Universidad de Oviedo

Departamento de Informática

Programa en Sistemas y Servicios Informáticos para Internet

An overlay network solution for delay tolerant video streaming over mobile ad-hoc networks

PhD Thesis

Sergio Cabrero Barros

Gijón, 2014

RESUMEN DEL CONTENIDO DE TESIS DOCTORAL

1.- Título de la Tesis	
Español/Otro Idioma: Solución basada en redes overlay para el streaming de vídeo sobre redes móviles ad-hoc	Inglés: An overlay network solution for delay tolerant video streaming over mobile ad-hoc networks

2.- Autor	
Nombre: Sergio Cabrero Barros	DNI/Pasaporte/NIE:
Programa de Doctorado: Sistemas y servicios informáticos para internet	
Órgano responsable: Departamento de Informática	

RESUMEN (en español)

Usar herramientas de comunicación eficientes mejora el rendimiento del personal que interviene en operaciones de emergencia. El vídeo es una de ellas. En concreto, las personas al mando tienen mucho interés de recibir vídeo desde el área de intervención. Los dispositivos y las redes móviles actuales permiten este tipo de comunicación. Sin embargo, la infraestructura de estas redes no está siempre disponible en emergencias. Por ejemplo, cuando éstas suceden en lugares aislados o son producidas por catástrofes naturales. En estos casos, una red móvil ad-hoc, también conocida como MANET por sus siglas en Inglés, puede ser una buena alternativa. Estas redes no necesitan infraestructura y podrían ser desplegadas utilizando los dispositivos móviles del personal de emergencias. Por el modo en el que el personal de emergencias se distribuye y se mueve, esta MANET sería poco densa. Por tanto, existirán particiones y desconexiones. Esto crea problemas en la distribución de vídeo que aún no han sido resueltos y en los que esta tesis se centra.

En una MANET poco densa, el origen y el destino del vídeo pueden estar en la misma partición de red o en particiones distintas. Además, esto puede variar con el tiempo; ya que los nodos de la red se mueven. Cuando ambos están en la misma partición, es posible transmitir (mediante "streaming") el vídeo sobre la MANET. Sin embargo, si están en particiones distintas, es necesario utilizar mecanismos de las redes tolerantes a retardos. Hasta ahora la investigación se ha centrado principalmente en el streaming sobre MANETs, ignorando los mecanismos de transmisión en redes tolerantes a retardos. En esta tesis se ha estudiado la movilidad del personal de emergencias, concluyendo que tanto el video streaming, como la transmisión de vídeo tolerante a retardos serían necesarios. Además, se investiga como enviar los paquetes de vídeo cuando existen particiones. Para ello, es necesario identificar nodos de la red que se muevan desde la partición del origen del vídeo a la partición del destino (delay-tolerant routing). También es preciso investigar como los nodos deben reenviar los paquetes de vídeo (forwarding) para reducir las pérdidas. Aunque estos problemas se solucionen, la capacidad de la MANET puede ser insuficiente para transmitir todo el vídeo generado. Por esta razón, también se investigan técnicas de adaptación en este tipo de redes.

Se propone utilizar una red superpuesta (overlay network) sobre la MANET, llamada MOMENTUM, que implementará las soluciones a estos problemas. MOMENTUM está formada por todos los nodos de la MANET y se ejecuta en el nivel de aplicación, sobre UDP. Es compatible con los protocolos de red y las aplicaciones de vídeo estándar. Esto facilitaría su despliegue en un entorno real. Se asume que la red utiliza el protocolo de routing OLSR, que MOMENTUM utiliza para descubrir la topología de la red. Esta información permite a MOMENTUM adaptar su comportamiento entre los modos de streaming en MANET y transporte de vídeo tolerante a retardos. Se ha construido un prototipo de MOMENTUM en Java, que se ha evaluado en un entorno de simulación de redes MANET. Al compararlo con una arquitectura cliente-servidor, se ha demostrado que MOMENTUM es capaz de adaptarse mejor a las desconexiones y particiones. Además, los recursos de red consumidos por MOMENTUM son despreciables en comparación con los recursos consumidos por el tráfico de vídeo.

MOMENTUM incluye un protocolo de routing tolerante a retardos que está especialmente diseñado para operaciones de emergencia. La idea principal es utilizar el conocimiento "a priori" extraído de los escenarios de emergencia. Nuestra hipótesis es que el routing tolerante a retardos puede ser más eficiente cuando se utiliza información conocida sobre el escenario de aplicación, por ejemplo si el dispositivo móvil lo lleva una persona o está en un vehículo. Para demostrarlo, se ha comparado nuestra propuesta con PROPHET, un protocolo ampliamente utilizado. Ambas alternativas se han implementado y evaluado utilizando MOMENTUM. Los resultados obtenidos son prometedores. Nuestra propuesta incrementa el número de paquetes de vídeo entregados al destino y además, reduce el retardo. Estos experimentos se han ampliado con otra serie de experimentos utilizando una red overlay distinta (Dts-overlay), más escenarios y otros protocolos de routing. Esto ha permitido confirmar algunas de las fortalezas de nuestra propuesta, pero también ha revelado debilidades. La más importante es que nuestra propuesta falla cuando la información "a priori" es incorrecta.
Las interrupciones temporales en las rutas de red causan perdidas de paquetes masivas. La razón más frecuente para estas interrupciones es el fallo en los protocolos de red; por ejemplo, OLSR declara una ruta, pero ARP no puede resolver las direcciones. Para solucionar esto, se propone un mecanismo de control de flujo en el reenvío (forwarding) de paquetes. Es un mecanismo sencillo que limita el número de paquetes que se pueden transmitir sin asentimiento entre dos nodos. La evaluación muestra que este mecanismo reduce drásticamente el número de paquetes perdidos por interrupciones temporales en la red. Por tanto, es una solución que mejora el rendimiento por defecto en las redes MANET poco densas, y que, además, funciona con protocolos de red estándar.
Aunque se consiga minimizar el número de paquetes perdidos en la transmisión, una MANET puede no tener capacidad suficiente para enviar todo el vídeo generado en el origen. Cuando se da esta situación, el destino recibe un vídeo incompleto, siendo probable que falten los paquetes que forman el final del vídeo. Es necesario adaptar el vídeo a los recursos disponibles. Sin embargo, es muy complicado estimar los recursos de red disponibles en este tipo de escenarios; ya que las personas o vehículos que portan los dispositivos móviles se mueven libremente. Por ello, se ha diseñado una técnica de adaptación que no precisa de estimación de recursos. Se propone adaptar los fotogramas por segundo (frame rate) del vídeo modificando el orden de transmisión de los paquetes. Los resultados muestran que es posible obtener un número de fotogramas por segundo proporcional a la capacidad de la red y un vídeo con duración aproximada a la original.

RESUMEN (en Inglés)

People in Emergency and Rescue (ER) operations perform better with efficient communication tools. One of these tools is video. In specific, ER officers are willing to receive video from the incident area in their command and control posts. Existent mobile networks and devices can be used it for this purpose. However, network infrastructure is often unavailable or collapsed in emergency operations. Especially, when these operations must take place in remote locations or after natural catastrophes. When this is the case, a good alternative is to deploy a Mobile Ad-hoc Network (MANET), which does not need network infrastructure. For example, a MANET can be deployed using mobile devices carried by ER personnel. ER personnel moves and is dispatched in a way that such a MANET would be sparse. This characteristic poses several unsolved challenges to video delivery, because the network is partitioned and suffers frequent disconnections. This thesis studies some of the problems that emerge in this situation.
Video source and destination can be in the same network partition or in different ones. As network devices move, the situation can change, resulting in partitions that split and merge. Video transport is different in each of these situations. When source and destination are in the same partition, video streaming over the MANET is possible. Whereas, it is not possible when source and destination are not in the same partition. Then, delay-tolerant video transport is necessary. Previous research focuses mainly in video streaming over MANETs, but it neglects delay-tolerance and the possibility of both situations occurring in a sparse MANET. Our study of mobility in real ER operations shows that partitions and disconnections occur. For that reason, this thesis addresses the problem of supporting video transport when disconnections and

partitions can happen. In addition, we investigate how to send video packets from the source's partition to the destination's partition. For this, the source must find nodes that move to the destination's partition and use them to carry the video. On the one hand, we study how to find these nodes in ER operations, i.e. how to find delay-tolerant routes. On the other hand, we analyze how to forward video packets between them in order to reduce packet losses and increase the amount of successfully delivered video. Even if the problems of routing and forwarding are solved, the capacity offered by the network may be not enough to deliver all the video. For that reason, we also investigate how to adapt the video stream to the available network capacity.

We tackle these issues with an overlay network, called MOMENTUM, formed by all nodes in the MANET. It works in the application layer, over UDP, and it is compatible with off-the-shelf video applications. Therefore, it could be easily deployed in existent mobile devices. We assume that the MANET runs the Optimized Link State Routing (OLSR) protocol, which MOMENTUM uses to gather information about the network topology. Then, it triggers mechanism for either MANET video streaming or delay-tolerant video transport. The support of partitioning and disconnection is tested with a MOMENTUM prototype. It is implemented in Java and evaluated over emulated MANETs with different mobility and density configurations. The results of this evaluation demonstrate the validity of our solution and the low overhead introduced when compared to the traditional client-server approach.

MOMENTUM implements a delay-tolerant routing protocol tailored for ER operations and video transport. The main idea behind it is to use "a priori" knowledge extracted from the study of ER operations. Our hypothesis is that delay-tolerant routing can be improved using known information, such as if a mobile device is carried by a person or embedded in a vehicle. To demonstrate this, we compare our proposal against PROPHET, a well-known state-of-the-art protocol, by implementing both in MOMENTUM. The results obtained from these experiments are promising, because our solution delivers more video and does it faster than PROPHET in most of the scenarios evaluated. In addition, another set of experiments is carried out with a different overlay network solution (Dts-overlay), with also different mobility scenarios and with more routing protocols. The performance of our solution is still better than PROPHET in most situations, but it can be improved against other protocols, especially in scenarios where the "a priori" knowledge assumed by our protocol is not correct.

Temporal disruptions in the network routes are the cause of massive packet losses. Malfunction of the network protocols are a frequent reason for disruptions, e.g. a route is declared by OLSR but ARP cannot resolve addresses. We propose to prevent packets losses using a flow control mechanism incorporated in MOMENTUM. It is a simple mechanism that limits the number of video packets that are forwarded until an acknowledgement is received. Experiments show that this flow control drastically reduces the number of packet losses. Thus, it is a feasible solution for sparse MANETs, with the advantage of using standard network protocols. Even with this mechanism, not all video packets are delivered to the video destination. The capacity of the network can be insufficient to deliver all the video recorded. When this happens, the video delivered to the user is incomplete, because some packets, often the ones containing the end of the video, do not reach the destination. Video adaptation is needed, but most adaptation techniques require resource estimation, which is hard in a sparse MANET formed by devices carried by people moving freely. We propose to adapt the video frame rate by forwarding packets in a different order. This proposal is compared with forwarding packets in the standard order, i.e., FIFO. The experiments show that video frame rate scales with network capacity and that the video received by the destination is longer than the one delivered by FIFO.

SR. DIRECTOR DE DEPARTAMENTO DE INFORMATICA
SR. PRESIDENTE DE LA COMISIÓN ACADÉMICA DEL PROGRAMA DE DOCTORADO EN Sistemas y servicios informáticos para internet

Universidad de Oviedo

Departamento de Informática

Programa en Sistemas y Servicios Informáticos para Internet

An overlay network solution for delay tolerant video streaming over mobile ad-hoc networks

PhD Thesis

Sergio Cabrero Barros

Gijón, 2014

Acknowledgements

This has been a too long process and many are to thank, but I am going to keep it brief and many names that should be included won't appear in here. First of all, I would like to thank my supervisors Professor Thomas Plagemann and Dr. Xabiel G. Pañeda. I have learnt many valuable lessons from both of you about research and life. I cannot say less of Dr. David Melendi and Dr. Roberto García, who were always there to help and encourage.

Lots of my gratitude also goes to the people in the Distributed Multimedia Systems research groups in Gijón (Alberto, Aurora, Laura…) and in Oslo (Daniel, Morten, Stein, Prof. Vera Goebel…). Thank you all for the many conversations, but especially for making easier the long working hours.

Last, but not least, my family and my friends. Thank you for being there to make the downs ups, even against my will. I don't need to say more, because you already know.

"I can live with doubt, and uncertainty, and not knowing. I think it's much more interesting to live not knowing than to have answers which might be wrong."

Richard Feynman

Abstract

People in Emergency and Rescue (ER) operations perform better with efficient communication tools. One of these tools is video. In specific, ER officers are willing to receive video from the incident area in their command and control posts. Existent mobile networks and devices can be used it for this purpose. However, network infrastructure is often unavailable or collapsed in emergency operations. Especially, when these operations must take place in remote locations or after natural catastrophes. When this is the case, a good alternative is to deploy a Mobile Ad-hoc Network (MANET), which does not need network infrastructure. For example, a MANET can be deployed using mobile devices carried by ER personnel. ER personnel moves and is dispatched in a way that such a MANET would be sparse. This characteristic poses several unsolved challenges to video delivery, because the network is partitioned and suffers frequent disconnections. This thesis studies some of the problems that emerge in this situation.

In a sparse MANET, video source and destination can be in the same network partition or in different ones. As network devices move, the situation can change, resulting in partitions that split and merge. Video transport is different in each of these situations. When source and destination are in the same partition, video streaming over the MANET is possible. Whereas, it is not possible when source and destination are not in the same partition. Then, delay-tolerant video transport is necessary. Previous research focuses mainly in video streaming over MANETs, but it neglects delay-tolerance and the possibility of both situations occurring in a sparse MANET. Our study of mobility in real ER operations shows that partitions and disconnections occur. For that reason, this thesis addresses the problem of supporting video transport when disconnections and partitions can happen. In addition, we investigate how to send video packets from the source's partition to the destination's partition. For this, the source must find nodes that move to the destination's partition and use them to carry the video. On the one hand, we study how to find these nodes in ER operations, i.e. how to find delay-tolerant routes. On the other hand, we analyze how to forward video packets between them in order to reduce packet losses and increase the amount of successfully delivered video. Even if the problems of routing and forwarding are solved, the capacity offered by the network may be not enough to deliver all the video. For that reason, we also investigate how to adapt the video stream to the available network capacity.

We tackle these issues with an overlay network, called MOMENTUM, formed by all nodes in the MANET. It works in the application layer, over UDP, and it is compatible with off-the-shelf video applications. Therefore, it could be easily deployed in existent mobile devices. We assume that the MANET runs the Optimized Link State Routing (OLSR) protocol, which MOMENTUM uses to gather information about the network topology. Then, it triggers mechanism for either MANET video streaming or delay-tolerant video transport. The support of partitioning and disconnection is tested with a MOMENTUM prototype. It is implemented in Java and evaluated over emulated MANETs with different mobility and density configurations. The results of this evaluation demonstrate the validity of our solution and the low overhead introduced when compared to the traditional client-server approach.

MOMENTUM implements a delay-tolerant routing protocol tailored for ER operations and video transport. The main idea behind it is to use “a priori” knowledge extracted from the study of ER operations. Our hypothesis is that delay-tolerant routing can be improved using known information, such as if a mobile device is carried by a person or embedded in a vehicle. To demonstrate this, we compare our proposal against PROPHET, a well-known state-of-the-art protocol, by implementing both in MOMENTUM. The results obtained from these experiments are promising, because our solution delivers more video and does it faster than PROPHET in most of the scenarios evaluated. In addition, another set of experiments is carried out with a different overlay network solution (Dts-overlay), with also different mobility scenarios and with more routing protocols. The performance of our solution is still better than PROPHET in most situations, but it can be improved against other protocols, especially in scenarios where the “a priori” knowledge assumed by our protocol is not correct.

Temporal disruptions in the network routes are the cause of massive packet losses. Malfunction of the network protocols are a frequent reason for disruptions, e.g. a route is declared by OLSR but ARP cannot resolve addresses. We propose to prevent packets losses using a flow control mechanism incorporated in MOMENTUM. It is a simple mechanism that limits the number of video packets that are forwarded until an acknowledgement is received. Experiments show that this flow control drastically reduces the number of packet losses. Thus, it is a feasible solution for sparse MANETs, with the advantage of using standard network protocols. Even with this mechanism, not all video packets are delivered to the video destination. The capacity of the network can be insufficient to deliver all the video recorded. When this happens, the video delivered to the user is incomplete, because some packets, often the ones containing the end of the video, do not reach the destination. Video adaptation is needed, but most adaptation techniques require resource estimation, which is hard in a sparse MANET formed by devices carried by people moving freely. We propose to adapt the video frame rate by forwarding packets in a different order. This proposal is compared with forwarding packets in the standard order, i.e., FIFO. The experiments show that video frame rate scales with network capacity and that the video received by the destination is longer than the one delivered by FIFO.

Resumen

Usar herramientas de comunicación eficientes mejora el rendimiento del personal que interviene en operaciones de emergencia. El vídeo es una de ellas. En concreto, las personas al mando tienen mucho interés de recibir vídeo desde el área de intervención. Los dispositivos y las redes móviles actuales permiten este tipo de comunicación. Sin embargo, la infraestructura de estas redes no está siempre disponible en emergencias. Por ejemplo, cuando éstas suceden en lugares aislados o son producidas por catástrofes naturales. En estos casos, una red móvil ad-hoc, también conocida como MANET por sus siglas en Inglés, puede ser una buena alternativa. Estas redes no necesitan infraestructura y podrían ser desplegadas utilizando los dispositivos móviles del personal de emergencias. Por el modo en el que el personal de emergencias se distribuye y se mueve, esta MANET sería poco densa. Por tanto, existirán particiones y desconexiones. Esto crea problemas en la distribución de vídeo que aún no han sido resueltos y en los que esta tesis se centra.

En una MANET poco densa, el origen y el destino del vídeo pueden estar en la misma partición de red o en particiones distintas. Además, esto puede variar con el tiempo; ya que los nodos de la red se mueven. Cuando ambos están en la misma partición, es posible transmitir (mediante "streaming") el vídeo sobre la MANET. Sin embargo, si están en particiones distintas, es necesario utilizar mecanismos de las redes tolerantes a retardos. Hasta ahora la investigación se ha centrado principalmente en el streaming sobre MANETs, ignorando los mecanismos de transmisión en redes tolerantes a retardos. En esta tesis se ha estudiado la movilidad del personal de emergencias, concluyendo que tanto el video streaming, como la transmisión de vídeo tolerante a retardos serían necesarios. Además, se investiga como enviar los paquetes de vídeo cuando existen particiones. Para ello, es necesario identificar nodos de la red que se muevan desde la partición del origen del vídeo a la partición del destino (delay-tolerant routing). También es preciso investigar como los nodos deben reenviar los paquetes de vídeo (forwarding) para reducir las pérdidas. Aunque estos problemas se solucionen, la capacidad de la MANET puede ser insuficiente para transmitir todo el vídeo generado. Por esta razón, también se investigan técnicas de adaptación en este tipo de redes.

Se propone utilizar una red superpuesta (overlay network) sobre la MANET, llamada MOMENTUM, que implementará las soluciones a estos problemas. MOMENTUM está formada por todos los nodos de la MANET y se ejecuta en el nivel de aplicación, sobre UDP. Es compatible con los protocolos de red y las aplicaciones de vídeo estándar. Esto facilitaría su despliegue en un entorno real. Se asume que la red utiliza el protocolo de routing OLSR, que MOMENTUM utiliza para descubrir la topología de la red. Esta información permite a MOMENTUM adaptar su comportamiento entre los modos de streaming en MANET y transporte de vídeo tolerante a retardos. Se ha construido un prototipo de MOMENTUM en Java, que se ha evaluado en un entorno de simulación de redes MANET. Al compararlo con una arquitectura cliente-servidor, se ha demostrado que MOMENTUM es capaz de adaptarse mejor a las desconexiones y particiones. Además, los recursos de red consumidos por MOMENTUM son despreciables en comparación con los recursos consumidos por el tráfico de vídeo.

MOMENTUM incluye un protocolo de routing tolerante a retardos que está especialmente diseñado para operaciones de emergencia. La idea principal es utilizar el conocimiento "a priori" extraído de los escenarios de emergencia. Nuestra hipótesis es que el routing tolerante a retardos puede ser más eficiente cuando se utiliza información conocida sobre el escenario de aplicación, por ejemplo si el dispositivo móvil lo lleva una persona o está en un vehículo. Para demostrarlo, se ha comparado nuestra propuesta con PROPHET, un protocolo ampliamente utilizado. Ambas alternativas se han implementado y evaluado utilizando MOMENTUM. Los resultados obtenidos son prometedores. Nuestra propuesta incrementa el número de paquetes de vídeo entregados al destino y además, reduce el retardo. Estos experimentos se han ampliado con otra serie de experimentos utilizando una red overlay distinta (Dts-overlay), más escenarios y otros protocolos de routing. Esto ha permitido confirmar algunas de las fortalezas de nuestra propuesta, pero también ha revelado debilidades. La más importante es que nuestra propuesta falla cuando la información "a priori" es incorrecta.

Las interrupciones temporales en las rutas de red causan perdidas de paquetes masivas. La razón más frecuente para estas interrupciones es el fallo en los protocolos de red; por ejemplo, OLSR declara una ruta, pero ARP no puede resolver las direcciones. Para solucionar esto, se propone un mecanismo de control de flujo en el reenvío (forwarding) de paquetes. Es un mecanismo sencillo que limita el número de paquetes que se pueden transmitir sin asentimiento entre dos nodos. La evaluación muestra que este mecanismo reduce drásticamente el número de paquetes perdidos por interrupciones temporales en la red. Por tanto, es una solución que mejora el rendimiento por defecto en las redes MANET poco densas, y que, además, funciona con protocolos de red estándar.

Aunque se consiga minimizar el número de paquetes perdidos en la transmisión, una MANET puede no tener capacidad suficiente para enviar todo el vídeo generado en el origen. Cuando se da esta situación, el destino recibe un vídeo incompleto, siendo probable que falten los paquetes que forman el final del vídeo. Es necesario adaptar el vídeo a los recursos disponibles. Sin embargo, es muy complicado estimar los recursos de red disponibles en este tipo de escenarios; ya que las personas o vehículos que portan los dispositivos móviles se mueven libremente. Por ello, se ha diseñado una técnica de adaptación que no precisa de estimación de recursos. Se propone adaptar los fotogramas por segundo (frame rate) del vídeo modificando el orden de transmisión de los paquetes. Los resultados muestran que es posible obtener un número de fotogramas por segundo proporcional a la capacidad de la red y un vídeo con duración aproximada a la original.

Contents

Chapter 1

Introduction

1.1 Motivation and context

The proliferation of mobile technologies has changed the way we communicate for work and leisure. Mobile phones are omnipresent in our everyday activities, allowing us to easily share all kinds of data. However, there are environments where the use of these technologies, although very appealing, is still complicated. Emergency and Rescue (ER) is one of these application domains. It is very interesting to leverage the potential of mobile technologies in ER operations, because information sharing is very valuable in this environment. However, the available systems cannot be always applied to this environment.

In specific, personnel in an ER operation can benefit from sharing video recordings. Video streaming services have become popular and shown their potential in many application domains. As humans, we have an outstanding capacity to understand visual information and video provides a huge amount of it. In ER, video can represent situations better than other means, such as oral descriptions through the radio. As a result, decision-making can be faster and more effective, which fortunately could turn out in more saved lives. The ideal situation for officers coordinating an ER operation would be to have real time video from the personnel in the incident area. For safety reasons, the Command and Control Center (CCC) of an ER operation is located far away from the incident areas. Thus, video could be a very useful tool to evaluate the situation in the incident area from the distance. Although live video would be the best-case scenario, Firefighters from the regional fire department in Asturias (112/Bomberos de Asturias) state that any type of video delivery can help. Currently, they use video delivering solution less sophisticated than real-time streaming. Sometimes they bring a camera inside the incident area, record one or several videos and carry the camera back to the CCC to watch the recorded video. They also use conventional TV cameras with radiofrequency transmitters when video is extremely necessary. This system needs repeaters when direct view between transmitter and receiver is not possible. A common practice is to install it in a helicopter using the height to overcome obstacles in the ground, but it is costly and keeps the helicopter busy in this task. These alternatives may be enough in some occasions, but they have a lot of room for improvement considering the recent advances on mobile communications and computing.

With the proliferation of mobile technologies, we devise a future where people and vehicles carry all type of sensors and communication devices. For example, a firefighter (see Figure 1-1) carries a camera, headphones and a microphone in the helmet or a small touchable screen in his forearm. These devices act as information sources and destinations, helping him and others in their duties. Video is recorded with the helmet camera and distributed over a communication network to his colleagues or the officers in the CCC. However, emergency services cannot rely on the existence of a

communication network. They often work in remote or underground locations, not reached by commercial cellular networks, or as disaster responders in places where the network infrastructure is destroyed. The use of packet switched radios to transmit voice (and sometimes small amounts of data) is the standard practice, but this is not enough for video distribution.

Figure 1-1 In the near future, ER responders will be equipped with several devices

In these situations, mobile devices as part of firefighters equipment and incorporated vehicles can be configured as a Mobile Ad-hoc Network (MANET). "Ad hoc" is a Latin expression literally translated as "for this" or "for this purpose". Wireless ad-hoc networks are decentralized networks in which every network node participates sending, receiving and forwarding data. Opposite to hierarchical networks, nodes act as routers and hosts. When a packet traverses a link between two nodes, it is called a hop. Hence, routes that traverse several nodes and links are called multihop routes. A single hop route requires that just the sender node transmits its packets to the network, while a multihop route needs nodes in between source and destination to forward them. For example, in a three hops route of Figure 1-2, the information is transmitted by the sender and then forwarded by two intermediate nodes. The fact that these networks do not rely on a preexistent network infrastructure of specialized equipment such as routers or switches makes them an interesting alternative for data communication in ER.

The movement of nodes changes the topology of the network. This poses different types of challenges that must be solved to use a MANET. Thus, it is important to identify which type of network topology is more likely to be found in ER operations. How often or how fast nodes move relatively to others determines how dynamic the network topology is. How nodes are distributed in the space also determines the links between them. Network density is the ratio between the links established in the network and the number of links if all nodes were connected. In a MANET, the distance between nodes and the communication range determine the existence of links. Hence, if there are many nodes concentrated in a relatively small area, they are likely to have many links between them, and the MANET is dense. On the contrary, if nodes are spread in a relatively large area, the MANET is sparse (see Figure 1-3). Sparse MANETs are prone to suffer partitioning. This means that there may be groups or clusters of nodes isolated from each other. In Figure 1-3b, the three nodes on the left and the two nodes on the right

form two separated partitions, because the communication range is not long enough to link them. Nodes in the same network partition are interconnected and can reach each other either by a single hop or multihop route.

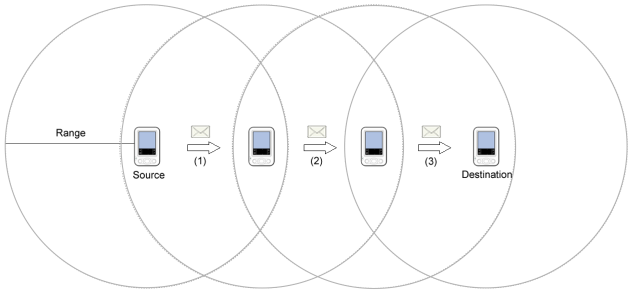

Figure 1-2 A three hops route requires two intermediate nodes to forward the packets

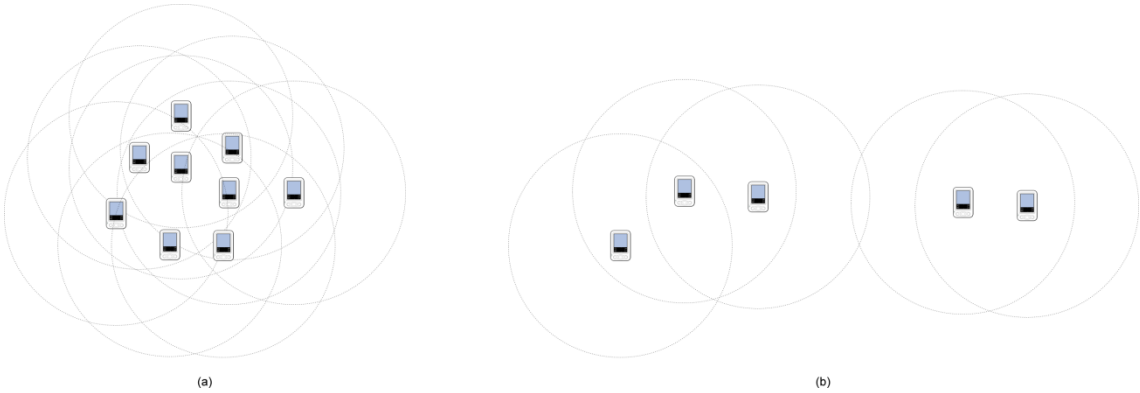

Figure 1-3 Dense networks (a) and Sparse networks (b)

Taking into account how emergency services act, the MANET they form is likely to be sparse and partitioned. When an ER operation occurs, personnel form teams that go to the points where something happened. For instance, in a fire inside a house, some firefighters will go inside the building and others remain outside with police, ambulances and other personnel. This behavior is likely to form network partitions that cannot communicate with each other through multihop routes in real time. Fortunately, personnel are likely to move and meet, which may make possible to leverage delay-tolerant networking. When a node moves from one partition to another, it can be used as data ferry. This is called the store-carry-forward paradigm [1], because a node receives data, stores it, carries it when moving and forwards it when it meets the destination. A sparse MANET using store-carry-forward is considered a Delay-Tolerant Network (DTN), because it supports communication with the delay produced by storing and moving with the data. This delay is much higher than the one produced by forwarding over a multihop route. The transport of video in such networks constitutes the core of our problem.

