Chapter 2 – Literature Study

2.1 Headers and different levels of encapsulation

In the OSI stack all layers have a specific purpose other than simply forwarding data. Most layers have a need to attach data along with the data it is forwarding to accompany it. This metadata provides important information regarding the accompanying data, so that the receiving party knows how to handle and interpret the data at each layer. The added data is called a header, or overhead. An example of layer adding data would be the network layer adding an IP address so that the network knows where to send the data.

Most layers also divide the data they receive into smaller parts that are more favourable to handle by the respective layer. The different parts of the data are then put back together at the receiving party's corresponding layer. Figure 2-1 illustrates how this process works.

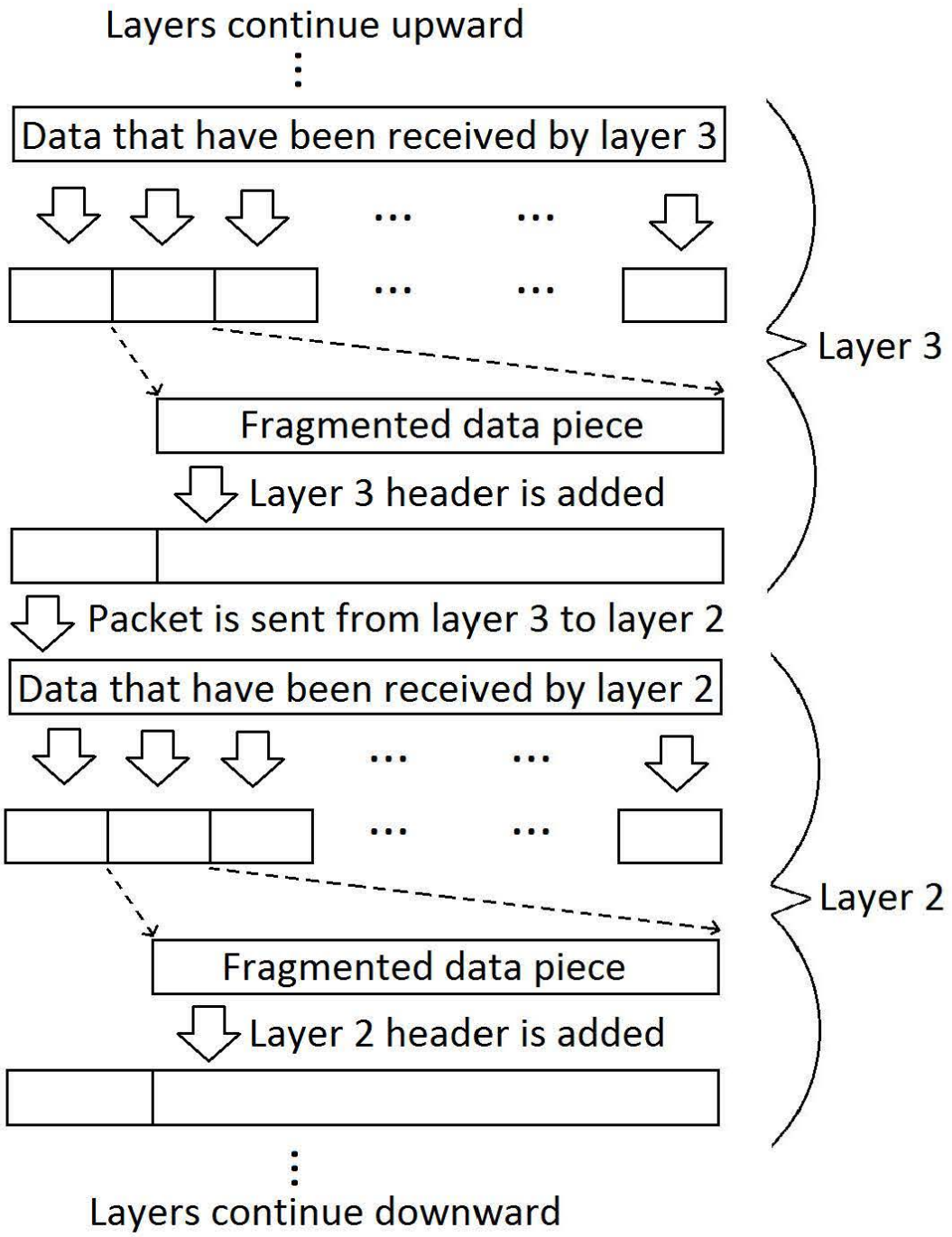

Figure 2-1: The encapsulation process

2.2 Different standards and protocols in the OSI stack for a wireless ad hoc network

This section gives a brief discussion of the purpose, as well as how each layer in the OSI stack works. Since 802.11 (Wi-Fi) is arguably the most successful [**29**] and most popular [**30**] of personal wireless technologies for ad hoc networks, the relevant standards of Wi-Fi at each layer will be discussed for this case. How network coding can be implemented in every layer is also discussed.

2.2.1 Physical layer

This layer represents the physical aspects of a network; this includes wires, connectors, antennae, transmission mediums and any other hardware. This is the determination point for a device on a network, and it is responsible for the physical signal being transmitted or received.

All the mechanical and electrical standards for the network are determined by the physical layer protocol used. Wi-Fi devices either function at a frequency of 2.4 or 5 GHz. Any antennae which are matched to these frequencies can be used [**31**].

There are three multiplexing techniques used in 802.11 [**32**,**31**] that are relevant to this study, each using different modulation strategies. The three are:

