapping $\tau : b_{ij} \rightarrow \mathbb{R}, \forall b_{ij} \in B$ which gives the starting time of b_{ij}. The transmission schedule must obey the following constraints:

1. The source node s must transmit at least once at time zero.

2. All nodes must follow precedence constraint, i.e a node can only transmit after receiving the packet from its parent.

3. Two arbitrary transmissions b_{ij} and b_{kl} can be scheduled together on an arbitrary channel i, *only and only if* the edge $(b_{ij}, b_{kl}) \notin G_{ci}$.

4. At most one transmission can take place at a time on one interface of any node.

Algorithm: The algorithm for transmission scheduling in MR^2-MC WMNs is mathematically described in Algorithm 5. The set of transmissions $B = \{b_{ij}\}, \forall b_i \in V_b, \forall j \in \hat{\mathcal{L}}_{b_i}$, the channel used $\lambda(b_{ij})$ by $\forall b_{ij} \in B$, and the latency $l(b_{ij})$ of $\forall b_{ij} \in B$, is given as the input to our scheduling algorithm. The current time *time* is initialized to zero, and the set E, containing eligible-transmissions, is initialized with different-latency transmissions of the source-node s. The set T, containing ongoing transmissions, is initialized as an empty set.

The algorithm then finds amongst all eligible transmissions—depicted as $\forall b_E \in E$ in Algorithm 5)—the transmission with the maximum '*Cardinal Value*' (CV). The CV of a node is defined as the worst-case '*latency distance*' to any of its downstream nodes. Our scheduling algorithm gives priority to transmissions which are more 'critical' or have higher CV values. The transmission with the maximum CV—let us denote this transmission by b_{ij}—is then deleted from the set of eligible transmissions E. It is then confirmed that the selected transmission b_{ij} does not interfere with any ongoing transmission, represented as b_T, on the channel $\lambda(b_{ij})$ used by b_{ij}. The number of ongoing transmissions p of the node transmitting b_{ij} (i.e. b_i) is then determined. If p is less than the number of radio interfaces Q, then b_{ij} is added to the set of transmissions taking place and its starting time $\tau(b_{ij})$ is decided as the current time *time*. The ending time of transmission b_{ij} is also decided as $time + l(b_{ij})$. However, if p is more than Q, it implies that node b_i has no free interface and all its interfaces are busy in transmitting. The transmission b_{ij}, therefore, has to be held-back until the next-round; the transmission b_{ij} is added to E_Next which is the set of eligible-transmissions for next-round.

Thereafer, *NextStop* is calculated as the earliest finishing time of any transmission in T. The transmission(s) *NextTrans* have the earliest finishing time of all transmissions in T. The transmissions enabled by the transmissions $NextTrans$ and the transmissions held-back during the current-round E_{Next}, are now added to E, as these transmissions are eligible for next-round. The transmission(s) $NextTrans$ are then deleted from T and E. The round finishes by adjusting the *time* to $NextStop$. The rounds continue until all transmissions have been scheduled and the start-time of each transmission has been calculated.

9.1 Transmission Scheduling for our example network (as shown in Section 6)

We refer to the example network in Section 6 to illustrate the working of our 'transmission scheduling' algorithm. The output of transmission scheduling for the CCA channel-assignment scheme is depicted in Figures 16(a), 16(b), 16(c) and 16(d) for the MSPT, MWT, LMT and PAMT trees, respectively. The node ID of the transmitting nodes is depicted on the vertical axis, while time is shown on the vertical axis. The red horizontal lines depict the time spent by a node transmitting, while the channel of transmission is depicted in blue *on* this horizontal line. The '*children nodes*' reached are shown *below* the line in black (or above the line in certain cases for readability).

Referring to the Figure 16(a), which contains the MSPT for CCA (Figure 4), we examine how the transmissions are scheduled. The source-node 1 starts with two transmissions, with latency 1 and 2, on channel 2 and 1, respectively. The nodes reached by the transmission (or the children nodes) with latency 1, are 2,3,4,5 and 8. Node 8 then starts transmitting in parallel with the the ongoing-transmission (with latency 2) of node 1. It should be noted that, at any given time, interfering transmissions cannot co-exist on the *same* channel. All transmissions interfere with each other for our example network due to its small size. Therefore, for any given channel, only a single transmission can take place at one time. The MSPT for CCA finishes transmitting to all nodes with a broadcast latency of 4.

We will now discuss the transmission scheduling for the MWT, LMT and PAMT tress, with INSTC as the channel-assignment. These trees are shown in Figures 8, 11, and 14, and their transmission schedules are shown in Figures 17(b), 17(c), and 17(d), respectively. For the MWT, the source-node starts with a transmission with latency 1, on channel 3, and reaches nodes 2, 3, 4 and 5. The node 2 then transmits at time 1, on channel 1 with latency 1, to reach nodes 8, 9 and 10. The nodes 8 and 6 follow with transmissions of latency 1 on channels 2 and 3, at time 2 and 3, respectively. The broadcast latency of the MWT using INSTC is 4 units. The scheduling of the LMT is identical to MWT's, since both trees are identical. The PAMT for INSTC, however, improves performance by *parallelizing* the transmissions of node 1 with latency 1, on channels 2 and 3. The node 8, rather than being covered by the transmission of node 2 as was the case in MWT and LMT, is now

Heuristic	CCA	INSTC	VCA
MSPT	4	4	7
MWT	3	4	4
LMT	3	4	4
PAMT	3	3	4

Table 2: Performance of the heuristics for the example topology in Sec. 6

covered by a transmission by the source-node. The node 8 can now start transmitting at time 1, and enable the transmission at node 6 to start and complete its transmission at time 2 and 3, respectively. This improves the performance of both MWT and LMT.

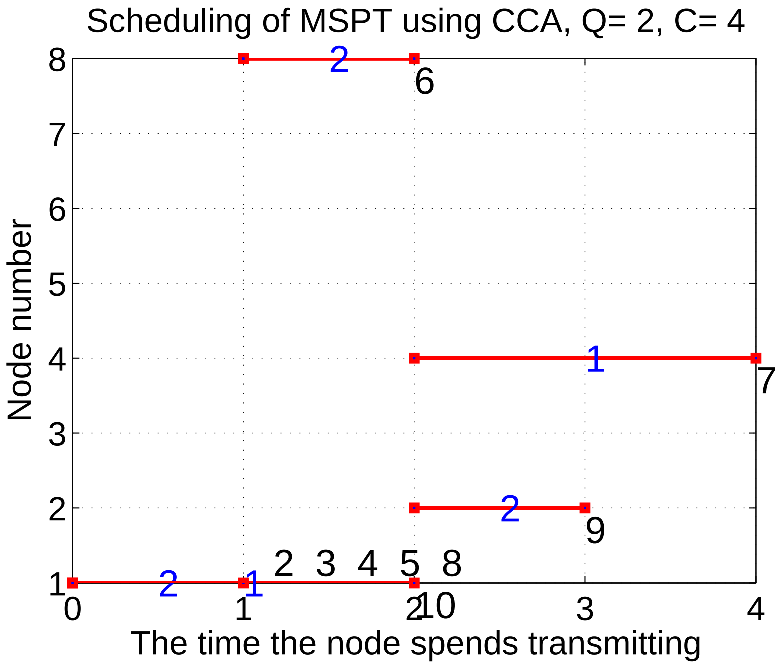
The scheduling for other trees and channel-assignments schemes is done similarly, and is shown in Figures 16(a) to 18(d). The broadcast latency of the trees and channel-assignment schemes are shown in Table 2.