1.2 Problem Statement

Video distribution is based on two fundamental techniques: video streaming or video download. Streaming consists on transporting video data through the network, ideally at the same speed as it is produced, to allow continuous and uninterrupted consumption. Thus, it requires a fairly stable connection between video source and destination, and the allocation of enough resources to support video stream forwarding. A video player application can reproduce the video as it is received. Download consists on transporting through the network one file as fast as possible. It is carried out using protocols, such as FTP, HTTP, or some P2P solutions. The video file is stored in the destination host and, after it has been downloaded completely, a video player can reproduce it. Video download can be resilient to disconnection between source and destination. However, video cannot be consumed as it is received. In a sparse MANET, continuous connectivity between source and destination is not guaranteed. Sometimes this connectivity is intermittent and sometimes it is necessary to use delay-tolerant networking. Therefore, conventional video streaming is not possible. However, in application domains like ER, it is desirable to produce the video as a stream, e.g. streaming live video from a camera. Users may also want to consume the video as it is received. Thus, video download is not a satisfactory solution either. None of these two techniques are adequate over sparse MANETs in ER operations.

The aim of this work is to develop new solutions to transport video streams in a sparse MANET. We focus on three key problems to achieve this. The first problem is how to adapt to different connectivity scenarios. Video source and destination in a sparse MANET can be in the same or different network partitions. Each of these situations requires different actions to deliver video, i.e. using DTN transport or streaming over the MANET. Thus, it is important to identify them and to investigate the mechanisms that must be triggered at each moment. There are several proposals for video streaming with source and destination in the same MANET partition. However, DTN scenarios still have many open issues. For that reason, our second and third problems focus on them. The second problem appears when video source and destination are in different partitions, i.e. the MANET is a DTN. Then, it is necessary to route video stream packets from the video source to the video destination using delay-tolerant routes. The third problem emerges if the throughput obtained over a delay-tolerant route is not enough to transport the complete video stream. In this situation, it is essential to investigate how to maximize the throughput, for example, by reducing packet losses. If throughput is still insufficient, the implications and possible solutions for this issue must be considered. This is a problem of video adaptation in a DTN.

1.3 Method

The approach used to address those problems is based on experimental computer science [2]. It consists on testing theoretical proposals and check if they behave as expected, i.e. solving the problems they are expected to solve. In specific, we carry out experiments with software prototypes. Each set of experiments is aimed to evaluate the validity of our ideas to solve a part of the presented problem. To build the prototypes, we have identified a set of requirements for our solution. These requirements are derived from the goal to design and implement a solution for ER operations. These requirements guide the design process. We use an iterative method, which resembles

agile methodologies, to discuss feasible solutions and build prototypes including them. Each prototype includes new functionality and fixes any relevant problem discovered in previous experiments.

We aim to study delay-tolerant video transport using current technologies as the starting point. As far as it is possible, the proposed solutions are based on state-of-the-art networking solutions. The goal is to have working implementations of our proposals as soon as possible. The advantage of this approach is that it is easier to demonstrate the feasibility of our proposals. However, implementation and integration processes are laborious. The first step in our research consists on discussing the best network architecture to solve our problems following the requirements established. We implement and evaluate the first prototype based on the design emerged from this discussion. Afterwards, we discuss delay-tolerant routing solutions for ER operations. For this purpose, we study mobility of emergency services. Our proposals are tested inside the proposed network architecture with a new version of the prototype. Then, we investigate solutions for video forwarding and adaptation over delay-tolerant routes. They are analyzed through a new version of our solution prototype and carrying out new experiments.

1.4 Thesis contributions

This thesis provides contributions in two topics. The core one is video delivery in sparse MANETs. It includes the design of an overlay network and, as a part of it, new networking mechanisms. In addition, the thesis also contributes to explore and create new tools, models and techniques that are useful in the evaluation process of similar solutions.

Video delivery in sparse MANETs

The first contribution is the design of an overlay network, called MOMENTUM, to support video transport on sparse MANETs. The overlay network is compatible with state-of-the-art networking and video applications. To make it work, each MANET node must execute a piece of software that works over the TCP/IP network stack and, in video source and destination, works as interface with video applications. Therefore, it is designed as a middleware between applications and conventional network protocols. The software architecture is component-based. Each component has particular responsibilities for video delivery. This approach makes it possible to evolve components with a great degree of independence. As far as our knowledge goes, nobody has attempted to design a solution for video delivery in sparse MANETs with the requirement of using state-of-the-art protocols and applications.

The experiments carried out with MOMENTUM show that the proposed architecture and mechanisms are feasible to deliver video over sparse MANETs. The overlay network uses the MANET routing protocol OLSR to trigger adaptation between delay-tolerant transport and streaming. The transport and delay-tolerant routing mechanisms are simple. However, they are effective in both streaming and using delay-tolerant transport. Besides, the overhead introduced by the overlay network is small. This contribution is published in:

S. Cabrero, X. G. Pañeda, T. Plagemann, V. Goebel, and M. Siekkinen, "Overlay solution for multimedia data over sparse MANETs," in Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly, 2009, pp. 1056–1061.

S. Cabrero, X. G. Pañeda, D. Melendi Palacio, and R. García Fernández, "En busca de un protocolo de transporte multimedia para redes móviles ad-hoc poco densas," in Conferencia de Ingeniería Telemática*, 2009.*

The second contribution is a delay-tolerant routing protocol, called Emergency Overlay Routing (EOR). Delay-tolerant routing depends on node mobility. Thus, we study mobility in ER operations and use this knowledge to design EOR. This novel protocol selects MANET nodes as ferries, which are used to transport video from video source to destination when they are in different partitions. EOR uses knowledge about mobility patterns in ER operations. It also aims for minimum delay, as it is beneficial for video stream transport. We compare its performance with PROPHET, a state-of-the-art delay-tolerant routing protocol. Both are implemented inside MOMENTUM. The decision mechanisms of EOR show its superiority in our evaluation. EOR is able to deliver more video and faster than PROPHET. However, video delivery is low for both approaches due to low connectivity and packet losses. Furthermore, EOR is studied in other networking architectures and compared with other protocols showing its advantages and limitations. These contributions are published in:

S. Cabrero, X. G. Pañeda, T. Plagemann, D. Melendi, and R. García, "An Overlay Routing Protocol for Video over sparse MANETs in Emergencies," Cadernos de Informática, vol. 6, no. 1, pp. 195–202, 2011.

M. Lindeberg, J. E. Haavet, S. C. Barros, V. Goebel, and T. Plagemann, "Message lost or message taken - on message ferry selection in DTNs -," in IFIP Wireless Days, 2012, pp. 1 –7.

The experiments with EOR show that temporal disruptions cause massive video packet losses. They affect video stream transport heavily, as packets are forwarded continuously. For that reason, the throughput of the overlay routes defined by EOR, and PROPHET, is very low. We propose to solve this with an Error and Flow Control (EFC) mechanism to forward video packets between overlay nodes. Although this is not a novel approach, we demonstrate that it successfully increases the throughput obtained over delay-tolerant routes defined by EOR. Results from the EFC evaluation are published in:

S. Cabrero, X. García Pañeda, T. Plagemann, D. Melendi, and R. García, "Towards reliable video transmission over sparse MANETs in emergencies," Informática na educação: teoria & prática, vol. 14, no. 1, Nov. 2011.

Even in a lossless network, mobility limits the capacity that can be obtained. For that reason, we propose a novel video adaptation technique for DTN, called Dynamic Temporal Scalability (DTS). DTS does not rely on resource estimation, which is not possible if nodes move freely. It is a part of the video forwarding system of MOMENTUM and, hence, cooperates with EFC. We demonstrate that DTS is able to adapt the video frame rate to the network capacity. We also discuss why adapting the

frame rate is the best alternative for video adaptation over DTNs, especially in application domains like ER. This contribution is presented in:

> *S. Cabrero, X. G. Pañeda, R. Garcia, D. Melendi, and T. Plagemann, "Dynamic Temporal Scalability: Video adaptation in sparse Mobile Ad-Hoc Networks," in IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications, 2012, pp. 349–356.*

Tools, models and methods for evaluation

In the process of creating the core contributions, this thesis has generated some useful tools, models and methods related to the evaluation of MOMENTUM.

First, although MOMENTUM aims to work with off-the-shelf video applications, these are not the best alternative to reproduce the video delivered. Off-the-shelf video players are focused on video streaming. A video player that is compatible with both delay-tolerant transport and streaming is more convenient. We propose a player that shows the user the parts of video received and the quality they have. In Appendix C we present a design and a proof of concept prototype of these ideas.

Second, node mobility is an important part of evaluating any MANET proposal. On the one hand, the application domain is important for mobility. We thoroughly study mobility in ER operations. In Appendix A we present an analysis of the mobility in an emergency service based on GPS traces. It draws interesting conclusions to understand how a MANET would have worked, if every node in the GPS traces had brought a network device. On the other hand, researchers need to understand and build mobility scenarios. We propose a tool for that purpose. It brings the possibility to visualize and modify mobility scenarios in an easy way. This tool is described in Appendix B and is published in:

> *S. Cabrero, X. G. Pañeda, D. Melendi, and R. Garcia, "MASS: Editor for mobile ad-hoc network scenarios," in 2010 IEEE GLOBECOM Workshops, 2010, pp. 1679–1683.*

Finally, this thesis explores new evaluation techniques. MOMENTUM is implemented using Java. Hence, real-time simulation (or emulation) is used to evaluate it. We explore the advantages and limitations of mixing real devices with network simulations, which resulted in several Master theses [3] and [4]. One key issue when building testbed that mix real and simulated parts is resource limitation. Our knowledge contributes to a more general methodology to build them:

> *A. Alvarez, R. Garcia, S. Cabrero X.G. Pañeda, D. Melendi, R. Orea, "In pursuit of massive service emulation: a methodology for testbed building," Communications Magazine, IEEE, vol.49, no.9, pp.162,168, September 2011.*

1.5 Outline

The remainder of the thesis is organized as follows.

- **Chapter 2** describes the background knowledge behind this thesis.
- **Chapter 3** analyzes the requirements of the proposed solution.
- **Chapter 4** describes the design of MOMENTUM, the overlay network proposed as solution for video transport.
- **Chapter 5** describes the implementation and evaluation of the first prototype of MOMENTUM.
- **Chapter 6** studies delay-tolerant routing for video transport in the context of ER.
- **Chapter 7** studies video forwarding and adaptation in delay-tolerant networks.
- **Chapter 8** presents the conclusions of the thesis and potential lines for future work.

Chapter 2

Background

This chapter introduces the main concepts used in this thesis. We describe the principles of MANETs and DTNs, as well as media streaming. We also present the evaluation techniques used in the experiments presented in next chapters. Finally, we thoroughly analyze the ER application domain.

2.1 Mobile Ad-hoc Networks

2.1.1 Overview

A Mobile Ad-hoc Network (MANET) is a wireless network that does not require an infrastructure to work. The origins of these networks are in the ALOHA protocol and network packet switching [5]. Most communication networks have hosts, which produce and consume information, and use an infrastructure to transport it. Nodes that carry out specialized functions, e.g. routers or access points, form this infrastructure. In MANETs, hosts assume these functions. They are characterized by the absence of base stations. Instead, devices discover others within their communication range to form a network. To send data beyond the transmission range of a single device, nodes collaborate forwarding packets on behalf of others. Thus, a MANET is a set of mobile nodes that dynamically form a wireless network without infrastructure and collaborate to establish and maintain the network.

Nowadays, the most common protocols for wireless networking belong to the 802.11 family. The ad-hoc mode of 802.11 is implemented in many mobile devices. Thus, MANETs are often deployed using these protocols. Most network simulation also uses them. We also use 802.11 protocols in our experiments. However, it must be taken into account that 802.11 mechanisms are not tailored for multihop routes [6], so it presents performance problems, such as lower bitrates than expected or collisions due to the hidden node problem. Thus, other wireless networking protocols may be useful in future MANET simulations and deployments.

Minimal configuration and quick deployment make ad-hoc networks suitable for applications such as ER operations, response to natural or human-produced disasters, military tactical networks, sensor networks and vehicular networks (VANETS). However, they have some general problems, for example:

- Mobility produces disconnections and partitions.
- Dynamic network topology causes unstable network routes.
- Communication is unreliable due to shared medium and unpredictable behavior of nodes.

- Resources are scarce because devices are limited in energy, memory, etc.
- A MANET is likely to suffer congestion if there is a high density of nodes in an area.

The severity of these problems depend on the characteristics of the MANET considered. Mobility of the nodes, speed, density, range and other properties emphasize different problems. For example, if node density is high, network congestion and collisions are more likely. If density is low, the network is likely to have more partitions. Traditional Internet solutions are not prepared to deal with MANET problems. For that reason, researchers propose tailored architectures, transport and routing protocols, some of which we describe next.

2.1.2 Transport protocols

Internet standard protocols, i.e., TCP and UDP, can be used in MANETs. Nevertheless, TCP shows performance issues, as is revealed in several studies [7], [8], and [9]. The main problem is that TCP does not detect network disruptions, such as disconnections or temporal route breaks, and assume it is path congestion. When sender and receiver disconnect temporally, the protocol executes the exponential back-off algorithm, leading to a decrease of the transmission window size. Hence, frequent route disruptions result in a very low throughput of the TCP connection. In addition, a permanent disconnection closes the connection. Therefore, TCP is not a feasible alternative for reliable transport over MANETs.

Research on MANET transport protocols focus on solving this problem. Two approaches are used: modifying and extending TCP, or defining a new transport protocols. An example is the Explicit Link Failure Notifications extension to TCP (TCP-ELFN), which is defined in [7]. It improves the performance of standard TCP versions over MANETs, but it is not fully reliable [10]. In the same line, [11] propose a new congestion control system for TCP to detect disconnections. The work in [12] defines a completely new protocol, although inspired in TCP. This protocol introduces an interesting idea that is the feedback of intermediate nodes. It breaks the end-to-end policy of TCP and makes intermediate nodes aware of the connection. This approach shows good results in MANETs.

2.1.3 Routing

In MANETs, node mobility produces changes in network links and, thus, network topology. This is opposite to the Internet where links are static and route changes seldom occur. Moreover, specialized components of the infrastructure, e.g. routers or gateways, simplify the process. However, MANET nodes are not specialized, they are hosts and routers at the same time. As a consequence, routing in MANETs is different to routing in the Internet. MANET routing protocols are designed to run in all network nodes, not only in routers. Furthermore, they are designed to cope with very dynamic network topologies.

Routing protocols for MANETs are classified in two main philosophies: reactive and proactive. The former comes from sensors networks, where resource consumption is crucial. The latter comes from the Internet routing community; where it is key to keep up to date topology. A reactive protocol is a protocol that sets-up routes on demand.

When a node wants to send a packet to another node, it starts a route discovery process. On the contrary, a proactive protocol tries to keep network routes updated. It exchanges periodic messages to calculate routes. When a node wants to send a message to another node, it just has to look in the local routing table. As a result, proactive protocols consume more resources than reactive protocols. However, reactive solutions add delay, which is due to route discovery. The reactive approach is therefore appropriate for systems with sporadic communication and scarce resources, e.g. sensor networks. Proactive protocols perform better if low delays are required. Finally, there are also hybrid protocols that combine both approaches. In Zone Routing Protocol (ZRP) [13], proactive routes are built for the closer nodes, a few hops away, and a reactive approach is used for the rest of the network. The division in zones resembles routing inside autonomous systems and outside them. This approach is convenient when nodes are more likely to exchange traffic with closer nodes. Finally, there are also protocols that use position of nodes, acquired with GPS or other systems, to find better routes, e.g. Location Aided Routing [14]. The position of MANET nodes is useful to build more reliable routes, because it determines the network topology. The main disadvantage of these systems is the dependence on an external positioning system.

Next, we briefly describe a reactive protocol (AODV) and a proactive protocol (OLSR). These are the most used routing protocols in MANETs and we build on them our solution. OLSR is used as MANET routing protocol and AODV inspires some mechanisms in the design of our overlay routing protocol.

AODV

The Ad-hoc On-demand Distant Vector (AODV) [15] provides on-demand routes using Route Request (RREQ) and Route Response (RREP) messages. Figure 2-1 shows a route discovery example. First, the source node (the camera icon) broadcasts a RREQ message looking for a node (the device with the flame icon). Every node that receives a RREQ for the first time forwards it adding its address in a list. If the same RREQ is received again, it is ignored. When a RREQ reaches the destination, it answers with a unicast RREP message to the source using the reverse path, which is contained in the message. During this process, all the nodes receiving the RREQ discover a route to the source. Nodes receiving the RREP also discover a route to the destination.

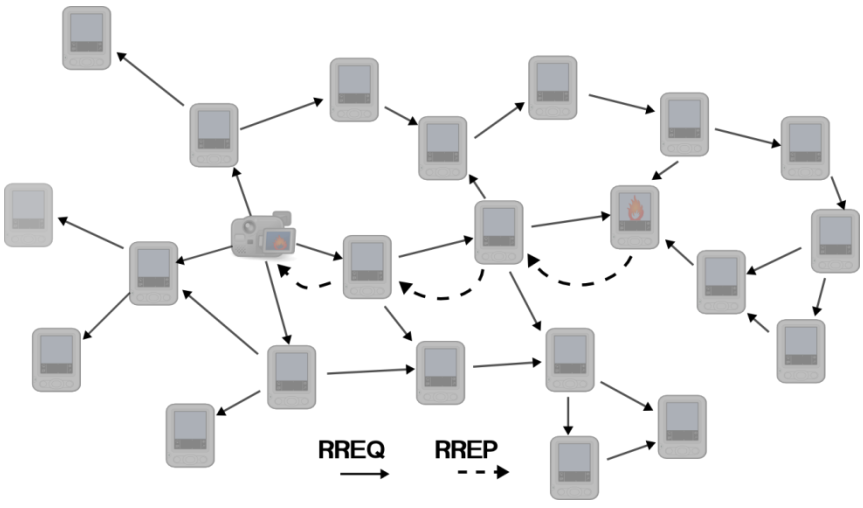

Figure 2-1 Example of AODV Route Request and Reply forwarding

In AODV and most reactive protocols, resource consumption is low. Routes are only stored temporally and messages are only sent when needed. These aspects make this protocol family especially appropriate for networks with devices with small memory or battery, such as sensors. Nevertheless, every route discovery process implies network flooding of the RREQ message, which demands energy and network capacity in that moment. In addition, the route discovery process introduces latency between the initialization to send a packet and the actual transmission. This delay can be an important drawback for services with real time constraints such as video streaming.

OLSR

Optimized Link State Routing (OLSR) [16] is a proactive and table-driven protocol designed for mobile ad-hoc networks. It looks for the shortest routes (in hops) using a variation of the Dijkstra algorithm [17]. OLSR works in a completely distributed manner without dependency of a central entity. Nodes periodically exchange messages containing topology information. This information is used to build a routing table containing all reachable nodes and the next hop towards them. This table is used to forward packets in the MANET.

Some messages of OLSR must reach the whole network. The simplest way would be flooding the network, but it is clearly inefficient. Instead, OLSR avoids network flooding using a special type of node: the MPR (**Multipoint Relay**). For each node, there is a set of nodes from the 1-hop neighborhood that are selected as MPRs. Only MPRs of a node forward its OLSR messages. This mechanism minimizes the overhead of flooding messages by reducing redundant retransmissions. Each node knows who its MPRs are and which nodes have selected it as MPR. Hence, it knows which OLSR messages it should relay and which ones it should keep only for itself. Furthermore, a node only retransmits a message if it is received for the first time. This way, OLSR avoids loops in the transmission of routing messages. The selection of MPRs, see Figure 2-2, takes into account parameters such as link quality or willingness to forward packets. MPRs of a node must cover the whole set of 2-hop neighbors. In other words, every 2-hop neighbor can be reached by at least one MPR.

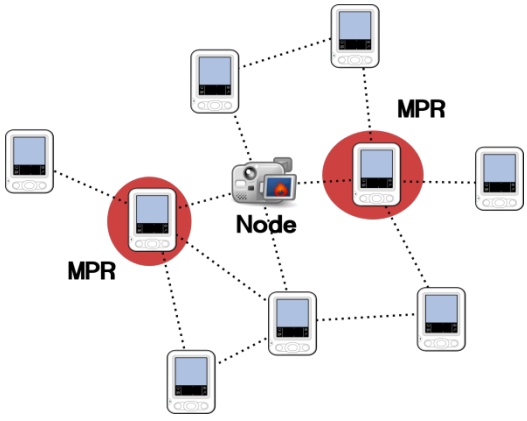

Figure 2-2 Multipoint relays must cover the full set of 2-hop neighbors

There are two important messages in OLSR, namely Hello and Topology Control. Hello messages provide local topology discovery. They are used for neighbor discovery. Topology Control messages give information about remote parts of the network, i.e. other neighborhoods.

Hello messages provide link sensing, neighbor detection and MPR selection signaling. These messages are not forwarded, so they only reach 1-hop neighbors. They contain a list of network links detected, and their status: symmetric, asymmetric, MPR (if this neighbor has been selected as MPR), lost link, etc. OLSR uses this information to discover available links, 1-hop neighbor and 2-hop neighbors. Moreover, it discovers which nodes have been selected as MPRs. The node selects its own MPRs processing Hello messages. The willingness parameter, indicated in the message, is important in the selection, because it expresses the readiness of a node to forward packets on behalf of others. Thus, nodes with higher willingness value are more likely to be selected as part of a route and also as MPRs. OLSR sends Hello messages periodically with an interval specified in a configuration parameter. If this parameter is set too high, the protocol reacts slowly on changes in the network. If it is set too low, more resources are spent in Hello messages. Thus, a good setup of OLSR influences the performance of the network.

Topology Control (TC) messages spread topology information through the network. Like Hello, a TC message contains a list of neighbors and it is sent periodically. In addition, a message has a sequence number that is used to detect outdated messages. TC messages are forwarded by MPRs. When a node A receives a TC message from a node B, A discovers the neighbors of B. Neighbor relationships are used to build a graph of the network topology and to calculate routes.

Every node stores information about the network in several sets and tables. This information is extracted from Hello and TC messages. We detail those structures that will be interesting for the design of our solution:

The neighbor set contains for each neighbor:

- The main address of the neighbor (a node may have several network addresses).
- The status of the link, if it is symmetrical or not.
- The willingness of that node to carry traffic on behalf of other nodes, in a 0-7 scale.

The 2-hop neighbor set contains for each 2-hop neighbor:

- The main address of a 1-hop neighbor.
- The main address of a 2-hop neighbor with a symmetric link to this 1-hop neighbor.
- The time when the entry expires.

The topology set contains for each entry:

- The address of a node.
- The address of other node that is 1-hop neighbor of the previous one.
- The sequence number of the message that carried this information.
- The time when the entry expires.

Finally, the routing table contains for each route:

- The address of the destination of the route.
- The address of the 1-hop neighbor that is the next hop in the route.
- The length of the route in hops.
- The local network interface of the node through which this 1-hop neighbor can be reached.

2.2 Delay-tolerant Networks

A Delay-Tolerant Network (DTN) is a network that supports connections with long delays. Often, long delays are due to a lack of continuous connectivity between its nodes, so they are also disruption tolerant. Disconnections and long delays can be due to many reasons, but in most applications it is because nodes move. Therefore, nodes can only communicate when they are close enough to do it, which depends on the network technology they use. As a consequence, the end-to-end principle [18], which is used for the design of most Internet protocols, is not suitable for DTNs. This principle states that communication functions should be implemented in the end hosts and not in intermediate nodes. Since end hosts are not connected in a DTN, intermediate nodes are needed to provide basic networking functions. By using intermediate nodes, delay-tolerant networking can overcome longs disruptions and disconnections. Therefore, communication that is not possible in real-time may still be possible using future links in a DTN with mechanisms like store, carry and forward.

The store-carry-forward paradigm [1] is a basic mechanism used in delay-tolerant networking. The nodes in the network store and carry them for some time, instead of forwarding packets instantaneously. In the example of Figure 2-3, the node A has a message for C, but they are not connected. However, A is able to know that B will meet C in the future. Thus, A sends the message to B and B stores it. Then, when B meets C the message is delivered. These routes, where all the links traversed by the packet are not available at the same time, are created “over time” [19]. They are delay-tolerant routes. The core problem of this mechanism resides in finding the data ferries, or mules, to build the route. In applications where movement is deterministic, such as space communications, ferries can be statically configured. In other situations, it is possible to proactively control or influence the movement of some nodes to carry out this task [20]. However, in many applications, movement of the nodes cannot be controlled nor known in advance. It is in this situation when delay-tolerant routing protocols, like PROPHET, are used, we discuss in Chapter 6 as part of the related work of EOR.

Internet protocols do not support network disruptions and store-carry-forward by themselves. Nodes drop IP packets and close TCP connections during long network disruptions. New networking mechanisms are needed. The Delay Tolerant Networking Research Group (DTNRG)[1] of the Internet Research Task Force (IRTF) focuses on developing them. They propose a DTN architecture [21], which includes the Bundle Protocol [22]. It basically groups data in bundles that have associated metadata. Bundles are forwarded following the concept of custody transfer [23]. In specific, the protocol

[1] www.dtnrg.org

provides mechanisms to ensure that bundles are reliably transmitted hop-by-hop. The bundle protocol has been implemented and deployed in some real scenarios, for example, to transfer sensor data from a satellite [24]. This deployment has revealed some limitations in the current design [25], such as the absence of detection of erroneous bundles or the absence of QoS mechanisms, which would be interesting for a video transport application.

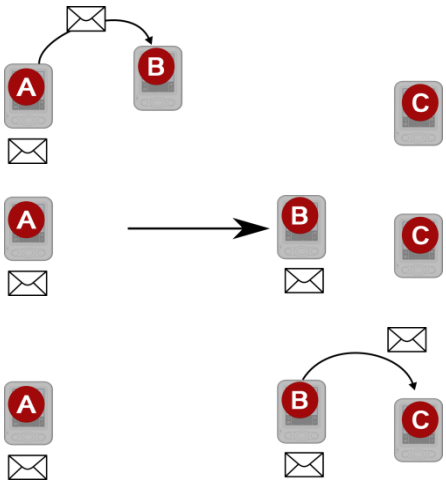

Figure 2-3 Store-carry-forward example in three steps

There are several other application domains for DTNs. One of them are space communications [26]. The goal is to build an interplanetary Internet between satellites, spaceships and base stations. They are continuously in movement and, hence, connect intermittently. Thus, there is a necessity to support delays in the communication. For example, if two satellites want to exchange data but a route between them is not available at that moment. Another application is Daknet [27]. It connects computers in remote villages with Internet using buses as data ferries. This network allows people to access to basic services, such as delay-tolerant web browsing and e-mail. There are more examples of applications using vehicles such as the UMassDieselNet used in [28], which is a live testbed of 35 buses. Moreover, delay-tolerant networking is useful in sparse sensor networks as well. For example, Zebranet [29] is a wildlife monitoring sensor network that utilize this paradigm. In the literature, the term opportunistic network is also applied to DTNs [30], especially when network nodes are carried by humans, e.g. smartphones.

2.3 Streaming

2.3.1 Overview

Streaming is the delivery of media that is consumed by one or several users at the same time it is received. A server application sends a media stream to the client application(s) over a communication network. At the receiving end, clients receive the stream and reproduce the content for the user. A streaming multimedia session consists of one or more media streams, e.g. a video stream and an audio stream, that are delivered to the user in a synchronize manner. By streaming media, the client does not need to download it beforehand. Thus, it is useful to deliver long media, as well as media produced live. There are two main types of streaming: on-demand and live.

On-demand streaming is the term used when media streams have been pre-recorded and stored. They are available to be sent whenever a client application requests it. Since media is stored, users can pause, rewind or fast-forward it. The experience is similar to the experience obtained by using a DVD or VCR.

Live streaming is the term used when the media is delivered at the same time it is created or, more precisely, with a small delay. Therefore, the interaction of users is more constrained. Some systems allow pauses and rewinds, but fast-forward is not possible. The experience is similar to the experienced obtained by watching broadcast TV.

Most streaming systems are deployed with independent control and transport planes. The control plane is used to negotiate and configure general session parameters between applications, including transport parameters. The transport plane is used to send video content through the network. Next, we briefly describe standard protocols used to provide these functionalities.

2.3.2 RTSP

The Real Time Streaming Protocol (RTSP) [31] is an application level protocol used in the control plane by streaming applications. The protocol is a widely accepted standard for this task and it is included in many streaming servers and clients. RTSP provides a general framework to control the delivery of media streams. Client and server applications use RTSP to configure session parameters and to interact. However, RTSP is not used to transport multimedia contents, other protocols, such as the Real-time Transport Protocol (RTP), serve this purpose. RTSP is a protocol with states; see the state machine in Figure 2-4. It uses messages to trigger changes between states. The client leads the communication and the server answers with the appropriate responses, as defined by the protocol. RTSP messages can be sent using TCP or UDP, but the former is the most common choice. The main reason is that client and server may be in different states after the loss of a RTSP message.

