J. Parallel Distrib. Comput. 72 (2012) 1295–1305

Contents lists available at SciVerse ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Channel switching control policy for wireless mesh networks

Xiaoguang Li, Jie Wu, Shan Lin *, Xiaojiang Du

Computer and Information Sciences, Temple University, Philadelphia, 19122, USA

ARTICLE INFO

Article history:
Received 13 September 2011
Received in revised form
4 June 2012
Accepted 20 June 2012
Available online 3 July 2012

Keywords:
Adaptive
Balanced control system
Dynamic channel assignment
Real-time
Wireless mesh network

ABSTRACT

Dynamic channel assignment algorithms allow wireless nodes to switch channels when their traffic loads exceed certain thresholds. These thresholds represent estimations of their throughput capacities. Unfortunately, the threshold estimation may not be accurate due to *co-channel interference* (CCI) and *adjacent-channel interference* (ACI), especially with high traffic loads in dense networks. When the link capacity is over-estimated, these channel assignment algorithms are not effective. This is because the channel switch is not triggered even with overloaded data traffic and the link quality decreases significantly as the channel is overloaded. When the link capacity is under-estimated, the link is under-utilized. Moreover, when link traffic load increases from time to time, channel switch occurs frequently. Such frequent channel switches increase latency and degrade throughput, and can even cause network wide channel oscillations. In this paper, we propose a novel threshold-based control system, called *balanced control system* (BCS). The proposed threshold-based control policy consists of deciding, according to the real time traffic load and interference, *whether to switch to another channel*, *which channel should be switched to* and *how to perform the switch*. Our control model is based on a fuzzy logic control. The threshold which assists to make the channel switch decisions, could be deduced dynamically according to the real-time traffic of each node. We also design a novel dynamic channel assignment scheme, which is used for the selection of the new channel. The channel switch scheduler is provided to perform channel-switch processing for sender and receiver over enhanced routing protocols. We implement our system in NS2, and the simulation results show that with our proposed system, the performance improves by 12.3%–72.8% in throughput and reduces 23.2%–52.3% in latency.

1. Introduction

Wireless mesh networks (WMNs) are gaining significant momentum as an inexpensive way to provide last-mile broadband Internet access. Recent studies [10,13] have shown that equipping each node with multiple interfaces can improve the capacity of WMNs. By equipping interfaces in different channels, a node can communicate with multiple nodes simultaneously. Each channel allows multiple data flow exchanges in both directions, as long as the traffic load does not exceed the link's throughput capacity, i.e., the maximum amount of traffic that the link can carry.

Many previous researches in WMNs usually assume static channel capacity. This simplified assumption does not hold in reality. The throughput capacities in real systems can vary dramatically with time and location due to fading, shadowing, and interference. As a result, protocols based on static channel capacity may not work well in real systems as channel throughput capacity (or simply link capacity) can be either over-estimated or under-estimated.

Static channel assignment algorithms that switch channels periodically or permanently [22,25], have been shown to achieve great performance with stable network traffic. However, with dynamic traffic loads, such algorithms are not effective due to the mismatch between dynamic channel throughput capacity and the real-time traffic load. To select a channel based on real-time traffic load, recent studies on dynamic channel assignment algorithms [2,26,29] can adaptively switch the channel on certain links in a distributed fashion. Accurate estimation of channel throughput capacity is very challenging, as it is notably influenced by both *co-channel interference* (CCI) [31] and *adjacent-channel interference* (ACI) [23], especially when the traffic load is high. When the link capacity is over-estimated, the channel saturates and the channel quality degrades before the channel switch is triggered. On the other hand, when the link capacity is under-estimated, the channel is not fully utilized. Also, when the traffic load experiences temporary increases, existing algorithms tend to switch the channel frequently. Such frequent channel switches degrade the network throughput and increase latency significantly. Moreover, the newly switched links cause interference to other nearby links, introducing link capacity variation on those links and triggering even more channel switches. In the worst case, it can cause network wide channel oscillation.

An intuitive example is shown in Fig. 1. With an over-estimated threshold, the channel switch is not triggered in all cases, even when the channel saturates and the link quality degrades. While

* Corresponding author.
E-mail address: shan.lin@temple.edu (S. Lin).

0743-7315/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
doi:10.1016/j.jpdc.2012.06.008

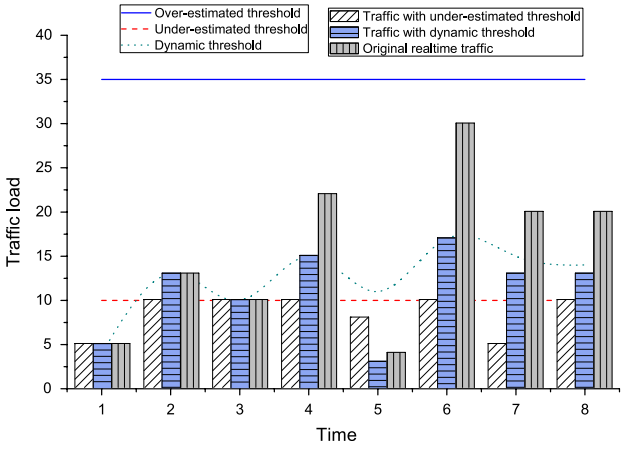

Fig. 1. Threshold vs. traffic load.

with an under-estimated threshold, the channel capacity is not fully utilized and channel switches occur frequently (at time 2, 4, 6, 7, and 8). If we can choose the link capacity threshold adaptively, the channel is better utilized than using the under-estimated threshold in all the cases. Moreover, the overhead in channel switching is reduced. We note that existing rate adaptation protocols [33,19] adjust transmission rate based on channel contention. Our work focuses on channel switching, which is an orthogonal issue to rate adaptation.

Our goal is to dynamically find a channel capacity estimation that fully utilizes link capacity and reduces unnecessary channel switching. In this paper, we propose a threshold based control system, called *balanced control system* (BCS). Unlike existing approaches that use static threshold estimation, our design features a fuzzy control loop to monitor the dynamic traffic load during runtime and adaptively adjust the channel switching threshold. This threshold serves as the bound for traffic load on this channel. The proposed threshold-based control solution consists of deciding, according to the real time traffic load and interference, *whether to switch to another channel, which channel should be switched to* and *how to perform the switch*. Our threshold control model is based on a fuzzy logic control. The threshold which assists to make the channel switch decisions could be deduced dynamically according to the real-time traffic of each node. Our control based design allows the dynamic threshold to approximate the runtime capacity accurately, therefore improving the channel utilization and reducing unnecessary channel switches. The contributions of our work are demonstrated as follows:

1. We propose a threshold-based balanced control system (BCS) in which each link in the network finds its own threshold according to the real-time traffic. We also offer a traffic metric model for our BCS. The metric model estimates the traffic load integrated with CCI and ACI problems.
2. We present a dynamic channel assignment scheme for the selection of the new channel. This algorithm fully utilizes variable channel capacities with reduced channel switching overhead. We also provide a channel switch scheduler to perform channel-switch processing for sender and receiver over enhanced routing protocols.
3. We implement our system in NS2. From our simulation results, we demonstrate that our proposed scheme outperforms the current techniques. Although channel switch algorithms for wired networks have been studied and practiced in industry [35]. In wired networks, static channel capacity models are highly accurate. Whereas in wireless networks, fading and interference (ACI and CCI) can cause significant channel throughput variations, resulting in frequent channel switch.

The remainder of this paper is organized as follows: Section 3 describes our proposed balanced control system. Section 5 provides evaluation results. Section 6 gives related work. The paper concludes in Section 7.

2. Preliminaries

In this section, we present the problem formulation, problem statement, and system overview.

2.1. Problem formulation

In this subsection, we describe the model formulation for our system in WMNs. Recall that WMNs consist of a set of stationary wireless routers, some of them acting as gateways to the Internet. Specifically, we do not require the presence of special gateway nodes, which could be the source or destination of all traffic in the network.

We define our system requirement as follows: (1) Two nodes that can communicate with each other should have at least one common channel. (2) We assume that every node in our system uses a single channel to communicate with each neighbor. (3) The common default channel is required for transferring control messages and is used as a temporary channel for data transfer. (4) Channels refer to different frequency bands. All of the channels are working on the half duplex mode.

2.2. Problem statement

Channel assignment algorithms in WMNs select channels for each link in the network in order to optimize network throughput and reduce latency. Recent channel assignment research explores a node's ability to dynamically switch channels. In essence, when the total amount of traffic load along this link exceeds the link capacity, nodes can either reduce the traffic load on this link or switch the channel. If the channel capacity is degraded due to CCI and ACI, it is desirable to switch from the current channel to another channel with higher bandwidth. However, channel switching incurs noticeable latency due to synchronization overhead between a pair of nodes, which also decreases the link throughput. Therefore, there is a tradeoff between the benefit of channel switching and its overhead.