10 Simulated Performance Evaluation

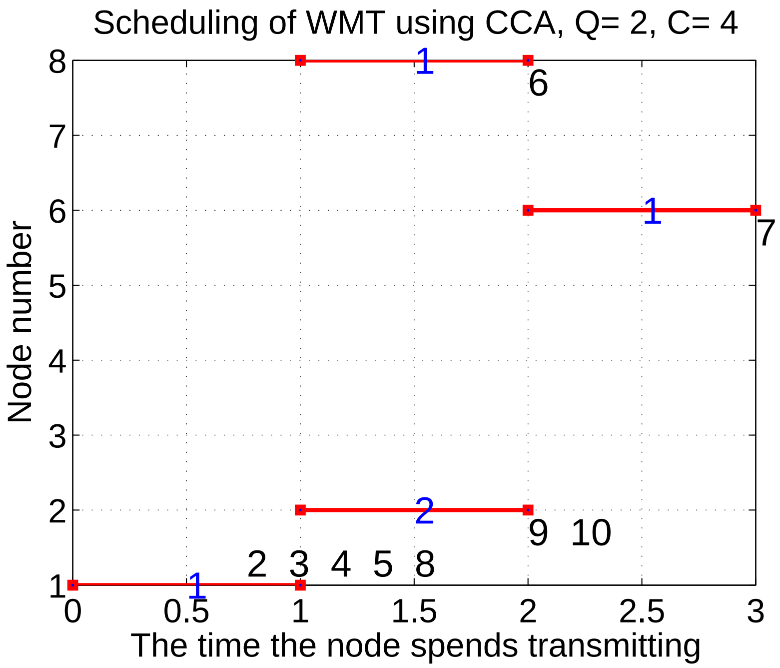
In this section, we evaluate the performance of our heuristic algorithms via simulations. We consider static wireless mesh networks with N nodes randomly located in a 1200 × 1200 m^2 region. The transmission rate/range relationship depicted in Table 1 is assumed. The interference range is assumed to be 520m. We have considered three channel assignment schemes in our current work: CCA, VCA and INSTC (discussed earlier in Section 2). The effect of the number of nodes in the network, the number of radio interfaces at each node, and the channel assignment strategy is observed on the broadcast latency when using our algorithms. We use the CCA channel assignment scheme for studying the effect of change in the number of nodes and interfaces on the broadcast latency in Section 10.1. Later, we show the effect of different channel assignment schemes on broadcast latency in Section 10.2.

10.1 Performance of our heuristic algorithms

We present the performance of our heuristic algorithms for the case of a *SR-SC multi-rate WMN* in Figure 19. The vertical axis shows the broadcast latency of our heuristics normalized against the broadcast latency of the Dijkstra tree with infinite number of Q and C. Since determining the actual optimal is NP-hard, we are using Dijkstra tree performance as a theoretical lower bound on the optimal achievable latency in a corresponding *wired* network. For the specific case of *SRSC multi-rate WMN*—MSPT compares unfavorably to our other three heuristics (Figure 19). These results are similar to those shown in [7]. MWT performs better than MSPT since it



Scheduling of MSPT using CCA, Q= 2, C= 4
Node number
The time the node spends transmitting
0
1
2
3
4
1
2
3
4
5
6
7
8
2
6
1
7
2
9
2
1
2 3 4 5 8
10



Scheduling of WMT using CCA, Q= 2, C= 4
Node number
The time the node spends transmitting
0
0.5
1
1.5
2
2.5
3
1
2
3
4
5
6
7
8
1
6
1
7
2
9 10
1
2 3 4 5 8

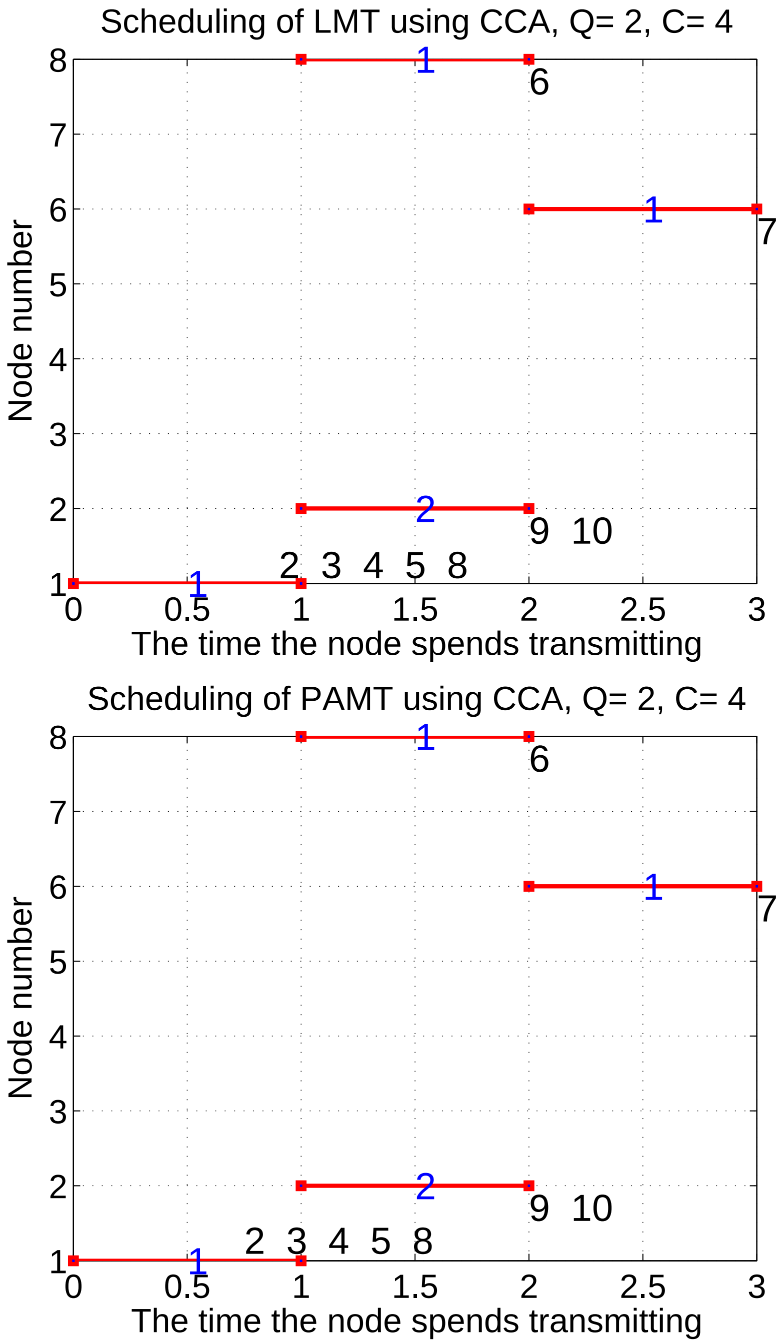
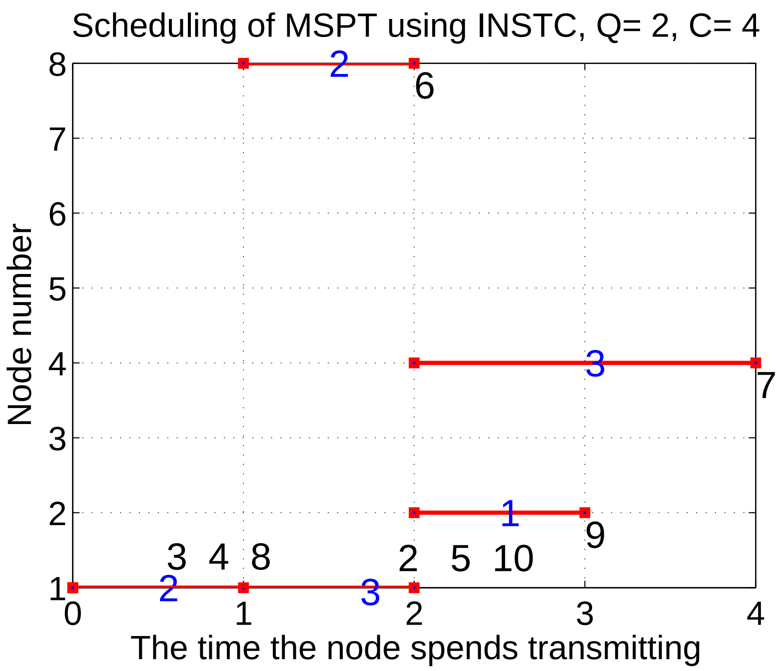
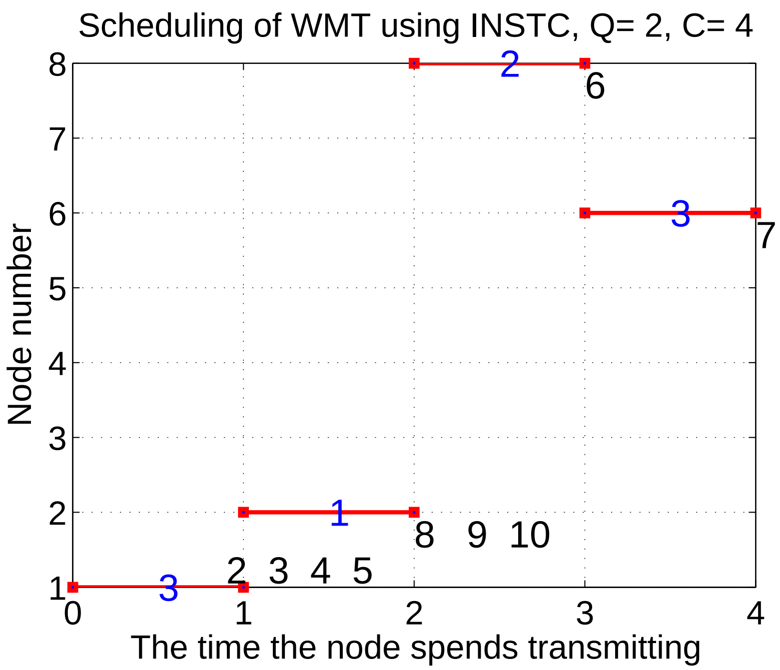