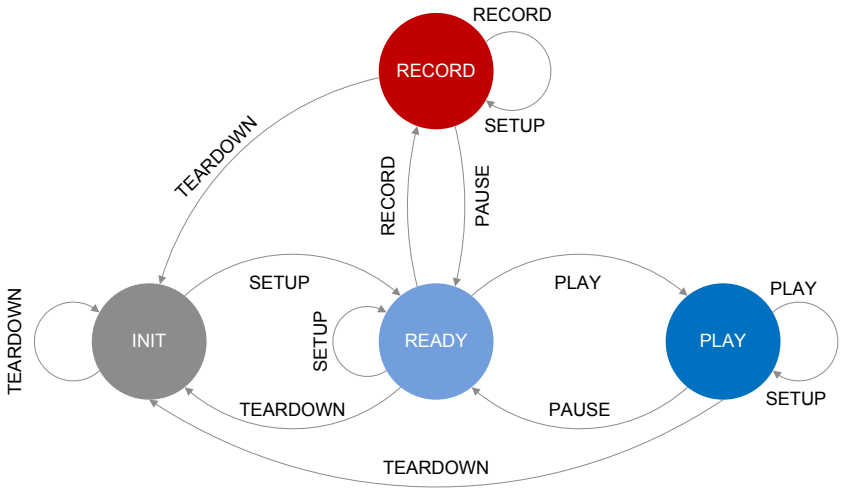

Figure 2-4 RTSP state machine

Next, we describe the most relevant messages and include an example from the RTSP RFC. The notation "`C->S`" indicates a message from the client to the server and "`S->C`" indicates a message from the server to the client. The protocol has been designed in

such a way that the messages are human-readable. Thus, examples help to clarify how the protocol works.

Options is used by the client to discover the messages supported by the server.

```
C->S: OPTIONS * RTSP/1.0
      CSeq: 1
      Require: implicit-play
      Proxy-Require: gzipped-messages

S->C: RTSP/1.0 200 OK
      CSeq: 1
      Public: DESCRIBE, SETUP, TEARDOWN, PLAY, PAUSE
```

Describe is used by the client to get the description of the media streams associated with an URL in the server, for example, a file containing audio and video streams. The response usually contains a description formatted with Session Description Protocol (SDP) [32].

```
C->S: DESCRIBE rtsp://server.example.com/fizzle/foo RTSP/1.0
      CSeq: 312
      Accept: application/sdp, application/rtsl, application/mpeg

S->C: RTSP/1.0 200 OK
      CSeq: 312
      Date: 23 Jan 1997 15:35:06 GMT
      Content-Type: application/sdp
      Content-Length: 376

      v=0
      o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
      s=SDP Seminar
      i=A Seminar on the session description protocol
      u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
      e=mjh@isi.edu (Mark Handley)
      c=IN IP4 224.2.17.12/127
      t=2873397496 2873404696
      a=recvonly
      m=audio 3456 RTP/AVP 0
      m=video 2232 RTP/AVP 31
      m=whiteboard 32416 UDP WB
      a=orient:portrait
```

Setup is used to exchange session parameters. The transport mechanisms that are used to send the media are negotiated with this message.

```
C->S: SETUP rtsp://example.com/foo/bar/baz.rm RTSP/1.0
      CSeq: 302
      Transport: RTP/AVP;unicast;client_port=4588-4589

S->C: RTSP/1.0 200 OK
      CSeq: 302
      Date: 23 Jan 1997 15:35:06 GMT
      Session: 47112344
      Transport: RTP/AVP;unicast;
      client_port=4588-4589;server_port=6256-6257
```

Play is used to start media streaming. The parameter *Range* in this message indicates the moment in which playback must start and, optionally, end. Thus, rewind and fast-forward interactions can be implemented using it.

```
C->S: PLAY rtsp://audio.example.com/audio RTSP/1.0
      CSeq: 836
      Session: 12345678
      Range: npt=20-25
```

Pause is used to temporally stop media streaming. The session remains open in the server, but media packets are not sent.

```
C->S: PAUSE rtsp://example.com/fizzle/foo RTSP/1.0
      CSeq: 834
      Session: 12345678

S->C: RTSP/1.0 200 OK
      CSeq: 834
      Date: 23 Jan 1997 15:35:06 GMT
```

Teardown is used to finish media streaming. The server frees all the resources assigned to this session.

```
C->S: TEARDOWN rtsp://example.com/fizzle/foo RTSP/1.0
      CSeq: 892
      Session: 12345678
S->C: RTSP/1.0 200
```

2.3.3 RTP/RTCP

The Real-time Transport Protocol (RTP) and the RTP Control Protocol (RTCP) [33] are a pair of protocols to send real-time data over the Internet. They are usually deployed over UDP. The main reason is that late packets are not useful in traditional streaming. If a packet is not received on time, it is discarded. TCP overhead and retransmissions can be inefficient. For that reasons, UDP has been traditionally preferred over TCP to transport real-time data. Nonetheless, this trend is changing in architectures based on HTTP [34].

RTP defines a packet format to carry the data. It adds a header in each packet with several fields. For our purposes, two are worthy to mention. **Sequence number** (16 bits) contains an identifier for the RTP packet. It is incremented by one in each packet sent. It is useful to identify lost packets. **Timestamp** (32 bits) represents the sampling time of the first byte in the payload. It indicates the receiver when the content of the packet must be played. The format of the payload in RTP is also important to deploy applications. Hence, some formats have been standardized, e.g. for h.264 [35]. The metadata in the RTP header and payload is essential to build complex video applications.

RTCP establishes a set of mechanisms to monitor and control data transport over RTP. Receiver and Sender Reports are relevant for these tasks. **Receiver Report** (RR) is a type of message used by client applications to inform the server and other receivers of the quality of the service. **Sender Report** (SR) is a type of message used by server applications to communicate statistics of the transmission of RTP packets. Both messages contain different statistics and measures, such as jitter, delay or timestamps, which can be exploited by the applications to manage resources and deliver a better quality. Furthermore, RTCP also support **Application-Defined** (APP) packets, which applications can leverage to support custom behaviors. Besides, extensions have been made to the original RTCP standard to provide more functionality, such as extended reports (XR) to improve the systems performance [36].

2.3.4 Video coding

A video codec is a piece of software used to compress and decompress video. Coding is an essential part of video streaming. On the one hand, it compresses video and determines its bitrate. Depending on what codec is used and how it is configured, a video stream consumes network bandwidth and client and server resources when it is coded and decoded. Most video codecs are lossy; they trade size and bitrate for quality. Hence, the less bytes the video needs for transmission, the lower the quality of the video once decoded. On the other hand, video codecs must produce content to be streamed. Video is divided in packets and played as received. This means that it must be decoded easily as it is being delivered. One solution is compressing each video frame separately, like M-JPEG does [37]. Then, the server sends each video frame separately and the client can decode it individually. The main advantage of this approach is that if a frame is lost, the rest can still be decoded.

More modern codecs, e.g. MPEG-2 Part 2 [38], increase video compression by considering similarities between consecutive frames. They produce three types of frames. The intra-coded frame (**I-frame**) can be decoded on its own, similar to a M-JPEG frame. The predicted frame (**P-frame**) needs the previous I-frame in the video to be decoded. The bi-directional frame (**B-frame**) needs the previous and the next P-frame or I-frame. Video codecs repeat a sequence of frames, called Group of Pictures (GOP), throughout the video. An example of GOP is IPBBPBBPBB. This technique increases video compression. However, it generates dependencies between frames. Thus, if a frame is lost during the transmission, it may affect the decoding of other frames. For example, an I-frame is needed to decode all the frames in the same GOP. This technique have been improved and combined with others in state-of-the-art codecs, such as h.263 [39] and h.264/AVC [40] that we use for our research.

In a streaming service, it is desirable to deliver the best possible quality with the available resources. Thus, it is convenient to have the same content in different qualities to send the one that matches better the current available resources. A typical solution is to create several versions of the video, e.g. implementations of HTTP Adaptive Streaming with DASH [34]. However, this implies storing several versions of the same content. Other codecs try to tackle this problem using layers. The video is coded using different layers, so each layer increases video quality. There are two main approaches, using hierarchical and independent layers. Multiple Description Coding (MDC) [41] uses the former approach. A video is coded in layers that can be decoded separately. The more layers are delivered, the better the video quality. An extension of h.264, called Scalable Video Coding (SVC) [42], uses hierarchical layers. There is a base layer that provides a minimum quality level. Then, additional layers enhance it. There are three types of layers depending on the quality parameter they scale: temporal, spatial and quality/fidelity. Temporal increases video frame rate. Spatial increases the frame size. Quality/fidelity improves the definition of each frame.

2.3.5 Streaming over MANETs

Video streaming over MANETs presents additional challenges to streaming over the Internet. In general, routes are less reliable, prone to suffer disruptions or congestion. The forwarding capability of mobile devices [43] is far from specialized routers. In addition, consecutive packets of the same stream are likely to collide over a multihop route. Therefore, research in this topic is mainly focused on fighting unreliability and

resource scarcity. Most of it is focused on dense MANETs, so delay-tolerant networking issues are not covered. A complete survey on this topic can be found in [44]. Here we review some of the most common techniques as they work as an inspiration for our proposal or are complementary to it.

Some solutions leverage video coding techniques to improve streaming in MANETs. For example, [45] define a scalable, non-causal predictive video encoder, which the authors claim as superior quality than other standard encoders. It works at low rates with different quality levels and it is tailored for error resilience, which is very important in MANETs. Besides, this work is oriented towards multicast transmission. They select multicast routes so transmission is done at the maximum available rate guessed by the signal level in each link. The opposite scenario to multicast is considered by [46]. They use MDC to distribute the same content from multiple sources streaming to one client in a MANET. The utilization of multiple sources implies that content must be previously distributed to them. They design an extension of the DSR (Dynamic Source Routing) [47] protocol to obtain disjoint routes[2] from sources to the client. Then, each source sends a set of sub-streams, so packet delivery is more reliable against route breaks. The concept of disjoint route is interesting, as they reduce the number of possible collisions when there are multiple streams in the network. MDC is also used by [48]. They combine it with multipath routing over static ad-hoc networks. Differently to [46], the sub-streams are now sent from one source to one client, but using several routes at the same time. To support this solution, they propose a variation of RTP, called MRTP [49]. They split RTP streams in many that travel through different network routes. In this case, they compare MRTP versus RTP over a dense MANET scenario. Results show that fewer packets are lost and the quality of the video improves with their approach. Multipath routing increases resilience against route breaks and also increments the available bandwidth. Hence, it is extensively explored in the literature. For instance, [50] propose a routing algorithm that looks for multipath routes to transport video using cross-layer information. In particular, it uses feedback from lower layers (i.e. MAC layer) to estimate the potential quality of each route. They implement this solution over the proactive routing protocol OLSR. [51] call this approach multimedia-centric routing. The weak point of these proposals is that they have been tested over static networks.

The authors of [52] propose a cross-layered architecture for video streaming. They measure the capacity of links at low layers (MAC, networks), then, it is used to perform smart scheduling of packets at the transport layer and the source of video that can be transmitted. Exchanging information between layers, but without breaking them, leads to a better optimization of protocols at all levels. Feedback from other layers helps protocols to adapt to new circumstances, which is critical in MANETs. Thus, cross-layer solutions are numerous in the literature. [53] proposes the Cross-layer Video Transmission Protocol (CVTP). CVTP selects an optimal path considering residual energy left in the nodes, bandwidth and hop count. Then, video encoding rate is adapted to these metrics. In their experiments, CVTP path selection improves AODV path selection. Vista-XL [54] is another cross-layer solution and proposes multipath transport too. It uses information from physical, MAC and network layers to find several paths from server to client. The interesting contribution is that instead of using MDC or SVC to generate several streams, two sub-streams are created one with I-frame

[2] Disjoint routes are routes that do not share nodes, so bottlenecks are avoided.

packets, and one with B and P-frame packets. In our opinion, this can be an interesting alternative when it is not possible to use layered video codification schemes, e.g. if the codec is executed in a node with scarce resources.

Finally, we remark two works that approach multimedia streaming from a different perspective. First, the authors of [55] propose a transport protocol for multimedia streaming, but inspired in the TCP congestion control protocol. It uses an algorithm able to differentiate congestion from route changes and other MANET issues. Hence, streaming improvement is shown over a dense network. Then, the study in [56] uses standard RTSP/RTP or HTTP streaming, proxies and OLSR. Standard video applications are isolated from the MANET using proxies, which deal with disconnections. OLSR is used to obtain feedback about the MANET that is used to manage the sessions. The idea is oriented to real environments, like ours, and it is tested in a real test-bed with three nodes.

2.4 Evaluation techniques

2.4.1 Techniques and tools

There are many techniques to evaluate a system. The most popular ones are simulation, emulation and testbeds. As defined in [57] "a testbed itself is a perfectly normal instance of the system that is under study in a particular experiment". In our case, a testbed is composed by mobile devices forming a MANET. Testbeds are the closest alternative to the real deployment of a system, but they are also expensive and complex. Simulation is in the other end of the spectrum. It uses models that behave like the system under test. Therefore, it is less realistic, but also cheaper. Simulations are also easy to repeat and replicate. Finally, emulation is in between simulation and testbed. It mixes modeled parts, but it must contain at least one real part. In the literature, the terms simulation and emulation are often mixed, especially when simulation is performed in real-time. In this thesis, we evaluate our prototypes over network simulators or emulators. They are used to model MANETs, but the system tested includes pieces of software that are the same in the real system, such as the overlay network software, video applications and routing protocol daemons.

There are several tools available to evaluate systems over MANETs. General network simulators include ns-2 [58], which is historically the most used one, and the more recent project ns-3 [59]. Then, there are tools specific to MANETs, such as NEMAN [60], or to DTNs, such as the ONE simulator [61]. They are good tools to carry out evaluations of new proposals. Nonetheless, they have different levels of abstraction and accuracy in how they model network protocols and physics. In addition, they all lack accurate models of the mobile devices in which the software would be executed. For example, the delay of packet forwarding by a mobile device can be significant [43] and it is important to consider it in a video streaming application. This is an interesting and open issue that has been tackled in [62] and [63]. Therefore, although simulation or emulation results provide good insights when developing new systems, they may be different in a real deployment due to the assumptions included in the modeled parts.

2.4.2 Mobility models

The mobility of nodes is a key part of evaluating any system for a MANET. This has been demonstrated in publications like [64], which shows the effect of different aspects of mobility in the performance of MANET routing protocols. For delay-tolerant routing protocols the effect of mobility exists too. For instance, it is easy to predict future connections of a node that always repeats the same movement, such as a bus; making it a good data mule. For that reason, it is important to understand, as well as possible, the mobility of the targeted application of a MANET. Nonetheless, this is not a simple task. Some mobility traces from real scenarios are publicly available, e.g. CRAWDAD [65]. However, they are only interesting if the evaluation aims the application domain of these traces. In addition, more than one mobility scenario is often necessary to carry out a good performance evaluation. For that reason, most systems are evaluated with scenarios generated by synthetic mobility models. These models are less realistic, but make it easier to generate multiple scenarios.

Random mobility models are a common choice of researchers when evaluating MANETs, i.e., Random Waypoint [47] or Random Walk [66]. Some models replicate particular situations like group mobility [67] or mobility in the streets of a city [68]. There are application oriented mobility models. They reproduce mobility in specific contexts, e.g. vehicular networks [69] or tactical networks [70]. In this group, the Disaster Area Mobility Model [71] is inspired in ER operations. Most of the previous models are synthetic models based on observations of the application scenario. When real mobility traces are available, it is possible to build trace-based models [72]. They overcome the limitations of using only traces and are more realistic than theoretical alternatives. Section 2.5.5 analyzes mobility in ER operations in more detail.

2.5 Emergency and Rescue

2.5.1 Overview

As the online version of Merriam-Webster's dictionary states: "An emergency is an unexpected and usually dangerous situation that calls for immediate action". This action is often referred to as an ER operation. In them, personnel from different emergency services act to save people and goods, as well as ending any potential danger. An emergency can be caused by nature, e.g. earthquakes or hurricanes, by human action, e.g. terrorist attacks or lost hikers, or by both, e.g. floods in houses built over a former riverbed. Depending on where these events occur, their impact in people and goods is different. Besides, the duration of the emergency and rescue operation is also variable. The recovery of big catastrophes can last weeks. However, most emergency operations are resolved in hours or days.

Emergency services are organizations in charge of carrying out ER operations. Depending on the country, there may be different divisions with different specializations. Common ones are Fire Departments, Police Departments and Paramedics. Sometimes there may be also other entities involved, e.g. specialized military units, or several organizations of the same specialization, e.g. local and national police. The coordination of several emergency services is not a simple task. Furthermore, ER operations are normally chaotic and stressful for both victims and rescuers. Hence, emergency plans [73] and manuals of best practices [74] are essential.

Emergency services use these documents and their experience to respond adequately to the circumstances. Nonetheless, decisions are not taken individually; hierarchy and coordination also place a major role. Emergency plans define the persons in charge of the different tasks, who they give orders to and who they receive orders from. Hierarchy is important to understand two important issues in the problem we tackle: information sharing and mobility of personnel in an ER operation. According to [73] firefighters control the incident area and nobody must act inside without their authorization and supervision. For example, if there are victims in a house on fire, the firefighters will carry them out where the paramedics are. For these reasons, we focus on firefighters actions.

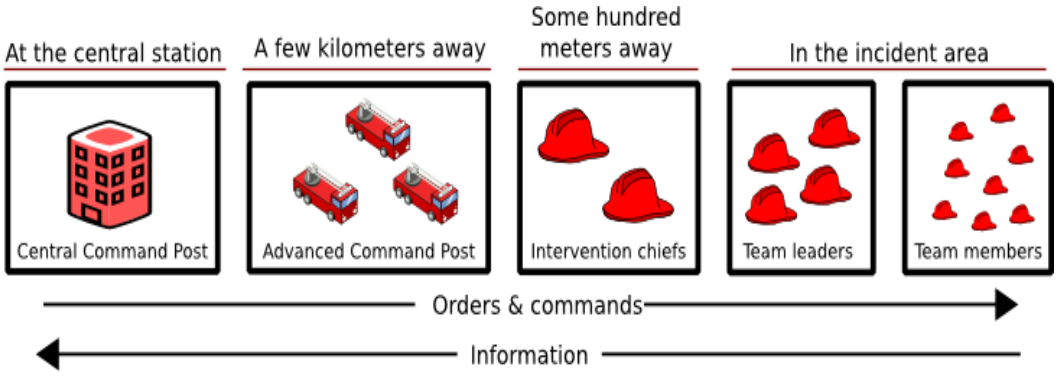

Figure 2-5 Hierarchy of information flow and commands

Figure 2-5 represents the hierarchy of firefighters in ER according to the Asturian Fire Department. The different posts are not necessary in all ER operations, but they are often present in big ones. The main coordination is carried out in a Central Command Post. It can be established in the central station where vehicles and human resources are based. The Advanced Command Post is closer to the incident area, but still in a safe place a few kilometers away. A big truck with audiovisual and communication systems can be used for this purpose. The location is selected strategically to have easy access to communication and power infrastructures. The Advanced Command Post is also used to coordinate the emergency services deployed in the area. This is often called Command and Control Center (CCC) by other sources. The leaders of firefighters, police, paramedics and civil administration meet there. They coordinate the actions of Intervention Chiefs, also called On-site-commanders (OSC), who are closer to the incident area. ER operations that affect large areas may be divided in several incident areas. Typically, there is an Intervention Chief in each incident area. Intervention Chiefs reports to the Advanced Command Post. He or she also commands the teams of firefighters in the incident area. In essence, the Intervention Chief is the link between the Incident Area and the external world. All the information going in and out must go through him. Teams of firefighters and commanded by team leaders are dispatched to carry out tasks in the incident areas. This hierarchy establishes the flow of information and command, as well as the position of the different entities relatively to the incident area.

Valuable information is gathered in the incident area through sources like reports from firefighters or victims, ambient sensors or Closed Circuit TV (CCTV) systems. All of them may be relevant to take decisions when going up the command chain. The amount of potential data sources is expected to grow as the use of technology in ER increases, e.g. by using wearable computing devices. Moreover, personnel deployed in the incident area can also receive interesting information in the form of maps or weather

forecasts. For that reason, information sharing in ER operations has been identified as a key area for research. Several projects have tackled it, such as MIDAS [75] and Infoware [76]. MANETs are used in many of these projects. Either alone or in combination with conventional infrastructure or mesh networks [77]. The eventual goal is to get the most of the available networks, even if they have heterogeneous technologies and paradigms [78]. Some authors use the name of Incident Area Network [79] for the networks deployed in an ER operation.

Emergency services have also demonstrated interest in using video, for applications such as surveillance or object recognition. The Video Quality in Public Safety Working Group[3] is part of Public Safety Communications Research in the U.S. Department of Commerce. They have done relevant efforts in this topic. They have published a set of technical requirements to use video in tactical applications [80]. They consider a wide range of applications such as telemedicine, robotics, infrared video to warn of spreading fire, tactical decision making, recorded surveillance to collect evidences, live surveillance and object recognition. It is interesting to quote two example applications from one of their reports for tactical and video surveillance [81]:

> *Video used to provide the incident commander with situation information, such as 1) a camera carried by a public safety practitioner, looking for victims, into a burning building; 2) a body-worn camera during a SWAT raid; and 3) an aerial camera following a suspect on foot.*
>
> *...*
>
> *A sweep of the whole incident scene to aid decision-making on how to deploy personnel.*

In addition, they carried out experiments for object recognition applications [82]. In their reports, they advise on values for the required parameters, e.g. bandwidth, delay, resolution or frame rate; in order to accomplish a given task. While some of these constraints are impossible to meet in a MANET or DTN, they are still useful guidelines. For example, delay requirements may be hard to achieve, but minimum video quality or frame rate may be possible to obtain. Furthermore, these studies demonstrate the clear interest of using video in ER operations and other public safety missions.

2.5.2 DT-Stream

The work in this thesis is part of the Delay Tolerant Streaming Services project (DT-Stream). This project aims to provide new solutions that enable AV streaming services over heterogeneous, mobile, and unstable networks which are found for instance in ER operations. DT-Stream is funded by the VERDIKT program of the Norwegian Research Council (project number 183312/S10) and by the Spanish National Research Program FUTURMEDIA (project TSI2007-60474). It has focused in four main sub-goals: (1) support for heterogeneity both at device and network level, (2) support for adaptation to dynamic networking conditions, (3) delay tolerant streaming transport, and finally (4) tools for development and performance evaluation. Furthermore, we have identified additional requirements at the application, network and transport layer. The need for these requirements is thoroughly discussed in a technical report [83].

[3] http://www.pscr.gov/projects/video_quality/vqips/vqips.php

The project has produced several outcomes in the form of PhD and Master theses, as well as conference and journal papers. They study problems complementary to the ones in this thesis or similar problems taking a different approach. The Master theses of [84], [85] and [86] cope with transport and routing challenges. The support of heterogeneous network is investigated in Rodriguez's work in [78]. Kristiansen's PhD thesis [87] focuses on the evaluation issues, such as the performance of real devices [43] and the accuracy of simulations [88], [63] and [62]. Several Master theses have been dedicated to improve evaluation techniques. One challenge was connecting mobile devices to network simulators [4] and [3] to measure and connect real and simulated entities, as an intermediate step towards a testbed formed solely by mobile devices. Other tools were also developed to ease the evaluation process, such as the network visualizer in [89] and the MASS editor [90]. A basic version of MASS is described in Appendix B and was improved by several other Master theses such as [91]. Finally, several attempts have been made in the application layer. Signaling protocols have been designed by [92] and [93]. Appendix C proposes a video application, ERPlayer, although other efforts have been done in the project, such as [94].

Next, we describe two different case studies in which the technologies developed by DT-Stream could be used by emergency services. The first illustrates an ideal application to respond to a big catastrophe, Hurricane Katrina. The second describes a smaller, incident based on a chemical accident trial.

2.5.3 Case study: Hurricane Katrina

Hurricane Katrina devastated several zones around the Gulf of Mexico in August 2005. The city of New Orleans was heavily affected with hundreds of fatalities and the partial destruction of its infrastructures. Buildings, roads and factories were destroyed by the strong winds and many neighborhoods were flooded (see Figure 2-6). As a consequence, the area suffered long-term power outages, as well as the shutdown of communication infrastructures. Katrina partially destroyed communications equipment and the absence of power impeded the normal functioning of the remaining devices. It took several days to recover and get minimum functionality. The communication failure deeply affected the ER operation that followed the hurricane. As a U.S. House of Representatives' report [95] mentions:

> *Massive inoperability had the biggest effect on communications, limiting command and control, situational awareness, and federal, state, and local officials' ability to address unsubstantiated and inaccurate media reports.*

Figure 2-6 Floods in New Orleans after Katrina

In such an ER operation, the lack of communication infrastructures could be partially mitigated by using a MANET. Mobile devices from ER personnel, and also from population, could communicate using ad-hoc technology. Since the affected area is large, the network would be partitioned. However, some of these partitions might be dense because people could gather in some zones (e.g. coordination points). Emergency services could use the MANET to improve their interoperability and their situational awareness. The population could also leverage it to receive alerts or send SOS messages. In addition, the technology investigated in the DT-Stream project can be applied in this situation. Aerial views taken from helicopters or drones are useful to evaluate damages and recognize the zone. Emergency services can take videos inside buildings to calculate the risk of collapse. Furthermore, trapped victims may use their mobile phones to record a video that may help rescuers in their task. Streaming and delay-tolerant transport of video over the MANET is necessary to leverage these applications.

This case study represents an ideal MANET in an extreme situation. Although most of the necessary technology exists, the challenges in such a big ER operation are many. Allowing everybody to participate in the MANET poses security, scalability and configuration issues. In addition, in some zones the network may be very dense, while in others it may be very sparse. These two opposite situations create different problems in the communication, e.g. congestion in the dense areas and, simultaneously, lack of connectivity in the sparse ones. Moreover, in the places where communication networks work, it would be convenient to leverage it. Beyond these challenges, this case study illustrates ideal long-term application of DT-Stream and MANETs in emergency and rescue.

2.5.4 Case study: Chemical Accident

This case study describes a trial carried out by the emergency services in Asturias (Spain). A toxic gas escaped from a tank in a factory that produces fertilizer. The trial started at 9:00am with a simulated call to the European emergency number (112).

Firefighters, police and medical services were mobilized. The first responders were the private firefighters employed by the company responsible of the scape. Next, personnel from the local fire and police departments joined. Finally, regional emergency services arrived. Firefighters established a security zone around the scape. The Intervention Chief established a coordination point at, approximately, 500 meters from the scape, where he had direct view to the scape. Police and ambulances cannot get inside the security zone, only firefighters. Police was used to alert the population in the close neighborhoods, block roads and reorganize traffic. Ambulances waited outside the security zone and carried victims to a near hospital. An Advanced Command Post was established in a truck parked in a public school a few kilometers away. The truck was used for meetings between officers of the different emergency services. It was equipped with communication and audiovisual equipment (see Figure 2-7). It took about one hour to deploy all the personnel in the incident area and a bit more to setup the Advanced Command Post in the truck.

For the firefighters, the first task was to stop the scape. Second, they extinguished the fire produced by it. Then, they watered the smoke to avoid the formation of toxic clouds. Meanwhile, they received calls from simulated victims that they had to rescue. These victims were intoxicated by the smoke, for example, while playing a football match in a close sports pavilion. The Intervention Chief managed these tasks in the coordination point by dispatching teams of two to four persons in one vehicle. Each vehicle was equipped with different tools, depending on the task. The teams carried out the task and came back to the coordination point. The ER operation ended at 13:00, after a total of four hours. Then, several meetings and briefings took place.

The hierarchy in the incident was clear, being the Intervention Chief the core of the operation. He reported to the Advanced Command Post that coordinated between emergency services. He also received information and alerts from them. Furthermore, he had to keep contact with the rest of firefighters. Figure 2-8 shows the Incident Chief and other firemen using several radios. One radio was used to communicate with the Advanced Command Post and another to communicate with the firefighters deployed in the area. They used different frequencies to keep the communications channel open. As there were several fire departments (private, regional and local), extra-radios were used. The situation would improve if technology that efficiently integrates all the communication channels were used. In these ER operations cellular networks collapse easily, so their use is discouraged. The amount of alerting calls by population is the main cause. Other networks are rarely available. Therefore, a MANET could be a suitable solution.

Video was used in the ER operation. A cameraman was transported by helicopter to the incident area. He recorded some minutes of footage and was carried to the Advanced Command Post. The video was shown to the officers in the TV screens of the Advanced Command Post. This is a manual version of delay-tolerant transport of video. The technology of DT-Stream aims to automatize processes like this one. Using a MANET to transport the video, it could be recorded by any firefighter deployed in the area using a helmet camera. Then, the movement of the different units can be used to transport the video so it reaches the Advanced Command Post, and also the Intervention Chief.

Figure 2-7 Inside of the Advanced Command Post

Figure 2-8 Firefighters in the coordination point using several radios

2.5.5 Mobility in ER operations

It is only in the recent years that emergency services have used GPS devices together with Geographical Information Systems. The mobility traces they recollect are useful for our purposes, but they are often difficult to obtain for security and privacy issues. When researchers can access them, they are sometimes incomplete and difficult to analyze. For example, they may contain the position of the vehicles, but not of the personnel. Nevertheless, GPS traces are very valuable even in suboptimal conditions. Fortunately, ER operations are scenarios where part of the mobility is dictated by emergency plans and the know-how of emergency services. Hence, the qualitative analysis of GPS traces can be complemented with the qualitative analysis of the many books and documents in this area. Obviously, none of these approaches, qualitative and quantitative, provides certainty of the movement in future ER operations, especially because humans are involved. Nonetheless, this analysis is crucial to reveal challenges and situations that a MANET is likely undergo.

Due to the mentioned difficulties, studies of mobility in ER operations are not abundant in the state-of-the-art. The Disaster Area Mobility Model, by [70], is the most noticeable contribution. It reproduces mobility of various emergency services (firefighters, police, paramedics, etc.) after a disaster. It defines four zones: incident site, casualty treatment area, transport zone and hospital zone, and then units moving in them and between them. Figure 2-9 illustrates this division. Units move strategically in the incident area, e.g. to carry casualties from the incident. Three case studies are implemented using this model, two small ones and a big one. They are based on real statistics of the incidents or trials, but not on movement traces. Although this represents the big picture of an ER operation and provides interesting insights, it is not adequate for our purposes. We focus on the transmission of video inside the area they call incident site. This is the most isolated area and where network infrastructure is more likely to fail and a MANET needed. Conventional communication networks are more likely to be available outside it or can be easily deployed, e.g. in zones like the so-called transport zone. For that reason, we are interested in studying specifically mobility of the incident site.