In this paper, we explore when to perform channel switching under dynamic channel throughput. Previous research on channel switch is usually based on analysis with static channel models. Although these works provide valuable insights on this problem, unfortunately, these assumptions may not be hold in real WMNs: First, wireless link capacity is sensitive to distance and surrounding environment. In WMNs with fixed topology, even though the distance between any pair of nodes is fixed, the link capacity may vary due to environmental changes. Second, the traffic loads sometimes experience transient increases, which will result in the decrease of the traffic load and interference. These problems are not well modeled in previous studies. Third, the traffic loads affect link capacity, especially in dense networks with high traffic load. When traffic load of a link increases, it can cause channel capacity to degrade on itself and other nearby links, even if they are assigned with different channels due to interference. Therefore, with the dynamic traffic loads and channel capacity, previous solutions may not work well.

2.3. System overview

Our solution to the above problem has four key components: a traffic metric model, a fuzzy control model, a dynamic channel assignment algorithm, and an enhanced routing algorithm. In

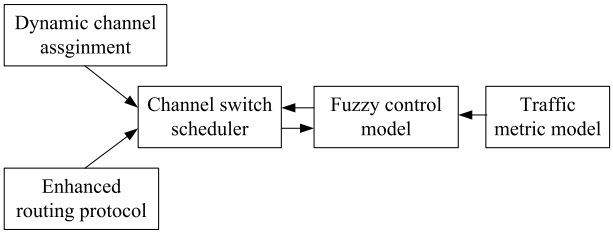

Fig. 2. System overview.

Table 1
Traffic metric model table.

Notation	Description
$pair(i, j)$	Link for nodes i and j
$bw^c(i)$	Traffic load of node i on channel c
F_t	Number of flows going through $pair(i, j)$
t, r	Flow id for transmitting/receiving the packets
$flow_t, flow_r$	Traffic load of outgoing flows t or incoming flows r
n_i	Neighbor id for node i
$bwt^c_{i,j}$	Total traffic load of $pair(i, j)$ on channel c
N_i	Number of neighbors of node i
γ	Interference ratio for ACI
c, c_n	Node's current channel and adjacent channel
$btt^c_{i,j}$	Traffic metric of channel c which integrates CCI and ACI

the fuzzy control model as shown in Fig. 2, a control loop is designed to monitor the channel capacity and dynamically adjust the threshold. The threshold serves as a condition for the channel switching algorithm. Our control model provides a reasonable fuzzy control mechanism for deciding whether channel switching should be performed and which channel should be switched to.

Based on the proposed model, the dynamic channel assignment algorithm performs channel switch during runtime in a distributed fashion. The conditions for selecting new channels are specified in the fuzzy control model. With the support of channel switch control and channel assignment, enhanced routing algorithms can achieve better performance than previous solutions.

3. Balanced control system

Our system is composed of a traffic metric model, a fuzzy control model, a dynamic channel switch scheme, and a channel switch scheduler. This section will present the details respectively.

3.1. Traffic metric model

There are significant amount works focusing on the model of interference level [17,30]. In this subsection, we offer an integrated model to estimate the level of traffic load and interference on each link (see Table 1) .

Assuming c is the current channel, and i is the current node, if j is the neighbor of node i, then, the traffic load between nodes i and j is the sum of all the outgoing flow $flow_t$ and incoming flow $flow_r$, as described by Eq. (1):

$$bw^c(i) = \sum_{t=1}^{F_t} flow_t + \sum_{r=1}^{F_r} flow_r. \tag{1}$$

To solve the CCI problem, we use the following Eq. (2) to describe the total traffic load on channel c in two hops for the $pair(i, j)$:

$$bwt^c_{i,j} = \sum_{n_i=1}^{N_i} bw^c(n_i) + \sum_{n_j=1, n_j \neq i}^{N_j} bw^c(n_j), \tag{2}$$

Table 2
Fuzzy control model table.

Notation	Description
x	Switching times
$\mu_{i,j}(x)$	Threshold for $pair(i, j)$
$\Delta t_{i,j}(x)$	Interval time between current and previous switches
$\phi_{i,j}(x)$	Control weight of threshold
$t_{i,j}(x)$	The time when $pair(i, j)$ switches to a new channel
$E(x)$	The value for deviation
$R(x)$	The variance ratio
$channelList$	Available channels in the network
$channel(i)(j)$	The channel on the interface j of node i

where N_i and N_j are the number of neighbors of node i and node j in the network, respectively. This equation is used to derive the total traffic load of $pair(i, j)$ and their neighbors, where they work on the current channel c.

The usage of adjacent channels can cause interference. Then, we can obtain the interference ratio γ according to the current channel measurement for ACI:

$$\gamma = 1 - \frac{|c - c_n|}{c_n} \tag{3}$$

where c is the current channel, which is used by node i and c_n is the adjacent channel corresponding to other interfaces of node i. $|c - c_n|$ is the distance of interference factor. It is similar to [23]. However, our comparison is based on two channels. The overall metric can be deduced as follows which integrates traffic load and adjacent channel interference factors based on two-hop neighborhood information:

$$btt^c_{i,j} = bwt^c_{i,j} + \sum_{l=1}^{N_i} (\gamma \times bwt^{c_n}_{i,l}). \tag{4}$$

Thus, we have offered our traffic metric model. Note that our traffic model is based on the estimation of traffic load and interference level.

3.2. Fuzzy control model

In this subsection, we present our system model. Our control model provides a reasonable fuzzy control mechanism for deciding which channel to be switched to. The goal is to find a per-link threshold for channel switching. An ideal threshold will approximate run-time channel throughput and reduces unnecessary channel switchings. In our fuzzy control model, if the threshold is set too small which indicates frequent switches, we will increase the threshold value. In contrast, if the threshold is set too large which prevents the system from performing beneficial channel switch, we will lower the threshold value accordingly. Therefore, we need to find the threshold adapting to the link capacity changes.

In this paper, we use fuzzy control to perform such an adjustment because it is difficult to accurately estimate the right threshold value a priori. Fuzzy control offers a convenient method for constructing nonlinear controllers via the use of heuristic information [18]. We present our fuzzy control model to adjust the threshold value (see Table 2). Accurate threshold adjustment is important because small errors in the thresholds can induce large channel-switch overheads. However, it is very challenging to directly estimate the exact threshold value since the environment is constantly changing. Thus, we use a fuzzy control model to demonstrate our strategy, which is similar to the automobile "cruise control" example in [36]. In our design, fuzzy interpretations are extended using the fuzzy set theoretic operations [12].

In our fuzzy control model, we use the interval time τ as a timer to monitor the network state. If the interval time between current switch and previous switch happens within τ, we will increase

the threshold after this switch. Otherwise, we will decrease the threshold. We use Eq. (5) to express this idea. There are two parts in this equation: the first part is that the current interval time is less than τ. When the current bandwidth is larger than constraint bandwidth $\mu_{i,j}(x)$, we need to switch the channel, and also increase $\mu_{i,j}(x)$; the second part is that the current time is larger than τ. This means there is no switching during this interval. We need to lower the threshold value:

$$\begin{cases} \mu_{i,j}(x) = \mu_{i,j}(x-1) + \phi_{i,j}(x-1), & \Delta t_{i,j}(x) \le \tau \\ \mu_{i,j}(x) = \mu_{i,j}(x-1) - \phi_{i,j}(x-1), & \text{otherwise} \\ \mu_{i,j}(0) = \mu_0, \qquad \phi_{i,j}(0) = 0, \end{cases} \tag{5}$$

where $\phi_{i,j}(x)$ is the control weight and $\mu_{i,j}(x)$ is the constraint bandwidth for $pair(i, j)$. $\mu_{i,j}(0)$ and $\phi_{i,j}(0)$ are the initial values of the two parameters. The threshold can be improved during τ. As we can see from this equation, the control weight is important for the controlling process. That is how much we need to increase or decrease the threshold. Moreover, when the switch should be performed is also an important issue. For this we use Eq. (6) to demonstrate the control weight $\phi_{i,j}(x)$:

$$\phi_{i,j}(x) = \alpha_{i,j}(x) \times E(x) + (1 - \alpha_{i,j}(x)) \times R(x-1) \tag{6}$$

where $E(x)$ is the value for deviation, and $R(x)$ is the variance ratio. $\alpha_{i,j}(x)$ is the weight for the balanced formula. $E(x)$ and $R(x)$ have different effects among different switches. Note that $E(x)$ should be much larger than $R(x)$. We use $\alpha_{i,j}(x)$ to adjust the threshold. Sometimes, we want to adjust this threshold quickly, thus, we increase $\alpha_{i,j}(x)$ for $E(x)$. Other times, we prefer that it changes slowly, thus, we decrease $\alpha_{i,j}(x)$ for $R(x)$. It is shown that $\alpha_{i,j}(x)$ is used to control the two parts. Next, we offer the following equation to obtain $E(x)$, $R(x)$ and the weight $\alpha_{i,j}(x)$.