- Direct-Sequence Spread Spectrum (DSSS). This technique uses an 11-bit Barker sequence, so each information bit (or symbol) is coded by an 11 chip sequence, yielding a chip rate of 11 Mchip/s. Either binary phase shift keying (BPSK) or quadrature phase shift keying (QPSK) is used to yield rates of 1 and 2 Mb/s, respectively. A modulation rate of 1 Msymbol/s is used with the chosen modulation scheme.
- Complementary Code Keying (CCK). Here a 16-bit sequence codifies either 4 or 8 information bits when it is sent on the channel, yielding a chip rate of 11 Mchip/s. It uses QPSK to modulate 1.375 Msymbol/s for a rate of either 5.5 or 11Mb/s with 802.11b.
- Orthogonal frequency-division multiplexing (OFDM). With this scheme a comb of 52 subcarriers with a spacing of 0.3125 MHz is used with a symbol duration of 4μs, for 12 Msymbol/s. Only 48 of these combs are used for data. A convolutional code protects each symbol with rates of either 3/4, 2/3, or 1/2. Quadrature amplitude modulation (*M*-QAM) is used with *M* being 2, 4, 16, or 64. OFDM is defined in 802.11a and also used in 802.11g and the above combinations result in signal rates of 6, 9, 12, 18, 24, 48, and 54 Mb/s.

Implementation of network coding in the physical layer is made possible by analogue network coding. [**19**] describes how to implement analogue network coding with practical modulation schemes such as M-PSK and MQAM.

An advantage of physical network coding is that it uses even less time to transmit the same data as logical network coding. On the other hand, an advantage of logical network coding is that if one node sends the coded message, multiple nodes can receive it. Physical network coding can also achieve this, but since multiple nodes must send simultaneously to combine a message, all the sending nodes must be in range of the receiving nodes, and this becomes a problem when more than two messages need to be sent to more than one receiver. The protocol necessary for organising the synchronisation of physical network coding is much more complex and causes more overhead than logical network coding. Another problem, unique to physical layer network coding, is network coding noise [**33**], which can sometimes worsen the performance of a network using cooperative communications to an extent that no network coding delivers better results. Physical layer network coding was first implemented in [**34**]. In [**35**], the authors explain how to adapt the modulation for analogue network coding, and propose a new modulation scheme.