Figure 16: Transmission scheduling for our heuristics with channel-assignment scheme being CCA (children nodes in black, channel used in blue)



Scheduling of MSPT using INSTC, Q= 2, C= 4
Node number
The time the node spends transmitting



Scheduling of WMT using INSTC, Q= 2, C= 4
Node number
The time the node spends transmitting

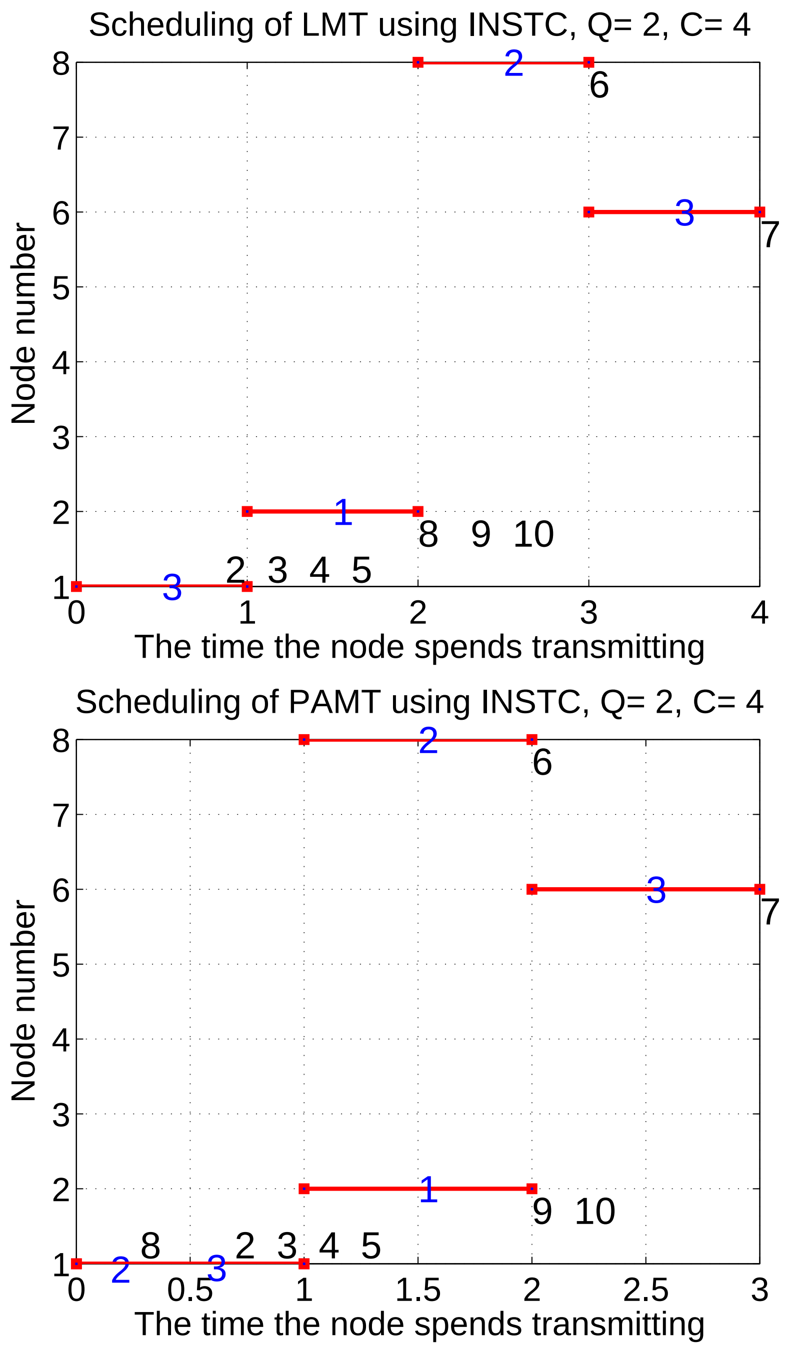
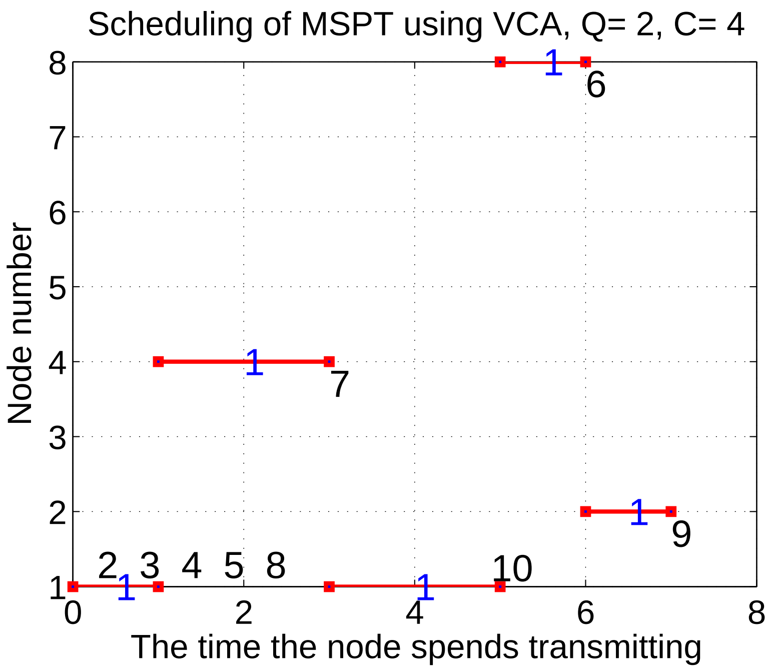
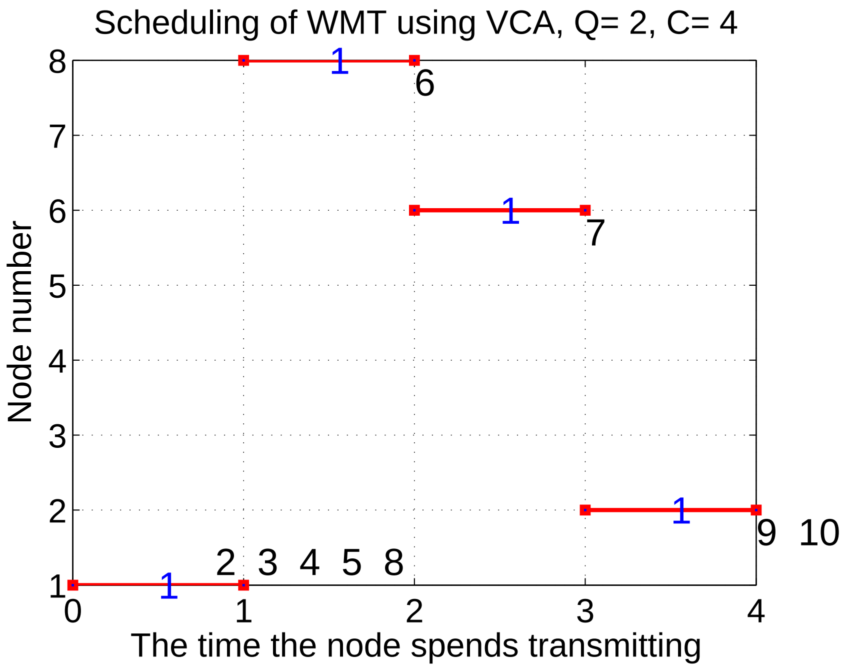