$$\begin{cases} E(x) = \mu_{i,j}(x) \\ R(x) = \dfrac{\phi_{i,j}(x)}{(t_{i,j}(x) - t_{i,j}(x-1))} \\ \Delta t_{i,j}(x) = t_{i,j}(x) - t_{i,j}(x-1) \\ t_{i,j}(0) = 0, \quad \forall i, j, x \in N,\ \tau > 1, \end{cases} \tag{7}$$

$$\begin{cases} \alpha_{i,j}(x+1) = \left| 1 - \dfrac{t_{i,j}(x) - t_{i,j}(x-1)}{\tau} \right|, & \Delta t_{i,j}(x) \le \tau \\ \alpha_{i,j}(x+1) = 0, \quad \text{otherwise} \end{cases} \tag{8}$$

where $t_{i,j}(x)$ is the recorded time when $pair(i, j)$ switches to a new channel. For the weight $\alpha_{i,j}(x+1)$, if the interval time between the current switch and previous switch is larger, the current threshold is close to the estimated threshold. Then, $R(x)$ is larger according to the weight $\alpha_{i,j}(x+1)$. Otherwise, $E(x)$ is larger.

Below, we summarize the whole process. At the start of the controlling process, the deviation $E(x)$ is more important. During this period, the control weight $\phi_{i,j}(x)$ can help the system quickly find the range of the threshold. A decrease in $\alpha_{i,j}(x)$ will make the variance ratio $R(x)$ become the main factor of the system and $E(x)$ become less important. Thus, it could adjust the value, and control the estimated threshold in this range.

3.3. A dynamic channel assignment scheme

We design a dynamic channel assignment algorithm (GNOC) to get the next optional channel for our system. The GNOC is used to select new channels for transmission. We will select the channel which is lower than the threshold and with minimum ACI. Table 3 offers the notation for the GNOC algorithm.

In Algorithm 1, *channelList* are the available channels in WMNs. The current channel, say c, is the current channel used for transmission. We first construct a temporary channel list, say *tempList*, from the available channel list of node i. Then, we get node i's neighbor list. To each neighbor of node i, we add the channels that the neighbors of node i are using to *tempList*. Then, we obtain the absolute value (abs) according to the channel in *channelList* and *tempList*. We set *diff* to be an extreme value of differences among channels. Then, the final channel is the channel with a minimum *diff*.

Table 3
Channel information of each node for GNOC scheme.

Notation	Description
$neighbor\_Set(i)$	Neighbors of the node i
$channelList$	Available channels in the network
$channel(i)(l)$	Channel id for node i on interface l
$interface(i)$	Interface information of node i
$channelState(i)(j)$	Channel bandwidth of $pair(i, j)$

Algorithm 1 GNOC

```
Let tempList ← cList(i);
for k ← 0 to N_i do
    if channel(j)(k) is not in the tempList then
        channel(j)(k) is added into tempList;
for j ← 0 to |channelList| do
    for k ← 0 to |tempList| do
        c ← kth channel from tempList;
        diff ← |channelList|/N_i;
        diff ← min(diff, |channel(i)(j) − c|);
```

The new channel we get is the channel with the least ACI. However, we are not sure about the traffic statement of the new channel. Therefore, we also check the current statement of the new channel according to our proposed routing metric. If it does not exceed the threshold $\mu_{i,j}(x)$, then the new channel is the next optional channel. Otherwise, we remove this channel and run Algorithm 1 again until we find it.

3.4. Enhanced routing protocols in WMNs

First, we demonstrate why the existing routing protocol cannot be used in our scheme. The existing routing protocols, such as AODV [21] and FSR [20] support multiple interfaces. However, these designs are typically used for the multi-home based protocol (such as SCTP and DCCP) instead of being used for multiple interface wireless mesh networks.

The main problem is that the ideal design of our system needs fast channel switching to solve the unstable system problem. As we all know, after one node, say node i, has sent the "hello" information, the neighbors who received the "hello" information will add node i into the routing table entry. Then, they establish the connection. This will remain for future use, until it expires. If node i has changed its channel, but the corresponding neighbor, say node j, does not switch to the new one, then, this means they cannot connect to each other after the switch. However, the routing table entry remains in both nodes. When they need the route, they still can find the route according to the routing table entry. However, the packets cannot be forwarded because the actual connection is broken. It will take more time to establish and find a new route. The time of transferring the packets has to be delayed.

We propose to introduce another field called "channel id" to the routing table entry. This "channel id" can be used among neighboring nodes to coordinate their channel selection processes. The two neighbors along a channel can talk to each other, when they switch to the same channel, specified by "channel id". Interface and channel information can be obtained during initialization. When the node has switched to another channel, and the "channel id" has changed, its routing table should be updated according to the new "channel id" information.

In our model, we propose modifications to the current routing protocols, AODV and FSR, so as to enable the discovery of channel information from a source to a destination. The proposed enhancement can help the node find the correct route, even if the nodes have switched channels.

E-AODV routing: When the source node requests to send the packet to the destination, it will send "hello" information. The broadcast packet RREQ will be flooded to every interface of the node. Channel information is also added into the packets, and the corresponding nodes that receive the notice will update the routing table entry. At that time, the "channel id" is updated, and the nodes will select the correct route to communicate.

E-Fisheye routing: Fisheye Routing is different from AODV Routing, which is routing on-demand. In the fisheye routing scheme, we update the routing table according to the channel table, which will be explained in the next subsection (Channel Switch Scheduler). If the channel table has been changed, the routing table needs to be updated as well. Algorithm 2 is offered to show the detailed process of updating the routing table.

Algorithm 2 Update routing table

```
1: for i ← 0 to num. of nics on node x do
2:   for y ← 0 to num. of nics on node y do
3:     if (channel(x)(i) = channel(y)(j)) then
4:       channelid ← channel(x)(i)
5: nodes x and y lookup the route, and update routing table with
   newchannelId;
```

3.5. Channel switch scheduler

In [11], the author proposes the idea of the channel switch being involved in the mesh networks. However, that idea is not flexible and did not consider the detailed routing issues in the channel switch period. In this subsection, we will provide the process of channel switch. The channel and interface information has been maintained by every node in WMNs. Table 3 details the information for every node.

Algorithm 3 Executed by sender

```
if btt^c_{i,j} < μ_{i,j}(x) then
  Forward packet;
else
  if The num. of common channels of pair(i, j) is larger than 1
  then
    Update the current routing table entry;
    Go to step 1;
  if Unused interfaces of node i is larger than 0 then
    newchannel ← interface(i);
    Go to step 1;
  newchannel ← the next optional channel according to GNOC;
  if btt^c_{i,j} < btt^{newchannel}_{i,j} then
    newchannel ← c;
  Step 1:
  Update μ_{i,j}(x) according to Eq. 5;
  neighbornodeList ← yList(x);
  Generate chr_packet;
  Update the current routing table entry;
  Send chr_packet to the nodes in ylist(x);
  End step 1;
  Forward packet using the new routing table;
```

The *neighbor_Set*(i) is the set for all the neighbors of the current node, say node *i*. The number of available channels contains all the channels that can be used for the whole network. Every node in this network can switch channels when it satisfies the threshold $\mu_{i,j}(x)$. Moreover, special conditions need be satisfied:

- Condition 1: There is another common channel for the *pair*(i, j). If the traffic load of that channel does not exceed threshold $\mu_{i,j}(x)$, then *pair*(i, j) will select that channel to communicate.
- Condition 2: If the new channel, taken from Algorithm 1, is already working for another interface of node *j*, then only node *i* switches to the new channel.
- Condition 3: There are other nodes that are neighbors of *pair*(i, j). All of them work on the same old channel. These nodes construct a temporary list called *yList*.

If a new pair is added to the transmission, the traffic load exceeds the threshold of $\mu_{i,j}(x)$ from Eq. (5). For example, the *pair*(i, j) in channel *c* first checks condition 1. If it is not satisfied, it will temporally take channel *c* from Algorithm 1. Then, it will check condition 2 and get the node to switch. If there is an available interface for the node, it will select the available interface.