2.2.2 Data link layer

This layer is responsible for reliable communication over the physical layer. It makes provision for frame synchronization or control which means that the layer controls the timing of when data is sent to the physical layer, the rate at which data is sent, the structure of the frames and error detection and correction. To achieve this, the layer is subdivided into two other layers:

2.2.2.1 Media Access Control (MAC)

The MAC sublayer provides the access to the transmission medium and control over when which frames are sent. For example in most wireless networks the MAC is responsible for making sure that the data that needs to be transmitted, is sent at the right time to avoid causing collisions in the channel. If a collision occurs, a collision signal is sent and all devices that are trying to use the channel back off for a random period of time, to ensure that the devices wanting to reconnect do not do so simultaneously. The MAC also takes into account the prioritisation of frames, and sends the frames with higher priority first. The prioritisation of data is useful because since data that is sensitive to delay e.g. Voice Over Internet Protocol (VOIP), can be sent with a higher priority than other data that also needs to be sent, resulting in lower delay.

For the 802.11 standard in an ad hoc environment, Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) is used. The CSMA ensures that no two stations within the same transmission range and operating on the same channel transmit simultaneously by looking at the channel signal energy. When it is above a certain value the channel is considered busy, and when below, it is free and the listening station can transmit. A station will not start transmitting if the channel is busy in

order to avoid collisions. The CA is a low level acknowledgement system. If a station sees a quiet channel and wants to transmit, it will send a request to send, on which a reply will follow if it is requesting a unicast session and the receiver is not engaged in another transaction. This helps to minimize the effect of the hidden node problem.

Many network coding applications require the ability to multicast messages. 802.11 does not support sending multicasts at the MAC layer, but uses broadcasts at its base transmission rate for most wireless multicasts routing protocols [**36**]. This has to be taken into account if network coding is to be implemented in this layer. Chen *et al*. proposes a system that enables multicasting in the MAC [**37**]. 802.11 was not originally created for ad hoc networks, and therefore there are performance issues [**38**] when it is implemented in an ad hoc network. These could be solved by designing an efficient network coding protocol in the MAC for 802.11.

2.2.2.2 Logical Link Control (LLC)

The logical link control provides the link between the upper layers of the stack to the actual underlying network. The LLC is responsible for finding and connecting to the rest of the network. To the upper layers, a connection made by the LLC looks to be a direct point-to-point connection. The LLC provides two basic modes of operation: the connection orientated (CO) mode creates and terminates a link and requires acknowledgements from the receiver so that the sender can know that all the data was successfully transmitted, or if an error occurred. The connectionless (CL) mode merely sends datagrams without a connection or acknowledgements at a best effort approach.

Network coding is often implemented in the data link layer because it is the first layer capable of implementing network coding logically – in other words combining two or more messages with a logical calculation rather than doing so by using interference, as is done with physical network coding. The reason one wants to implement network coding in a layer as low as possible is that the lower the layer, the more efficient network coding is. Each layer has to send its own overhead and the higher network coding is in the stack, all the extra data produced by the lower layers are not network coded but sent normally. Since the data link layer has no routing abilities, routing has to be built into the data link layer if network coding is to be applied here. The reason that the network layer cannot handle the routing any more in this case is because only if the data is decoded at the receiving node can the data make sense again, so the network layer's data will be encoded and will be irrational to the whole network except the sender and receiver. Applying network coding in the data link therefore nullifies the network layer when network coding is applied, and makes the data link layer very complex. The work in [**10**] suggest that opportunistic network coding in a one-hop area for each node can yield gains of up to 5 times that of a similar network without network coding. This gain is even more than the gain expected from theoretical results. In [**23**] a scheme is presented that improves the performance over TCP connections, but network coding itself is done in the MAC.

2.2.3 Network layer

The network layer is responsible for creating and clearing a connection between the two transport layers of the sender and receiver in the network. To do this, addressing and routing are handled by the network protocol. The Internet Protocol version 4 (IPv4) is by far the most popular network layer protocol used today, and although a newer version; version 6 is available (Version 6 was designed because version 4 does not support enough addresses for today's needs). IPv4 is still today mostly used in conjunction with NAT or Network Address Translation: meaning several devices can share one address. NAT provides the problem that a device is not always traceable in a network because several devices use one IP address. With today's ever increasing need for security, an attempt to convert networks to IPv6 is being made because IPv6 has enough unique addresses for all devices on the internet.