Figure 17: Transmission scheduling for our heuristics with channel-assignment scheme being INSTC (children nodes in black, channel used in blue)


Scheduling of MSPT using VCA, Q= 2, C= 4
Node number
The time the node spends transmitting
2 3 4 5 8
10
7
6
9


Scheduling of WMT using VCA, Q= 2, C= 4
Node number
The time the node spends transmitting
2 3 4 5 8
6
7
9 10

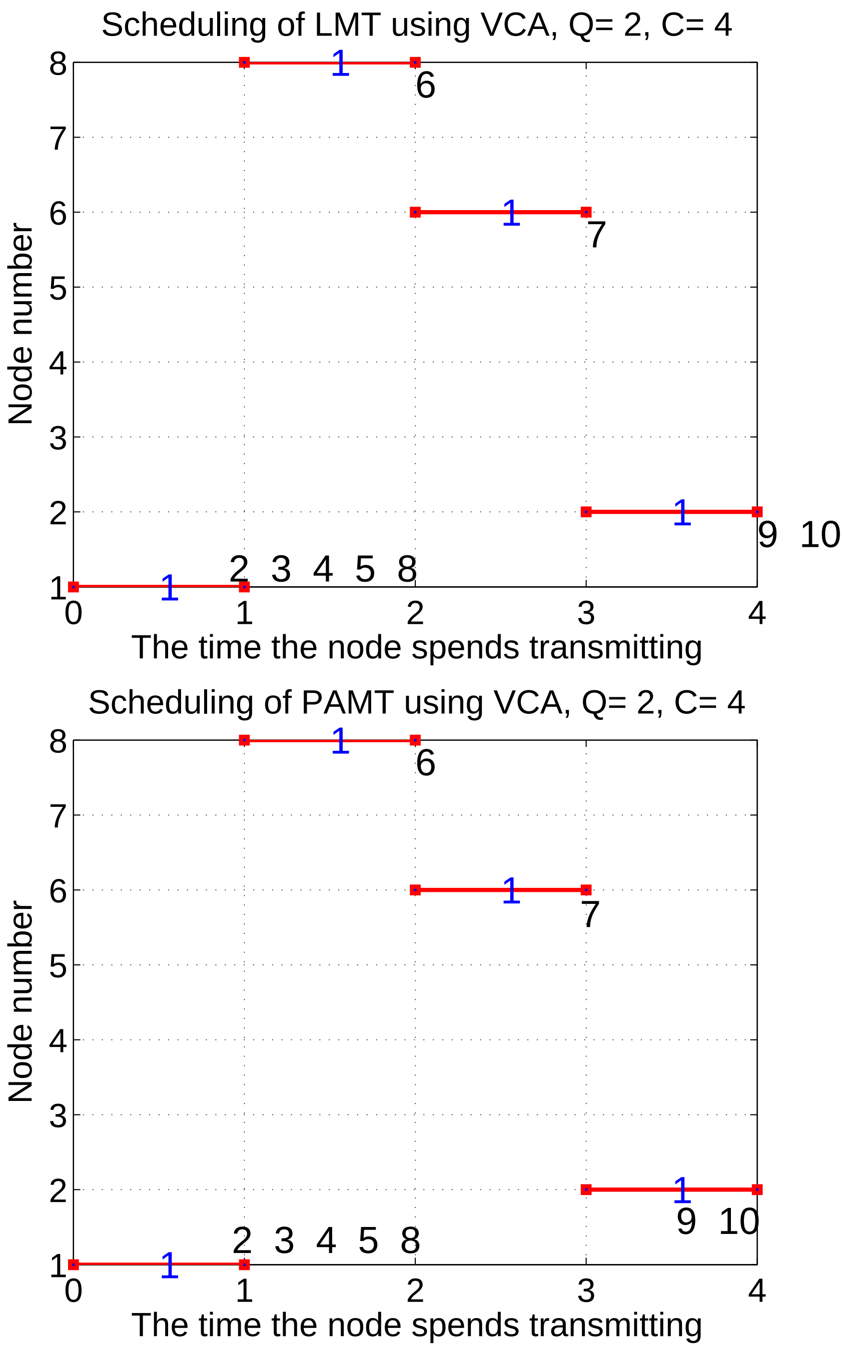


Figure 18: Transmission scheduling for our heuristics with channel-assignment scheme being VCA (children nodes in black, channel used in blue)

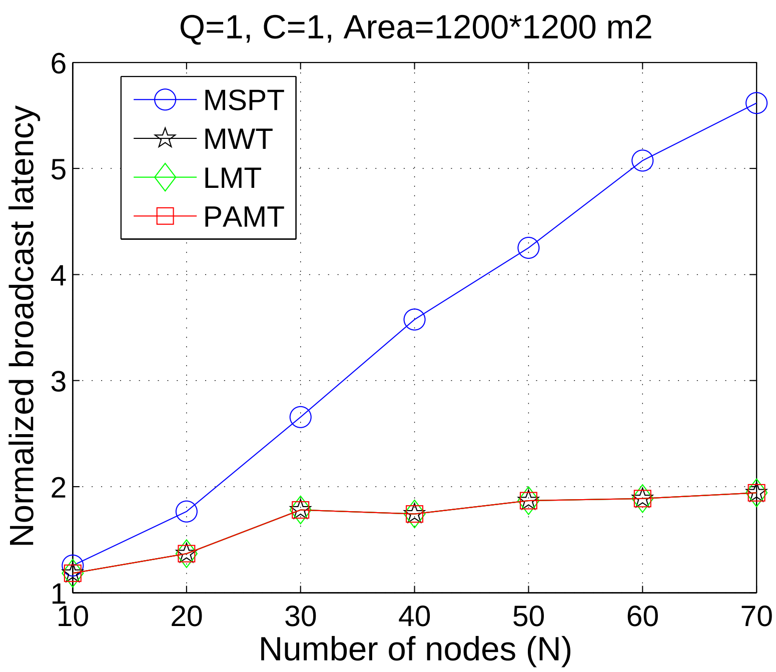


Figure 19: Normalized broadcast latency against varying number of nodes N for Q=1, C=1

considers both the WBA and the multi-rate nature of the mesh (Figure 19). The LMT and PAMT algorithms, both adapted from MWT, can only *match* and *not improve* the performance of MWT (Figure 19) in SR-SC multi-rate scenarios, since both cannot find *alternative* channel paths to '*parallelize*' transmissions on. *Thus for SR-SC multi-rate WMN, the performance of LMT and PAMT is exactly the same as WMT.*

For the cases of MR^2-MC multi-rate meshes where $Q > 1$, all of our proposed heuristics improve their performance. This is true both for small networks (N=10, Figure 20) and for large networks (N=70, Figure 21). The Figures 20 and 21 display representative performance of different heuristics for MR^2-MC multi-rate meshes across the range of radio interfaces from Q= 2 to Q=8.