In this enhanced routing protocol, the channel information is stored on every node's neighbor table. We employ the standard mechanism [22,8,27,34] widely used to deal with topology dynamics, such as nodes joining the network. If a new node joins the network, it can notify other nodes in the network by broadcasting a hello message via a common control channel. In unicast, the current node will collect the channel state information through the channel information before transmission. The current traffic load can be obtained from Eq. (4). Then, we apply the channel switch algorithm, shown in Algorithms 3 (sender) and 4 (receiver). If the current traffic load exceeds the threshold $\mu_{i,j}(x)$, it will switch to another channel. Before that, the node should send the message to every node in the network. A field *chr_packet* (see Table 4) is added to a packet to carry the information. If the other node in the *neighbornodeList* receives the *chr_packet*, and it does not satisfy condition 2, then it is required to switch to the new channel *c*.

We first apply an initial random channel assignment according to the topology generator. The GNOC strategy is used to select the next optimal channel for this switch. The enhanced routing agent is used for the channel switch scheduler and the proposed balanced controlling model is used to make the decision of the switch.

Algorithm 4 Executed by receiver

```
Receive a chr_packet;
if (nodeId ∈ neighbornodelist) then
  Switch channel and update μ_{i,j}(x) according to Eq. 5;
  Update the current routing table entry;
  Update channel table;
else
  Update channel table;
```

Table 4
CHR packet for channel switch scheduler.

Notation	Description
nodeId	The current node to be switched
othernodeId	The other node of the pair to be switched
oldchannelId	The old channel of the pair
newchannelId	The new channel to be switched for the pair
neighbornodeList	The neighbors sharing the same channel

4. Stability analysis

In this part, we will provide the stability analysis. We offer the following theorem to show that our system can finally find the threshold.

Theorem 1. *If the control weight can be infinitely close to* 0*, then our system could finally find the estimated threshold* $\mu_{i,j}(x)$*. Thus, we need to prove the following conclusion:*

$$\lim_{\Delta t\to\tau, x\to\infty} \phi_{i,j}(x) = 0.$$

Proof. From Eq. (6), the problem can be converted to two parts:

$$\lim_{\Delta t\to\tau, x\to\infty} \alpha_{i,j}(x+1) \times E(x) = 0,$$
$$\lim_{\Delta t\to\tau, x\to\infty} (1-\alpha_{i,j}(x+1)) \times R(x) = 0.$$

Here, we know:

$$E(x) < C, \quad \text{such that } C \in \Re.$$

Although C could be sufficiently large, C actually is a finite number. Thus, for the first part, we only need to prove:

$$\lim_{\Delta t_{i,j}\to\tau, m\to\infty} \alpha_{i,j}(x+1) = 0.$$

From Eq. (7), we have:

$$\lim_{\Delta t_{i,j}\to\tau, m\to\infty} \alpha_{i,j}(x+1) = \lim_{\Delta t\to\tau} \left| 1 - \frac{\Delta t(x)}{\max(\Delta t_{i,j}(x), \tau)} \right|.$$

According to the first part of Eq. (5), we know that the interval time $\Delta t_{i,j}(x)$ is increased after the switch, since the threshold is larger and more difficult to meet switching conditions. However, in the second part, the proof is obvious, since $\frac{\Delta t_{i,j}(x)}{\max(\Delta t_{i,j}(x),\tau)} = 1$. Thus, we prove that

$$\lim_{\Delta t_{i,j}\to\tau, x\to\infty} \alpha_{i,j}(x+1) = 0.$$

Next, we need to verify the following assumption:

$$\lim_{\Delta t\to\tau, x\to\infty} (1-\alpha_{i,j}(x+1)) \times R(x) = 0.$$

Because we have:

$$\lim_{\Delta t\to\tau, x\to\infty} (1-\alpha_{i,j}(x+1)) \times R(x) \le \lim_{\Delta t\to\tau, x\to\infty} R(x) = \lim_{\Delta t\to\tau, x\to\infty} \frac{\phi_{i,j}(x)}{\Delta t_{i,j}},$$

$\phi_{i,j}(x)$ could be sufficiently large. It is actually bounded by a finite value $\phi_{\max}$, such that $\phi_{i,j}(x) < \phi_{\max}$. According to the Cauchy series [16], a monotone sequence converges if and only if it is bounded. Since $\Delta t_{i,j}$ is increased during the controlling process, then:

$$\lim_{\Delta t\to\tau, x\to\infty} \frac{\phi_{i,j}(x)}{\Delta t_{i,j}} = 0.$$

Thus, this concludes the proof of our solution. □

Examples:

In this part, we will offer an example of our proposed model. We will formalize the theory results according to our analysis. We first change Eq. (6) to another form. We set:

$$\begin{cases} A = \alpha_{i,j}(x+1) \\ B = \dfrac{1-\alpha_{i,j}(x+1)}{t_{i,j}(x) - t_{i,j}(x-1)}. \end{cases}$$

Then, Eq. (6) can be deduced:

$$\phi_{i,j}(x+1) = A \cdot \left(\mu_{i,j}(0) + \sum_{k=0}^{x-1} \phi_{i,j}(k) \right) + B \cdot \phi_{i,j}(x). \tag{9}$$

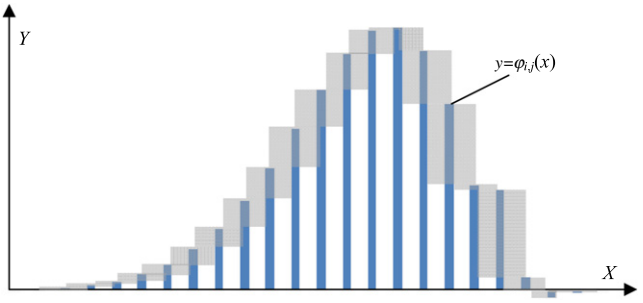

Fig. 3. Formula $\phi_{i,j}(x)$ in the control model.

Fig. 3 demonstrates theoretical results of formula $\phi_{i,j}(x)$ (see Eq. (9)). Because other values could be deduced after each switch, we only need to consider A and B of Eq. (9), because $\Delta t_{i,j}(x)$ of B is the unknown result measured from the real-time record. We set the value of $\Delta t_{i,j}(x)$ from 1 to 20. The increase is 1 each time. These values could be different each time, but they must be incremental. The reason is that each time the threshold increases, it becomes more difficult to meet the channel switch condition. Without loss of generality, we set $\tau = 20$ and $\mu_{i,j}(0) = 100$. From Fig. 3, we can see that $\phi_{i,j}(x)$ first increases and then decreases as time goes on. This confirms the validity of our design. The purpose of the control system is to dynamically find the threshold and make the system stable. To achieve this goal, we should continuously reduce the selected area of the constraint $\mu_{i,j}(x)$ until we find the exact value $\mu_{i,j}(x)$. In other words, we first reduce the area as quickly as possible, and then adjust it. The increased process of $\phi_{i,j}(x)$ is to find the smallest area. In addition, the decreased process is to adjust the value in the small selected area. That is the reason we need the control weight $\phi_{i,j}(x)$ to be first increasing and then decreasing.

Discussion:

According to our analysis, we know that our method can dynamically adjust the threshold value according the traffic demand. If the amount of traffic increases dramatically beyond the threshold, the threshold will be adjusted higher more aggressively, and vice versa. The node might also be switched to use the channel which has less traffic load. More specifically, when τ becomes larger, the increase of $\phi(x)$ scales in an exponential way. We present this analytical result in Fig. 4 by varying the parameter τ from 10 to 20. A larger τ with the same interval will incur higher ϕ. This will result in larger threshold value. In this figure, the largest values for points on each curves vary from 699.67 to 23710. We plotted these largest values in Fig. 5. From this figure, we can see that the threshold will increase in an exponential way as τ value increases. We also provide an analysis subject to changes of $\Delta t_{i,j}(x)$ in Fig. 6. When $\Delta t_{i,j}(x)$ is set to increase as 0.5, 0.75 and 1 separately, the increase of $\phi(x)$ is exponential. This indicates that if $\Delta t_{i,j}(x)$ is smaller, the traffic become more dynamic and the increase is larger.