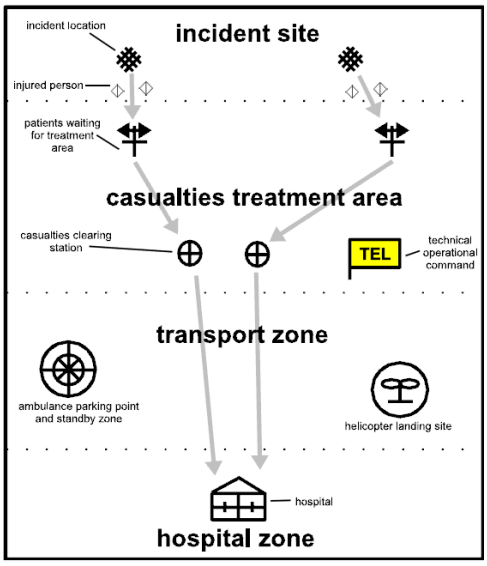

Figure 2-9 Zones in the Disaster Area Mobility Model

For our study, we ask for the collaboration of Bomberos de Asturias (the regional fire department) and 112 Asturias (the regional entity coordinating emergency services). Bomberos de Asturias is tracking their vehicles location. We had access to one year of GPS traces. We carried out a general study of these data traces (see Appendix A We analyze them as a whole and also taking a wildfire as example scenario. These analyses reveal interesting characteristics of a hypothetical MANET deployed with the vehicles of this emergency service. In specific, it reveals that the network is sparse, but that there are delay-tolerant routes to connect nodes. It also shows insights on repeating connectivity patterns. However, these general properties do not fully represent a potential MANET during an ER operation, because mobile computing devices carried by personnel are not represented.

We completed the analysis of raw GPS traces with a qualitative study. Our sources include reports from some incidents that we compared with GPS traces. Most of them come from wildfires, as they are by far the most common incident in the region of Asturias. Furthermore, we used the mentioned emergency plans and books of tactics. We have also been able to interview firefighters and officers and to assist to emergency trials, e.g. a chemical escape trial. All of it gave us a better understanding of how they move. These studies lead us to some general patterns of the organization and mobility in ER operations. It is important to take into account both, as organization conditions position and mobility. There are three patterns that we want to emphasize, as they will be especially useful in our further research:

Zones: The Intervention Chief is located in a position outside the Incident Area. The orography determines where the Incident Chief is, especially in wildfires in the forest. In an ideal situation, he must be able to see the incident, but still being in a safe place. For example, a good location is a hill from the burning forest is visible. In the chemical accident, see GPS traces in Figure 2-11, two areas concentrate most of the activity. One is the location of the Intervention Chief and the other is a hot point inside the Incident Area. In this case, the distance between them was in the order of hundreds of meters, enough to form a network partition. In addition, Figure 2-10 shows the location of various fire fronts and of vehicles frequent stops, labeled as P1 the Intervention Chief location. Thus, most ER operations show at least two zones where personnel are located. This pattern is sound with previous research like the Disaster Area Mobility Model. Moreover, the Intervention Chief is in most cases at a considerable distance of the Incident Area. This is important because it is likely to imply a network partition in a MANET.

Teams: Typically, firefighters work in teams and move together. A common structure is four firefighters and one vehicle. They use the vehicle to go to the points commanded and back to the Intervention Chief location. Since we do not have traces from people, it is not possible to know how exactly they move when they are outside the vehicle.

Intervention time: Teams of firefighters have a limited time of work in the Incident Area. They are dispatched from the Intervention Chief location and return back when their task is finished or they need rest. This has been confirmed by several of our case studies and interviews. The type of task they carry out and the equipment they need limits the time they can be in the area. Sometimes the time is set, such as when they need to use an oxygen tank, and sometimes varies more, such as when they extinguish a fire with the water in the fire truck. Often, when a team is resting, another is dispatched to substitute it. In Figure 2-11, the traces located in between these two areas draw the

path of the road between them, because teams of firefighters were driving back and forth.

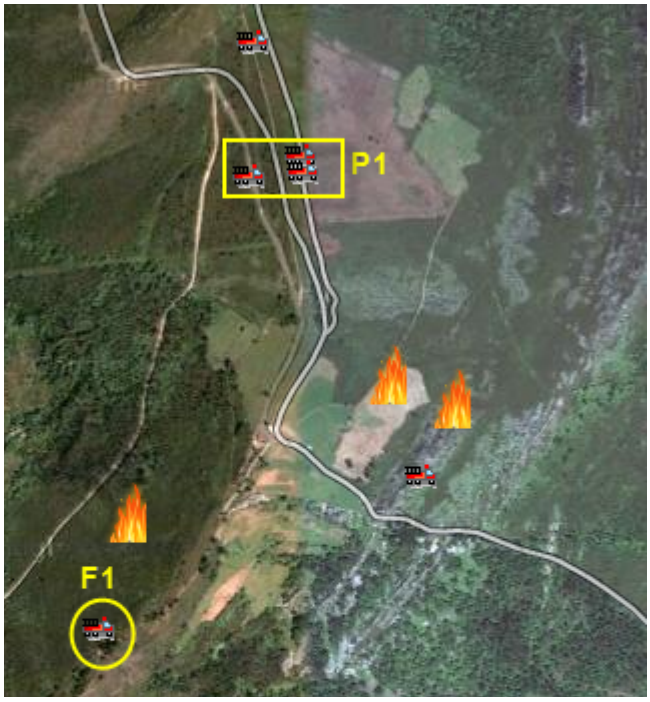

Figure 2-10 Zones identified in a wildfire

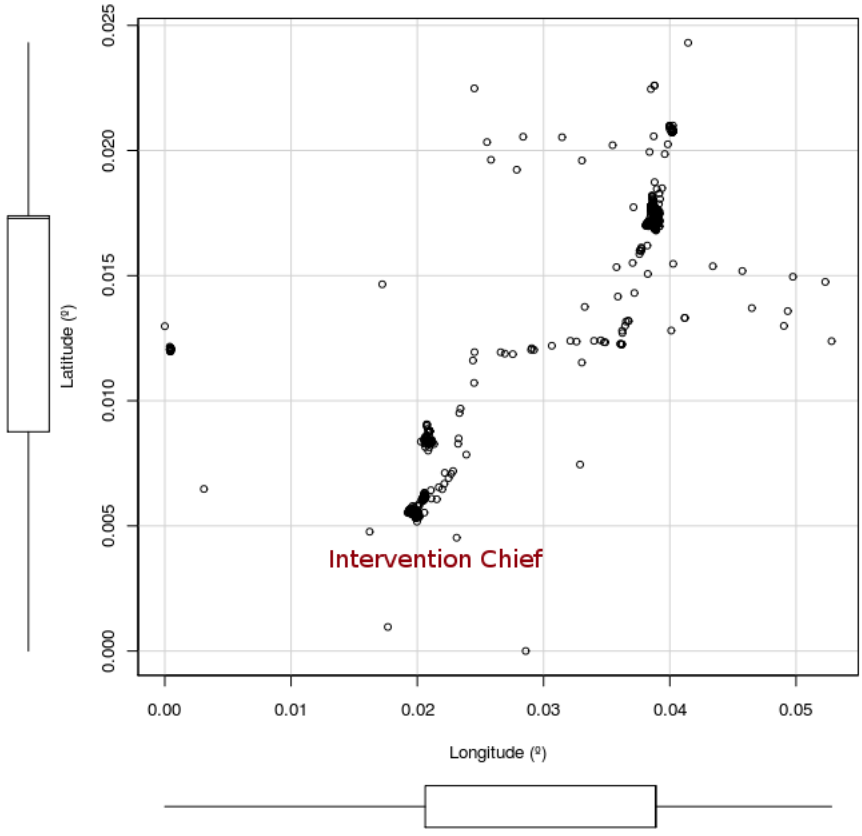

Figure 2-11 GPS traces from a chemical accident maneuver

The following Chapter establishes a set of requirements to solve the problems of this thesis. They are based on the technology described in this chapter and the ER application domain.

Chapter 3

Requirement Analysis

In this Chapter, we define a set of requirements for an ideal solution to the problems stated. The application domain, ER, is also used to make them more specific. They will be used to analyze related work and to design a solution. We have divided them into four groups. First, we have set requirements related to the support of mobility and the different connectivity configurations that may come as a result. Second, we explain the convenience of integration between the desired solution and existing network and video technologies. Third, we describe desirable resource management requirements, as it is a key factor for scalability. Finally, we discuss Quality of Service and Quality of Experience related requirements.

3.1 Adaptation to mobility and connectivity

Mobility and density of nodes determine connectivity and, hence, network topology. As we mentioned in the Introduction, a sparse MANET can lead to two different networking paradigms: conventional MANET/multihop routes and delay-tolerant routes. The ideal solution should support both of them.

Requirement 1.1 - MANET Support: The solution must support scenarios where video source and destination are connected through multihop routes. In this case, video streaming is possible. The behavior should be similar to streaming in the Internet, but with the additional unreliability and resource constraints imposed by the MANET. As a result, the user should be able to see the video as it is received, if there are enough available resources.

Requirement 1.2 - DTN Support: The solution must support scenarios where video source and destination are in different network partitions. Streaming from the source is not possible, because the destination is likely to receive the video from ferry nodes using store-carry-forward. However, ferry nodes may be able to stream video to the destination, which the user may reproduce with the delay introduced by the delay tolerant communication.

Requirement 1.3 - Paradigm transitions: The solution must support transitions between MANET and DTN scenarios. Mobility may separate source and destination in different partitions or put them together. The solution must detect these transitions and adapt to the new networking paradigm. It must also differentiate a temporal disruption and a transition from MANET to DTN scenario. Temporal route disruptions are common issues in MANETs and are normally solved by the routing protocol.

3.2 Integration and compatibility with existing technologies

We establish requirements of integration with existing technologies. The main reason is to preserve compatibility with existent hardware and software. In addition, working with off-the-shelf devices and applications eases the evaluation of proposals in real or realistic environments. However, these requirements limit the possible solutions, which may also complicate the design process.

Requirement 2.1 - Off-the-shelf video applications: The solution must integrate with off-the-shelf video applications and devices. Existent devices, such as cameras, use state-of-the-art codecs and protocols. The ideal solution must be able to support video generated by these sources. Moreover, it must be able to deliver video to existent client applications. Fulfilling this requirement implies that there is no need in creating new video inputs or outputs as far as we can reuse existent ones.

Requirement 2.2 - Off-the-shelf networking: The solution must be integrated with existent networking technologies. In specific, it must coexist with the standard TCP/IP protocol stack. There advantages are compatibility with other services in the network, possibility of using standard hardware and operating systems, and easy interconnection with other networks, such as the Internet.

3.3 Resource consumption

Resource management and resource consumption are important for any solution. The general goal is to keep them as low as possible, so resources are dedicated to deliver content and to provide good user experience.

Requirement 3.1 - Low network consumption: The solution must produce low network consumption. Not only MANET resources are likely to be scarce, but also video bandwidth requirements tend to be high. Therefore, the ideal solution should aim to deliver the video contents consuming as little network resources as possible.

Requirement 3.2 - Low overhead: The solution must introduce low traffic overhead. All network traffic that is not video is overhead and the aim should be to minimize it.

Requirement 3.3 - Low CPU and storage usage: Although mobile devices are becoming more powerful every day, keeping CPU and memory consumption low implies less energy consumption.

3.4 Quality of Service and Quality of Experience

Quality of Service (QoS) and Quality of Experience (QoE) are tightly bounded concepts that must be taken into account in any multimedia service. MANET and especially DTN video transport impose a QoS different from the Internet and a different QoE as well. The goal must be to leverage resources and mobility in order to get the best possible quality.

Requirement 4.1 - High video packet delivery: The ideal solution must deliver to the destination node all video packets that can be delivered. Although conventional video streaming may drop late packets, our ideal solution should still deliver them. In the ER application domain, video may be watched live, but also reviewed afterwards. Delivering late packets can enhance the quality of crucial video parts.

Requirement 4.2 - Lowest delay possible: The ideal solution must aim to minimize the possible delay. In a multihop route, packet delay is determined by packet forwarding and it must be low enough to provide good user experience. In a DTN, it is constrained by node mobility. Then, the solution must aim to find the fastest delay-tolerant routing alternatives.

Requirement 4.3 - Low video packet losses: The ideal solution must not drop or delete video packets. Those packets not delivered to the destination node should be stored in other nodes. The reason is that these video packets may be merged with the delivered ones and be used in offline analysis of the ER mission.

Requirements 4.4 - Highest possible quality: The ideal solution must aim maximum video quality for the video delivered. We consider quality in a wide sense, not only image definition, but also frame rate, frame size or video duration. Due to scarce resources, it may be impossible to provide the ideal video quality. Therefore, application domain characteristics and user preferences should be taken into account to decide which quality parameters must be preserved or penalized.

Chapter 4

MOMENTUM

This Chapter describes the design proposed to support video transport over sparse MANETs. First, we justify why an overlay network is the best alternative to solve this problem. Then, the overall architecture and every component on it are described. The components of this architecture are a result of analyzing the requirements from the previous chapter.

4.1 Overlay network

4.1.1 Why using an overlay network?

A sparse MANET can present two different scenarios for video transport. On the one hand, source and destination can be in the same network partition, the standard MANET scenario. On the other hand, they can be in different partitions, a DTN scenario. The fundamental problem is that DTN and MANETs are incompatible networking paradigms. The latter forwards over multihop routes relying on end-to-end connectivity; while the former uses the store-carry-forward paradigm to overcome network partitions. As we stated in the connectivity and mobility requirements (1.1, 1.2 and 1.3), we seek a solution that can transport video in both situations.

Most of the related work studies one category or the other, MANETs or DTNs. The MANET research community has focused on optimizing video streaming over multihop routes. They tackle issues such as unreliable links, low available bandwidth, congestion or routing and propose different schemes to solve them. However, these proposals are not prepared to cope with disruptions and network partitioning. On the contrary, DTN researchers have mostly studied transport and routing using store-carry-forward. For example, [96] study reliable data streaming using the Bundle protocol [22]. These proposals are not designed to work with end-to-end multihop routes and video transport is not considered either. All this research is useful to solve specific problems in MANETs or DTNs, but it does not cope with both networking paradigms. Hence, we must look for new alternatives that combine them. Several approaches can be taken. Standard network protocols could be modified in order to better support delay-tolerance and also multihop routes, like DtsOverlay in [97]. DtsOverlay is an overlay network to transport video in MANETs. It increases network reliability with a set of mechanisms that improve the MAC layer. It may be also possible to design a new networking architecture from a clean slate point of view, e.g., the HOP protocol [98]. Nonetheless, these options go against our requirement of being compatible with off-the-shelf networking (2.2). Another possibility is to modify client and server applications to adapt to DTN and MANET scenarios. Although feasible, this approach has many limitations. For instance, it is only possible to deliver video when source and destination are in the same partition. Using other network nodes, additionally to source and destination, is a

good alternative to overcome these limitations. These nodes can form an overlay network over the sparse MANET that adapts to the networking scenario and leverages network nodes to transport video.

Therefore, we propose to use an overlay network. It is deployed with a piece of software, called MOMENTUM, running in each overlay network node. The overlay network is used to coordinate several nodes in order to transport video from source to destination according to network conditions and available resources. This approach has been successfully used in sparse MANETs for purposes different than video transport. DENS [99] is used for event notification in ER operations. It proposes a publisher/subscriber solution to send and receive discrete events in a sparse MANET. The solution described in [100] enables delay-tolerant networking over a MANET running OLSR. It deals with network partitioning using the concept of structure mesh overlay network (SMON) for routing and service discovery. One interesting consequence of using SMON is that not all of the nodes in the MANET have to be capable of delay-tolerant transport. Data is forwarded in Bundles using the TCP convergence layer for DTNs [101]. Both proposals, DENS and SMON, introduce interesting concepts to deal with sparse MANETs, but they do not tackle video transport issues.

In our case, an overlay network has the potential to fulfill all our requirements. It is compatible with using off-the-shelf networking and applications. Moreover, it provides the necessary flexibility to implement adaptable transport policies. On the down side, an overlay network is less resource efficient than other approaches. Part of this inefficiency comes from constantly running the overlay network software in the nodes and from the generation of network traffic to support the overlay network. In addition, if the overlay network processes video packets, an additional delay will be introduced. Our design decisions must be aware of these drawbacks to minimize their effect.

4.1.2 Overlay network membership

Member management is a crucial part of the design process. Overlay network nodes must know what others are available for video transport. First of all, it is a question of how many nodes should be part of the overlay network. The more the overlay nodes, the more the resources available for video transport, but also the more overhead produced by the overlay network. It is a tradeoff between potential resource consumption and availability. The optimal decision is to take only those nodes that are essential to transport video efficiently. However, it is impossible to know it in advanced, or even to estimate it. We must also determine how nodes join and leave the overlay network and how they identify themselves. Membership could be dynamic, so nodes can join and leave the overlay network. This requires a protocol to manage these events and notify them to network nodes. It also requires mechanisms to store or discover members when they are needed. On the contrary, membership could be static. A predefined set of nodes could be established as permanent members. This option is much easier to implement, but implies that others cannot take part of the overlay.

In the context of ER, the video transported by the overlay can be mission-critical. In order to maximize odds of success, we have decided that all the nodes in the MANET are members of the overlay network. This approach maximizes available resources and simplifies overlay network design. First, membership management is unnecessary as well as setting any membership conditions. Nodes do not have to exchange messages to

discover, add or remove nodes from the overlay network. They do not have to store a list of members. Each node knows that every other node in the network is part of the overlay network and can communicate with it. This eases the design of other mechanisms too. For example, if a node has to look for a ferry node, all nodes in the same partition will be candidates. We consider feasible to use all MANET nodes in an ER operation. At this point and for security reasons, we assume that all devices that join the MANET are under our control. Emergency services own the devices used for the MANET, thus, they can be configured a priori. In our case, this implies that every MANET node runs MOMENTUM.

4.2 MOMENTUM architecture

This section is dedicated to explain how the architecture of MOMENTUM was designed. First of all, it has to be flexible and easy to evolve as our understanding about the problem progresses. For that reason, we have opted for a component-based architecture. We apply the principle of Separation of Concerns to determine what components are needed. The analysis of requirements draws a set of aspects that the desired system must accomplish. Ideally, these aspects involve functionalities that can be isolated in components and studied individually. It is our aim that components can be studied and evolved separately, as for instance the Internet layers do. This should simplify system design, implementation and evaluation. Figure 4-1 illustrates the components resultant from this analysis.

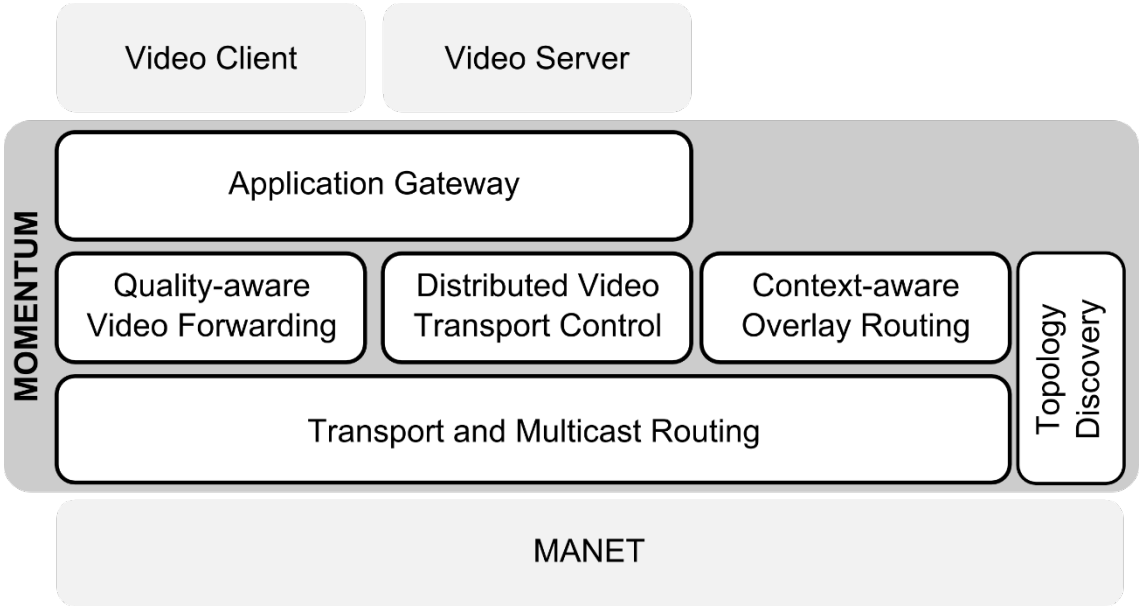

Figure 4-1 MOMENTUM Architecture

Connectivity and mobility requirements (1.1, 1.2 and 1.3) imply that the overlay network must operate over MANETs and DTNs. For this purpose, MOMENTUM must determine the networking context (DTN or MANET) taking into account which node is the destination of the video. Then, the overlay network must adapt to these conditions and pursue the best possible QoS defined by requirements 4.1, 4.2 and 4.3. One of the key advantages of using an overlay network is the possibility to divide end-to-end communication between source and destination. Overlay network nodes are used to contribute to video transport towards video destination. This is in many occasions

necessary in both MANETs and DTNs in order to increase video packet delivery (requirement 4.1), reduce delay (requirement 4.2) or improve reliability (requirement 4.3). However, the method and criteria to select these intermediate nodes are not trivial as they may depend on factors such as context, network topology and available resources. The Context-aware Overlay Routing component decides what is the context (DTN or MANET) and which overlay nodes can be used to get the best QoS accordingly.

In order to achieve interoperability with current multimedia technologies, we have stated that off-the-shelf applications are in the ends of the communication path (requirement 2.1). Most video servers and client/players are designed to operate in the Internet. Therefore, they are unlikely to be prepared for the challenges of MANETs and DTNs. In addition, there are many different multimedia technologies in the form of video codecs, transport protocols, control protocols and so on. Although many of these are standardized, particular implementations can differ from one another. In order to make MOMENTUM as independent as possible from the particular multimedia technology or applications used, we introduce the Application Gateway component as an interface with applications. It aims to separate external multimedia technologies and internal overlay network actions. Thus, if applications were changed, ideally only the Application Gateway component would need modifications.

Video applications exchange not only video packets, but also signaling information related to user actions, preferences or video stream properties. For example, many standard applications use RTP to stream video packets, RTCP to report about the streaming process, RTSP to express user or server actions and SDP to describe the streams. All these metadata accompanying video cannot be ignored in the overlay network. On the one hand, we need to preserve end-to-end signaling between applications to fulfill requirement 2.1. On the other hand, signaling has to be taken into account inside the overlay network. Control messages generated by the applications are important for resource management (requirements 3.1 to 3.3) and also to provide video quality adapted to user expectations (requirement 4.4). In addition, overlay network nodes may also need to produce their own signaling. The component called Distributed Video Transport Control manages signaling inside the overlay network and defines the transport policies according to it.

Video packets are likely to be forwarded several times between overlay nodes before they reach the destination. These intermediate steps heavily impact QoS and QoE. Due to the heterogeneity of ad-hoc networks, each of them may be carried out in completely different network contexts and with uneven available resources. In addition, forwarding must be sound with the policies defined by Distributed Video Transport Control as, for instance, a consequence of user preferences. The Quality-aware Video Forwarding component encapsulates the mechanisms to carry out this process and to achieve the desired quality (requirements 4.1 to 4.4).

Overlay network nodes must communicate using off-the-shelf networking (requirement 2.2). This poses a challenge because standard network protocols have problems over MANETs and DTNs. Moreover, as we have seen in our previous argumentation, we have two different traffic types: video streams and messages for signaling or other overlay management tasks. The requirements for them are different, but they ask for some common mechanisms. For example, an acknowledgement may be necessary for different types of messages. In addition, video packets and messages may be distributed using multicast, e.g. to increase resilience. Instead of replicating the same mechanisms

in different components and letting each of them cope with the underlying network separately, we introduce the Transport and Multicast Routing component. This component gathers all mechanisms to successfully send and receive data through the network and at the same time provides enough flexibility to adapt to the specific requirements of each component. Furthermore, a single point to manage traffic between the node and the network leverages other possibilities to enhance QoS, such as establishing priorities and assigning network resources to traffic types.

Knowing the network topology is required by several components for tasks such as determining DTN or MANET contexts, overlay routing or multicasting. Therefore, we introduce the Topology Discovery component. It is in charge of finding out network topology and making it easily available to the rest of the components.

As a result of this analysis, we have six collaborating components. They can be grouped according to different criteria. On the one hand, Context-aware Overlay Routing and Distributed Video Transport Control define transport polices, while Quality-aware Video Forwarding and Transport and Multicast Routing implement them in specific mechanisms. On the other hand, the Application Gateway, Topology Discovery and Transport and Multicast Routing are interfaces with the outside world; while the others determine the core of the overlay network behavior. Finally, all these components and their interactions must be designed taking into account requirements related to resource management (3.1 to 3.3). The resultant system must aim low resource consumption and overhead.

4.2.1 Context-aware Overlay Routing

The Context-aware Overlay Routing component identifies network context as DTN or MANET and selects nodes that help in delivering video to its destination. Using the Topology Discovery, any node can determine whether there is a path towards another node or not. Thus, any node that stores video can determine if it is connected to the destination of this video. If the destination is connected, the overlay network operates in MANET node. Video is distributed using the paths over the MANET. Otherwise, the overlay network operates in DTN mode. Video is distributed using delay-tolerant mechanisms such as store-carry-forward. The nodes selected to transport video have different roles depending on the operation mode.

In MANET mode, the node that stores video and its destination are connected. A MANET route can be used to forward video. Nonetheless, it is well-known that reliability decreases with route length. Therefore, it is useful to avoid long routes to increase reliability and resilience. A **relay node**, or just relay, is a node that receives video packets, forwards them immediately and stores a copy. The advantages of using relays emerge from splitting long routes towards the destination into shorter ones. Shorter routes are less likely to lose packets and if a packet is lost, it will be retransmitted from the relay. As a result, resilience improved and network resource consumption decreases because the retransmission is done closer to the destination. However, relays introduce overhead over using just the network path, because they store a copy and forward each packet at the application level. In addition, when the relay is not in the shortest route towards the destination, additional hops are introduced.

In DTN mode, the destination is disconnected from the node that stores the video. Thus, the options to deliver it are two: waiting for the possibility of connecting to the

destination in the future or finding other nodes that may connect to it. A **ferry node**, or just ferry, is a node that receives packets, stores them and forwards them, but not immediately like a relay. Instead, a ferry forwards packets when it connects to the destination of the video, i.e. switches to MANET mode, or when it meets with a better ferry. Using ferries also introduces overhead, but in a DTN it may be the only alternative to deliver video to the destination.

Video may traverse several ferries and/or relays to reach the destination. Therefore, the selection criteria for these nodes are fundamental to achieve the best possible QoS and QoE. For example, when there are several ferry candidates, they may reach the destination at different times and produce higher or lower delays in video packets. Ideal criteria should take into account factors from resource availability to network topology and node mobility. At this point, it is complex to consider all these parameters. Therefore, we focus on criteria based on network topology and mobility and assume that there enough available resources in the network (e.g. bandwidth) and in the nodes (e.g. battery, storage and processing). Future research should consider available resources too and benefit from the criteria developed in this thesis.

Relays have minimum overhead when they are in the shortest route towards the destination. Thus, they must be selected according to their position in the network. As it is described later, Topology Discovery can provide a list of nodes that are in the around the shortest path between two given nodes. These are good candidates for relays. In the next chapter, we provide a selection technique based on this idea as proof of concept of relay functionality. Although beyond the scope of this thesis, the potential of relays could be higher and can be used to build more complex video distribution techniques the MANET, e.g. disjoint multipath routes.

Ideally, the best ferry is a node that can reliably deliver video packets with the lowest delay, because it will meet soon with the destination or with other good ferries. Hence, ferry nodes must be selected according to their expected mobility. This is not trivial, because it requires estimating future mobility, unless there are nodes with set movement. In the next chapter, we use random selection as proof of concept for ferry nodes. In Chapter 6 , we analyze this problem in more detail for the ER application scenario and propose a delay-tolerant routing protocol.

The overlay network can use these criteria to select ferries and relays in a centralized or a distributed fashion. The centralized approach has some advantages in networking applications (e.g. controllers in software defined networking), because an entity can have a global view of the network status and take better decisions. In our case, it would mean that one node decides for all. However, this approach is not feasible in a mobile network that is likely to be partitioned. Then, there are several distributed mechanisms. For instance, the network is divided into clusters, for instance one cluster per partition. Then, a node inside the cluster acts as cluster head and decides for the others. This combines the advantages of the centralized approach and overcomes the problem of partitioning. Nevertheless, the formation of clusters and the selection of cluster heads are complex tasks on its own and an ongoing research topic. Another option is to decide between several nodes, for example the source and destination collaborate to select relays in MANET mode. The problem is that finding consensus in the selections can also be difficult. To avoid these problems, the best option is that each node selects ferries and relays individually. All nodes use the same criteria, together with local and remote information requested from other nodes. This is feasible because the information using in the selection criteria can be available to all nodes. This mechanism avoids any

type of negotiation and its associated overhead. In a real system deployment, this may produce the situation of selecting a node that it is not willing or not capable of acting as relay or ferry. Nevertheless, this can be done safely in our environment since we have assumed enough available resources in the nodes and the network.