The improvement seen in MR^2-MC performance can be attributed to two main reasons: Firstly, the usage of MR^2-MC minimizes the interference in the network and allows interfering transmissions to be transmitted simultaneously using orthogonal channels. This improvement factor called '*interference reduction factor*' is *general* and applies to all our proposed heuristics. The '*interference reduction factor*' substantially improves performance when the heuristic constructed tree involves many transmissions (e.g. as in MSPT). Secondly, a heuristic broadcasting algorithm that *parallelizes* its transmission, according to the number of available interfaces

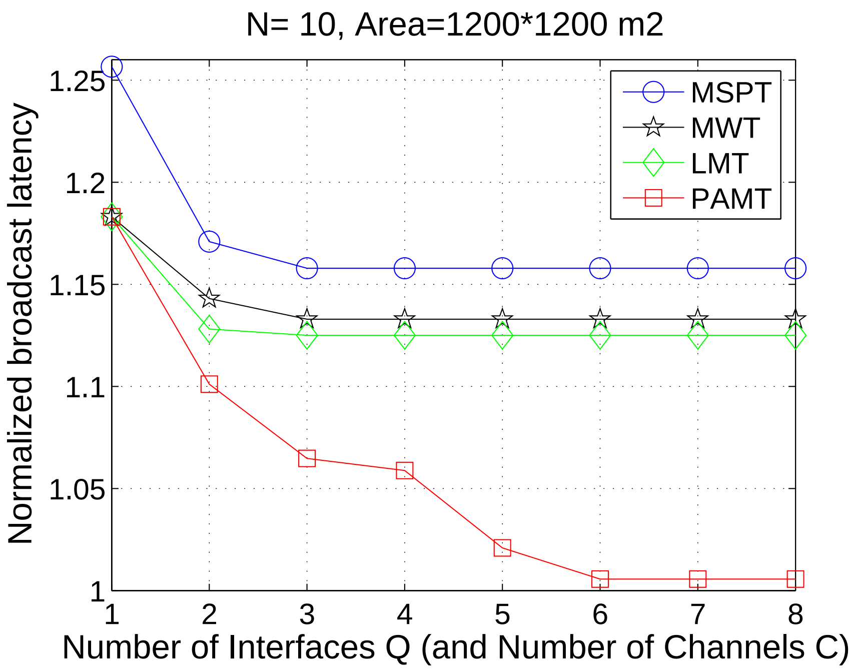


Figure 20: Normalized broadcast latency against varying number of radio interfaces Q $(C = Q)$ with $N = 10$ (Area=1200*1200 m^2)

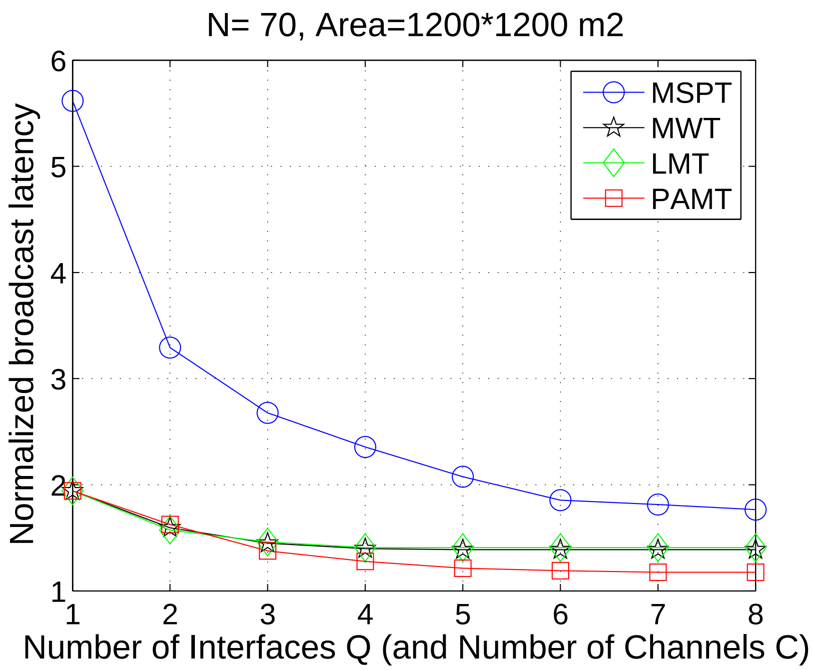


Figure 21: Normalized broadcast latency against varying number of radio interfaces Q $(C = Q)$ with $N = 70$ (Area=1200*1200 m^2)

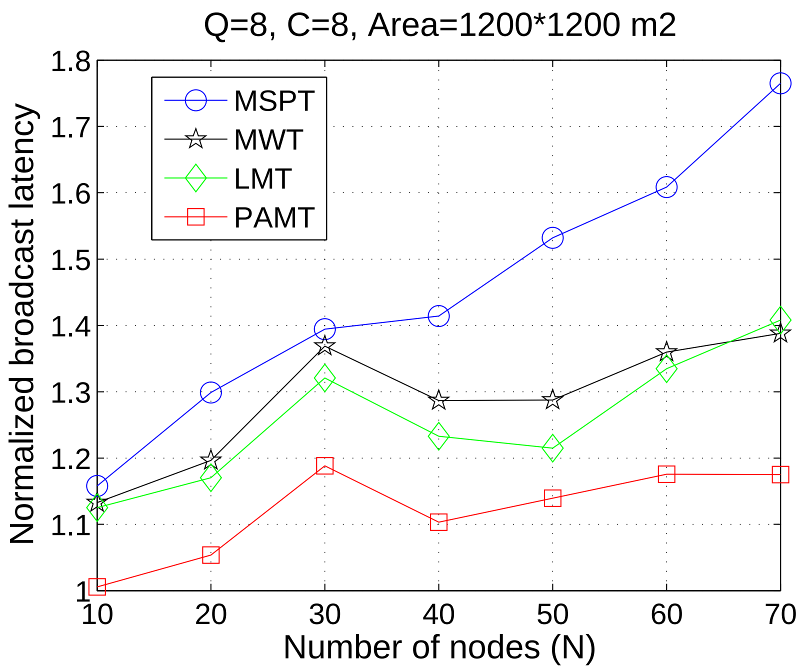


Figure 22: Normalized broadcast latency against varying number of nodes for Q=8, C=8