5. Performance evaluation

In this section, we evaluate the performance of our proposed system through simulation. We implemented our solution: channel switch control and dynamic channel assignment algorithm in NS-2 simulator. As NS-2 provides rich physical layer models, the NS-2 based simulation has been widely used in research studies. We test the enhanced routing algorithms: AODV and FSR on top of our solutions, which are default routing protocols according to 802.11s [32]. AODV [21] is a reactive routing protocol while FSR [20] is a proactive routing protocol. FSR controls its update overhead using a policy of non-uniform frequency for update. The

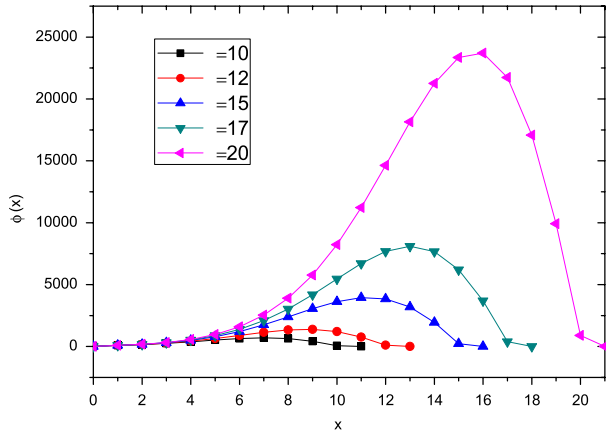

Fig. 4. Comparison of different τ.

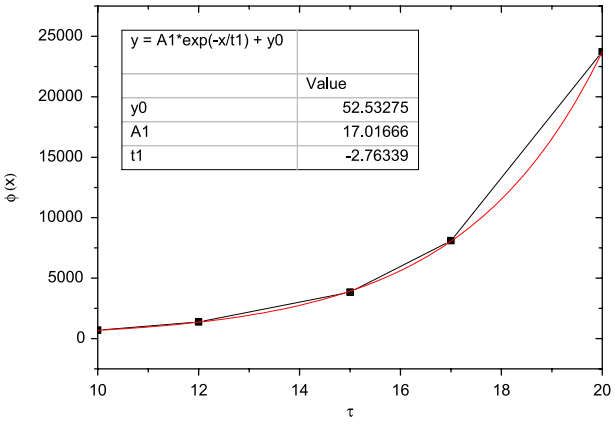

Fig. 5. Comparison with fitting function.

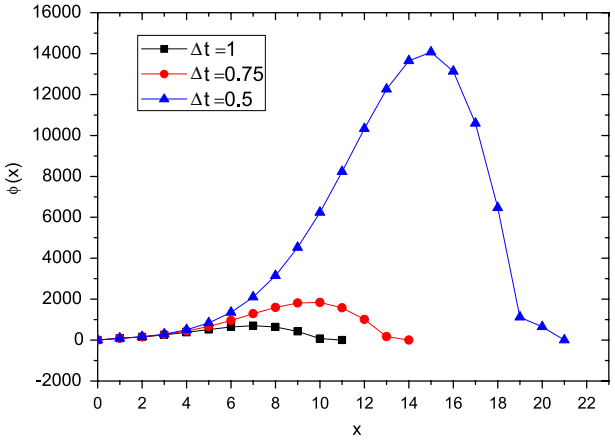

Fig. 6. Comparison of different Δt.

inner scope nodes are updated more frequently (and hence have more accurate information) than the outer scope nodes. Our solutions can also work with other routing protocols in WMNs.

In the simulation of WMNs, there are several available extensions [25] for M2WMNs. We extend the existing work with switching abilities using NS2. Extensively simulation results demonstrate that our algorithm outperforms existing solutions without control [14]. Overall, our solution improves existing solutions by 12.3%–72.8% in throughput and reduces 23.2%–52.3% in latency.

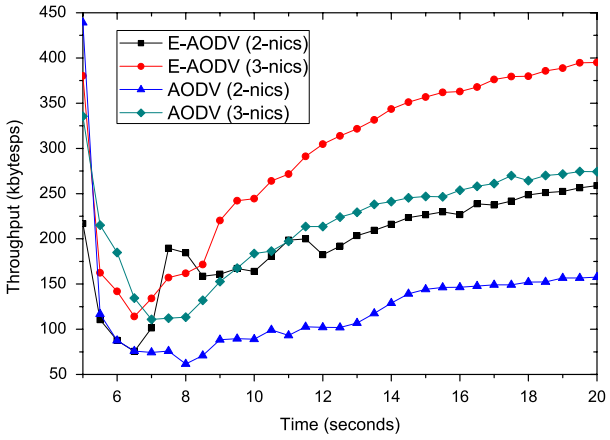

Fig. 7. Throughput comparison with AODV and E-AODV.

5.1. Simulation setup

We select the two-ray ground reflection model. The transmission range is 22 m, so two nodes that are 22 m apart can communicate with each other. The listening range is 44 m, so nodes that are within 44 m can cause interference to each other. We adopt KN-CA in our evaluation. There are 12 channels in the 802.11b network.

We evaluate our BCS with two enhanced routing protocols: AODV and FSR. With the 802.11b environment, the actual maximum throughput B_{th}, with MSDU size of 200 bytes, is 1.21 Mbps. The range of $\mu_{i,j}(x)$ is from 0.49 to 3.848 Mbps according to interfaces per node. We select 491,510, 891,510, and 1,291,510 bytesps (bytes per second) as the initial values. This traffic profile is fixed for all the simulations. There are 25 nodes in this simulation, and each node has up to five interfaces. Four of these interfaces can be switched for data transmission and one interface is fixed as the default control channel. Besides the default control channel, we test two interfaces (2-nics) and three interfaces (3-nics) for data transmission in Section 5.2. In the rest of the evaluations, we test four interfaces (4-nics) for data transmission.

5.2. Performance evaluation of routing protocols

We consider different number of interfaces as our evaluation study. Due the space limitation, we focus on comparison between E-AODV and AODV. The comparison result between E-FSR and FSR has the same trend. We use an 802.11b network environment. We adopt an interference based static channel assignment (KN-CA) and AODV routing in a simulation study. The evaluation consists of the three interfaces (3-nics) and 2 interfaces (2-nics). Fig. 7 gives the comparison results of the proposed AODV routing and enhanced AODV routing (E-AODV).

As Fig. 7 shows, the throughput of our enhanced AODV routing is higher than the original AODV. The 3-nics AODV is better than the 2-nics AODV. The reason is because with enhanced AODV, the selected shortest route is calculated by the common channel. All the pairs of the route are working in the same channel. With 30 heavy traffics, the 3-nics AODV can partake in three different interfaces. The throughput can also be improved.

We can also see that our E-AODV achieves higher reliability than AODV, both in 2-nics and 3-nics cases in regards to the packet loss rate comparison (see Fig. 8). From the above simulation results, we observe that our enhanced scheme for routing protocols performs better in multiple-channel multiple-interface environments. Then, we evaluate our balanced controlling system in the following sections. We will evaluate the BCS with both E-AODV and E-FSR routing protocols, separately.

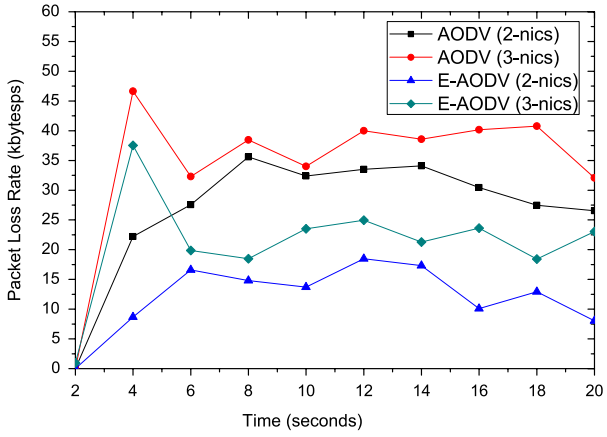

Fig. 8. Packet delay comparison with AODV and E-AODV.

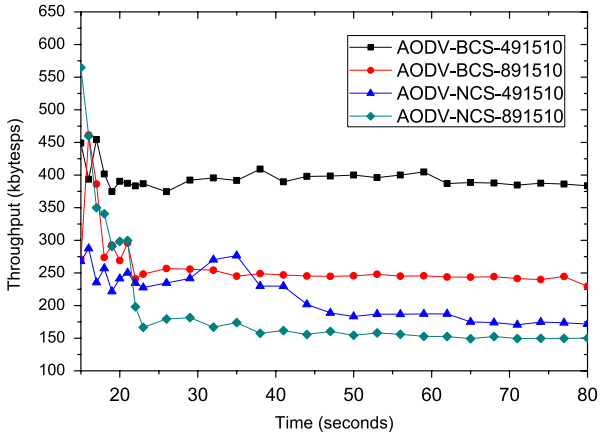

Fig. 9. Throughput comparison between BCS and NCS using E-AODV.

5.3. Performance evaluation with enhanced-AODV

In our solution, if the traffic load is high, the constraint $\mu_{i,j}(x)$ should be updated for that link. This link also needs a new channel that has less interference.