4.2.2 Application Gateway

The Application Gateway component is the interface between MOMENTUM and off-the-shelf video applications. These applications are a video server, e.g. in a camera, and a video client, e.g. a video player. There are plenty of video codecs and protocols that applications use. The design and implementation of this component depends on them. For that reason, we focus on applications that use RTSP/RTP/RTCP protocols and MPEG family encoders (MPEG-2, h.264, etc.). These are well-known standards with several open source applications, which eases the integration of the applications and MOMENTUM. However, there are incompatibilities between the design of these technologies and MANET or DTN networks. The Application Gateway does not only solve incompatibilities, but also provides the necessary functionalities for other MOMENTUM components to work properly, i.e. Distributed Video Transport Control and Quality-aware Video Forwarding.

RTSP/RTP/RTCP were conceived for video streaming over the Internet and they are not disconnection resilient. A client uses RTSP to request a video from a server and to negotiate transport parameters. For example, the client sends Options, Describe and Setup messages in order to establish the basic streaming parameters with the server. Then, RTP and RTCP are used to transport video streams and exchange transport control messages respectively. They will fail in a DTN or a MANET. For example, RTSP typically needs an open TCP connection and a video server will stop streaming otherwise. Moreover, applications use keep-alive mechanisms, such as RTCP Receiver Reports, and if they are not received, the server will stop as well. Therefore, the Application Gateway must simulate a connected network for the applications. The mechanisms needed to maintain client and server alive differ from application to application, but the main tasks are to keep TCP connections open, to answer messages that ask for a response and to generate messages expected by applications, such as RTCP Receiver Reports. In the next chapter, we discuss the specific implementation details for the applications selected for our first prototype.

Another problem is how applications locate and request videos in the network. In the Internet, a client application requests a video with an URL typically composed by the name of the video and an IP of the host where it is located. This needs to be different in our environment. On the one hand, server and client can be in different partitions. On the other hand, MOMENTUM must be in the middle between the applications. A possible solution is forcing applications to use a different system to locate resources. For example, request video to the local MOMENTUM instance. Then, MOMENTUM must have its own mechanisms to find contents. This alternative is complex and may need the collaboration of users, which we would like to avoid. Therefore, we have selected to make MOMENTUM transparent to the applications and the users. The main advantage is that applications and users can use the original video URL. A client requests videos as if it were connected to the server, but MOMENTUM silently intercepts and distributes the traffic they exchange. This is feasible because most operating systems allow this type of functionalities using firewalls. As we explain in the next chapter, this was implemented using IPTables in Linux.

The Application Gateway also provides functionalities to other MOMENTUM components. The Quality-aware Video Forwarding and the Distributed Video Transport Control manage the video packets and control messages that applications exchange. These components are independent to the technology used by applications. Thus, the Application Gateway translates from RTSP, RTP and RTCP to the protocols used by them and vice-versa. In addition, Quality-aware Video Forwarding implements transport mechanisms supported by metadata associated to content, e.g. the type of frame contained in a packet, which depends on the video encoder used. This component carries out this task as well.

Finally, off-the-shelf video players are not the best option for video delivered over a DTN, as we will see in following chapters. For that reason, in Appendix C we describe a prototype of a tailored video player and the Application Gateway.

4.2.3 Distributed Video Transport Control

The Distributed Video Transport Control component contains a control/signaling protocol for MOMENTUM. The main aim is to express user actions and preferences and use them to apply transport policies. However, this requires a thorough knowledge of the services that can be deployed, the user requirements and the possible transport policies, which we lack at this point of our research. This thesis focuses on transport and routing mechanisms and leaves signaling as future work. Nevertheless, basic functionality is needed to get a working MOMENTUM prototype. The current design of the Distributed Video Transport Control component covers the minimum functions to do it. They are, first, distributing essential signaling information from the applications and the nodes and, second, controlling the Quality-aware Video Forwarding component.

Video applications use RTSP as signaling protocol. They expect their messages to arrive at the other end; so they can carry out actions such as play or stop, inform about video metadata and, eventually, maintain the session status. This component must ensure that the signaling generated by the application at one end is reflected at other end. Moreover, application signaling is also interesting for the overlay network as it helps in defining transport and resource management policies. For example, pause or fast forward actions can be used to drop packets that are not interesting for the user. Thus, this component shares signaling information with nodes running client and server applications, but also with ferry or relay nodes in charge of video transport. RTSP on its own is not sufficient to fulfill this task inside the overlay network, because it is designed for the client/server paradigm and does not support ferries or relays. For that reason, we need to find an alternative solution.

Signaling protocols used in the Internet, such as those for P2P networks or SIP, need a connected network. Protocols designed for MANETs may be used or adapted to carry out these tasks. For example, the event notification system DENS [99] has some similarities with the requirements for a signaling protocol. Nevertheless, these solutions are too complex to cover just the basic functionalities needed in the current MOMENTUM design. For that reason, we have chosen to extend RTSP with the fields needed inside the overlay network. The Application Gateway extends RTSP messages from the applications and passes them to this component. Then, they are distributed to all the ferries and relays that take part of video distribution and to the source or destination node, depending on which node generated the original RTSP message.

Messages are distributed using the Transport and Multicast Routing component to any node that has received video packets.

Now, we discuss the necessary fields to extend RTSP in the overlay network. First, there may more than one pair of applications exchanging video and control messages in the network, i.e. several video sessions. It is necessary to differentiate them. In addition, ferry and relay nodes need to identify where the applications are located. For simplicity, we assume that client and server exchange only one video stream, although our design can be easily extended to support several. One possible solution for this problem is to generate a unique identifier for each pair of applications. Then, each message belonging to this pair of applications is tagged with this identifier. However, this would mean that every node has to be informed about it and inconsistencies may occur if a node receives a message with an unknown identifier. Then, there is the problem of how to generate unique identifiers. For that reason, we have chosen a simpler solution. It is to add client and server locators (IP addresses) in every message. To distinguish between several video sessions between the same nodes, an integer is also added. It is generated with a counter in the source node, so the combination of source and integer is unique. This is likely to consume more resources, as the size of these three fields may be bigger than a single identifier. However, its simplicity is enough to justify its use. Furthermore, messages must be delivered to every node that has received video packets, i.e. everyone selected as relay or ferry. As nodes select relays and ferries independently, this list is likely to grow over time. To keep track of the current relays and ferries, a simple alternative is to add the list of them as a field in every message.

This component manages Quality-aware Video Forwarding to ferries or relays that are determined by Context-aware Overlay Routing. We use RTSP state machine as a guide of whether to forward video or not. If RTSP state is Play, this component retrieves ferries or relays from Context-aware Overlay Routing and activates video forwarding. Otherwise, it remains idle. To implement this policy, nodes must be aware of the state of RTSP. In the applications this is tracked by the messages received, but this is not possible in the overlay network. Messages can be received in the wrong order or not received by some nodes. For that reason, we incorporate another field in RTSP messages that indicates the current state.

4.2.4 Quality-aware Video Forwarding

The Quality-aware Video Forwarding component manages video packet forwarding between overlay network nodes. Packets are passed from the source node to the ferries, relays and eventually to the destination node. As we have described, Distributed Video Transport Control establishes when video is forwarded and Context-aware Overlay Routing determines towards what overlay nodes. The decisions of when and where to forward affect QoS and QoE. For example, delay is heavily influenced by ferry selection. Nevertheless, packet forwarding has a great influence on quality as well. In specific, we study how to use forwarding to cope with packet losses produced by disruptions and to adapt video to available network capacity.

Disruptions occurred during video packet forwarding produce intensive packet losses, as we demonstrate in Chapter 6 One of the causes is that some protocols, such as ARP [102], react slowly to mobility. This affects video forwarding especially because video packets are transmitted as a continuous stream. Thus, a disruption causes the lost of many consecutive packets. We propose to solve this problem using an Error and Flow

Control mechanism for video packets. This mechanism is specific to video packet forwarding and works over the common ones implemented in Transport and Multicast Routing.

Even if packet losses were completely avoided, packet delivery is limited by network capacity. In a sparse MANET, network capacity is largely determined by mobility. Nodes move and connect intermittently, which limits the amount of information that can be exchanged. In addition, there are other limitations, e.g. inefficient delay-tolerant routing that cannot find the best ferries. For that reason, it is interesting to get the best possible quality from the video packets that can be delivered. This is a video adaptation problem, but its constraints are different from the constraints of video adaption in the Internet. Again, mobility causes the main challenge, because it is often impossible to predict and, as a consequence, it makes impossible to predict network capacity as well. In Chapter 7 we study this problem and propose a solution based on adapting video frame rate through the forwarding order of the packets.

Finally, this component must also manage video packets of different streams and sources. Most of the points in the discussion for Distributed Video Transport Control apply also here. To work smoothly with current standards, we have chosen to extend RTP with the necessary fields. In this case, it needs to identify the stream with the identifier generated for Distributed Video Transport Control. This component manages both RTP and RTCP packets, another header field is added to indicate if a packet is RTP or RTCP. A more thorough discussion about how to extend RTP, which goes beyond the scope of this thesis, can be found in our paper [103].

4.2.5 Transport and Multicast Routing

The Transport and Multicast Routing component is the interface of the overlay network with off-the-shelf networking. We choose to work with the most extended protocols. First, we consider that the MANET is deployed using a protocol from the 802.11 family, as WiFi network cards are available in most mobile devices nowadays. Second, the network works using IP. While deploying the overlay network over IP is a possibility, it makes the implementation of this component more complex. Therefore, we choose to deploy it over an existent transport protocol, being TCP and UDP the standard options. TCP is an attractive alternative, as it provides several interesting features. In the Internet, TCP is not always used for streaming as it introduces additional delay and retransmissions of late packets are not interesting. Nevertheless, TCP reliability can be useful over a MANET. The problem for TCP arises when disconnections and route breaks appear. Then, its performance drops drastically [7], which makes it useless. UDP does not suffer from these issues, as it does not have any advanced protocol mechanism. Therefore, the overlay network is deployed using UDP. This component sends and receives UDP datagrams between overlay nodes.

UDP is a simple protocol and lacks the transport mechanisms that MOMENTUM components need. This component provides several functionalities to be the common entry and exit point for them. First, it provides an interface for them to send and receive their packets. Components have different transport requirements depending on the situation or the type of data that is sent. Therefore, this interface is flexible so components decide the mechanisms that are applied in each occasion. Second, components may want to send packets with different relevance. For example, two video packets affect user quality differently depending on their content. A useful mechanism

to cope with them is transport prioritization. Components assign a priority level to the packets they send and it is reflected on how network resources are assigned. For example, the most important packet is sent the before the others. Third, some packets also need reliable transport to ensure that they are delivered. This component must implement a mechanism to cope with reliable transport. In addition, some packets have to be delivered to several nodes. For example, signaling messages that must be delivered to all ferries and relies. For that reason, it is convenient to implement multicast. Finally, other transport mechanism are not strictly needed, but they can improve system performance by saving resources using message fragmentation and aggregation or reducing packet loses with flow control.

Transport protocols for MANETs are an active research field. Proposals exist to create completely new protocols or to improve TCP. Nevertheless, most of them are not solid proposals yet. Moreover, none of them provides the flexibility needed in our case, so we would be bound to use a different protocol depending on the situation. For that reason, we decide to design our own transport and multicast routing mechanisms and implement them inside this component. Nonetheless, the architecture of MOMENTUM supports that this component is complemented or substituted by future proposals. Chapter 5 thoroughly studies the design, implementation and evaluation of this component.

4.2.6 Topology Discovery

The Topology Discovery component provides an interface for other components to find out about network topology. This information is crucial to implement store-carry-forward or to build overlay routes. The first challenge in the design of Topology Discovery is how to discover the topology of the network. On the one hand, this component could discover the topology itself. For example, sending and answering broadcast packets to find out connected nodes. However, the overlay network works over off-the-shelf networking and an already working MANET. Therefore, this network needs a routing protocol that already can carry out topology discovery. Hence, the best option for this component is to extract information from the routing protocol and avoid the implementation redundant techniques.

Another question rises at this point: what routing protocol the MANET should use? This is an essential decision in the design of this component. At this point of our research, we assume that we can choose the routing protocol. Otherwise, the design of this component would have to be changed, but the other components would need only minimal or no modifications at all. Reactive protocols have showed efficiency under in some conditions, e.g. very big networks. Nonetheless, they present a delay in the route discovery process that may hinder video transmission. Furthermore, our application scenario is a sparse MANET, which are likely to have a small number of nodes in each partition. In this case, the fast response of proactive routing protocols seems to be a better alternative. Proactive protocols build routes to all the nodes in the network partition, which is convenient for our purpose of building overlay routes. Among the existing protocols, OLSR is the most evolved one. The MANET Working Group has released several versions of its specification, e.g. [16]. Furthermore, one of its implementations, called olsrd, allows the extension through plugins, which is useful for our purpose of extracting information from the routing protocol.

The plugin inside the routing daemon and this component establishes a cross-layer connection from which we can extract information. OLSR has available a list of the 1-hop and 2-hop neighbors of the node, as well as the routing table containing destination, next hop and number of hops. Another data structure contains the 1-hop neighbors of any other node in the partition (we call them remote neighbors), because this is used to populate the routing table by calculating shortest paths. All these data is used to select relay and ferry nodes or to plan multicast routing. For these purposes it is also interesting to calculate the nodes in the path towards another node. This information is not directly available from OLSR, but can be estimated using the remote neighbors sets. Therefore, Topology Discovery also implements an algorithm to calculate them.

In the next Chapter, the design of the overlay network is used to implement a first prototype that can transport video over DTNs and MANETs. For that purpose, we propose adaptation and transport mechanisms and analyze them.

Chapter 5

MANET and DTN adaptation

The main goal of this chapter is to demonstrate that MOMENTUM adapts to MANET and DTN scenarios, as it is established in the Requirements 1.1 to 1.3. Adaptability is a core characteristic of MOMENTUM. For that reason, its components are designed with this aim. MOMENTUM detects the network context and selects the appropriate transport alternatives. Transport and Multicast Routing is a key component to achieve this goal, because it implements common transport mechanisms that can be used by the other components in one or both contexts. This component, together with the basic design of the others, is enough to build a first MOMENTUM prototype in Java. Then, we evaluate it in a network emulator using different density and mobility configurations of random mobility scenarios and comparing it with the standard client/server solution. The results show that the overlay network adapts to mobility/density variations that form MANETs and DTNs. In addition, the experiments reveal design weaknesses and strengths, implementation challenges, and relevant insights to prioritize next steps to improve MOMENTUM. Part of these results where published in [104]

5.1 Transport and Multicast Routing

Transport and Multicast Routing is used by other MOMENTUM components as the interface with off-the-shelf networking (UDP). In addition, it implements a set of flexible transport mechanisms that can be used by other components. They pass their packets to Transport and Multicast Routing and select a configuration for these mechanisms. They also receive packets through this component. This section describes the detailed design of these transport mechanisms. To support them, a header with additional fields is added in front of the packets generated by the other components. This header is only visible to Transport and Multicast Routing, so packets are encapsulated as other protocols, such as IP, do.

5.1.1 UDP interface

Transport and Multicast Routing sends and receives UDP datagrams. UDP is a simple protocol, so the main design challenge is how to manage UDP ports. Ports are basically an identification mechanism to separate the communication of different applications. One or several ports can be used. Furthermore, if several were used, they could be assigned dynamically or statically. The main advantage of using several ports, e.g. one for the packets of each component, is that the operating system assigns a separated buffer for each. Thus, the problem of buffer overflow, which is mentioned later, is less likely, although not necessarily solved. In addition, the traffic can be easily monitored, which is an advantage in the evaluation process. However, it requires a complex port management mechanism, especially if ports are assigned dynamically. Moreover,

MOMENTUM may collide with other services that may want to use the same ports. We consider that the advantages of using several ports do not pay for the additional complexity they introduce. Therefore, Transport and Multicast Routing uses one UDP port. This makes configuration, design and implementation easier, but requires some additional mechanisms, which in our opinion are not as complex as port management. First, buffer overflow must be considered carefully and, for that reason, we include a flow control mechanism. Second, MOMENTUM needs an internal mechanism to relate packets and components, because port number is the same for all packets. A field in each packet identifying its type is used for packet dispatching, as we explain later.

5.1.2 Reliable packet delivery

Communication over MANET is not reliable; packets may be lost for several reasons, e.g. collisions produced in the shared medium. UDP lacks mechanisms to correct and prevent packet losses. However, MOMENTUM must ensure that some packets are delivered. While the loss of a video packet may affect performance slightly, the loss of a signaling packet produces affects it heavily. For example, a signaling packet requesting a video must reach the video source, otherwise video transport will not start. Moreover, not only signaling can benefit from reliable transport, but also overlay management messages and even video packets containing important data. For that reason, Transport and Multicast Routing must provide a common mechanism to ensure that packets are delivered.

Reliable transport starts with preventive actions, e.g. avoiding congestion. MOMENTUM applies them in different components, such as flow control in this component and in Quality-aware Video Transport, as Chapter 7 explains. In addition, the selection of relays and ferries by Context-aware Overlay Routing also influences reliability. Nevertheless, these preventive mechanisms do not ensure that all packet losses are avoided. Corrective mechanisms are needed as well. The most common solution is to use acknowledgements and retransmission. Acknowledgements can be positive (ACK), which are sent when a node receives a packet, or negative (NACK), which are sent when a node expects a packet but is not received. Using NACKs is not feasible for us, because not all packets require reliable transport. Hence, it is difficult to know whether a packet loss should trigger a NACK or not. Positive acknowledgements fit better as a solution. They can be used for each packet or for a group of packets. The latter implies using one ACK to acknowledge several received packets. This approach reduces the number of ACKs, which saves network resources. However, it opens the challenge of deciding when to send the ACK. For that reason, we opt for the simpler approach of using one ACK per packet.

An ACK must identify the packet that was received. Therefore, packets must contain a unique identifier in the overlay network. This identifier can be generated in several ways. A common solution is to use a counter that is incremented with every packet sent, i.e. a sequence number. This counter is included in the packet and used by the ACK. Assuming the counter is big enough, this generate a sequence number unique for the packets generated in a node. However, packets generated in different nodes may still have the same identifier, which is a problem in the overlay network. At some point a node may have two packets with the same sequence number from two different nodes. One alternative is using a distributed system to assign sequence numbers. Nonetheless, it is too complex for our purposes. Instead, our solution uses the IP address of the node that generated the packet. If sequence numbers generated by each node are unique, the

combination of sequence number and IP address is also unique. Thus, both the sequence number and the original source node of the packet are included in the header of each packet. For example, a video packet is identified by the IP address of the node where the video server is and the sequence number assigned by Transport and Multicast Routing in that node.

Retransmissions are needed either if the packet is lost or the ACK is lost, because there is no way for the sender to distinguish between these scenarios. A timer is a common mechanism to trigger retransmissions. In protocols like TCP, this timer is calculated using the Round Trip Time (RTT) and using estimations to adapt to get the most of the channel. A short timer may produce unnecessary retransmissions, whereas a long timer may increase the delay unnecessarily. In a MANET, the calculation of this timer is complex and falls out of the scope of this thesis. For that reason, our current design considers a timer of a fixed duration that can be configured with a parameter of MOMENTUM. Another design challenge is the maximum number of retransmissions allowed. Limiting this number implies that a node desists on trying to forward a packet at some point. This may be useful in some situations. For example, if a node in the route is congested, retransmitting dropped packets may produce more congestion. The ideal number of retransmissions depends on network conditions, which is again a complex problem out of our scope. For that reason, we opt for a configurable parameter that set a maximum of number retransmission. Infinite retransmissions, i.e. not limiting the number of retransmissions, are also possible.

Acknowledgements and retransmissions are applied in every hop in the overlay network, for example, between the video source and a ferry node or a relay node and the video destination. This ensures that packets are successfully delivered to the next node in the overlay network route. Nevertheless, there is no way for the original source of the packet to determine if it was delivered to its final destination. This option is interesting in some situations. For example, it would be interesting for the video server to know if the client received a video packet or for the video client to know if the server received a signaling packet. Therefore, we add an additional feature to this scheme, the possibility of sending an ACK to the original source of a packet. This feature also implies that an ACK must be delivered to a node in a different partition. Thus, it may also require store-carry-forward.

Finally, a node that receives a packet must determine the type of action required. It may require an ACK to the previous node in the overlay network route or it may require an ACK to the packet source node. It may require both of them or none of them as well. Hence, a mechanism is necessary to identify among these possibilities. A field in the packet header is included for this purpose.

5.1.3 Flow control

Standard operating systems assign a sending buffer to each UDP port. When an application sends a datagram, it is stored in this buffer before the operating system sends it to the network. A buffer overflow occurs when an application tries to fill this buffer over its capacity. In the Internet, this problem is not as frequent as buffer overflow in the receiver. However, this is a relevant problem for MOMENTUM, due to the combination of store-carry-forward and streaming. For example, the video source generates video packets that stores. Then, a route between source and destination appears, so there is a path to deliver video packets. Sending packets at the highest rate

possible by the application is likely to produce a buffer overflow. On the contrary, sending packets at a slower rate (e.g. the streaming rate) may not get the most of the transmission opportunity. Thus, the ideal situation is to send at the maximum rate that avoids buffer overflow in both the sender and the receiver.

This is essentially a congestion avoidance problem, but UDP does not implement any mechanism to solve this. An elaborated solution is out of the scope of this thesis. For that reason, we propose a simple approach that is enough for our current purposes. If needed, a more complex mechanism can be added in future versions. We have selected to limit the sending bitrate of MOMENTUM. A process of Transport and Multicast Routing schedules packet sending according to the maximum rate configured. If the maximum sending rate is set correctly, buffer overflow is avoided. We are aware that network bandwidth is likely to be underutilized, which may be translated in less video delivered in our experiments.

5.1.4 Store-carry-forward

Delay-tolerant networking support relies on using store-carry-forward. This paradigm is implemented in several MOMENTUM components in different ways. Transport and Multicast Routing implements it as well. MOMENTUM can send packets to one or several overlay network nodes. Packets must be delivered to them, but this only makes sense if a network route exists according to the routing protocol, i.e. OLSR. Otherwise, if a UDP datagram is sent towards a node that is not in the routing table, it will be dropt. To avoid this, Transport and Multicast Routing checks whether a node is connected before triggering any other transport mechanisms and forwarding the packet to it. For this purpose, it asks with the Topology Discovery component. Transport and Multicast Routing stores a copy of the packet until it has been forwarded to all the nodes established by multicast routing.

5.1.5 Transport priority

Packets from different components have different influence in resource management policies or the QoS and QoE experienced by users. For instance, a signaling message from the client shutting down video transport would imply that video packets are dropped in overlay network nodes. If the transport of this message were prioritized over others, resources would be freed and made available for the transport of other videos, with the potential QoS improvement. Furthermore, this effect also occurs with packets from the same component. For example, video packets containing I-frames impact QoE more than packets containing B-frames. For that reason, this component must support flexible transport priorities assigned by the component sending the packet. Note that the success of this mechanism depends strongly on the policies used in other components to assign transport priority.

In the big picture, transport priority means that more resources are allocated for the handling of the most important packets. In specific, the current design of Transport and Multicast Routing applies transport priority to decide the forwarding order of the packets. This component is likely to have several packets stored, waiting to be forwarded, either because they are sent at the same time or as a result of the store-carry-forward mechanism. Queues are the standard option to order packets for forwarding, but their design is flexible. First, one or several queues can be used. Using one queue that considers priority implies that new elements must be inserted in the right order, before

the packets of lower priority and after the ones with higher one. Reordering elements in a queue has a computational cost that adds delay each time a new packet is inserted. Delay is an important factor for us, so this effect is not desirable. Thus, several queues are used, one for each transport priority. This simplifies inserting new elements, but it increases memory usage, as references to the queues must be stored.

After packets are ordered in the queues, the next step is to select one packet to be forwarded. The aimed solution is to forward first the packets with the highest priority. There are several alternatives to implement it: from polling the queues in different ways to using interruptions when packets of high priority appear. The design of a system based on interruptions is interesting, but more complex to design and implement. Thus, we propose an iterative queue polling that may be refined in future versions. Queues are polled from highest to lowest priority. For each queue, this component attempts to forward its packets in FIFO order. It may happen that the destination of the packet is in a different partition, so it is not forwarded. Packets that cannot be forwarded are kept in the queue to be forwarded when the destination is connected.

5.1.6 Packet dispatching

MOMENTUM components receive packets through Transport and Multicast Routing. For that reason, this component implements a mechanism to dispatch packets to other components. There are two main alternatives to implement packet dispatching. On the one hand, this component could assume an active role. When a packet is received, this component could call the destination component and pass it the packet. The advantage of this approach is that the delay of dispatching the packet is low. However, Transport and Multicast Routing must have a reference to all the components in MOMENTUM. On the other hand, it could have a passive role. Received packets are stored and receiving components request them. This is the approach used in transport protocols of layered architectures, i.e. TCP or UDP. This component also follows this approach. Hence, Transport and Multicast Routing works as a new protocol layer over UDP. For the sake of flexibility, this component implements blocking and not blocking calls to dispatch packets. Thus, components have two ways of getting their packets. They can call a function that returns when a packet is received or they can call a function that returns a packet or nothing if there are not packets stored. Finally, it is necessary to add a mechanism so components request packets, equivalent to ports in UDP or TCP, because MOMENTUM uses a unique UDP port. We add a field, called message type, to the packet header for this purpose.

5.1.7 Fragmentation and aggregation

It is known that packets of certain size improve MANET reliability and resource utilization. For example, the results from [105] show that 250 or 750 bytes are better network frame sizes in their simulations. The main reasons behind this are the characteristics of lower layers and the competence for the shared medium. Therefore, it may be interesting to shape the size of the packets sent by MOMENTUM to get the most of the underlying network. The solution is to enable packet fragmentation and aggregation. The former implies splitting a single MOMENTUM packet in several UDP datagrams. The latter means that several MOMENTUM packets are grouped in the same UDP datagram. We believe this mechanism is interesting in real deployment, although it is yet not implemented in any of our prototypes.

Fragmentation can be supported with an additional field to the Transport and Multicast header. This field must indicate if the current UDP datagram contains a fragmented packet and if it is the first, the last or an intermediate fragment. Then, each fragment is sent in an individual UDP datagram repeating the header fields in each of them but with the right fragmentation field. In addition, it is possible to support aggregation by adding the length of the payload of the packet. The receiver uses this length to distinguish between consecutive MOMENTUM packets that arrive in the same UDP datagram.

5.1.8 Multicast routing

Multicast routing is a mechanism to increase transport efficiency when a packet must be delivered to several nodes, e.g. a signaling packet that is sent to all nodes participating in video transport. Components pass packets to Transport and Multicast Routing indicating one or several nodes that must receive it. If the packet is sent to just one node, this mechanism is not necessary. However, if there are several destination nodes, multicast routing is automatically triggered. This mechanism ensures that the packet is delivered to all the nodes in an efficient way. The simplest option is to send one copy of the packet to each node, but this may not be the most efficient alternative in terms of network utilization. Another simple option is to use one copy of the packet, which can be subsequently forwarded through all the nodes in the form of chain. However, this may imply that a packet traverses the same network links many times, which is also inefficient. A better solution is to use multicast routing trees. Routing trees organize all the destinations of a packet, with the source of the packet as root. They improve network resource utilization when some destinations are in the network route to others. For example, if a node N1 is in the way to a node N2, the source sends a packet to N1, which forwards it to N2. Moreover, route trees have the advantage that acknowledgements and retransmissions are leveraged from a closer node. If the message is lost from N1 to N2 and it has to be retransmitted, this can be done by N1. Routing trees can provide significant resource savings in the overlay network, e.g. when several relay or ferry nodes are used. Furthermore, future work can apply this mechanism to video transport from one source to several video destinations.

The main challenge of using route trees is to calculate them. The straightforward solution is to build them in the source node. The Topology Discovery component provides information about the network topology that a node can analyze to build the trees. However, MANET mobility implies that the knowledge about network topology is less reliable the farther a node is. In other words, nodes closer to a change in topology are the first to detect it. Therefore, the source may be efficient calculating the first nodes in the tree branches, but less efficient as the depth of it grows and nodes are far away. For that reason, we improve this approach by optimizing multicast route trees in each hop. The source establishes who are the first nodes of the tree branches and passes each of them a list of nodes as its successors. Then, each node repeats the process with its successor, i.e. selecting the first node in the branch and passing the successors for it. This recursive process is illustrated in Figure 5-13: (a) indicates how the source plans the routing tree originally, (b) represents the multicast routing tree once it is fully planned.

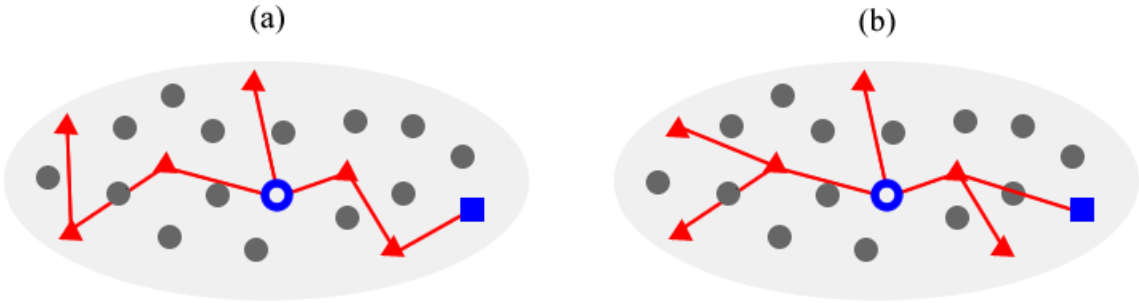

Figure 5-1 Multicast routing trees are planned recursively

Three elements are needed to support this solution: a protocol to notify routes, an algorithm to organize successors and a way to associate packets with routing trees. The latter is the easier feature to design. A node may be part of several multicast routing trees. Thus, route trees need to be uniquely identified in the overlay network in order to establish clear forwarding rules. In addition, a node can be root of several trees. For that reason, we use the same solution used to assign unique sequence numbers to packets. Each route tree is identified with a number and the IP address of the root of the tree. When a node receives a packet, it needs to know if it must be forwarded through an existent routing tree. Furthermore, we expect that several packets use the same routing tree. For that reason, we add a field to the Transport and Multicast Routing header indicating the routing tree id. Since the source of the tree is the same as the source of the packet, which is already in the header, no more fields are need. All the packets send to the same nodes use the same multicast routing tree.