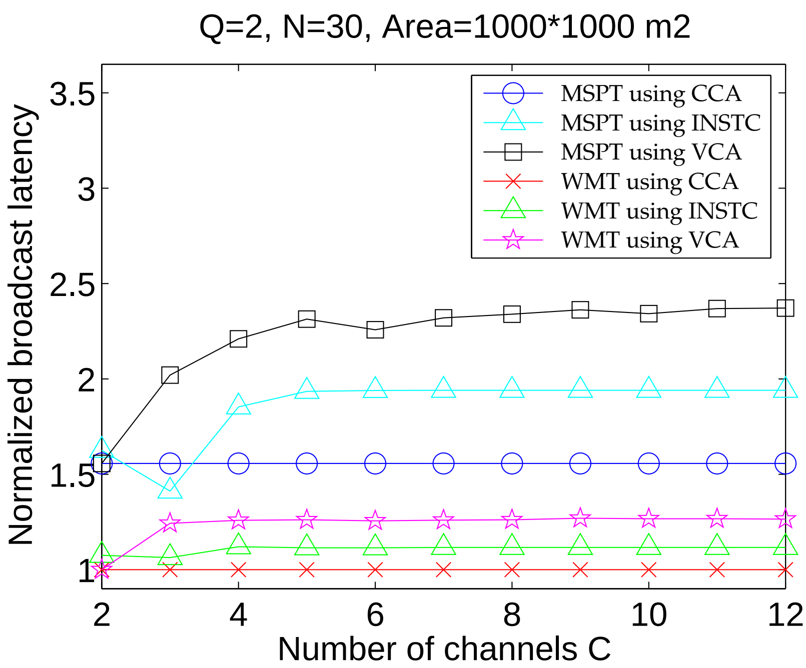


Figure 23: The impact of channel assignment on MSPT and MWT algorithms for Q= 2, N= 30 and Area=1200*1200 m^2)

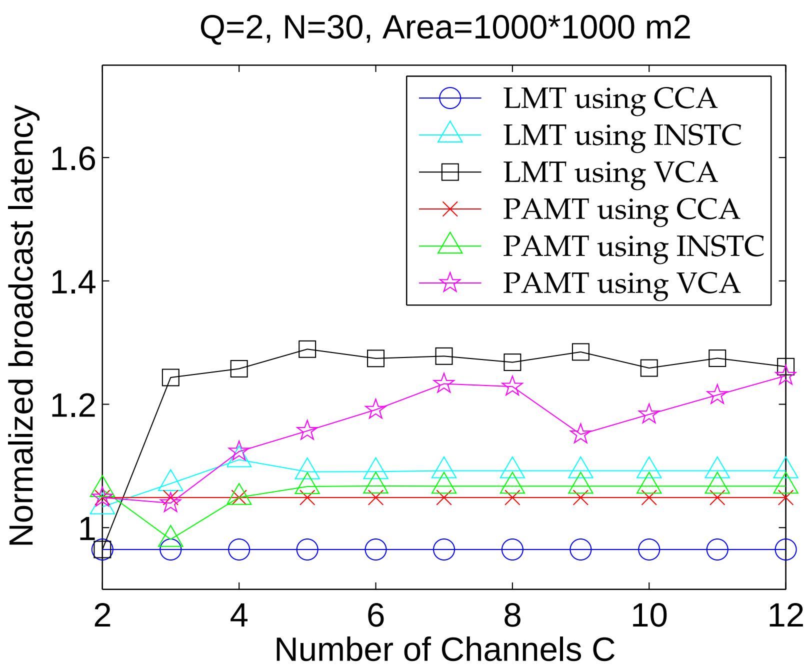


Figure 24: The impact of channel assignment on LMT and PAMT algorithms for Q= 2, N= 30 and Area=1200*1200 m^2)

and channels, reaps extra benefits by efficient usage of the resources available. This improvement factor called the *'radio adaption factor'* is specific to broadcasting algorithms such as LMT and PAMT.

We will now discuss the performance of each of our heuristic in MR^{2}MC wireless meshes with increasing Q and C. The performance of MSPT improves with increasing radio resources due to '*interference reduction factor*'—however, in our considered range of nodes (10 to 70) and interfaces (1 to 8), its performance compared to other proposed heuristics is modest (Figures 20, 21 and 27). MSPT's poor performance is explained by its *lack of accounting for WBA* during its construction, which in turn implies that too many transmissions are involved in a MSPT. Another reason is its *lack of adaption to the available radio resources.* It must be pointed out that although MSPT's performance is poor in the practical range of values of Q, its performance with the non-practical value of $Q = \infty$ corresponds to optimal achievable performance.

The performance of MWT can be seen in the Figures 20, 21, 22 and 28. It is worth noting that the performance of MWT improves with increasing Q, till a point, beyond which increasing Q does not produce any noticeable gain. Note in the Figures 20, 21 and 28, that although good gains are achieved when increasing the Q from 1 to 3, increasing Q further does not produce any gain. This is because MWT does not *parallelize* its transmission by *adapting* to increasing number of interfaces unlike LMT and PAMT. Thus for MWT, like MSPT, only '*interference reduction factor*' is relevant and the '*radio adaption factor*' does not apply.

It is interesting that both LMT and PAMT improve upon MWT's performance when Q and C are increased, as depicted in the Figures 20, 21, 22, 11, and 30. This implies that both these algorithms are *adaptive* to the available radio resources, and can therefore benefit from both the '*interference reduction factor*' and the '*radio adaption factor*'. The LMT algorithm is the best performing heuristic for $Q = 2$ and $N = 70$ (Figure 21). In such large networks with limited resources (in the considered case, $Q = 2$), the effect of interference is dominant and the trees that transmit less generally perform better. Since LMT is more conservative than PAMT in adding *parallel* links, it performs slightly better than PAMT in this case.

PAMT is generally the most *adaptive* of our algorithms to the available Q and C. The broadcast performance of PAMT is very close to optimal for small networks *and/or* large Q (Figure 20). PAMT also performs consistently well across all ranges of Q and N. Interestingly, PAMT can approach the performance of MSPT with $Q = \infty$ with relatively few radio interfaces in most instances.

Finally, we point out the performance gain due to multiple radio interfaces in MR^2-MC meshes over SR-SC multi-rate meshes. Referring to Figures 20 and 21, we see that for Q as less as 3 or 4, the broadcast latency decreases by about 30-40% compared to the scenario where well-designed heuristics are used and by as much as 80% when poorly designed heuristics (e.g. MSPT) are used for Q=1.

10.2 Impact of Different Channel Assignment Strategies

The graphs of the performance of different channel assignment schemes (CCA, VCA and INSTC) are shown in Figures 23 and 24, and Figures 25 and 26, for the cases of Q= 2 and 3, respectively.The results shown are representative of similar results seen across different values of Q. The vertical axis in the graphs show broadcast latency of the algorithm normalized against the WMT algorithm with channel assigned through CCA. All the channel assignment schemes considered have different *connectivity* and *interference* characteristics. As noted earlier, the topology given as input to our heuristics greatly affects the broadcast performance; with the input topology being defined by the channel assignment scheme, broadcast performance is closely affected by the channel assignment scheme chosen.

In CCA, a set of common channels are shared amongst all nodes; hence both the connectivity and interference are maximum. In VCA, although connectivity is ensured by tuning one interface at all nodes to a common channel, with the other interfaces being assigned from channels randomly from the remaining channels in C, the connectivity can suffer at the cost of reduced interference. INSTC, like VCA, reduces interference in the network by increasing channel diversity at the risk of reducing its connectivity and possibly mitigating the WBA. An ideal channel assignment algorithm has to balance the two *conflicting* requirements of low interference and high connectivity. In the presence of low interference, more transmissions can be scheduled simultaneously resulting in reduced broadcast latency. Similarly, with large connectivity there are increased opportunities of availing the WBA.

From Figures 23 and 24, it can be seen that for values of C only slightly larger than Q, VCA and INSTC can outperform CCA. This is because in such a scenario, the effect of reduced interference outweighs any reduction in connectivity. However, with further increase in C, the reduced connectivity can adversely affect the broadcast latency of the heuristics by neutralizing the WBA. This leads to generally more transmissions (not availing the WBA), and higher broadcast latencies. The characteristic of reduced interference in VCA and INSTC schemes have a more pronounced effect on the performance of MSPT, LMT and PAMT than on WMT, since these algorithms generally involve more transmissions (on possibly interfering channels). As we can see from Figures 23 and 24, the best performing channel assignment scheme for broadcast generally is CCA (which performs poorly for unicast flows [6]). Although the channel assignment scheme INSTC gives improved performance for unicast traffic, it is not necessarily the best performing channel assignment scheme for broadcast. Thus, we make an important observation that *a channel assignment scheme designed for unicast flows may sometimes perform poorly for broadcast/multicast flows.*

11 Conclusions and Future work

In this paper, we have studied the problem of minimum latency broadcasting in MR^2-MC wireless meshes. We have presented four heuristic algorithms, the first (MSPT) does not exploit the WBA, the second (MWT) exploits WBA but not the availability of multiple interfaces on the same node, while the other two (LMT and PAMT) differ in how they exploit both WBA and the interface diversity on individual nodes. Interestingly, both PAMT and WMT perform fairly close to the theoretical optimal, resulting in latencies that are on average only $\sim 10-20\%$ higher.

