July 24, 2018

CONCURRENT TRANSMISSION MECHANISM TO IMPROVE THROUGHPUT OF BOTTLENECK DEVICES IN LOW-POWER AND LOSSY NETWORKS

Dechen Xu

Huimin She

Xiang Fang

Chuanwei Li

Xueying Yang

CONCURRENT TRANSMISSION MECHANISM TO IMPROVE THROUGHPUT OF BOTTLENECK DEVICES IN LOW-POWER AND LOSSY NETWORKS

AUTHORS:
Dechen Xu
Huimin She
Xiang Fang
Chuanwei Li
Xueying Yang

ABSTRACT

Techniques are described herein for designing a concurrent channel mechanism to cause mesh network border router devices or key/bottleneck devices to communicate with multiple devices concurrently. This may reduce collision probabilities and thus enhance the scaling ability of a network.

DETAILED DESCRIPTION

In wireless device communication mechanisms such as mesh networks, frequency hopping technology is a method of transmitting radio signals by rapidly switching a carrier among multiple channels using a pseudo-random sequence known to both transmitter and receiver. Advantages to frequency-hopped transmissions include resistance to interference/collisions and being difficult to intercept.

As illustrated in Figure 1 below, in a mesh network the border router device and the extender device may create a bottleneck as the scaling increments.

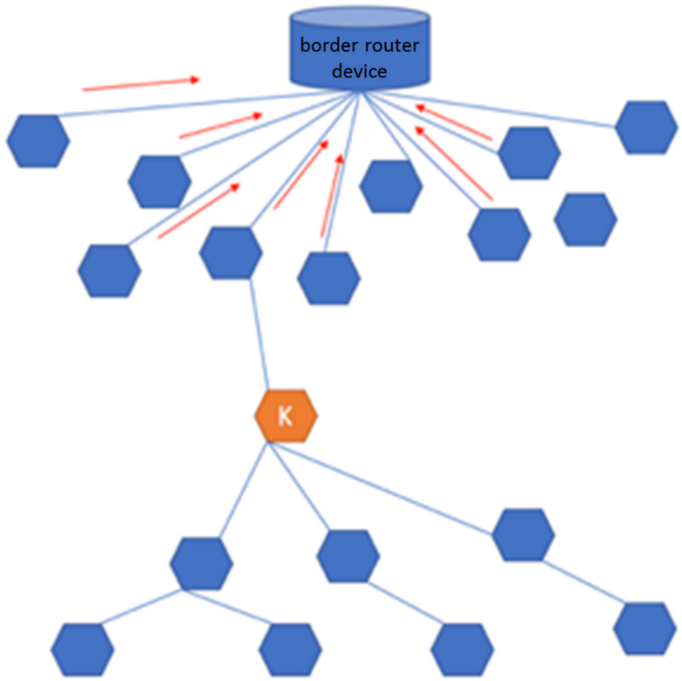

Figure 1

In large-scale multi-hop networks, when many first hop nodes send data to a border router device simultaneously, there are high probabilities of collisions. Other nodes need to wait for back-off time and retransmissions because the border router device is only able to receive data on one channel in one time slot. Moreover, in a real deployment field, in some areas there might be several nodes deployed. As such, the extender device K may be used as a relay, which can also create a bottleneck.

Accordingly, a concurrent transmission mechanism is described herein to improve the bottleneck throughput for large-scale low-power and lossy networks. This mechanism may cause the key/bottleneck devices to support receiving/transmitting data on multiple channels concurrently.

In order to support the concurrent data receiving/transmitting, on the hardware side, the interface may have the ability to receive/transmit on different channels concurrently. There are multiple options to implement this. The first option uses multiple Radio Frequency (RF) chips and one Multipoint Control Unit (MCU) for control. The MCU coordinates the RF chips' working channels and combines the data received on multiple channels. The second option involves a Software Defined Radio whose antenna receives the whole radio signal frequency range and use the software to filter each channel and demodulate the signal, and vice versa. The third option includes Application Specific Integrated Circuits (ASICs) to support the concurrent channels.

In Institute of Electrical and Electronics Engineers (IEEE) 802.15.4 mesh frequency hopping networks, when a device needs to send data to its neighbor, the device calculates the receiving channel of its target neighbor device and sends the data on that channel. That is, if two or more devices send data to the same target simultaneously, these devices send the data on the same channel, after which the collision occurs.

Figure 2 below illustrates how to cause the devices to receive on multiple channels simultaneously.