Nodes belonging a multicast routing tree must be notified. Notifications start in the root and being forwarded following the tree. They must contain the route id and a list of successors. We call these packets Route Announces. The notification process is illustrated in Figure 5-2. The source node S sends Route Announces to its children that do the same with theirs. In this case, the tree has only two linear branches. Thus, the nodes in the tree have not optimized it. However, MANETs are dynamic, so multicast route trees may become inefficient soon after planning them. Sometimes it may be necessary to reorganize the full tree or a part of it. Figure 5-3 illustrates a situation in which nodes 2 and 3 move and 1 reorganizes them using Route Announces.

Since route trees may become inefficient after some time, there is the challenge of when to reorganize, because it produces overhead. The ideal solution should evaluate changes in the network topology and analyze whether reorganization is worth its cost. Studying this specific problem is interesting, but falls out of the scope of this thesis. Thus, we have opted for a provisional solution: to reorganize routes periodically, e.g. every 10 seconds the tree is planned again by the source. Evaluating our system with this simple approach allows us to understand its advantages and disadvantages and improve them in future designs. Reorganizing a multicast routing tree makes it more efficient. Nonetheless, it creates a problem, because there is a possibility of delivering duplicated packets to some nodes. If a node moves from a branch to another, its new parent does not know the packets it has received. A simple solution to this problem is to use a packet that we call Reception Report. This packet contains the sequence numbers of all nodes received through this routing tree; so no more duplicated packets are received. This method is useful to avoid wasting network resources on packets already delivered. Nevertheless, better solutions to the whole multicast routing tree reorganization issue need to be studied in future work.

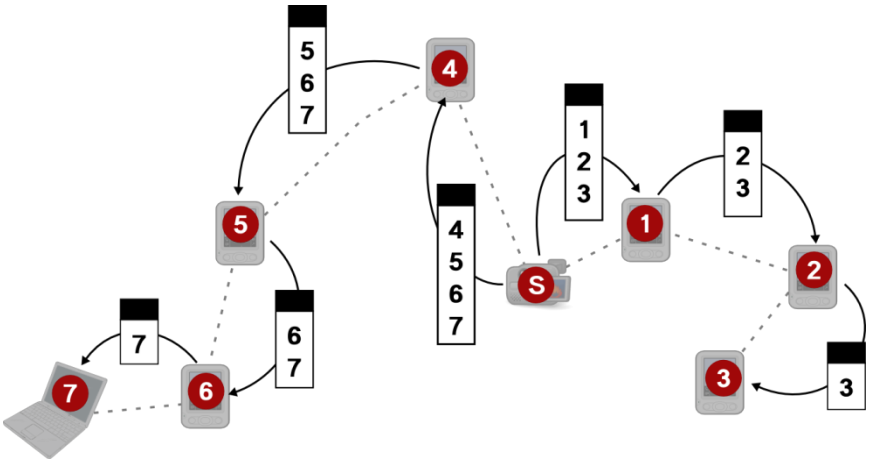

Figure 5-2 Route announce process

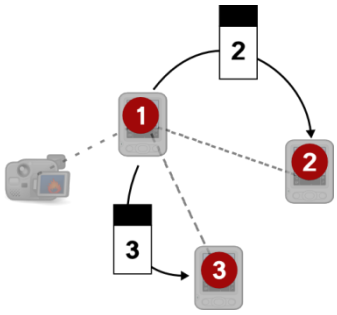

Figure 5-3 Route Announce of an optimization

Finally, we need an algorithm to organize the nodes as a tree. Our idea uses the intermediate nodes obtained from Topology Discovery. When Transport and Multicast Routing is ordered to send a packet towards several destinations and there is not an existing routing tree, the node calculates a new one. First, the root executes the algorithm with all the destinations of that packet. Then, each node that receives a Route Announce executes it again with the list of successors received as if they were the destinations. Given the set of destination nodes, the farthest one is selected. Then, Topology Discovery is asked for the intermediate nodes between the source and the farthest node. The intermediate nodes are examined in order to find any of the given destinations. All the destinations found and the farthest node, ordered by distance, form the first branch of the route. The process is repeated until all destinations have been added to a branch. If there are nodes in the list of destinations that are not in the same network partition, these are added as children of every node. Thus, when a node finds another that was listed as disconnected, it will receive the stored packets. The pseudo code of the algorithm is:

```
s = source node (node executing this algorithm)
D = Destination nodes list
R = Route tree
B = Route branch

while(D is not empty)
{
      f = Farthest node of D in hops
```

```
        I = Intermediate nodes between s and f
        P = new empty path

        for(each i in I)
        {

                for(each d in D)
                {
                        if(d==i)
                        {
                                add d to B
                                remove d from D
                        }
                }
        }
        order nodes in B by hop distance
        add B to R
}
```

5.1.9 Header format

As a result of the design of these mechanisms, all packets exchange by Transport and Multicast Routing instances include the following header fields (see Figure 5-4). We have also made an initial estimation of their length for our current prototype.

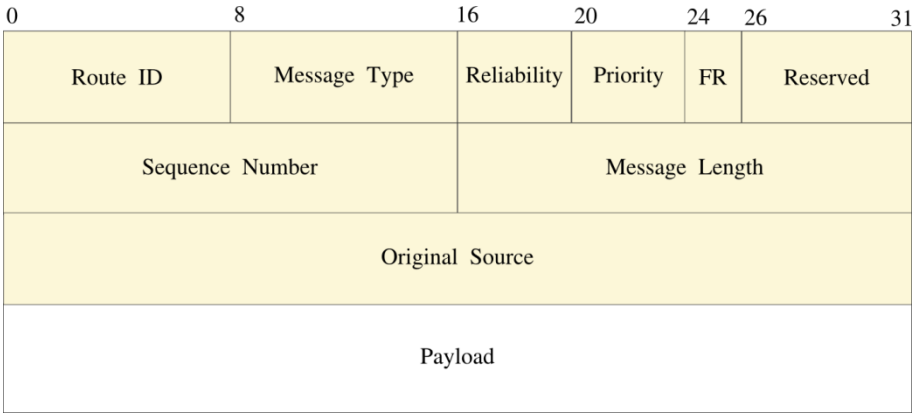

Figure 5-4 Header format example

- **Route ID** (8 bits) indicates the unique identifier of the route the packet must follow. There are special identifiers: 0 is for Broadcast and 1 is for unicast.
- **Message Type** (8 bits) specifies the type of the message. Each MOMENTUM component can use one or more message type identifiers.
- **Reliability** (4 bits) denotes the reliability actions that nodes must apply to this packet.
- **Priority** (4 bits) expresses the priority of this packet.
- **FR** (2 bits) are the fragmentation bits.
    - 00 indicate the packet is not fragmented.
    - 01 indicate this packet contains the last chunk of a fragmented packet.

- 10 indicate this packet contains the first chunk of a fragmented packet.
- 11 indicate this packet contains an intermediate fragment.

- **Sequence number** (16 bits) is the identifier that the original source node gave to the packet.
- **Message length** (16 bits) indicates the length of the payload in bytes.
- **Original source** (32 bits) is the IP address of the node that generated the packet.

5.2 MOMENTUM prototype

This section thoroughly describes the functionality of the components in the first MOMENTUM prototype. We analyze fine grain design decisions and implementation challenges. This prototype will be evaluated in its ability to adapt to different mobility and density networks with random mobility. Hence, some design decisions are targeted to these scenarios.

5.2.1 Context-aware Overlay Routing

This component selects ferry and relay nodes for video transport. MOMENTUM design establishes that every node that has video stored must select relays or ferries towards the video destination. This is the best choice for a real deployment, but it is not scalable with our current prototype due to the simplicity of some mechanisms, e.g. signaling and video forwarding. If each node could select its own ferries and relays, the video may get fragmented among all MANET nodes, with the consequent resource consumption. For that reason, we limit relay and ferry selection to the video source and the video destination nodes. Thus, the source selects ferries and relays, they receive video packets and, then, deliver them to the destination. In addition, control packets travel both ways using the same overlay routing mechanism.

The selection is triggered when there is a context change between MANET and DTN. The Topology Discovery component is used to monitor the route between source and destination. When there is a route, MOMENTUM is in MANET mode and this component selects relay nodes. Otherwise, MOMENTUM is in DTN mode and this component selects ferry nodes. The selection of nodes is independent every time it is done. Therefore, different ferries or relays can come out from every selection. When there is video stored, Distributed Video Transport Control requests the current selected nodes to forward them the video.

A good choice of ferries and relays depends on the good understanding of node mobility. Our current prototype will be evaluated in scenarios with random mobility. Therefore, little assumptions can be made about the nodes future behavior. Initially, all nodes have the same chances of being good or bad relays or ferries. Due to random mobility there are not solid criteria to base relay and ferry selection. For that reason, this version uses a partially random selection of relay and ferry nodes. Relay nodes are selected among the nodes around the shortest path between video source and destination. This component uses the intermediate nodes calculated by Topology Discovery and a select random node every two hops. For instance, if the destination is two hops away from the source, it will select two relays: one, two hops away and

another, four hops away. The selection of ferry nodes is triggered when source and destination are disconnected. As we will describe in the next chapter, patterns in mobility help selecting ferries. Since mobility is random, this prototype select ferry nodes randomly between the one-hop and two-hop neighbors, which are obtained from Topology Discovery. We chose to select half of these neighbors as ferries, to increase the probability of successful video delivery.

5.2.2 Application Gateway

The video client and server applications determine the Application Gateway implementation. For this prototype, we use openRTSP and live555[4] as client and server. These applications follow the RTSP, RTP and RTCP standards. They are open source and have good support from their developers, which eases troubleshooting. Finally, they are easy to use through the command line, which benefits the automation of experiments.

This component intercepts traffic from these applications. The goal was to make the overlay network transparent to the applications. Hence, we have implemented this functionality using a firewall (IPTABLES) and intercepting TCP/UDP ports used by the applications in the local interface. The Application Gateway opens additional TCP and UDP ports to receive the traffic there. This is repeated in both client and server. Then, this component simulates the communication with the other end to keep the session alive. The first necessary measure is to keep the TCP connection for RTSP alive, which is done automatically. The server application (live555) also requires periodic RTCP Receiver Reports. These are generated and sent by the Application Gateway in the server side. The Application Gateway can support several RTSP/RTP applications simultaneously.

5.2.3 Distributed Video Transport Control

RTSP packets captured by the Application Gateway are extended with the additional fields justified in the previous chapter: client and server IP addresses, session status and a list of ferries and relays used. Then, this component passes these packets to Transport and Multicast Routing indicating as destinations all ferries and relays, as well as the client or server, depending on which generated the packet. Note that these packets are sent not only to the nodes currently selected as ferries and relays, but also to the nodes previously selected for these tasks. This component must also select the transport mechanisms that must be used. As we have mentioned, signaling can be important to manage resources and also to achieve good QoE. Hence, we consider that these packets must have a high priority, i.e. 14 in our current design. For the same reasons, we want to ensure that packets are delivered. Therefore, ACKs to the parent in the multicast routing tree are required. By activating these mechanisms, this component leaves the task of distributing signaling messages to Transport and Multicast Routing.

This component also controls video forwarding, i.e. acting as an intermediary between Context-aware Overlay Routing and Quality-aware Video Forwarding. According to the status of the session, this component requests the current relays or ferries from Context-aware Overlay Routing and triggers video forwarding towards them. In this prototype,

[4] http://www.live555.com

we opt for a resource consuming, but reliable, video distribution technique. Video is forwarded to all the ferries, relays and to the destination node. Again, Transport and Multicast Routing is responsible for the transport of video packets after this. As a result, several copies of the video packets exist in the overlay network, with the consequent resource consumption. However, the goal is to increase the chances of delivering video to the client application over sparsely connected scenarios with random mobility.

The implementation of this component and of the Application Gateway does not interfere in the signaling between applications. They behave according to the RTSP messages they exchange. This means that video streaming does not start or stop in the server until it receives the appropriate RTSP message from the client, i.e., a PLAY message and all the previous configuration messages. In a DTN, this initialization process may produce significant delay.

5.2.4 Quality-aware Video Forwarding

The implementation of this component deals with video packet forwarding. Packets from the Application Gateway are extended with identifiers of the stream and the protocol (RTP and RTCP) they carry, as discussed in the previous chapter. At this point, there are two main issues to be decided: the specific video forwarding mechanisms implemented in this component and the mechanisms from Transport and Multicast Routing that are applied to video packets. In this prototype, we focus on achieving adaptability to different network conditions. We will not look into QoE or QoS, which are more sensible to packet forwarding policies, as we will see in Chapter 7 For that reason, we apply a simple forwarding policy: FIFO (First In First Out). Video packets are forwarded as received from the Application Gateway in the source and in the order received from the source in relays and ferries. Nonetheless, the video client may still receive packets disordered under some circumstances. For example, a ferry receives packets 1 to 10 from the source. Another receives packets 11 to 20. If the second ferry connects with the destination before the first does it, the destination will receive packets 10 to 20 before packets 1 to 10. We are aware that this affects QoE, but it is irrelevant for our current goals.

Video packets are passed to Transport and Multicast Routing to be sent to the nodes indicated by Distributed Video Transport Control. Their priority of the packets must be lower than for signaling packets, but still higher than for other packets. There is also the possibility to apply different priorities to different types of video packets, e.g. packets containing I-frames, P-frames and so on. However, the implementation of the Application Gateway in this prototype is not able to identify video packet payload. Thus, the priority assigned to all video packets in our experiments is the same, i.e. 13. Finally, the acknowledgements policy must also be decided. The loss of video packets influences QoE, but not the behavior of the overlay network, as it may happen with a signaling packet. In addition, the vast majority of packets managed by the overlay network are video packets. Hence, using ACKs for all of them can significantly increase resource consumption. For that reason, ACKs for video packets are not sent in this version.

5.2.5 Transport and Multicast Routing

All mechanisms, with the exception of Fragmentation and Aggregation, have been implemented as discussed in the previous section. Thus, the remainder issues are the

setting of specific parameters. These parameters affect the performance of the overlay network. Moreover, they depend on the scenario where MOMENTUM is deployed. As we have discussed, the ideal situation may be to use adaptable mechanisms in the future. For the current prototype, we carried out preliminary experiments to adjust them. One parameter is in the flow control system. We have limited it to 11 Mbps, as it is a typical value in 802.11 networks and send buffer overflow is not produced. Furthermore, it provides enough capacity to stream a video if client and server are connected. The value in a real deployment would depend on many factors. On the one hand, a higher transmission rate may be possible in theory. On the other hand, there are studies [43] that show lower forwarding rates in testbeds with mobile devices. Another parameter is the timer to update multicast route trees. We have set it 10 seconds. The update of the routing tree is only carried out if there have been changes in the network topology. Finally, the acknowledgement and retransmission scheme uses two parameters. First, a timer to trigger retransmissions is set to two seconds. If the ACK for a packet is not received in two seconds after it is sent, this component will forward it again. Second, ACKs need a transport priority. An ACK avoids a retransmission; therefore, it is a resource saving strategy to give them a high priority. In this version, ACKs have the highest priority, i.e., 15.

5.2.6 Topology Discovery

Topology Discovery is the interface with the MANET routing protocol. We have selected the OLSR implementation: olsrd. This implementation eases the first implementation challenge: extracting information from OLSR. It supports being extended with plugins, which can access a copy of its internal data structures. Thus, we have implemented a plugin that accesses the routing data stored by olsrd. Another problem is the communication between this plugin and Topology Discovery. Topology Discovery runs inside MOMENTUM and the plugin inside olsrd, which are two different processes. There are several alternatives to share data between processes, but the implementation of some of them may be dependent on the operating system. We need a reliable channel that is available in most operating systems. Hence, we use a TCP connection. The plugin opens a TCP server socket and Topology Discovery connects to it through the localhost interface. Every time the network topology changes, the plugin sends the routing data structures. The format used to exchange data resembles XML, but it is easier to parse without specific libraries. Therefore, it is easy to implement, but also easy to read by humans for debugging purposes. This is an example:

```
data

routing_table
...
route 10.0.0.1    10.0.0.3 3
...
/routing_table

neigbours
...
neighbor    10.0.0.3
...
/neighbors

neighbors_2hop
...
```

```
neighbor_2hop     10.0.0.2
/neighbors_2hop

tcset
...
set 10.0.0.2
destination 10.0.0.3
destination 10.0.0.1
/set
...
/tcset
/data
```

Topology Discovery processes these data and stores it. Most required functionalities, e.g. neighbors of a node, have a straightforward implementation. For example, a node is in the same partition if there is a route in the routing table. However, some components also require the calculation of intermediate nodes. They are used to select relays and also to plan multicast routing trees. Intermediate nodes are not only the nodes in the shortest path between two, but also those surrounding them. The value of the former nodes is clear, but also the latter ones may be useful in some situations. For example, there may be several options to choose a relay that is almost as well placed as the ones in the shortest path and that may have more available resources. We have designed and implemented an algorithm that analyzes links between nodes in the partitions, which are received as *tcset*, to find out which nodes are topologically in between two others; we call it the Intermediate Nodes algorithm

This algorithm calculates the intermediate nodes between the node executing it (the Source, S) and another node in the same partition (the Target, T). The algorithm uses information of 1-hop, 2-hop and Remote Neighbors to find the solution. This solution is not the exact network route, but a set of nodes that form many possible paths between S and T. Figure 5-5 illustrates an example of intermediate nodes result.

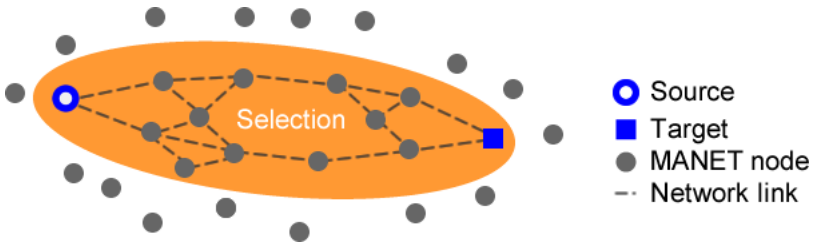

Figure 5-5 Example of the intermediate nodes algorithm result

The Intermediate Nodes algorithm is inspired in how OLSR builds network routes. When we run this algorithm in S, we know the neighbors of T and their neighbors too. We also know the number of hops from S to every node in the network partition from the routing table. This way we go from T to S adding nodes to a set. We keep as intermediate nodes the neighbors closer to S and discard the others. We iterate over neighbors of neighbors filtering with the number of hops to S, until we find S itself as a neighbor. This is the pseudo code used in the intermediate nodes algorithm:

```
S: Source node
T: Target node
IN[d]: List of nodes at d hops from S

h = number of hops from S to T
IN[h] = T + (neighbors of T at h hops from S)
```

```
while (h > 0)
{
      for(each n in IN[h])
      {
      R = remote neighbors of node n
            for(each r in R)
            {
                  if (hops from S to r == h-1)
                  {
                        add r to IN[h-1]
                  }
            }

   h--
}
```

5.3 Evaluation

5.3.1 Goals

The aim of this evaluation is to demonstrate that MOMENTUM fulfills the mobility and connectivity requirements. In specific, MOMENTUM is able to transport video over DTN and MANET scenarios, as well as in scenarios that change between both paradigms. For that purpose, we evaluate the prototype transporting a video stream over random mobility scenarios. We compare its performance with just using a client and a server application, without the overlay network. In order to identify the regions in the mobility/density space in which the different approaches are preferable, we aim to cover in these experiments the entire (realistic) mobility/density space. We force the scenarios to vary in density by changing the number of nodes and in mobility by modifying the average speed. We expect that the traditional client-server solution will work more efficiently than the overlay network in fully connected and more static networks, but with increasing amount of route changes and network partitionings and mergings the advantages of the overlay approach should overcome the additional overhead introduced by it. Furthermore, it is obvious that there are certain scenarios with very low density and mobility in which none of these solutions are feasible.

5.3.2 Emulation environment

The selection of an adequate evaluation platform must follow a few directives. First, it must be compatible with the MOMENTUM prototype, which is implemented in Java. Second, it must support using olsrd. Third, it must be able to simulate connection and disconnection of nodes according to the mobility of the scenarios used. Finally, it should have low resource consumption and be easy to setup. NEMAN [60] is a network emulator that fulfills these requirements. It reproduces the connectivity of a MANET in real time using virtual network devices of a computer (TAP interfaces) associated to nodes. Thus, we bind MOMENTUM, olsrd and video applications with these virtual network devices. Moreover, NEMAN accepts standard ns2[5] scenario files as input,

[5]ns-2 is one of the most renowned simulator/emulator. It can be found at http://www.isi.edu/nsnam/ns/. ns-3 was released after this experiments where done.

which is convenient due to the large amount of scenario generators for this platform. Experiments can run on single computer if it has enough resources. There are only a few differences between this emulated environment and a real scenario, but they are not relevant at this point of our research:

- The sockets opened by the application must be bound to a TAP device associated with a node. Since Java lacks of native support for this, we have developed a C library to do it.
- Resource availability and consumption is different, because we run all processes on the same machine. All of them share the machine resources, which limits the number of nodes that can be emulated at the same time. We must keep resource consumption of each experiment below the maximum capacity of the machine where they are run. Otherwise results could be affected.
- Network is emulated in a connected/disconnected basis. Thus, there is no bandwidth limitation like in a real network. The limitations come from the hardware and operating system running the experiment. Although NEMAN can reproduce some physical layer features, we have not activated them, for simplicity.

5.3.3 Experiment description

Our knowledge about mobility in ER operations is not enough at this point of our research to build realistic scenarios. We use random mobility, because we can obtain the different situations to cover a large mobility/density space. In specific, we vary node speed and number of nodes to cover a broad and realistic set of situations. Node speed affects the stability of network routes and also the probability of connecting different partitions in DTN scenarios. Maintaining the size of the area in which nodes move, we vary the number of nodes to obtain different degrees of density. The more nodes in the area, the more likely they are connect, the higher the density and vice versa.

We use Random Waypoint mobility to generate the scenarios. All of them last 1000 seconds. We use two area sizes, labeled as Normal and Big. Normal ones have a size of 1000x600 meters and contain nodes ranging from 4 to 20 nodes. Big ones cover a larger ground of 3000x1000 meters and have 30, 40 or 50 nodes. For all of them, speed goes from 1 to 20 m/s. We have selected rectangular areas, because they are more prone to route breaks. In addition, we have chosen area sizes that are closer to sparse MANETs than to dense ones. Specifically, if we compare with some experiments performed by related work such as [46], we have chosen a similar area with a lower number of nodes. Nodes communication range is 250 meters. Although it is a long range for popular wireless technologies such as 802.11, this value is often used in other studies. Furthermore, results for other range values could be extrapolated if we divide range, area and speed of the nodes. As far as possible, the selection of parameters attempts to replicate real application scenarios. In this case, it is difficult to compare with other proposals, due to low repeatability in previous MANET research, as is stated by [106], and to the fact that we aim sparse MANETs. The configuration of OLSR is the default provided in its RFC [16].

As workload we use a 1200 seconds long action video encoded in MPEG-2. The video is composed of a single stream, with the same priority for all the packets in it. It is longer than the experiment in order to reproduce a live video situation. The application

used as video server, live555, is streaming video during the whole experiment. To receive the video stream, we use OpenRTSP as client. In one set of experiments, these applications run directly over the MANET. In another set, they run with MOMENTUM in the MANET nodes. There is only one video source and one destination and in each scenario they are always located in the same nodes, i.e., Node 1 and Node 2 as generated by the Random Waypoint mobility model. Therefore, we can compare client-server and MOMENTUM under the same conditions: same mobility, same source and same destination. Every scenario is repeated five times. Finally, random packet losses and collisions due to physical layer are not enabled in the emulator. However, packet losses can occur due to network disruptions. For example, if a packet is sent to an unreachable destination. Cross-traffic has not been introduced. Table 5-1 summarizes the configuration of the experiments.

	Normal scenarios	**Big scenarios**
Area	1000 x 600 square meters	3000 x 1000 square meters
Nodes	4 to 20	30, 40 and 50
Mobility model	Random Waypoint	
Range	250 meters	
Emulation Time	1000 seconds	
Video	MPEG-2 video stream	
Node mean speed	1 to 20 m/s	
Runs	5 runs per scenario	

Table 5-1 Random Waypoint scenarios properties

5.3.4 Metrics

Delivered Video Bytes measures the number of video bytes received by the client application. It is calculated by adding the payload size of the RTP packets that are delivered to the client. In MOMENTUM these are the packets forwarded by the Application Gateway.

Delivered Video Bytes Ratio measures the number of video bytes received by the video client relatively to the video bytes sent by the video server application. The calculation is similar to the previous metric, but also measuring video going out from the server.

Video Packet Delay is the time elapsed between the instant when the server sends a RTP packet and the moment when the client receives it. It is calculated by monitoring

RTP traffic in the server and the client. Therefore, this metric includes the delay introduced by the overlay network and the MANET.

Video Packet Arrival Time indicates the time when a RTP packet is received by the client application. As Video Packet Delay, it is calculated by monitoring RTP traffic in the client. From this metric we can also calculate the number of packets received by the client at a given time, which is useful to describe video delivery.

5.3.5 Results and discussion

In this section, we discuss the results obtained in our experiments. First, we provide an overview of the video delivered in all the scenarios evaluated. Based on this first analysis, we examine with more detail some representative cases. Finally, we discuss the role of ferries and relays.

Overview

In the big picture, the ideal outcome is that MOMENTUM delivers at least as much video as client-server in every situation. This should be more obvious when the network has disconnections, because client-server should fail. In addition, MOMENTUM should be able to deliver as much video as the client-server alone approach in scenarios with good stable connectivity, i.e. high density and low mobility (low speed). In these situations, we expect MOMENTUM to introduce overhead in form of extra packets and delay. Nevertheless, overhead resource consumption should be small compared with the resource consumption of video transport. On the contrary, delay can be important in case video is played as received. Next, we analyze whether MOMENTUM meets our expectations, as well as the reasons why they are or are not met.

Figure 5-6 and Figure 5-7 compare Delivered Video Bytes of client-server and MOMENTUM in the normal and big scenarios respectively. Axis X indicates the number of nodes in the network and axis Y their average speed. Therefore, density increases to the right of the graph and mobility increases upwards. Every scenario is repeated 5 times for each alternative, client-server and MOMENTUM. Delivered Video Bytes in each run is measured and averaged, getting one value for each system in every scenario. Then, these two values are subtracted. Hence, we obtain a positive or negative number for each scenario. The circle surface is proportional to the absolute value of this number. Green filled circles represent the experiments where MOMENTUM delivered more video. White filled circles with black border represent the experiments where client-server delivered more video than MOMENTUM.

In Figure 5-6, MOMENTUM outperforms client-server in the majority of scenarios. There are a few cases where client-server is very superior, such as 13 nodes and 15 m/s. They are a consequence of either mobility favorable to client-server or errors in one or several experiment runs with MOMENTUM. Apart from these outliers, we can draw some conclusions from examining the big picture. Low-mobility and high-density scenarios, in the bottom-right corner, represent networks with almost static topologies. In these cases, client and server nodes may be in the same partition or not. If they are connected, video will be transmitted by any of our alternatives. If they are in different partitions, MOMENTUM has more chances, but mobility is so slow that it is difficult to find ferry nodes to carry video. For that reason, scenarios in the bottom-right corner have similar video delivery for both approaches. On the contrary, scenarios with low-

density and low-mobility are very disconnected. For example, in a scenario with 5 nodes moving at 1 m/s in average, it is likely that video source and destination will not meet in the 1000 seconds of experiment. Because we use random mobility and select source and destination nodes randomly, achieving a good result is also random. For example, sometimes source and destination nodes meet at the beginning of the experiment and client-server is able to deliver some video. Because MOMENTUM is more resilient to disconnections, it has more success when mobility increases but density is still low. With a few exceptions, MOMENTUM dominates the rest of the density versus mobility domain. As we have mentioned there are several random parameters in the experiments. This influences results, as this randomness is likely to benefit one of the alternatives. This is a weak point in our analysis that could have been solved generating several mobility scenarios for each nodes-speed pair of values. Nevertheless, we also think that this is compensated, because there are scenarios with similar parameters. For instance, although client-server is superior in the 13 nodes moving at 15 m/s experiments, MOMENTUM is clearly better in similar scenarios, such as 14 nodes and 15 m/s and 12 nodes and 15 m/s. The small variation in the parameter values compensates the lack of several scenarios with the same parameters.