Two common routing protocols[**4**] for ad hoc networks are Ad Hoc on demand Distance Vector (AODV) and Dynamic Source Routing (DSR). Hop-count is a term used to describe a routing metric. It counts the relays or hops from the source to destination and uses the number as indication of which route would be the shortest (not necessarily always the fastest). Both AODV and DSR use hop-count to find the shortest route in a network. Both are also reactive, meaning they find a route to the destination when it becomes necessary. For ad hoc networks one can also be proactive on the other hand. This means one can make provision for route establishment before a new route is needed. With proactive protocols like Open Shortest Path First (OSPF), each node in the network has a complete list of connectivity for each forwarding station in the network, and can thus immediately start sending data when a transmission is requested – it does not have to find a route first. Proactive protocols are thus faster, but do not expand well in larger networks because the overhead for updating connectivity lists becomes too much. Many new protocols [**39**] emerged that are based on basic AODV or DSR. Usually they modify the old protocol by making it to take other link parameters into account such as delay or signal strength. These newer protocols can make the network yield greater throughput, give feedback regarding QoS of routes, or have less delay, etcetera. The authors in [**5**] give an overview and performance comparison of the most popular routing protocols, showing each protocol's strengths and weaknesses. Hybrid proactive and reactive protocols are proposed in [**40**] [**41**][**42**]. A performance evaluation on the different routing protocols for ad hoc networks is given in [**4**], and suggests that AODV performs best overall.

The fact that the network layer handles routing, also makes it a good choice for implementing deterministic network coding because it has to take the routes that messages follow into account before it can combine messages. In other words, deterministic network coding and routing go hand in hand. It would be a good idea to build network coding into an already existing routing protocol for ad hoc networks, since the basic mechanism for route establishment could be used as a base and then building thereon, so that network coding can be implemented. The problem is that finding a route is no longer as simple as with a point to point link because of the restrictions described in the network coding theory section, and one has to take the routes of multiple senders into account

simultaneously. As the number of users that want to send data increases, or the network size increases, the algorithms for calculating the routing for the network coding increase drastically. A solution to this would be to implement random network coding. Combining random network coding with direct diffusion [**43**] is proposed as a solution in [**44**]. An analysis of routing for network coding is provided in [**45**].

2.2.4 Transport layer

The transport layer provides a protocol that can transport messages from one process to another, and ensure that the right messages are supplied to the right process. The network layer takes into account how the network architecture looks because it must handle routing between sender and receiver. The transport layer, on the other hand, is not concerned with the network topology; it is responsible for making sure that the right messages get sent to the right processes on the sender and receiver. For example: if one is using two network applications on one computer, the transport layer is responsible for sending the right data which it receives from the network layer to the right application on the computer. The transport layer is also responsible for setting up a link from the sender to the receiver's transport layers, addressing, flow control and handling a broken link. There are two prominent transport layer protocols used in the internet today: TCP and UDP.

2.2.4.1 TCP

Transmission Control Protocol (TCP) is a connection-orientated protocol designed to use over IP. Since IP is an unreliable transfer mechanism, TCP has to provide a means to ensure that any lost or corrupted data is replaced before it is delivered to the upper layers. It does so by requesting any lost or corrupted data to be sent again. Processes are handled by giving each a port through which it can connect, and therefore it can handle several processes simultaneously, using the network interface. Since the process's communication are handled by establishing a connection with the receiving end's TCP, each process can only communicate with one other process through the established link. The error and flow control make TCP ideal for applications which have to have a reliable means of data transfer to ensure that all data are delivered error-free.