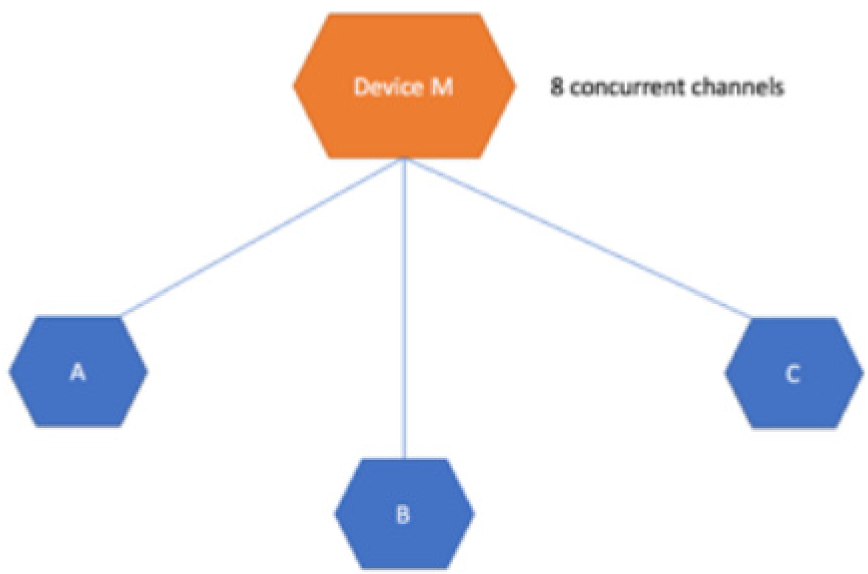

Figure 2

Figure 2 above shows the example topology, with device M supporting the ability of eight concurrent channels.

In a first example step, the device M adds an Information Element (IE) in 802.15.4 frames, including the number of concurrent channels it supports. When its neighbor devices receive the frame and read the IE, they determine that device M supports concurrent channels as well as the number of the concurrent channels.

In a second example step, the original frequency hopping algorithm is used to calculate the channel schedule time slot. The channel number is treated as the base channel. For example, since device M supports eight concurrent channels, another seven channels may be chosen as extended available channels. In order to reduce the interference between two channels, the adjacent channels may not be used. Here, the channel number spacing may be calculated with the supported channel divided by the concurrent channel capacity. For device M, the channel spacing is 64/8=8. Another seven extended channels may then be calculated. Figure 3 below illustrates an example table.

PHY Mode Channel Number = 64
Channel Range = 0-63

	1	2	3	4	5
Base Channel	7	16	21	63	58
	15	24	29	7	2
	23	32	37	15	10
	31	40	45	23	18
	39	48	53	31	26
	47	56	61	39	34
	55	0	5	47	42
	63	8	13	55	50

Figure 3

In a third example step, when device A needs to send data to device M, device A has multiple choices of channels that can be used to send the data since device A knows how many concurrent channels device M supports and the base channel. It can use a hash algorithm with the node's Extended Unique Identifier (EUI) address as input to calculate an index to choose the channel in order to make a different node choose a different channel number. As illustrated in Figure 4 below, devices A, B, and C send data to device M concurrently on different channels based on the hash result.

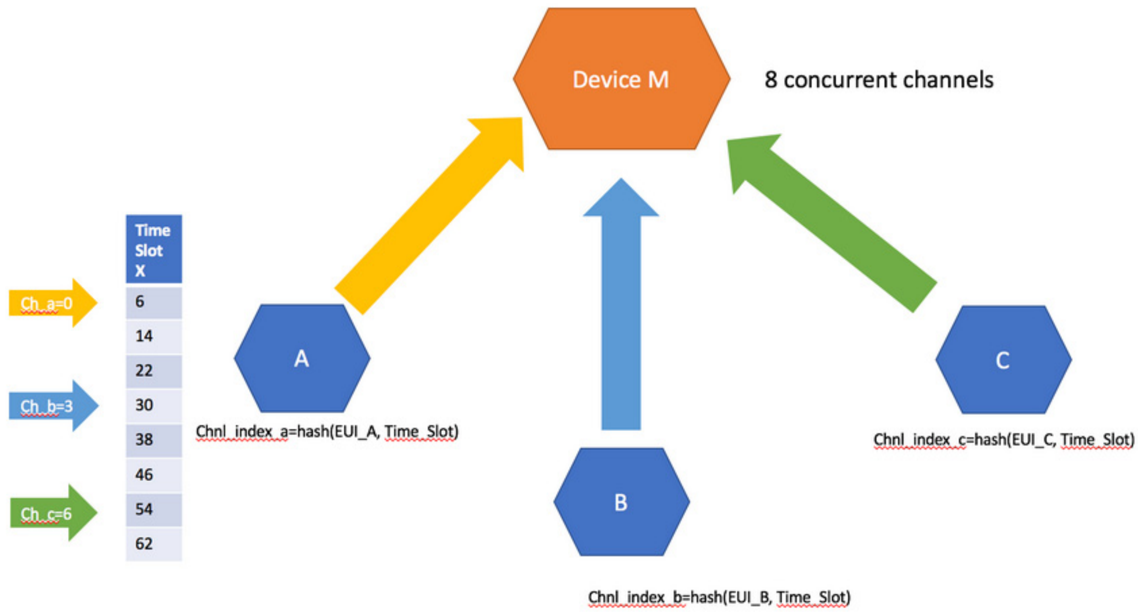

Figure 4

In this case, all three devices are able to send data to device M concurrently and collisions are avoided.