The simulation results and performance studies also show the impact of channel assignment strategies on broadcast latency, due to the conflict between greater connectivity and lower channel contention. Perhaps a more important observation is that a channel assignment scheme designed for unicast flows may sometimes perform poorly for broadcast/multicast flows. In our simulations, the performance of CCA (which generally performs poorly for unicast flows) is generally better than both VCA and INSTC.

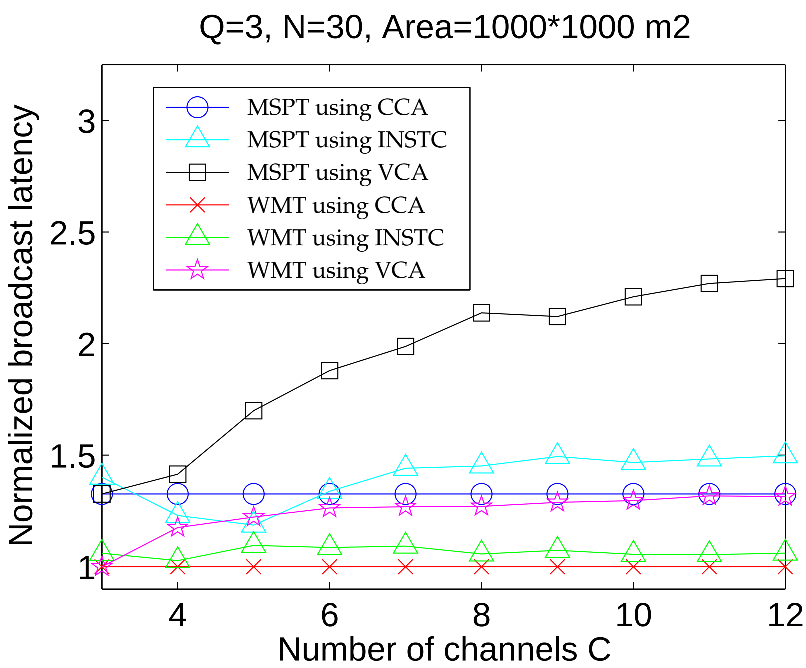


Figure 25: The impact of channel assignment on MSPT and MWT for 3 Radios, N= 30 and Area=1200*1200 m^2)

In our immediate future work, we plan to modify our heuristics to support QoS-aware broadcasting, where the algorithms consider both the existing traffic load on different interfaces and the data rate of the newly arriving broadcast flow. We shall also study the performance of the algorithms with a distributed 802.11-style contention-based MAC. In the long term, the problem of finding channel assignment strategies that perform well with a mixture of unicast and broadcast traffic is an important challenge in WMN design that we plan to address.

References

[1] Ian F. Akyildiz, Xudong Wang, and Weilin Wang. Wireless mesh networks: a survey. *2004 Elsevier B.V.*

[2] Bhaskaran Raman and Kameswari Chebrolu. Design and evaluation of a new mac protocol for long-distance 802.11 mesh networks. In *Proceedings of 11th Annual International Conference on Mobile Computing and Networking paper (MOBICOM)*, Sep 2005.

[3] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, IT-46(2):388–404, March 2000.

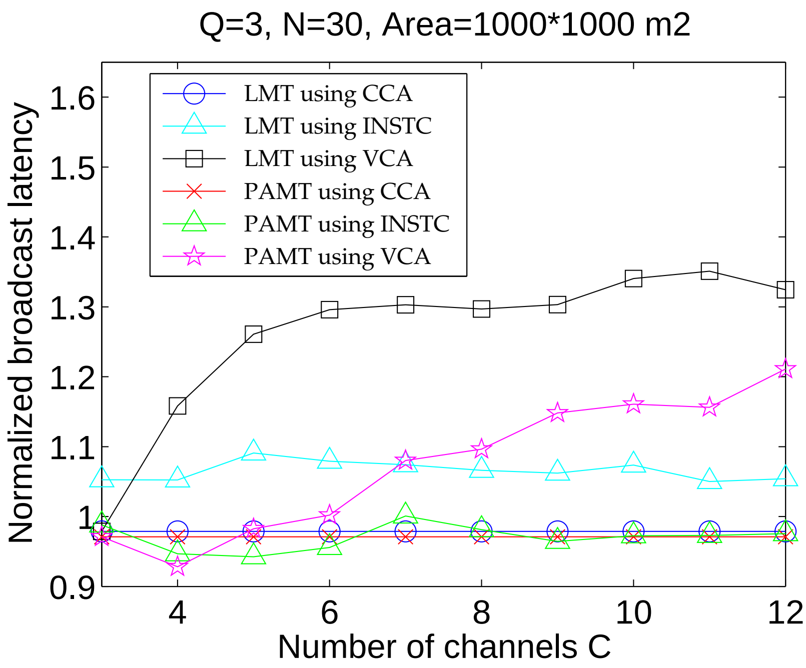


Figure 26: The impact of channel assignment on LMT and PAMT for 3 Radios, N= 30 and Area=1200*1200 m^2)

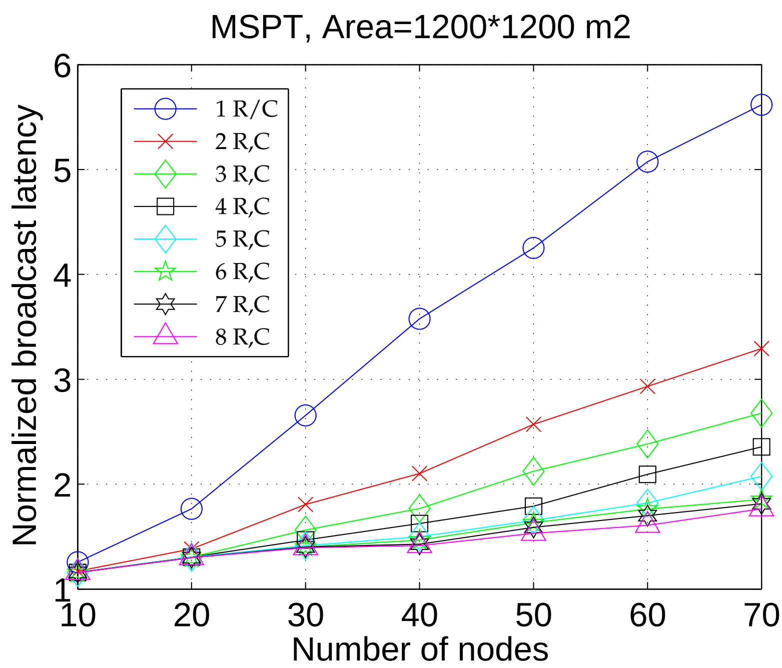


Figure 27: Normalized broadcast latency for different range of radio interfaces for varying nodes using MSPT (Area=1200*1200)

[4] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Mobicom*, pages 114–128, 2004.

[5] Ashish Raniwala, Kartik Gopalan, and Tzi cker Chiueh. Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks. *Sigmobile Mob. Comput. Commun. Rev.*, 8(2):50–65, 2004.

[6] P. Kyasanur and N.H.; Vaidya. Routing and interface assignment in multi-channel multi-interface wireless networks.