2.2.4.2 UDP

The User Datagram Service (UDP) uses a connectionless mechanism for processes from higher layers to communicate. The fact that UDP is connectionless has the advantage that it adds less overhead to all the data it sends. The disadvantage is that UDP has no error detection or flow control. UDP is faster because it sends less overhead and doesn not have to establish and destroy connections, and therefore has less delay than TCP. This makes it ideal for some real-time applications such as Voice

Over Internet Protocol (VOIP), which cannot afford the delay caused by retransmission of lost data. Broadcasts are sent over UDP too, since creating connections would not be appropriate.

The transport layer is designed to be point to point orientated between the sender and receiver, not taking the rest of the network into account and therefore, is not an ideal layer for implementing network coding with the intent of gaining faster throughput. Since the transport layer is responsible for reliable transmissions, random network coding can be used to create a more robust and secure link on the other hand. TCP uses acknowledgement packets to ensure reliable point to point communication. In a random network coding environment these acknowledgement messages could become a significant amount of data on the network. A scheme implementing network coding with reliable multicast sessions by means of acknowledgement messages, and which causes minimum delay, is presented in [**46**]. A technique which takes into account the queue size of both the ends of a link over the transport layer, and accordingly implements distributed flow control works well with random network coding [**47**]. A network coding scheme that is built into TCP is presented in [**48**]. It uses a form of random network coding with a specialized acknowledgement system.

2.2.5 Session layer

The session layer is responsible for synchronisation between pairs of processes, organising whose turn it is to send. One always sends data to someone, and thus a sending and receiving pair is formed when a transmission starts. The session layer at each side ensures that each member of the pair knows when to send data, thus ensuring the orderly exchange of data.

This layer is not favourable for the implementation of network coding. It plays a supportive role to the application layer and is not designed for routing or transportation of data.

2.2.6 Presentation layer

The presentation layer ensures that a pair of communicating users agree on the syntax that is used while application processes are communicating. Certain standards are needed to ensure that the data sent, can be understood by the receiving party. The following are examples of services provided by the presentation layer: file transfer, virtual terminal protocols, compression, code transformation and compression.

This layer is not favourable for the implementation of network coding. It plays a supportive role to the application layer and is not designed for routing or transportation of data.

2.2.7 Application layer

The application layer is the link from the OSI stack to the user and to any other process, application or program. If a user or process wants to send data over a network, the application layer is there to receive the data, and send it down the stack and into the network. It is also there to present or hand over requested data to the user or program. Examples include distributed data bases, electronic mail, or a browser.

This layer supports the applications to send data over a network. It is possible to implement network coding in the application layer, but once again, this layer is not designed for routing, and all intermediate nodes have to always run a network coding application for it to work. Network coding would work more efficiently in the lower layers, but development would be easier because most programmers are familiar with application development but not necessarily with development on lower layers. Also, since each of the lower layers is responsible for many other functions other than network coding data, when network coding is implemented in each layer the function of network coding is built upon other functions. When a specific network coding application is developed, this is not the case and development could be simpler since any existing network could probably be used without it needing to be altered. Avalanche [**49**] is a system developed by Microsoft, and makes use of random network coding in the application layer. If a node in the system wants to download a large file, it downloads linear combinations of fragments of the file from other nodes over a peer-to-peer overlay network. It collects fragments until it can recover the complete file. The results in [**49**] show increased robustness against sudden changes in the network and reductions in download time. For each packet that is sent using random network coding, coding coefficients have to be put in the header, and the larger the packets, the smaller the ratio of the headers to data. In wireless networks however the packet lengths have to be restricted and the percentage of overhead in the network increases [**21**]. The fact that random network coding has to wait for enough linearly independent combinations of the data that has potential to take longer than sending data without network coding. The fact that the decoding process has high complexity and may take significant time to complete, has raised questions about the feasibility of real-time applications using network coding. The work in [**50**], however, shows promising results concerning this problem, and proposes a live peer-to-peer streaming scheme called R^2. R^2 is compared to conventional peer-to-peer applications without network coding, and shows clear advantages above conventional peer-to-peer applications.