Data transmitting is described with relation to two different cases. In the first case, the target device does not support the concurrent channel. In this case, the target device only supports receiving data on one channel in a time slot. Here, even if the source device supports concurrent channels, it still only can send data on one channel. But considering there should be more data in the buffer that is ready to be sent, if the next data is ready to be sent to different targets and on different channels, that data can be sent concurrently. Figure 5 below illustrates how a device sends data to multiple targets concurrently.

Seq	Target
1	A
2	B
3	B
4	A
5	C
6	C

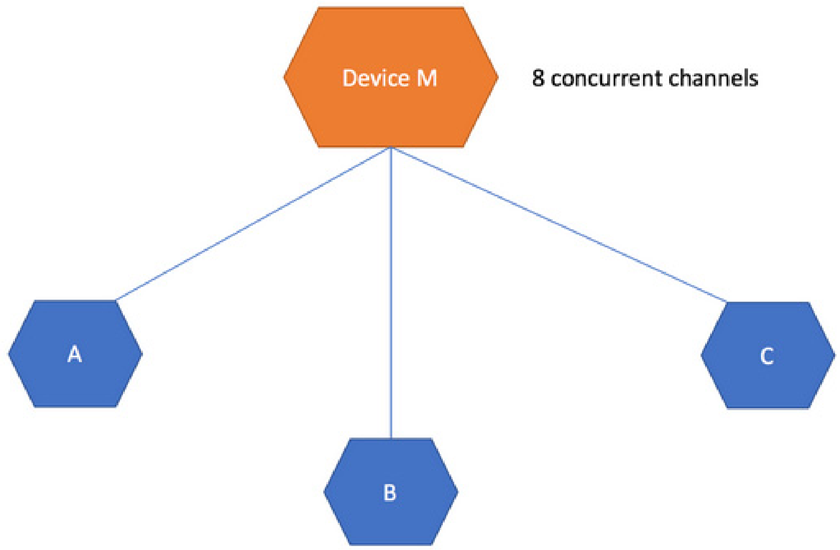

Figure 5

In a first example step, in the device M buffer, there are six data items to be sent to different devices. Device M obtains the next up to the concurrent channel number data items which are ready to be sent to different targets. In this example, data items 1, 2, and 5 are selected and placed in a temporary table.

In a second example step, the channel number is calculated for each target node. If the channel numbers are the same, an item may be randomly removed from the table, thereby ensuring all the channel number are non-repeating. The sending table may then be created.

In a third example step, all the data in the sending table is sent simultaneously.

Figure 6 below illustrates an example transmitting procedure.

Send Buffer Before Sending

Sequence	Target
1	A
2	B
3	B
4	A
5	C
6	C

Temp Table

Sequence	Target	Channel
1	A	5
2	B	18
5	C	5

Final Sending Table

Sequence	Target	Channel
1	A	5
2	B	18

Send Buffer After Sending

Sequence	Target
3	B
4	A
5	C
6	C

Figure 6

In the second case, the target device supports concurrent channels. Here, if there is continuously data for this target in the send buffer, this data may be sent simultaneously. Following the mechanism introduced in the first case, if the target supports concurrent channels, the data may be selected repeatedly. A hash algorithm may be used to calculate the channel number index and select the available channel for the target device, which supports the concurrent channel. As represented by the green numbers shown in Figure 7 below, target device A supports concurrent channels.

Send Buffer

Sequence	Target
1	A
2	B
3	B
4	A
5	C
6	C
7	A
8	A
9	A
10	A

Temp Table

Sequence	Target	Channel
1	A	5
2	B	18
4	A	5
5	C	7
7	A	13
8	A	21
9	A	29
10	A	37

Sending Table

Sequence	Target	Channel
1	A	5
2	B	18
5	C	7
7	A	13
8	A	21
9	A	29
10	A	37

Figure 7

In summary, techniques are described herein for designing a concurrent channel mechanism to cause mesh network border router devices or key/bottleneck devices to communicate with multiple devices concurrently. This may reduce collision probabilities and thus enhance the scaling ability of a network