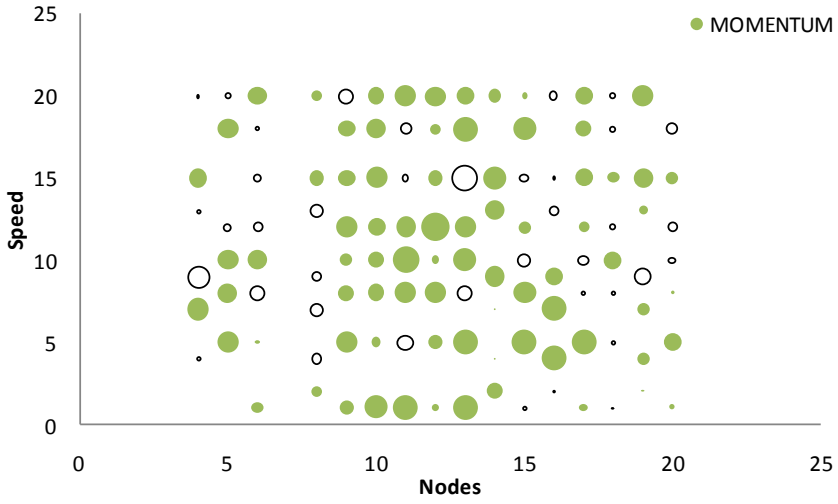

Figure 5-6 Video delivery comparison for normal scenarios

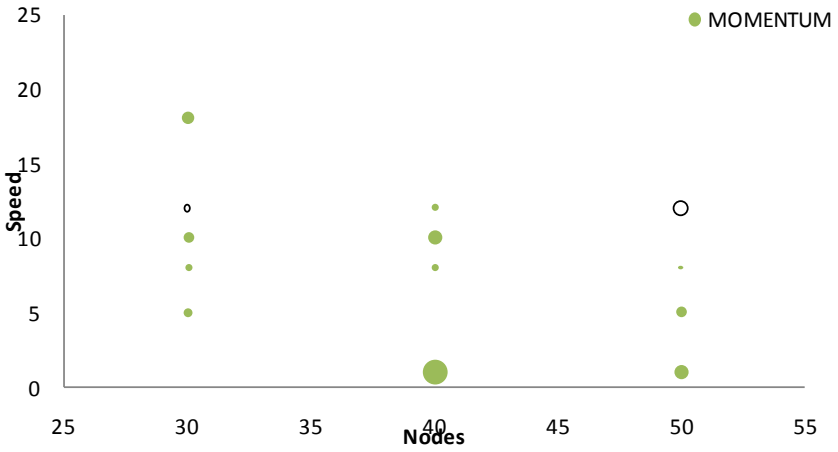

Figure 5-7 Video delivered comparison for big scenarios

We carried out a set of experiments using larger scenarios, which are represented in Figure 5-7. MOMENTUM delivers more video in most of them, even when some of them have stable topologies that are good for client-server. The main reason for the best performance of MOMENTUM is the existence of long network routes. The more hops an ad-hoc route has, the less reliable it is. MOMENTUM splits network routes by selecting relays that are in between source and destination. In scenarios such as 40 nodes and 1 m/s, this seems to pay off very well, while in others it supposes a small difference. As it happened with the normal size scenarios, randomness plays is role in the results. Therefore, if client and server are most of the time connected by a short route, client-server will perform at its best without the overhead introduced by MOMENTUM, like in 50 nodes and 12 m/s.

Table 5-2 contains the Delivered Video Bytes, in Megabytes (MB), and the Delivered Video Bytes Ratio (%) by MOMENTUM and client-server for six relevant scenarios. We have selected representative scenario runs where significant events take place. How video is delivered is thoroughly studied in the following sections for three of them (Streaming, DTN, Bad Relay). The other three: Best, Error and Disconnected help us to explain remarkable events in the experiments. In the last row, we provide the sum of megabytes delivered by client-server and MOMENTUM in all the experiments. This allows us to compare the big picture of MOMENTUM and client-server. As we can see, the client receives roughly 58% more video megabytes using MOMENTUM.

Label	**Nodes**	**Speed**	**Client-Server (MB)**	**%**	**MOMENTUM (MB)**	**%**
Streaming	*18*	*1*	22,37	100	22,35	99,90
DTN	*4*	*7*	1,62	7,24	14,33	64,06
Wrong relay	*11*	*18*	18,30	81,79	15,50	69,28
Best	*12*	*12*	0,04	0,19	20,64	92,23
Error	*13*	*15*	15,85	70,83	0	0
Disconnected	*13*	*18*	0	0	16,43	73,45
…						
Total	*	*	1090,54	-	1723,77	-

Table 5-2 Video delivered in relevant scenarios and in total

The experiments labeled as Error, Disconnected and Best represent remarkable events that have been detected. On the one hand, Error represents a situation where MOMENTUM failed to deliver video. In this case, there was a failure in the communication with the olsrd plugin in the source node that left it blind. It was not aware of the topology, so the source node stored all the packets but forwarded none.

This type of error illustrates one of the potential problems of using MOMENTUM. The client-server approach is simple and depends on just the video applications and the routing protocol. However, MOMENTUM introduces more elements and involves more nodes, which increases the potential points of failure. This is something that should be taken into account. In a real deployment, there must be mechanisms to increase resilience by preventing, detecting and recovering from possible failures. For example, a mobile device can reboot if MOMENTUM or olsrd are failing. On the other hand, the client-server approach fails in Disconnected and Best scenarios. In the former, the problem is that source and destination node are never connected, so client and server applications cannot even start video transport. MOMENTUM leverages ferries to overcome this situation. In the latter, client and server start streaming video, but there is a long lasting route break. The server application is not aware of the situation and goes on sending packets without feedback from the client. The server does not receive RTCP Receiver Reports and closes the session unilaterally. Hence, a disruption that last long enough shutdowns the session and it will not be recovered even if the disruptions ends. These were two expected problems that MOMENTUM solves successfully.

MANET: video streaming

In these scenarios MOMENTUM receives 0.1% less video than client-server. This is a connected scenario where nodes move slowly; hence, there are few disruptions. Server and client establish the RTSP session successfully. Then, the server application is able to stream video packets during the whole experiment. Because packet losses and collisions were not activated in NEMAN and the route was stable, all video packets generated were delivered. The circumstances are similar to those found in the Internet. Figure 5-8 shows received packets according to their Video Packet Arrival Time. The number of packets is in the Y-axis and the time when the packet is received, and therefore can be reproduced by the client, is in the X-axis. Cyan squares are for client-server and red crosses for MOMENTUM.

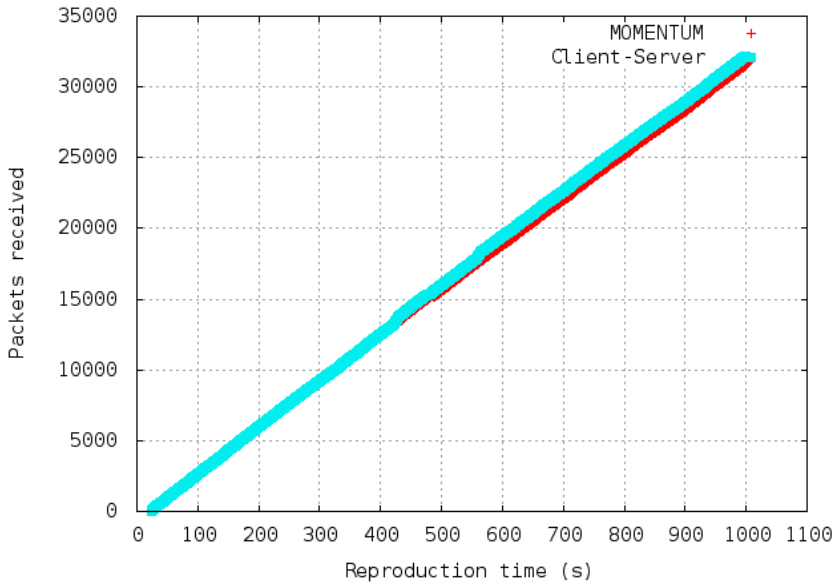

Figure 5-8 Packet Arrival Time

Packet delivery is the same during the first 400 seconds. After, MOMENTUM goes a little behind client-server that delivers packets as the server generates them. The

overhead of the overlay network is the reason for the slightly worse performance of MOMENTUM. Figure 5-9 illustrates Video Packet Delay in both alternatives. MOMENTUM delivers packets with more delay than client-server. Except for one packet, this delay is below 0.25 seconds. Therefore, we can conclude that MOMENTUM performance in a connected scenario is close to the performance of client-server and the overhead introduced is not significant. However, we must also be aware of the effects of these overhead and aim to minimize it.

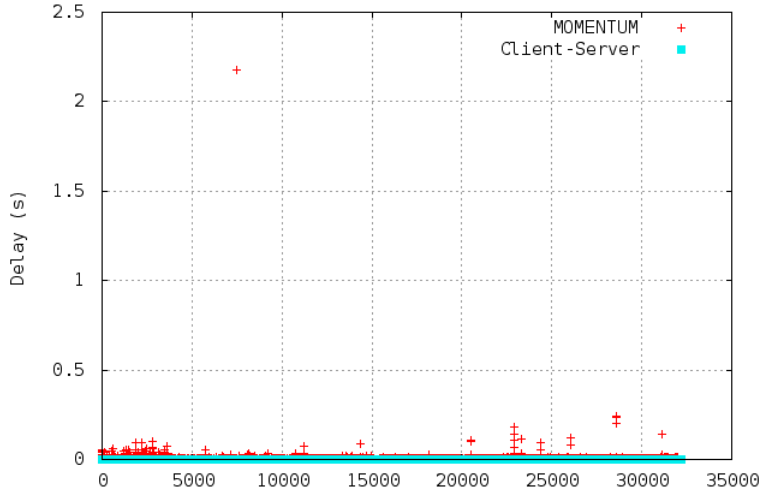

Figure 5-9 Packet Delay

DTN: frequent disruptions

This scenario exemplifies situations where MOMENTUM performance is clearly superior. Client-server delivers video packets to the client, but there are several network disruptions that it cannot overcome. On the contrary, the disconnection resilience mechanisms of MOMENTUM leverages that more packets are delivered despite route breaks. Figure 5-10 shows packets delivered during the experiment. Both sessions are established after 50 second of emulation, before that client-server were disconnected. Video streaming is possible for a while. Then, there is a long disconnection between server and client approximately in second 130. Client-server fails to recover the session after it. However, MOMENTUM is resilient to this disconnection and later ones. On the one hand, MOMENTUM keeps the video session open in the server and stores video packets. On the other hand, the client receives video packets from ferries and from the source node using store-carry-forward. Instead of streaming video, it is delivered using delay-tolerant mechanisms. Video Packet Delay, represented in Figure 5-11, reflects these disconnections. Maximum delay surpasses 35 seconds for some packets. Packets with low delay have been delivered directly by the source node as they are generated by the server application. Packets with high delays have been delivered using store-carry-forward either by ferry nodes or the source itself.

These two ways of delivering video are important because they affect the client application. Not only applications (and users) should be able to receive and reproduce streaming as it comes, but they should support receiving delayed packets. Most of the existent client application does not support this feature. The common behavior is to discard late packets, but they may contain useful information in video service for ER

operations. In addition, packets that arrive using delay-tolerant transport are delivered in a different way. Conventional video streaming is temporized, e.g. one packet is sent by the server every 200 milliseconds. However, video packets delivered with store-carry-forward can be sent as fast as the network capacity allows it. This can be observed in the burst of packets delivered after the second 500 in Figure 5-10. A client application suited for this environment should support both video delivery patterns.

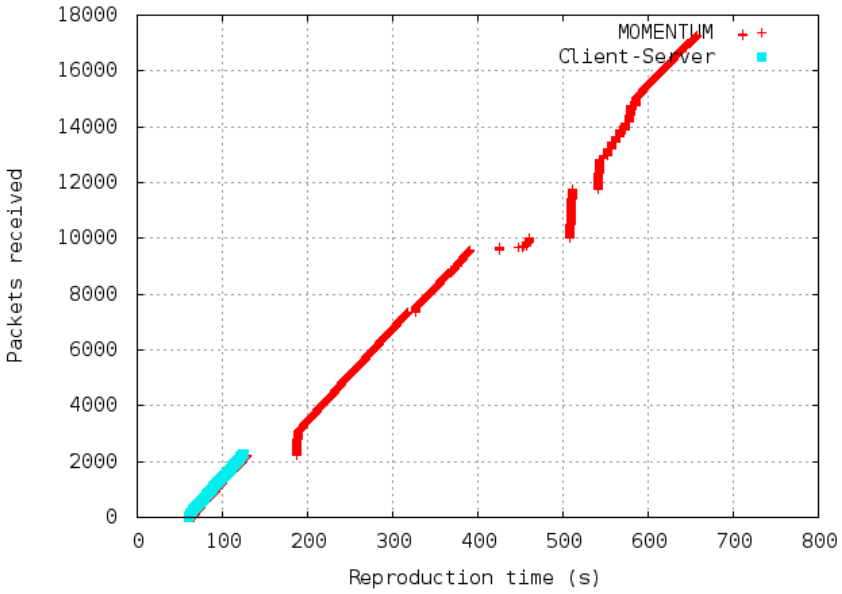

Figure 5-10 Packet delivery in a DTN scenario (4 nodes, 7 m/s)

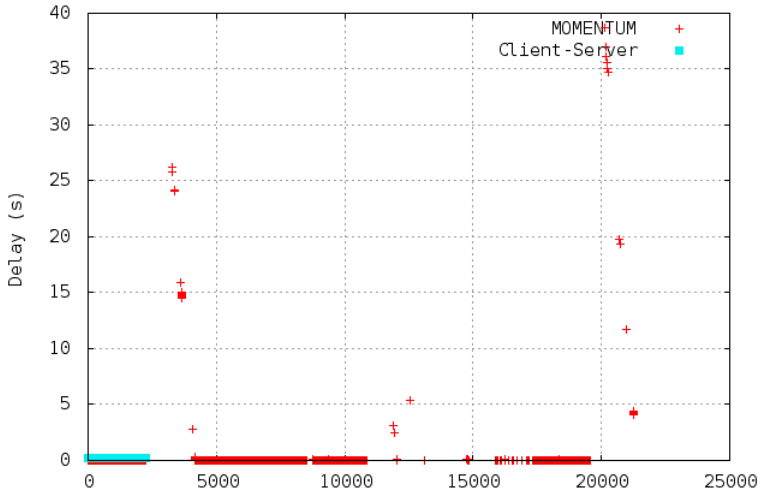

Figure 5-11 Packet delay in a DTN scenario (4 nodes, 7 m/s)

MANET: wrong relay selection

This experiment illustrates a situation where MOMENTUM selection of relays does not work as expected. The network route between source and destination node is stable during the session. Therefore, client-server can stream packets, as can be seen in Figure 5-12. However, MOMENTUM behaves differently. Its underperformance is also perceived in Figure 5-13. Some packets reach the client with a delay of several seconds, which should not happen if source and destination are connected.

Unexpectedly, MOMENTUM is affected by the high mobility of nodes. Source and destination nodes are in the same partition during the experiment, but the route that connects them changes frequently. Client-server is affected by these temporal breaks, so it is unable to deliver packets until OLSR recalculates a route. MOMENTUM also depends on OLSR, but it selects relays to split network routes. These relay nodes are selected in the route between source and destination. This is a policy that increases reliability in other scenarios, but not appropriate with high mobility. Relays move fast, as source and destination nodes do. Thus, a selected relay is likely to become useless quickly, because it abandons the zone between source and destination. In this scenario, MOMENTUM is affected by this situation. It trusts in a relay node that works at the beginning, but after a while is not a good one. The chosen relay hinders packet delivery and introduces extra delay. When the source node discards it, after 800 seconds, it is too late to recover the situation.

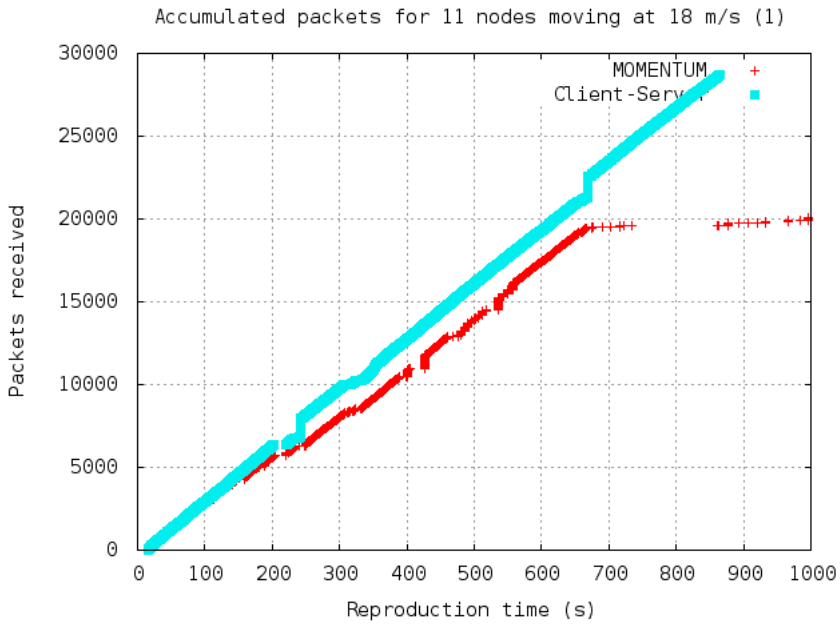

Figure 5-12 Packet delivery with wrong relay selection (11 nodes, 18 m/s)

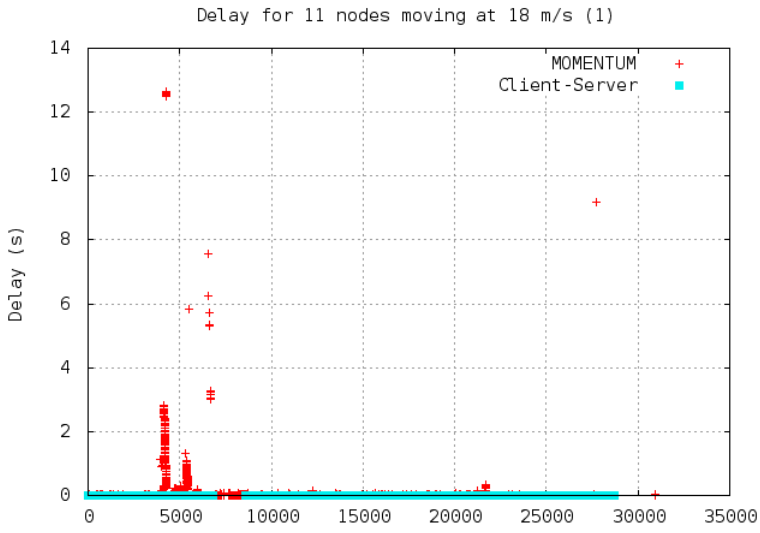

Figure 5-13 Packet delay with wrong relay selection (11 nodes, 18 m/s)

This scenario exemplifies how a wrong relay selection can hinder video delivery. In this case, our first error was to use the same policies independently of the characteristics of the mobility. Our second error was the lack of real time mechanisms to measure overlay route performance and detect possible failures. Good relay node selection should adapt to mobility and real time performance.

Video delivered by ferries and relays

In this section, we analyze the contribution of ferries and relays to video delivery. MOMENTUM delivers more video than just client and server applications. Store-carry-forward and the isolation of applications from the MANET are key parts for this success. However, the influence of relays and ferries is still low. We have accounted that only 2% of the video packets are delivered by them and not by the source, as seen in Figure 5-14. However, we still believe this approach has great potential if the right selection mechanisms are developed. In disconnected networks, ferries are the only possibility to deliver packets. The algorithm proposed is simple enough to give acceptable results in some scenarios, but we need better selection policies. Moreover, our results reveal that random selection over random mobility often chooses useless nodes. One key to elaborate better policies is to obtain a better understanding of node mobility.

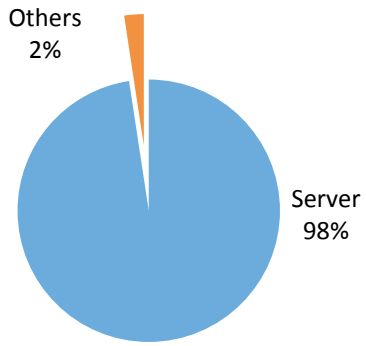

Figure 5-14 Packets delivered to client by ferries and relays in all the scenarios (%)

5.4 Conclusions

We extract three important conclusions from the evaluation described in this chapter. First, we have demonstrated that MOMENTUM fulfills the requirements for mobility and connectivity in sparse MANETs. Our proposal also integrates nicely with existing technologies. Second, we have a better understanding of the best video streaming/transport solutions for the different mobility versus density scenarios. Third, we establish solid foundations for the research described in the next chapters of this dissertation. These are our proposals in delay-tolerant routing, video forwarding and video adaptation to improve MOMENTUM, as well as a video client application for this environment and a tool to analyze mobility scenarios, i.e. MASS.

The results demonstrate that our overlay network supports a great deal of the connectivity situations that are expected in sparse MANETs. The set of scenarios

covered realistic mobility and density combinations. On the one hand, MOMENTUM is able to deliver video in scenarios where client-server cannot. This includes connected (MANET) and disconnected (DTN) situations and transitions between them. The overlay network is also resilient to route breaks. Adding up all the experiments, it delivers approximately 58% more video than client-server. On the other hand, it performs almost as well as client-server when there is connectivity, but it introduces a small overhead. This overhead is especially relevant in the form of delay in the video packets. In addition, our proposal integrates with existent technologies. We use standard video protocols and applications in the source and destination nodes. This design eases the connectivity with external networks as well. Finally, the overlay network can coexist with other services in the network, as long as they can use OLSR as well, because it does not modify how the network works.

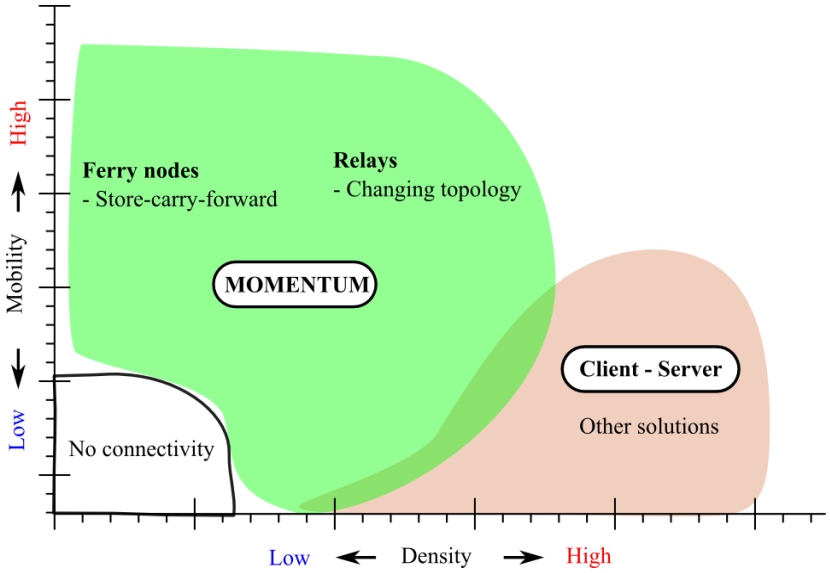

Figure 5-15 MOMENTUM application domain in the density vs. mobility space

Figure 5-15 summarizes video delivery solutions in the mobility versus density domain. There is an area where no solution can deliver video because connectivity is very low or inexistent. Then, there is a zone with a fairly high density of nodes that move slowly. It represents networks with good connectivity that changes infrequently. Connectivity is almost as stable as in the Internet, although other problems may exist, such as unreliability of ad-hoc links. The traditional client-server architecture works here, either those designed for the Internet or those improved specifically for streaming over MANETs. Finally, there are scenarios where connectivity exists but it is not stable. Either nodes move fast and/or produce sudden and frequent route changes, or the network is partitioned and nodes visit different partitions connecting them. This is the type of network expected in many ER scenarios. MOMENTUM is a suitable solution in these circumstances as our experiments have proved.

These experiments helped us to establish key research challenges to improve the performance of MOMENTUM. First, opportunistic and delay-tolerant connectivity is frequent in the targeted scenarios. In these situations, it is essential to select ferries wisely, because video delivery depends on them. Getting the maximum network throughput depends on being able to predict node movement and plan delay-tolerant network routes. As we have seen, this is a weak point in the current MOMENTUM architecture, because ferries and relays deliver only 2% of the video packets. In

addition, random mobility scenarios are not realistic. We must study how mobility is in ER scenarios and then study ferry node selection in these situations. Second, we have not studied the effect of packet losses that are not due to mobility, such as collisions and so on. We must carry out experiments using more realistic network simulation to study this issue. Finally, we have studied how video packets are delivered in some particular scenarios. When disruptions occur or ferries are used, the packet delivery pattern is different to common video streaming. Packets are received in bursts. Existent video players are not the best applications in this environment. Thus, we study new video client applications tailored to this type of networks.

Chapter 6

Delay-tolerant routing

This chapter discusses delay-tolerant routing for video transport in ER operations. We use our knowledge on the mobility in ER operations to build a mobility model of these scenarios. Then, we propose a solution to build delay-tolerant routes, Emergency Overlay Routing (EOR). EOR is implemented inside MOMENTUM and compared with other strategies, i.e., PROPHET. In addition, EOR is tested in the DTS-overlay architecture from the related work. Our experiments show that EOR is superior to PROPHET in packet delivery and delay. However, we have also discovered some weaknesses in its design; especially when the "a priory" information does not match the mobility in the experiment.

The results presented in this chapter are published in [107] and [108].

6.1 Delay-tolerant routing

Delay-tolerant routing consists on finding paths to connect nodes that are in different network partitions. Thus, a delay-tolerant route is a route formed by ferry nodes, which are not simultaneously connected and store the packets before forwarding them. For this reason, it has been also called as routing in time in [19], opposed to routing in space, which is the task of standard routing protocols such as OLSR. Delay-tolerant routing is a crucial part of video transport in a sparse MANET, because source and destination are likely to be in different partitions. Then, ferry nodes using the store-carry-forward paradigm can be leveraged to overcome partitions. The main challenge is how to find these ferries that can carry packets from the video source to the destination. In MOMENTUM, Context-aware Overlay Routing is the component in charge of this task. Our previous prototype solved delay-tolerant routing by selecting random nodes as ferries and by limiting ferry selection to the source and the destination. Our goal in this chapter is to get a better understanding of this problem in the context of ER and explore feasible solutions.

Several strategies are used in the state-of-the-art to cope with delay-tolerant routing art [109]. The most basic protocols use a brute-force approach, spreading multiple copies of the information they must disseminate, such as epidemic routing does in [110]. Data delivery is maximized with this approach, but it is an inefficient alternative resource-wise. If MOMENTUM used it, video packets would be broadcasted to all the nodes in the partition and stored until they could be delivered. The consumption of bandwidth, energy and storage would be high. Hence, flooding the network with video packets would compromise scalability. For that reason, we look for a routing solution that keeps limited copies of each video packet in the MANET, one in the best case. This means limiting the number of nodes that receive and store the packets. Depending on how well mobility is known, there are several approaches. Their success has a tight relationship

with understanding mobility in a particular scenario. If mobility is known, delay-tolerant routes are also known in advance. For example, a network node in a public bus can be used to connect nodes in close neighborhoods. In this situation, static routes easily solve the problem, as in [1]. On the contrary, if mobility is completely random, the simple solution of selecting ferries randomly, as we have done in the previous chapter, might be good option. However, in real situations mobility may follow patterns and nodes may form clusters, especially when we talk about human mobility. This issue has been analyzed before in studies such as [111] and exploited by protocols like PROPHET [112]. These protocols look for mobility patterns and predict future mobility using them. They use these predictions to generate routing metrics, e.g. future contact probability, to define the best hops in the delay-tolerant route.

State-of-the-art ferry selection strategies are aimed to general scenarios or application domains different than the ER operations tackled in this thesis. In Chapter 2 we described some interesting patterns of mobility in ER operations that can be leveraged to improve delay-tolerant routing. In addition, previous work on delay-tolerant routing is not evaluated for video transport applications. Most of the opportunistic protocols are evaluated with messages, not data streams, and focus on delivery ratios. While delivery is important for us, delay must be also considered. For these reasons, we aim to design a delay-tolerant routing protocol that, first, leverages our knowledge on mobility and, second, is tailored for video transport. This protocol is compared against state-of-the-art proposals and other tailored proposals with interesting results.

6.2 Emergency Overlay Routing

6.2.1 Overall design

Emergency Overlay Routing (EOR) is a delay-tolerant routing protocol designed for video transport in ER operation. It leverages the knowledge of mobility extracted from our previous studies. Chapter 2 discusses the role of the Intervention Chief and identified him/her as the most likely video destination in a real situation. For that reason, EOR aims to transport video from a source node located in the Incident Area to the Intervention Chief in a DTN scenario. This implies finding a delay-tolerant route of one or several ferries that connect them. EOR is part of Context-aware Overlay Routing. It is used in DTN context, i.e. when the destination is not connected.

The first design challenge is who defines the route, which could be done by the source or hop-by-hop. In the previous chapter, ferries were selected randomly by the video source. However, this is a limited design in real ER operations. Routes of several successive ferries could be needed. For example, a video recorded by a firefighter may need to be forwarded to another firefighter, then to a vehicle and finally to the destination. Furthermore, the source would have to select a succession of nodes that, acting as ferries, would store, carry and forward video packets. Nonetheless, it is unlikely that the source would have enough information to define the whole route. Selecting the route hop-by-hop is a better alternative.

Overlay network nodes with video packets stored use EOR to look for potential candidates to be the next hop in the delay-tolerant route. In other words, they look for ferries towards the Incident Chief. Thus, we have to decide which nodes can be ferries and how many of them are used. Initially, any node in the network partition is a

potential ferry candidate, because OLSR provides connectivity inside the partition and Topology Discovery the appropriate mechanisms to discover connected nodes. As we have seen, several delay-tolerant routing proposals rely on spreading multiple copies of each packet to increase its chances to be delivered. The previous MOMENTUM prototype used a similar approach. Video transport benefits from replicating packets, because it might reduce their delay as well. Nevertheless, resource consumption increases, which hinders the scalability of MOMENTUM to support several videos over in MANETs with limited resources. Therefore, we opt for a resource conservative approach for the next prototypes. MOMENTUM keeps only one copy of each video packet in the overlay network. In other words, when a node forwards a packet to another node, it is removed in the sender. Even if one copy is kept for every packet, there is also the possibility for EOR to select more than one next hop at the same time. Several delay-tolerant routes could be used to forward video packets through different paths. This could be advantageous as it may increase the probability that at least some video packets make it to the destination. However, it also requires policies to divide video packets and criteria to select ferries. Thus, EOR selects just one next hop, one ferry, at the time. In specific, EOR in every overlay node with video packets looks for the best ferry in the network partition. This requires establishing a criterion to choose the best ferry. A protocol to exchange the data that supports ferry selection is also needed.