[7] C. T. Chou, A. Misra, and J. Qadir. Low latency broadcast in multi-rate wireless mesh networks. *IEEE JSAC special issue on wireless mesh networks*, 2006.

[8] Chun Tung Chou and Archan Misra. Low latency multimedia broadcast in multi-rate wireless meshes. In *First IEEE Workshop on Wireless Mesh Networks, Held in conjunction with SECON-2005.*

[9] The status of the IEEE 802.11s standard project. http://grouper.ieee.org/groups/802/11/reports/tgs_update.htm.

[10] Rajiv Gandhi, Srinivasan Parthasarathy, and Arunesh Mishra. Minimizing broadcast latency and redundancy in ad hoc networks. In *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 222–232, New York, NY, USA, 2003. ACM Press.

[11] C.T. Chou, A. Misra, and J. Qadir. Low latency broadcast in multi-rate wireless mesh networks. Technical Report ftp://ftp.cse.unsw.edu.au/pub/doc/papers/UNSW/0514.pdf, 2005.

[12] Jeffrey E. Wieselthier, Gam D. Nguyen, and Anthony Ephremides. Energy-efficient broadcast and multicast trees in wireless networks. *Mob. Netw. Appl.*, 7(6):481–492, 2002.

[13] J. Cartigny, D. Simplot, and I. Stojmenovic. Localized minimum-energy broadcasting in ad-hoc networks. In *INFOCOM*, page 2210, 2003.

[14] W. Lou and J. Wu. On reducing broadcast redundancy in ad hoc wireless networks. *IEEE Trans on Mobile Computing*, 1:111, 2002.

[15] P. M. Ruiz and A. F. Gomez-Skarmeta. Heuristic algorithms for minimum bandwidth consumption multicast routing in wireless mesh networks. In *ADHOC-NOW, pp. 258-270, 2005.*

[16] C. Gui and P. Mohapatra. Scalable multicasting in mobile ad hoc networks. In *INFOCOM*, page 2119, 2004.

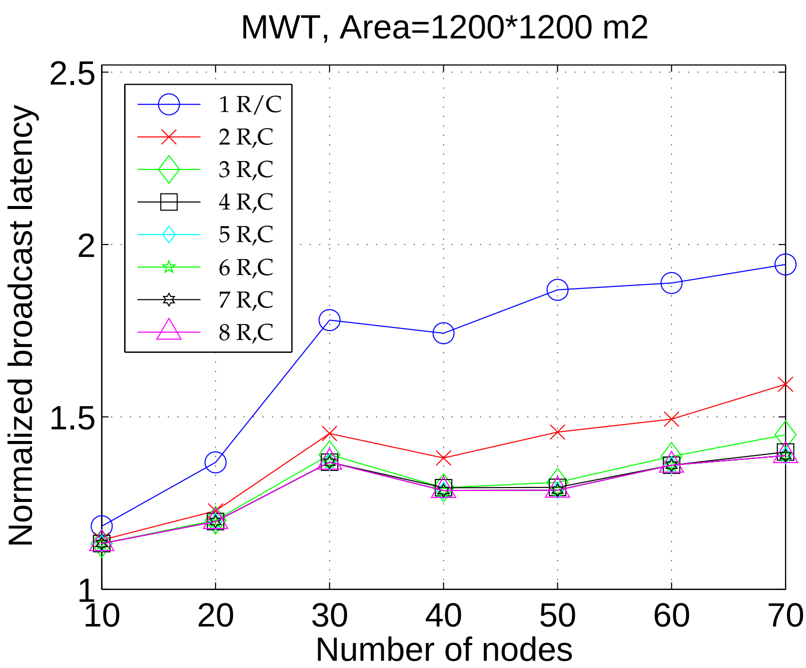


Figure 28: Normalized broadcast latency for different range of radio interfaces for varying nodes using MWT (Area=1200*1200)

[17] Jian Tang, Guoliang Xue, and Weiyi Zhang. Interference-aware topology control and qos routing in multi-channel wireless mesh networks. In *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 68–77, New York, NY, USA, 2005. ACM Press.

[18] Scalable Networks Inc. http://www.scalable-networks.com/.

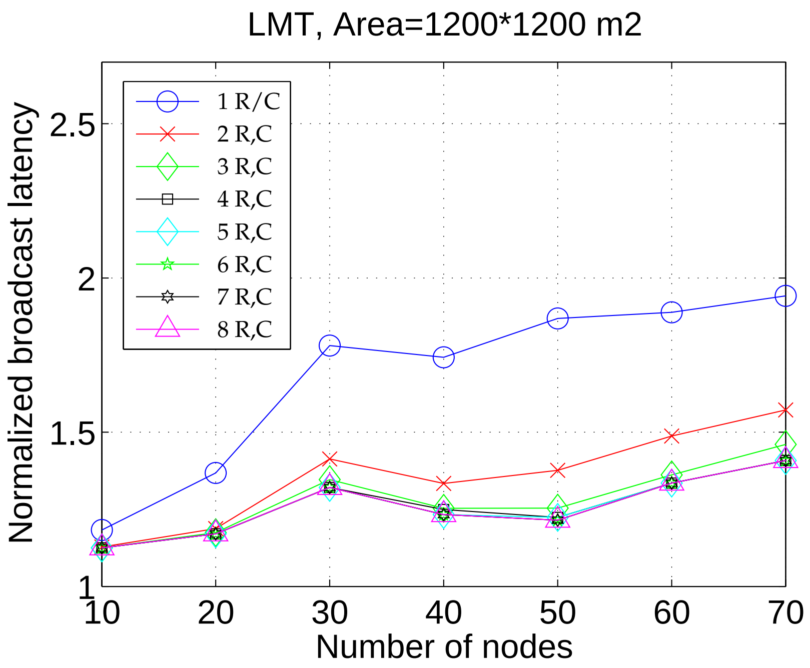


Figure 29: Normalized broadcast latency for different range of radio interfaces for varying nodes using LMT (Area=1200*1200)

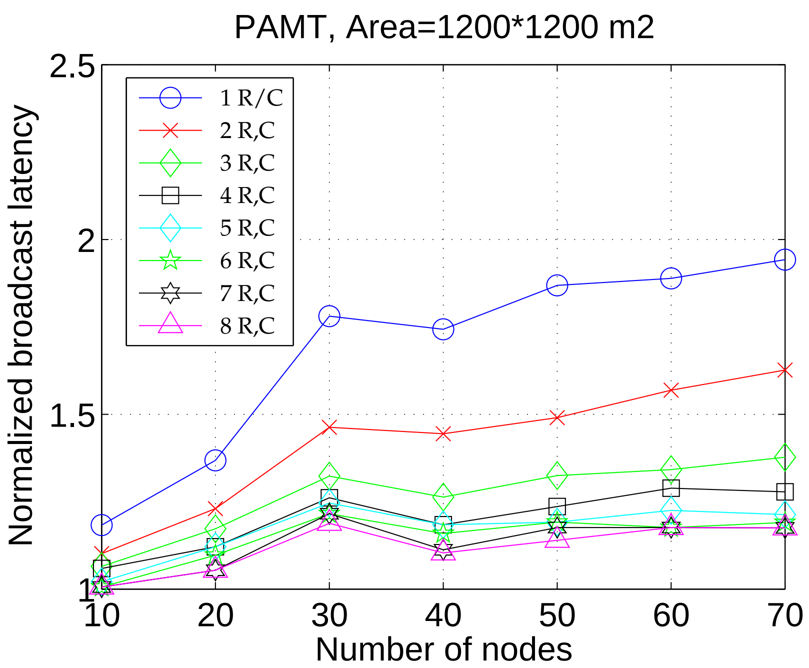


Figure 30: Normalized broadcast latency for different range of radio interfaces for varying nodes using PAMT (Area=1200*1200)