However, it is not easy to determine the exact value $\mu_{i,j}(x)$. So, BCS will try to get $\mu_{i,j}(x)$ as quickly as possible, according to Eq. (5). The simulation time is 80 s, and 30 heavy traffics are added separately, with the interval 0.4 s. Fig. 9 shows the comparison results of the BCS, and without the channel switch control (NCS). The value behind the name in the figures is the initial value of $\mu_{i,j}(0)$.

It is obvious from the throughput comparison that with the same traffic profile, when the system uses the balanced control strategy, the network performance is stable, and also better than the system without the BCS by 40%–70%. With different initial $\mu_{i,j}(0)$, the value of 491,510 bytesps is better than 891,510 bytesps. The reason is that the smaller $\mu_{i,j}(x)$ causes more switches, and the larger value $\mu_{i,j}(x)$ is difficult to achieve. This smaller $\mu_{i,j}(0)$ causes the default channel to take over some of the traffic.

We also give the comparison of packet delay (see Fig. 10). Our BCS is also more efficient than the system without the BCS. The reason for this is that the network is involved in the control system, preventing invalid channel switching. This is desirable because these invalid channel switches decrease the network performance. We also investigate the other parameters, such as packet delay and packet loss rate. With a BCS and the parameter 491,510 bytesps, the system achieves the best performance using the same traffic profile.

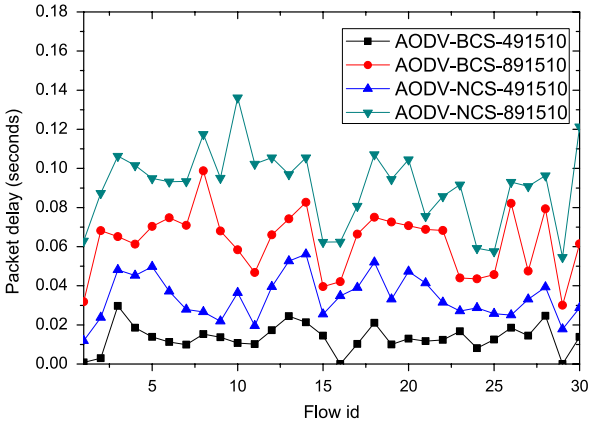

Fig. 10. Packet delay comparison between BCS and NCS using E-AODV.

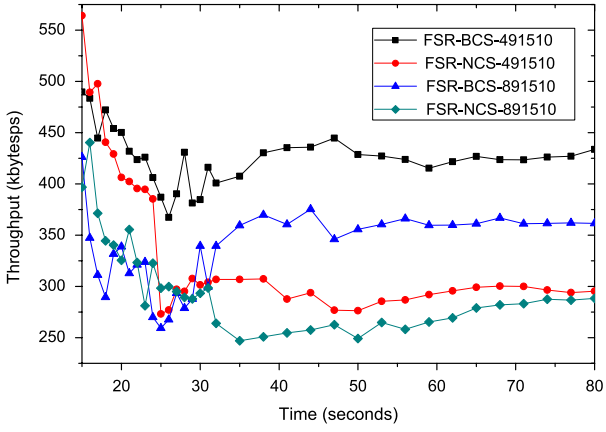

Fig. 11. Throughput comparison between BCS and NCS using E-FSR.

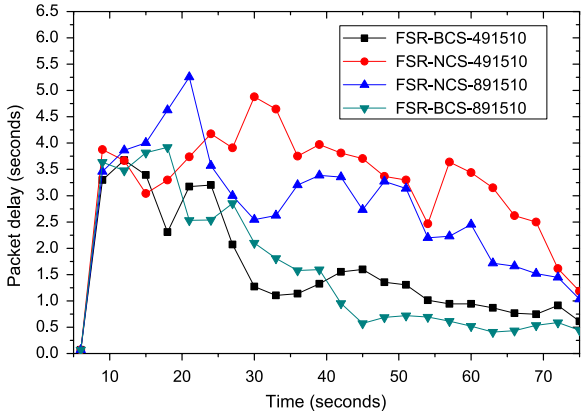

Fig. 12. Packet delay comparison between BCS and NCS using E-FSR.

5.4. Performance evaluation with enhanced-FSR

As Fig. 11 demonstrates, our control system is also better when compared to NCS after 15 s. The performance of $\mu_{i,j}(0)$ is better than $\mu_{i,j}(0) = 891{,}510$. The reason is that 491,510 bytesps is the value that can be easily achieved, thus, some of the traffic is divided to the default channel.

Fig. 12 gives the comparison results of the packet delay. We can see that the packet delay increased before 15 s. The performance is similar among the four situations. This is because the traffic is added as time goes on. Therefore, the packet delay increases. We can also see that when all the traffic is stable after 15 s, the BCS will balance the traffic. It is clear that after 25 s, the performance is

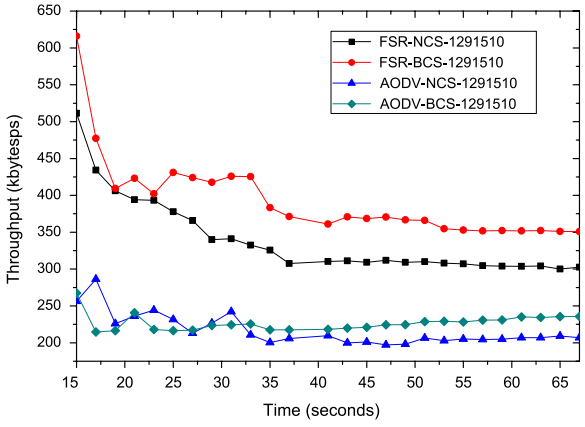

Fig. 13. Throughput comparison between E-AODV and E-FSR.

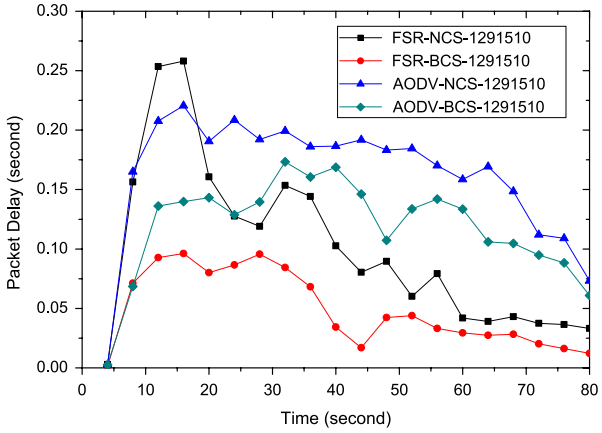

Fig. 14. Packet delay comparison between E-AODV and E-FSR.

better when it is used with the BCS. Also, the system of $\mu_{i,j}(0) = 491{,}510$ is outperforming $\mu_{i,j}(0) = 891{,}510$, both with the BCS and without the BCS.

5.5. Performance comparison with different enhanced routing protocols

This part will demonstrate the comparison between enhanced AODV and FSR routing protocols with the parameter $\mu_{i,j}(0) = 1{,}291{,}510$. We have shown the results of $\mu_{i,j}(0) = 491{,}510$ and $891{,}510$. We can roughly see the throughput results above in Figs. 9 and 11. The performance of the FSR protocol is better. To further verify the result, we select another parameter $\mu_{i,j}(0) = 1{,}291{,}510$ and compare its performance to the one without channel switch control.

Fig. 13 shows the throughput comparison with different routing protocols: AODV and FSR. The performance of the FSR routing protocol is better than AODV routing protocol.

We also present the packet delay and packet loss rate comparison in Figs. 14 and 15. Since our solution continuously adjusts the channel according to the bandwidth and interference, the packet delay has also been decreased. The packet delay is increased before 15 s, and decreased thereafter. The reason for the situation is that we add the traffic, flow by flow, with 0.4 s intervals. After 15 s, 30 flows are stable in the network system, and no more traffic will be added in. But, our solution still adjusts the traffic until no more bandwidth exceeds $\mu_{i,j}(x)$. With the same condition, the channel switch control can quickly find the right parameter, making it more efficient. Also, the packet delay is lower with the FSR routing protocol.

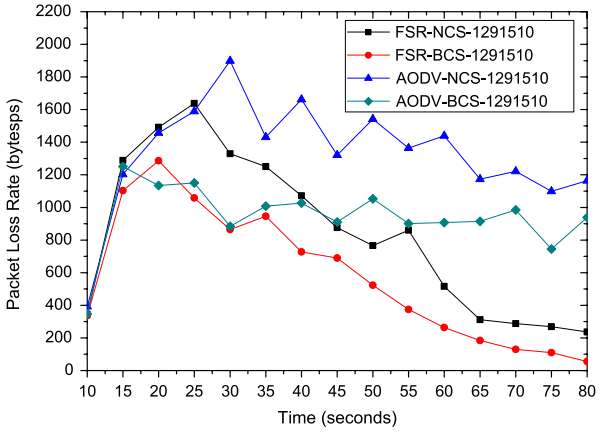

Fig. 15. Packet loss rate comparison between E-AODV and E-FSR.