2.3 Influences of cross-layer capabilities and adding a network coding layer

Sometimes network coding requires information from more than one layer of the OSI stack, and implementing network coding in only one layer could leave functions in other layers redundant or simply limit its full potential. Also when data is coded, it is inaccessible before it is decoded. So when network coding is implemented in a lower layer in the stack for instance, the upper layers become invisible to the network, and since the network generally needs routing details or the data content for prioritizing frames, some functionality might be lost when network coding is implemented in a layer lower than the ones responsible for routing or prioritisation. This leads to questioning which layers must be left unaltered, which layers will be network coded and thus invisible to the network, and which layers are altered to implement network coding? Two solutions are suggested for wireless networks: a cross-layered approach [**51**,**52**,**18**,**53**,**54**,**55**], and introducing a specialized network coding layer [**10**]. The cross layered approach works by allowing different layers to communicate with each other, and by doing this, one can take all the layers into consideration when implementing network coding. The drawback that this has, is that all communicating layers have to be altered to support this functionality, and the whole idea of having independent layers for ease in development, is lost.

The idea of putting in a whole new network coding layer appears to be an optimum solution in one sense, because it does not have to take all the other functions performed by the other layers into account, or change the way the layers function. This could also be a disadvantage because one cannot build on the existing structures or algorithms used in a relevant layer. If one implements network coding in the network layer for instance, the layer already has mechanisms for routing. These could then be used to aid the implementation of network coding. When a new layer is created, the functions in other layers are not available.

2.4 Issues in a wireless network

Some problems that could be found in a wireless network are now explained and discussed. There are many more problems that could be encountered in a wireless network, but the issues of latency, delay, and the hidden node problem, were observed in the experiments, reported in chapter 4, and therefore are relevant to this study.

2.4.1 Interference

Interference occurs when a signal gets distorted because of other electromagnetic signals. These other electromagnetic signals can have their origin in other wireless devices trying to communicate, or other electric devices causing unwanted noise in the applicable frequency band. The same signal following different routes will arrive at the destination out of phase, and can also cause interference. The last type is called multipath interference and limits the length of transmission distances. Wireless devices have frequency filters, so they only pick up signals at specifically chosen frequencies. This is how it is possible to have multiple wireless devices functioning in different frequency bands at the same time without being affected much by each other. Even though wireless devices try to avoid sending at the same time, their efforts are not always enough. The following example explains how interference could occur in a wireless network:

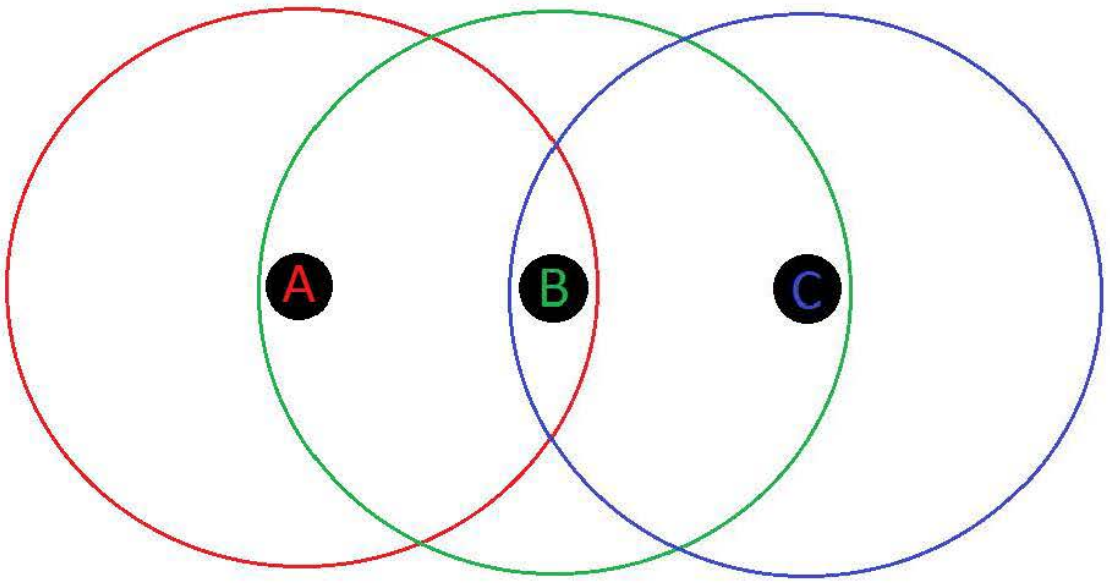

Figure 2-2: The hidden node problem

The small black circles represent three wireless nodes: A, B, and C. Each respectively has an effective transmission/reception range indicated by the large red circle for node A, the large green circle for node B, and the large blue circle for node C. If node A is busy transmitting data to node B, node C is too far away from node A to detect the channel that node A is using, is occupied for both nodes A and B. Node C could try to transmit data to node B at this point, and if this happens, interference would result at node B. This problem is commonly referred to as the hidden node problem.

2.4.2 Latency

The delay between the transmission of data to the reception thereof is known as latency. A delay can be caused by either the physical time it takes for an electromagnetic signal to travel from source to destination, or the time it takes to process and route data. Even though electromagnetic signals travel at the speed of light, some communication systems are so vast, that the time it takes for a signal to travel the distance to the receiver can be long enough to have a significant influence, and can therefore, not be neglected. This mostly happens with satellite communication and is negligible for smaller wireless systems. The factor responsible for most delay in systems is the time taken to process the data on the way through the network. This can include multiplexing/demultiplexing, compressing/decompressing, routing, waiting for channel or hardware availability.

2.5 Applying network coding

This section discusses a few network coding concepts and systems which present information and characteristics on network coding which is valuable for this study. These systems were the forerunners for more implementations of network coding, and they lay down fundamental concepts necessary for making systems practical.

2.5.1 Practice and theory

Before [**56**], network coding was mostly a theoretical concept, not taking practical problems and overconfident theoretical assumptions into account. Chou *el al.* proposed the first practical system [**56**] to implement network coding in a MANET opportunistically. This system has no need for central knowledge of the graph topology, the encoding and decoding functions, and also has no need for synchronisation, which is often very difficult to achieve practically. This scheme buffers the incoming messages, and can then use these previously received messages to code or decode sent and received messages. This paper also discusses the buffering model necessary for the implementation.

A system that builds upon the work done in [**56**], but is easier to implement and has fewer overhead is presented in [**57**]. Even though the systems in both [**56**] and [**57**] were created with the purpose of being practical, they were only simulated and not yet implemented physically.

COPE [**10**] is the first system in which network coding was implemented physically. In this system each node in the network keeps track of what information its adjacent nodes have received. COPE uses this knowledge to code data together with new data that it has to send in a manner so that all

its adjacent nodes can decode the message. It is random in a sense because it spreads linear combinations of its data across the network until the receiver has received enough combinations to decode the message that was intended for it. It is also deterministic in a sense, because it takes into account what data its adjacent nodes have and therefore can decode. So the system has the benefits of deterministic network coding, namely that redundant data will not be sent, but it still applies network coding opportunistically.

The routing used in COPE does not take into account that opportunities for network coding in a particular route will make that route faster. A system built on COPE is presented in [**45**]. In this system, the routing algorithm it uses, does take network coding opportunities into account and delivers better results than COPE. This shows that a new metric namely network coding opportunities, should be taken into account for future routing protocols for network coding systems.

2.6 Conclusion

This chapter serves as a literature study. It discusses how the structure of encapsulation of data in a network works and the different standards and protocols in the OSI stack are discussed for wireless networks. Chapter 2 also presents an overview of the work done on network coding in the different layers of the OSI stack. Alternatives to using existing layers are also provided, namely a cross layer approach, or creating a new network coding layer. Issues which are relevant to this study for wireless networks are explained, and the difference between network coding practice and theory is pointed out. The important characteristics of the different types of network coding are also presented and explained in this chapter. The next chapter explains how we created a platform for the comparison and evaluation of the different types of network coding.