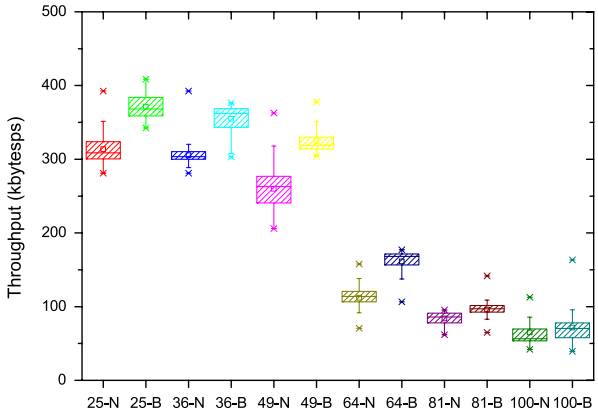

Fig. 16. Throughput comparison with different network sizes.

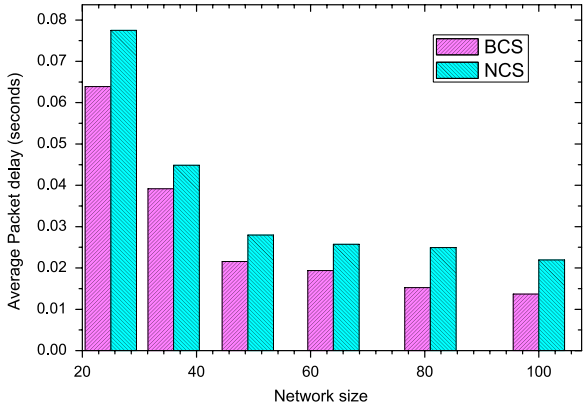

Fig. 17. Packet delay comparison with different network sizes.

5.6. Performance comparison with different network size

In this subsection, we offer the simulation results of different network size. We use the parameter $\mu_{i,j}(0) = 891{,}510$. Since we still want to maintain the connectivity, the grid topology was adopted into the work. We use 5×5–10×10 as the network size. All of the experiments are conducted with the same traffic profile.

Fig. 16 shows the simulation results regarding to different network size, where the label with N and B stand for NCS and BCS, respectively. The results are collected after 240 tests for all the simulations. As the traffic load are the same, the number of packets for each node is smaller. Thus, the throughput is lower. Using our BCS, the throughput can be improved by 10.5%–34.2%. Similar results of average packet delay are also presented in Fig. 17.

Table 5
Confidential interval for throughput (kbps).

Node size	Low (NCS)	High (NCS)	Low (BCS)	High (BCS)
25	272	355	342	400
36	278	332	307	381
49	203	316	294	355
64	83	141	129	193
81	68	100	73	119
100	28	102	37	106

The packets in larger network size have lower latency due to the less traffic load. With BCS, the packet delay can decrease by 14.6%–32.9%. Table 5 shows the confidential interval of throughput within 95%.

The simulation results are summarized as follows:

1. Both E-AODV and E-FSR have superior performance than regular AODV and FSR, respectively.
2. Our proposed balanced control system makes the system more efficient than the normal system in a dynamic environment. The simulation results show that throughput, packet loss rate and packet delay are all better than the system without control.
3. The initial parameter is usually difficult to decide. However, in our simulation study, the system performs better when working with a lower parameter $\mu_{i,j}(0)$.
4. Considering different routing protocols working with our channel switch control, the FSR routing protocol performs better than the AODV routing protocol.

6. Related work

Extensive studies have been done to utilize multiple channels in WMNs. Some works focus on changing MAC protocols [15,6]. In [15], a busy tone is used to show the channel reserving information. However, this MAC protocol cannot be applied directly because it is not compatible with commodity hardware. Protocols of [6] seek to use one interface to exploit multiple channels to improve network performance.

So and Vaidya [28] propose an architecture for multi-channel networks that uses a single interface. Each node has a default channel for receiving data. A node with a packet to transmit has to switch to the channel of the receiver before transmitting data. However, the proposal does not consider the effect of channel switching. The packet has to wait for the delay of the transmission.

A common default channel is introduced in [22,8,27,34] to handle the network partition caused by the dynamic channel assignment, and to facilitate channel negotiation for data communications. To assign channels to the interfaces, [22] presents a localized greedy heuristic based on the interference cost function defined for pairs of channels. [8,27] consider WMNs with main traffic flowing to and from a gateway, which is also in charge of the channel computation. In their channel assignment to a non-default radio, nodes closer to the gateway and/or bearing higher traffic load receive a better quality channel. In DCA [34], the default channel is used as a control channel. For each node, one of the radios stays on the control channel to exchange control messages, and other radios dynamically switch to the data channels for transmission. In this case, the utilization of the control channel could be small, even though the data channels can be fully utilized.

Raniwala et al. [25,24] devise routing and interface assignment algorithms for mesh networks. The protocols are designed to be used in WMNs, where traffic is directed toward specific gateway nodes. Raniwala's protocols assume traffic load between all nodes is already known unlike our work. Moreover, with the load information, interface assignments and route computations are intelligently computed.

When these models are applied in real systems, the impacts of dynamic environment and interference can cause link capacity estimation to be inaccurate, resulting in unstable performance.

Wu et al. [35] describe the design, implementation and evaluation of a WMN system. That system supports both dynamic channel switching and load-balancing/fault-tolerant routing, and successfully runs on low-cost commodity IEEE 802.11-based access points. Azz et al. [1] proposed a new adaptive algorithm, called *Enhance and Explore* (E&E). It maximizes the utility of the network without requiring any explicit characterization of the capacity region. However, the threshold does not feature the channel switching problem. Kim and Shin [7] proposed an autonomous network reconfiguration system (ARS) that enabled a multi-radio WMN to autonomously recover from local link failures.

Ding et al. [5] proposed a hybrid multi-channel multi-radio wireless mesh networking architecture, where each mesh node has both static and dynamic interfaces. In [4], the authors studied a new framework to mitigate the effects of interference in 802.11b/g mesh networks by fully exploiting the spectrum resource. This method utilized both non-overlapping channels and partially overlapping channels. Chieochan and Hossain [3] considered opportunistic listening into the joint problem of channel assignment, network coding, and scheduling in multi-radio WMNs.

In this paper, our attempt is to study fuzzy control and integrate this model into our system to dynamically find the threshold according to the real-time traffic. The aim of our system is to incorporate expert human knowledge in the control algorithm. In this sense, a fuzzy controller may be viewed as a real-time expert system to balance the unstable network. The only work related with fuzzy logic is discussed in [9]. However, that work is based on the QoS considerations for multimedia transmission.

7. Conclusion and future work

In this paper, we proposed a novel threshold-based channel switching system, called balanced channel control system. In our design, the threshold could be dynamically deduced according to the real-time traffic and corresponding throughput of each node. Our threshold control model is based on a fuzzy logic control. We also designed a novel dynamic channel assignment scheme, which is used for the selection of a new channel. To perform a channel switch between a pair of neighboring nodes, we designed a channel switch scheduler. The channel switch scheduler is used to perform channel-switch processing for sender and receiver over enhanced routing protocols. We evaluated this system in NS2, and the simulation results showed that our BCS improves the throughput by 12.3%–72.8% and reduces the latency by 23.2%–52.3% over existing solutions. Our work is not confined to channel switching over a single link, it can be extended to ensure stability over a local region, and hence, the network as a whole. It is well known that interference in reality is very complicated. We plan to conduct real system experiments with our balanced control system. A detailed study of this extension will be our future work.

References

[1] A. Aziz, J. Herzen, R. Merz, S. Shneer, P. Thiran, Enhance & explore: an adaptive algorithm to maximize the utility of wireless networks, in: Proc. of the 17th Annual International Conference on Mobile Computing and Networking, 2011.
[2] P. Bahl, R. Chandra, J. Dunagan, SSCH: slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad hoc wireless networks, in: Proc. of ACM Mobicom, 2004.
[3] S. Chieochan, E. Hossain, Channel assignment for throughput optimization in multi-channel multi-radio wireless mesh networks using network coding, IEEE Transactions on Mobile Computing (2011).
[4] Y. Ding, Y. Huang, G. Zeng, L. Xiao, Using partially overlapping channels to improve throughput in wireless mesh networks, IEEE Transactions on Mobile Computing (2011).
[5] Y. Ding, K. Pongaliur, L. Xiao, Channel allocation and routing in hybrid multi-channel multi-radio wireless mesh networks, IEEE Transactions on Mobile Computing (2011).

[6] N. Jain, S. Das, A. Nasipuri, A multichannel CSMA MAC protocol with receiver based channel selection for multihop wireless networks, in: Proc. ICCCN, 2001.
[7] K. Kim, K. Shin, Self-reconfigurable wireless mesh networks, IEEE/ACM Transactions on Networking (2011).
[8] B.-J. Ko, V. Misra, J. Padhye, D. Rubenstein, Distributed channel assignment in multi-radio 802.11 mesh networks, in: Proc. of IEEE WCNC, 2007.
[9] C.-F. Kuo, H.-W. Tseng, A.-C. Pang, Fuzzy-based cross-layer transmission scheme with QoS considerations for wireless mesh networks, in: Proc. of IWCMC, 2009.
[10] P. Kyasanur, N.H. Vaidya, Capacity of multi-channel wireless networks: impact of number of channels and interfaces, in: Proc. of ACM Mobicom, 2005.
[11] P. Kyasanur, N. Vaidya, Routing and interface assignment in multi-channel multi-interface wireless networks, in: Proc. of IEEE WCNC, 2005.
[12] K. Lee, First course on fuzzy theory and applications, in: Advances in Soft Computing, 2005.
[13] J. Li, C. Blake, D.S. De Couto, H.I. Lee, R. Morris, Capacity of ad hoc wireless networks, in: Proc. of ACM Mobicom, 2001.
[14] X. Li, C. Xu, Joint channel assignment and routing in real time wireless mesh network, in: Proc. of IEEE WCNC, 2009.
[15] R. Maheshwari, H. Gupta, S. Das, Multichannel MAC protocols for wireless networks, in: Proc. of IEEE SECON, 2006.
[16] H. Matsumura, Commutative Algebra, Addison Wesley, 1998.
[17] J. Padhye, S. Agarwal, V. Padmanabhan, L. Qiu, A. Rao, B. Zill, Estimation of link interference in static multi-hop wireless networks, in: Proc. of ACM SIGCOMM Conference on Internet Measurement, 2005.
[18] K.M. Passino, S. Yurkovich, Fuzzy Control, Addison Wesley, Longman, 1998.
[19] I. Pefkianakis, Y. Hu, S.H.Y. Wong, H. Yang, S. Lu, Mimo rate adaptation in 802.11n wireless networks, in: Proc. of ACM Mobicom, 2010.
[20] G. Pei, M. Gerla, T. Chen, Fisheye state routing in mobile ad hoc networks, in: Proc. of IEEE ICDCS Workshop on Wireless Networks and Mobile Computing, 2000.
[21] C. Perkins, E. Belding-Royer, S. Das, Ad hoc on-demand distance vector (AODV) routing, Request For Comments, RFC, 2003.
[22] K.N. Ramachandran, E.M. Belding, K.C. Almeroth, M.M. Buddhikot, Interference-aware channel assignment in multi-radio wireless mesh networks, in: Proc. of IEEE Infocom, 2006.
[23] V. Raman, N. Vaidya, Adjacent channel interference reduction in multichannel wireless networks using intelligent channel allocation, Technical Report, 2009.
[24] A. Raniwala, T.-C. Chiueh, Architecture and algorithms for an IEEE 802.11 based multi-channel wireless mesh network, in: Proc. of IEEE Infocom, 2005.
[25] A. Raniwala, K. Gopalan, T.-C. Chiueh, Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks, in: ACM SIGMOBILE MC2R, vol. 8, 2004.
[26] J. Shi, T. Salonidis, E.W. Knightly, Starvation mitigation through multi-channel coordination in CSMA multi-hop wireless networks, in: Proc. of ACM Mobihoc, 2006.
[27] H. Skalli, S. Das, L. Lenzini, M. Conti, Traffic and interference aware channel assignment for multi-radio wirelesss mesh networks, Technical Report, Institutions Markets Technologies, 2006.
[28] J. So, N. Vaidya, A routing protocol for utilizing multiple channels in multi-hop wireless networks with a single transceiver, Technical Report, University of Illinois at Urbana-Champaign, 2004.
[29] J. So, N.H. Vaidya, Multi-channel MAC for ad hoc networks: handling multi-channel hidden terminals using a single transceiver, in: Proc. of ACM Mobihoc, 2004.
[30] A. Subramanian, H. Gupta, S. Das, Minimum interference channel assignment in multi-radio wireless mesh networks, in: Proc. of IEEE SECON, 2007.
[31] J. Tang, G. Xue, W. Zhang, Interference-aware topology control and QoS routing in multi-channel wireless mesh networks, in: Proc. of ACM Mobihoc, 2000.
[32] Wireless medium access control (MAC) and physical layer (PHY) specifications: ESS mesh networking, IEEE P802.11s/D1.00, 2006.
[33] S.H.Y. Wong, H. Yang, S. Lu, V. Bharghavan, Robust rate adaptation for 802.11 wireless networks, in: Proc. of ACM Mobicom, 2006.
[34] S.-L. Wu, C.-Y. Lin, Y.-C. Tseng, J.-L. Sheu, A new multi-channel MAC protocol with on-demand channel assignment for multi-hop mobile ad hoc networks, in: Proc. of the International Symposium on Parallel Architectures, Algorithms and Networks, 2000.
[35] G. Wu, S. Singh, T. Chiueh, Implementation of dynamic channel switching on IEEE 802.11-based wireless mesh networks, in: Proc. of the 4th Annual International Conference on Wireless Internet, 2008.
[36] J. Zhu, S. Roy, 802.11 mesh networks with two-radio access points, in: Proc. of IEEE ICC, 2005.

Xiaoguang Li received her B.Sc. in Information and Computational Science from Nanjing University of Science and Technology in 2004, and her M.Eng. in Software Engineering from Southeast University in 2006. She is currently working toward a Ph.D. degree in the Department of Computer and Information Sciences at Temple University. Her current research interests mainly focus on wireless sensor networks, delay tolerant networks and wireless mesh networks.

Jie Wu is Chair and Professor at the Department of Computer and Information Sciences at Temple University, USA. He is an IEEE Fellow. He is on the editorial board of IEEE Transactions on Mobile Computing. He was a Distinguished Professor in the Department of Computer Science and Engineering, Florida Atlantic University. He served as a Program Director at US NSF from 2006 to 2008. He has been on the editorial board of IEEE Transactions on Parallel and Distributed Systems. He has served as a Distinguished Visitor of the IEEE Computer Society, and is the chairman of the IEEE Technical Committee on Distributed Processing (TCDP). His research interests include wireless networks and mobile computing, parallel and distributed systems, and fault-tolerant systems.

Shan Lin received his B.E. degree in computer science and engineering from Shanghai Jiao Tong University in 2004, and his Ph.D. degree in Computer Science from the University of Virginia in 2010. He joined Temple University as a Tenure Track Assistant Professor in 2010. He was awarded as the SAIC scholar in 2009. Lin's primary research interests are in the areas of cyber physical systems, networked embedded systems, and wireless sensor networks. He has been investigating feedback control based approaches to networked system designs, including wireless networking and system composition. He is also involved in building wireless sensing systems for pervasive medical care and fire fighting. His major papers have been published in ACM MobiSys, ACM SenSys, IEEE RTSS, IEEE INFOCOM, ACM TECS, and ACM IPSN. He is a member of IEEE and ACM.

Xiaojiang Du is currently an associate professor in the Department of Computer and Information Sciences at Temple University. Dr. Du received his B.S. degree in electrical engineering from Tsinghua University, Beijing, China in 1996, and his M.S. and Ph.D. degree in electrical engineering from the University of Maryland College Park in 2002 and 2003, respectively. Dr. Du was an Assistant Professor in the Department of Computer Science at North Dakota State University between August 2004 and July 2009, where he received the Excellence in Research Award in May 2009. His research interests are security, wireless networks, computer networks and systems. He has published over 100 journal and conference papers in these areas, and has been awarded more than $2M research grants from the US National Science Foundation (NSF) and Army Research Office. He serves on the editorial boards of four international journals. Dr. Du is the Chair of the Computer and Network Security Symposium of the IEEE/ACM International Wireless Communication and Mobile Computing conference 2006–2010. He is a Technical Program Committee (TPC) member of several premier ACM/IEEE conferences such as INFOCOM (2007–2012), IM, NOMS, ICC, GLOBECOM, WCNC, BroadNet, and IPCCC. Dr. Du is a Senior Member of IEEE and a Life Member of ACM.