6.2.2 Ferry selection

EOR is aimed to video transport in ER operation. Thus, ferry selection must take into account these two goals. Low delay and high delivery are desirable for video transport, even in a DTN scenario. Hence, EOR looks for ferries that are likely to introduce minimum delay. In order to increase video delivery and minimize resource consumption, reliability is also taken into account in ferry selection. For instance, avoiding ferries connected through long network routes. In addition, EOR leverages the knowledge acquired by studying mobility in ER operations. EOR selects ferries using "a priori" information about the scenario in which it will be deployed. This approach is hybrid between static routing and pure probabilistic routing protocols, such as PROPHET. In [108], we call it knowledge-driven routing. For the current version of EOR, we make two assumptions that emerge from our knowledge of ER operations. First, mobility is different for different units. A firefighter moves different than a vehicle. A water pump moves different than a helicopter. Some units may have movement patterns that are useful for data transport. Second, we consider that the node that has been longer in the Incident Area is more likely to meet the Intervention Chief first. This is a simple but realistic assumption for many scenarios, especially small ER operations.

Ferry Value (FV) is the routing metric that EOR uses to decide the best ferry. The node in the network partition with highest FV is considered as the best ferry. This triggers MOMENTUM to forward packets towards it, unless the highest FV correspond to the node storing the packets. Hence, each node must be able to calculate its own FV. There are many alternatives to calculate a routing metric, from simple alternatives such as the number of hops to more complex ones such as estimating the available bandwidth of the route. In delay-tolerant routing protocols, it is common to measure past connections with nodes and process them to predict future connections, as PROPHET does. FV is calculated based on the previously explained characteristics of video transport and mobility in ER operations. For that reason, we use the following input parameters:

Node type (T). In an ER operation, nodes are carried by different types of units from persons, to fire trucks or helicopters. Tagging nodes with the identity of who carries them is a useful input for routing decisions. A vehicle is more reliable as ferry than a person for several reasons, i.e. less battery constraints, less probability of breaking the network device during the intervention, or more likely to go back to the Intervention Chief. In addition, they build more stable network links when parked. A fire truck is likely to stay in the Incident Area until its water tank is emptied. A helicopter moves much faster than any of the other vehicles. A good knowledge of the movement patterns of each unit increases the value of this parameter. The type of unit can be configured in each network device.

Seconds since the last encounter with the Intervention Chief's node (S). Our current knowledge about mobility in ER operations indicates that a node that has been working in the Incident Area for a long time is likely to go back to the Intervention Chief's location soon. The more seconds have elapsed since last encounter with the Intervention Chief's node, the more likely they will meet again soon. Of course, this may be different for different units, but Node Type can be used to modulate this. This parameter can be obtained from the OLSR routing table using the Topology Discovery component. In each node, a timestamp is saved when a network route to the Intervention Chief's node exists.

Number of hops to the ferry (H). The longer the network route is, the less reliable it is. Moreover, this is a bigger problem for video streams than for single packets. We introduce the number of hops in the ferry selection to add reliability in video forwarding. This parameter indicates the number of network links that video packets must traverse to reach a ferry. In some occasions, it pays off to use a closer ferry and avoid packet losses, even if it is not the best according to the other variables. This parameter can also be obtained from OLSR. Note that when a node calculates its own FV, this variable is equal to 0.

These three input parameters are processed to calculate FV. There are two main alternatives to do so: dynamic/adaptive or static. The former is a powerful alternative as the way to calculate FV could be better fitted while EOR is running. For example, a machine learning algorithm could be used to identify the best ferry according to past successful ferries. Although this is advantageous, it is too complex for the current state of our research. Among other things, it would need complete mobility traces from real scenarios or a real deployment to be evaluated. For that reason, we opt for a simpler static calculation of FV. We combine the input parameters in a formula and introduce a scaling factor.

Scaling factor (K). It is a rational number used to weight the influence of the previous variables in the formula. For example, a Scaling Factor of 2 divides by two the weight of hops in the ferry value, which is useful if network routes are considered reliable.

All these inputs are combined in the following formula:

$$FV = K \cdot (T \cdot V)/(H+1)$$

Thus, each node can calculate FV for every other node and also for itself, but it needs to obtain the parameters T and V. For that purpose, EOR implements the protocol described in the next section.

6.2.3 Protocol

The EOR protocol has the task of gathering the input parameters to calculate FV. We aim a scenario where the Incident Chief is the only destination. Therefore, each node has stored the value of the time elapsed since the last contact with him (V). Each node also stores its own Node Type (N). The nodes that look for a ferry in the same partition, i.e. the nodes that have video packets stored, need these parameters. Their transmission can be done proactively or reactively. In the proactive approach, every node would periodically broadcast its own N and V to the others. In the reactive approach, a node would send them under request. In our application scenario, only some nodes in the partition are interested in receiving these parameters. Therefore, using a reactive protocol consumes fewer resources and provides the same functionality. The main disadvantage is the response time that increases delay.

The design of this protocol resembles AODV, one of the state-of-the-art reactive routing protocols. It uses a Route Request message and a Route Response message. Requests reach all nodes in the network partition. Responses are unicast messages to the node that sends the original request. The protocol is triggered as follows. First, Distributed Video Transport Control asks Context-aware Overlay Routing for a node to forward video when there is video stored in the node. This action initiates a request by EOR if the destination is not connected (DTN mode) and the best ferry has not already been found. Figure 6-1 illustrates the process of sending a request (EORReq) (1); waiting for the responses (EORRes) (2) and, once a ferry has been selected, forwarding the video (3).

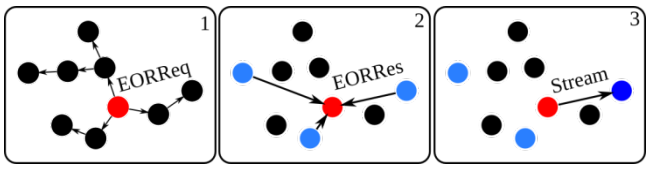

Figure 6-1 EOR protocol stages

Route Requests must reach all nodes in the network partition. To avoid flooding the network, which is the simplest alternative, EOR uses the nodes selected as Multipoint Relays (MPRs) by OLSR. OLSR selects MPRs of a node among its 1-hop neighbors. They are chosen in such a way that all 2-hop neighbors can be reached when they broadcast a packet. The mechanism to forward Route Requests works this way. The requester broadcasts the request reaching all its 1-hop neighbors. Then, it is only broadcasted by MPRs of the requester reaching 2-hop neighbors. Next, MPRs of these MPRs repeat the broadcast and so on. This process ensures that every node in the partition receives the Route Request, but consuming fewer resources than flooding the network. This mechanism is the same that OLSR uses to spread its Topology Control messages. Topology Discovery is used by EOR to know the MPR relationships, so a node knows if it is MPR of another node. Then, it decides if it must broadcast the message or not. Although MPRs are used, a node may receive the same request message several times. Broadcasting it every time is a waste of resources and can create broadcast storms under some circumstances. The source of the message on its own is not enough to decide if a request should be broadcasted, because a node can create several requests. For that reason, each Route Request contains a unique sequence number generated by the node creating it. Therefore, EOR uses two criteria to broadcast a request: being MPR of the node from it is received and being the first time it is received.

A Route Response is a unicast message destined to the node that initiated the Route Request. They contain the parameters T and V of the node sending it. The simplest approach is that every node receiving the request sends a response. This implies one message for each node in the partition. This solution can be improved by reducing the amount of responses generated. The enhancement consists on including the values of V and T of the node looking for a ferry in every Route Request. Then, a node receiving the request calculates its own FV and the FV of the route requester from the point of view of the requester. This can be done by setting the parameter H to the values the requester would use, i.e. 1 for itself and the distance between the two nodes for the ferry candidate. Hence, only nodes with a higher FV answer with a Route. This optimization avoids useless responses, which reduces the overhead introduced by EOR. Figure 6-2 summarizes the full process triggered when receiving a request.

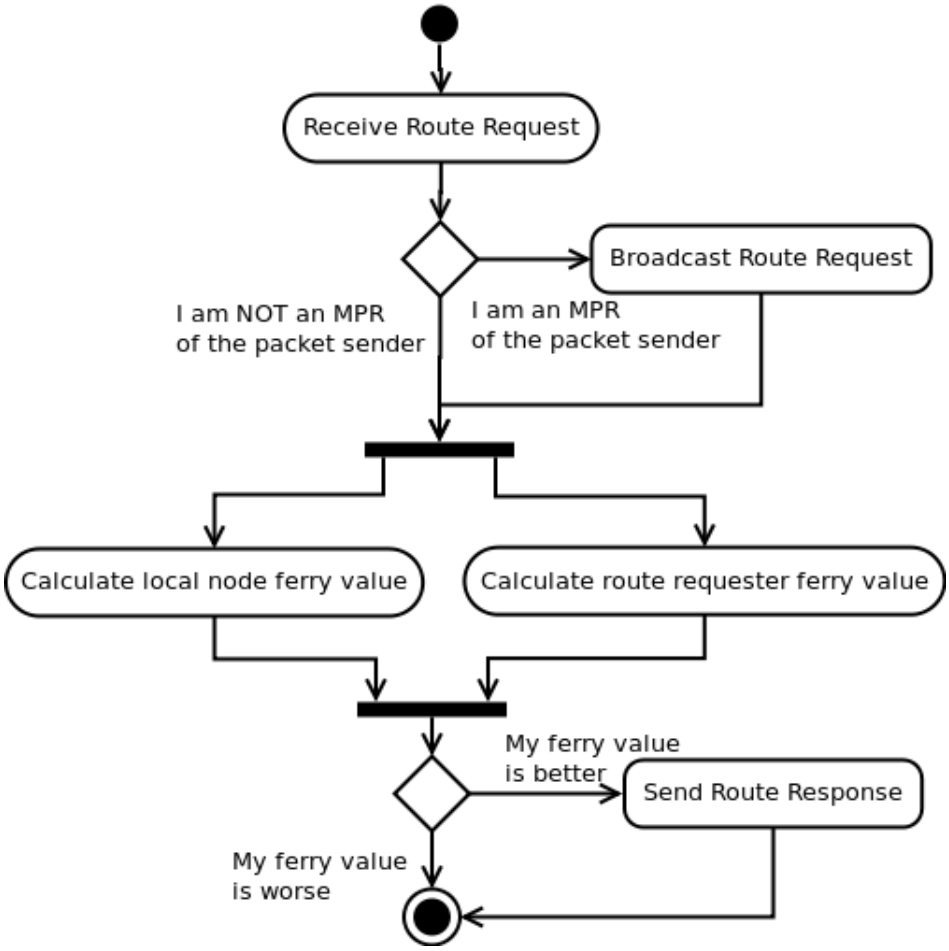

Figure 6-2 A Route Request triggers this activity diagram

As Route Responses are received, a node calculates the FV of potential ferries. The node with the highest FV is selected as ferry. However, responses do not arrive simultaneously. EOR must know when to make a decision. If a winner is declared too soon, the response of a better ferry may be received afterwards. Then, changing to a better ferry can be beneficial, but frequent changes can hinder QoE. Video packets would be spread between many nodes and reach the destination separately. Then, the video player would have problems decoding the video, because the video might be too fragmented in different nodes. On the contrary, waiting a long time to select a ferry can be a waste of time and available network capacity. The available time to forward video

to the ferry is limited and it is better to start soon. Thus, EOR should wait for a time in the middle of these extremes. However, the optimal value is difficult to find, because it depends on, among other factors, the round trip time between the route requester and the rest of the nodes. By now, we leave it as an implementation parameter.

After a ferry has been selected, changes in the network topology may occur. Nodes may join and leave the partition and they may change their position in the topology. Topology Discovery detects these changes and notifies EOR. Then, it sends a new Route Request. If a better ferry is found, video packets are forwarded to this new ferry. This has a collateral effect. Several nodes in the network partition can select the same node as ferry, e.g. the video source and a former ferry. This situation is feasible as all nodes can select ferries and use the same criteria. The positive side is that video packets end up stored together in the same node, which could produce a better QoE if this node delivers them to the destination. Nonetheless, in extreme situations, the network surrounding the ferry and the ferry itself can become congested, because many nodes forward it video. Since we currently target small scenarios with only one video stream, this problem is not likely. However, it should be considered in bigger systems. In that case, Distributed Video Transport Control could be used to coordinate video forwarding between nodes to avoid congestion.

Transport and Multicast Routing is used to send both Route Request and Route Responses. Route Requests are sent to the network broadcast address. Store-carry-forward is not applied for broadcast packets; they are always transmitted to the air and reach 1-hop neighbors, if any. Route Responses are unicast packets. In the current version, acknowledgements are not used. A lost Route Request is likely to arrive through another neighbor node. The possible loss of a Route Response is assumed, because Route Requests are repeated when the topology changes. Thus, the node may be selected as ferry then. The priority of EOR packets is set to a low value, because they should not be prioritized over video packets or signaling, which constitute the essential traffic for the service. This way, if video packets are being forwarded over a known route, route updating mechanisms will not hinder this process.

6.3 MOMENTUM prototype

This section describes the modifications of the MOMENTUM prototype. Their goal is to include EOR and to simplify the prototype for the evaluation presented in the next section. In addition, we describe the implementation of EOR and PROPHET as ferry selection strategies for our experiments.

6.3.1 General modifications

In the previous prototype, video applications exchanged RTSP messages that were managed by Distributed Video Transport Control. One of the side effects was that video transport did not start until applications had initialized the RTSP session. This is a realistic situation in a real application, but once observed it does not provide any benefit in further experiments. However, it produces undesirables traffic and startup delay. Therefore, we have simplified the prototype by completely removing signaling between the applications and in the overlay network. The location of the video applications is hardcoded in our prototype. In addition, the applications are configured as a source and

a sink of RTP packets, without RTSP or RTCP. Therefore, Distributed Video Transport Control just carries out the task of managing Quality-aware Video Forwarding.

The former prototype also included the selection of relays. Relay nodes are tailored for video forwarding in MANET scenarios. Since we focus now on the evaluation over DTN scenarios, they are not needed. We simplify the prototype by removing relays for our following experiments. Therefore, Context-aware Overlay Routing only carries out the context detection and the selection of ferries.

Transport and Multicast Routing remain the same, but some of its functionalities are not used. The previous two simplifications, together with the decision of having only one copy of each video packet in the network, produce that multicast routing is not necessary. In addition, none of the packet types (video packets and EOR messages) need acknowledgements.

6.3.2 EOR

EOR is implemented as part of the Context-aware Overlay Routing component. There are a few implementation and configuration issues that must be solved. In the following experiments, we consider four types of nodes: the camera, firefighters, cars and the Intervention Chief's node. The first three node types have a preconfigured Node Type (T) of 0, 1 and 2 respectively. These values discard the video source as a ferry and give double value to the cars than to the firefighters. The parameter V, which measures the last contact with the destination, needed small modifications in Topology Discovery. In every overlay node, this component stores the timestamp of the last contact with the Intervention Chief's node. Then, when a Route Request or Route Response is generated V is calculated by subtracting the current time and the stored timestamp. Hence, nodes do not need a synchronized clock to compare the time of their last contacts with the destination. Topology Discovery is also used to obtain the number of hops (H). Finally, we set the scaling factor (K) equals 1.

6.3.3 PROPHET

PROPHET is a widely accepted probabilistic protocol. It is inspired by the movement patterns of people and assumes that nodes often in contact with each other have higher probability of encountering each other again in the future. We aim to compare the knowledge-driven approach of EOR with the probabilistic ferry selection of PROPHET. Although there are working implementations of PROPHET, they are not compatible with the MOMENTUM architecture. For that reason, we implement a ferry selection strategy based in PROPHET inside MOMENTUM. Instead of Ferry Value, EOR uses the Encounter Probability defined in PROPHET. The higher the probability is, the better the ferry. The remainder EOR mechanisms remain the same. For simplicity we have not implemented the transitivity properties of PROPHET.

The Encounter Probability considers past encounters to predict future node connections. So each time the Intervention Chief is in the same partition the encounter probability is updated as:

$$P=P_{old}+(1-P_{old})\cdot P_o$$

Then, at any given moment the Encounter Probability is calculated as:

$$P=P_{old}\cdot\gamma^{V}$$

P is the Encounter Probability with a node (in our case the Incident Chief). P_o is a constant representing the initial probability. P_{old} is the value of the probability calculated in the last encounter with the node. Gamma (γ) is the ageing constant and V represents the time elapsed since last encounter (as in ferry value). Values of gamma and the initial probability have been taken from the values recommended in [112]:

$$P_o=0.75,\ \gamma=0.98$$

6.4 Evaluation

6.4.1 Goals

The main goal of these experiments is to demonstrate the advantages of using a knowledge-driven strategy, like EOR, in a scenario resembling the mobility of an ER operation. Although it is not possible to use real mobility at this stage of our research, we evaluate MOMENTUM using scenarios that contain the same patterns found in real ER operations. Then, we compare the performance of EOR ferry selection, which uses "a priori" information, against PROPHET, which uses a pure probabilistic approach. We expect EOR to leverage higher video packet delivery and to achieve lower packet delay. However, there may be some situations where PROPHET is able to detect the existing patterns and perform well.

A secondary goal is to evaluate MOMENTUM in a more realistic protocol stack. The previous evaluation with NEMAN demonstrated the capacity of the overlay network to adapt to MANET and DTN context. However, it ignored factors that are important in a real deployment. Layer protocols below MOMENTUM were not modeled in detail. Hence, packet losses due to problems in these layers, e.g. shared medium collisions, were not present. For the current experiments we aim a better modeling of lower layers, because they are important to evaluate EOR and MOMENTUM. On the one hand, packet losses affect EOR, e.g. what happens when a Route Response is lost. On the other hand, the selection of a particular ferry can produce packet losses, e.g. creating network congestion. In addition, a realistic simulation of the full protocol stack can challenge the other MOMENTUM mechanisms, revealing weaknesses and strengthens of our design. These issues are not only important for a real deployment, but also to prioritize necessary future work.

6.4.2 Emulation environment

We use real time simulation with ns-3[6]. It is a network simulator widely used in the research community. Among many other features, it includes realistic models of wireless networks. They can be used to simulate a MANET. In our setup, each ns-3 network node is connected to a light virtual machine[7] (lxc) through virtual network interfaces (taps). Virtual machines handle independent protocol stacks. Thus, a process

[6] http://www.nsnam.org

[7] https://linuxcontainers.org/

running in them only sees the network simulated inside ns-3. One MOMENTUM instance and one olsrd instance run in each machine.

Since we no longer use RTSP in the applications, we use VLC[8] as client and server application, instead of openRTSP and live555. VLC is a flexible and well-documented application that suits better our current purposes.

6.4.3 Incident Area Mobility Model

The evaluation of ferry selection strategies must be carried out considering realistic mobility. The best option is to use real scenarios, e.g. GPS traces of emergency services mobility. However, the traces available to us represent only vehicle mobility and are, therefore, incomplete to analyze specific ER operations. For that reason, we rely on scenarios generated by mobility models. Models based on random mobility, as Random Walk for the previous chapter, are not useful to evaluate ferry selection strategies. Disaster Area Mobility Model is the only alternative to provide realistic scenarios in our application domain. However, it does not represent the mobility in the Incident Area in the detail we require it. To solve this problem, we have designed a simple mobility model, called Incident Area Mobility Model, which is based on the patterns observed in our study of real mobility. The Incident Area and the Intervention Chief are taken into account. In addition, firefighters are grouped in teams and stay for a limited time in the Incident Area. The model allows us to generate multiple scenarios with realistic mobility.

The main input parameters of the model are:

- The coordinates and size of the Incident Area.
- The coordinates of the Intervention Chief location.
- The number of teams in the ER operation. There is one vehicle for each.
- The number of members in each team.
- The Rest Time is the interval between the maximum and minimum time a team can rest.
- The Intervention Time is the interval between the maximum and minimum time a team can work.
- The Task Time is the interval between the maximum and minimum time a person carries out a task.
- Walking Speed is the interval between the maximum and minimum speeds of a person.
- Vehicle Speed is the constant speed at which vehicles move.

One node is located in the Intervention Chief location. Then, one node for each vehicle and team member is created and initially located in this point too. The activity diagram

[8] http://www.videolan.org/

in Figure 6-3 defines the movement of each team after this. Each team waits for a random time, computed from a Uniform Distribution between the Rest Time limits, before beginning to move. Then, a parking point is selected inside the Incident Area. This point can be a prefixed point; so all the vehicles will park in the same place, or selected using a Uniform Distribution over the Incident Area. Next, the car moves in a straight line at Vehicle Speed. When the parking point is reached, the team members in the vehicle start moving inside the Incident Area following the Random Waypoint Mobility Model. We use this model to simulate that firefighters carry out tasks inside the Incident Area. Each firefighter node moves to a random point in the Incident Area, stays there for a time selected using a Uniform Distribution between the Task Time limits and then goes to another random point. The movement between points is done at a speed randomly chosen, again using a Uniform Distribution, in the limits of Walking Speed. Team members move freely until the time of intervention, which is a random number from the Uniform Distribution of Intervention Time, is over. Then, every node goes towards the vehicle position. When all the team members are in the vehicle position, they move together towards the Intervention Chief location. This process is repeated independently for each time and random numbers are only used once, e.g. parking points, new rest times or intervention times are generated for each iteration. The scenario runs as long as it is specified. Finally, for our experiments, we have added a node that moves following Random Waypoint inside the Incident Area, which represents the camera.

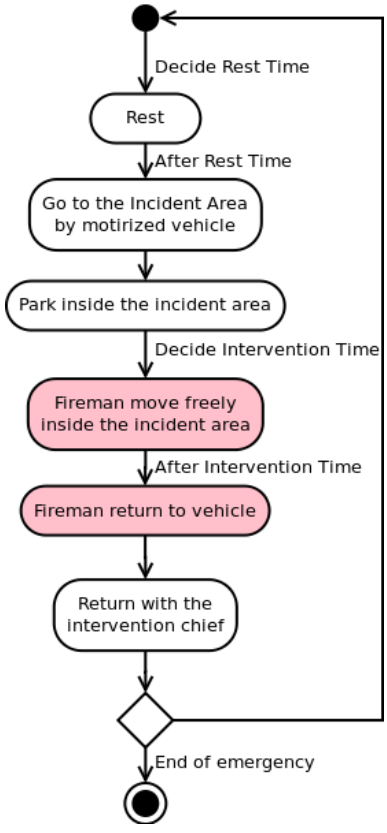

Figure 6-3 Incident Area Mobility Model Activity Diagram

Note that there are several differences between our model and Disaster Area Mobility Model. First, we consider group mobility when moving from and to the Incident Area.

Disaster Area assumes that personnel and vehicles moving between zones are independent. Second, our model reflexes more accurately the workflow of firefighters in the Incident Area. We have used Random Waypoint to model firefighter mobility, assuming that firefighters carry out tasks independently in any point of the Incident Area. Depending on the type of ER operation, it is meaningful to model this mobility as a type of group mobility or using hot spots where firefighters gather. The most important feature of this model is that it models the most challenging part of the communication, from firefighters in the Incident Area to the Intervention Chief. Therefore, we can assume using Random Waypoint inside the Incident Area, because in this stage it is more important how they go in and out of the Incident Area than how they move inside it.

We have implemented the Incident Area Mobility Model in ns-3 and MASS.

6.4.4 Experiment description

The ns-3 scenarios in our experiments use the Incident Area Mobility Model. Each scenario contains one node for the Incident Chief and one node for the camera. Then, one node for each vehicle and firefighter. Each team has one vehicle and four firefighters, so five nodes per team. We generated ten scenarios with different input parameters for the model, which fall evenly into two categories: connected and sparse. Each scenario is identified by the seed used for random number generation in ns-3, i.e. seeds 1 to 10. Seeds 1 to 5 are connected scenarios and seeds 6 to 10 are sparse scenarios. Table 6-1 summarizes the parameters used to generate them. Connected scenarios are based on the dimensions of the chemical accident trial we assisted. On the contrary, sparse scenarios have a bigger Incident Area, which it could represent a wildfire. These two categories produce different conditions for routing and video delivery. In the connected scenarios, communication between the Incident Area and the Incident Chief is mainly delay-tolerant, but multihop routes are possible. In addition, nodes inside the Incident Area are likely to form a unique network partition. Sparse scenarios pose a completely different situation, because nodes in the Incident Area are likely to form a heavily partitioned MANET. Although there are more teams in the scenarios of this category, they are not sufficient to produce the same network density. In addition, the Incident Chief is too far away with the purpose of forcing delay-tolerant routing towards it. The mobility of the teams is modulated by the Intervention Time and Rest Time parameters. The ranges have been chosen so there are several contacts between the teams and the Intervention Chief in the duration of the scenario (6000 seconds). These time values have been selected for practical reasons. However, nor these ranges or the total duration of each experiments are realistic, because an ER operation can last several hours and teams can be at work for longer periods of time. We have reduced them, because long experiments suppose a big computational and memory effort. In addition, this decision should not influence the comparison between ferry selection strategies, although absolute values for delay may be unrealistic. Finally, each of the ten scenarios is repeated three times to avoid random errors to affect our results.

We want realistic simulation of low layers. Thus, each node is configured with a network device with the DSSS 11Mbps version of 802.11b. The Friss propagation model is used for losses and the Constant Speed propagation model for delay. To obtain realistic communication ranges the parameter RxGain of the WiFi physical layer is set to 16, which establishes a range of approximately 100 meters. The configuration of OLSR is the default provided in its RFC [16].

The workload is a video stream called Coastguard sequence from the Video Trace Library[9]. It is encoded in MPEG-2 at 500 Kbps and repeated in a loop by VLC until the experiment finishes.

Category	*Connected*	*Sparse*
Seed	1 to 5	6 to 10
Incident Area	200x200 m^2	1000x1000 m^2
Distance to the Incident Area	100 m	400 m
Teams	2	4
Intervention Time	U [300, 1500] s	
Rest Time	U [300, 600] s	
Node Range	~100 m	
Duration	6000 s	
Runs with each seed	3	

Table 6-1 Scenario parameters

6.4.5 Metrics

The current evaluation focuses on comparing EOR and PROPHET ferry selection in terms of packet delivery. We use tcpdump[10] to monitor the packets generated by the server and delivered to the client. Then, we calculate the following metrics:

Delivered Video Packets is the number of packets that arrive at the client in the Incident Chief's node.

Delivered Video Packets Ratio is the number of packets delivered to the client divided by the number of packets that are generated by the video server.

Video Packet Delay is the time that it takes for a delivered packet to reach the Intervention Chief's node since it is generated by the video server application. There are thousands of packets delivered in each scenario; thus, we use mainly the mean and the standard deviation of these values, as well as their frequency distribution.

[9] http://trace.eas.asu.edu/

[10] http://www.tcpdump.org/

6.4.6 Results and discussion

This section discusses the results obtained. We start with an overview of all the scenarios and then we examined each of them in more detail.

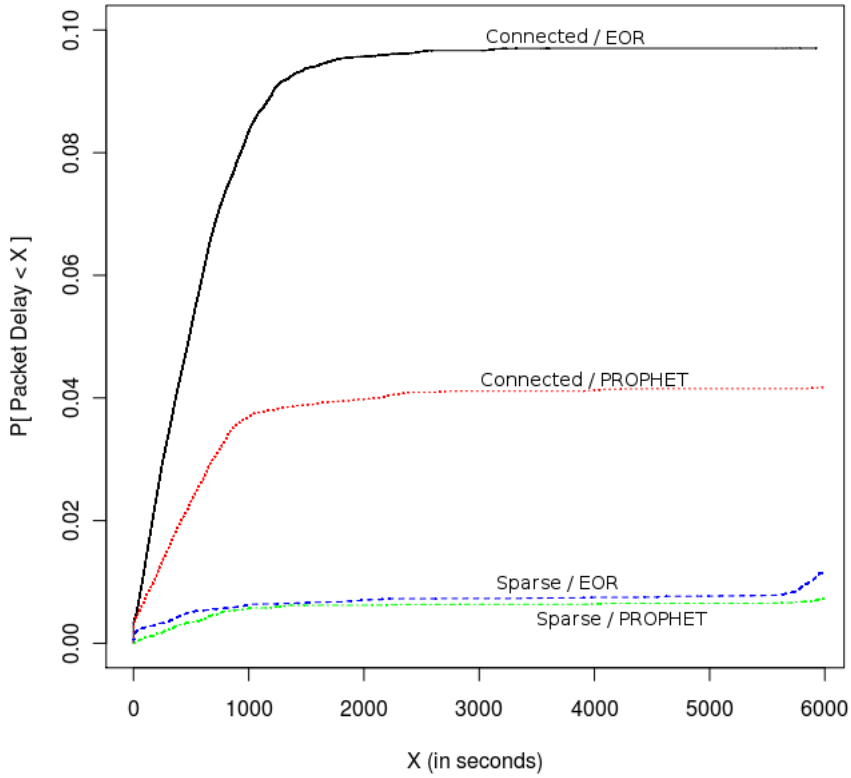

Figure 6-4 ECDF of Video Packets Delay

We have summarized the results of all the scenarios and classified them by scenario category and ferry selection strategy. Figure 6-4 shows the Experimental Cumulative Distribution Function (ECDF) of Video Packet Delay for all the packets in all the experiments. The Y-axis indicates the percentage of packets produced by the video source that arrived before the seconds, represented in the X-axis, passed. The video server sends packets in a very similar way in all experiment runs, so distributions can be directly added and compared. Infinite delay is assigned to not delivered packets, but they are not in the figure. Hence, it is easy to compare both delay and delivery. The first noticeable result is that packet delivered adding all scenarios is low for both strategies and scenario types. When the scenarios finish, it is below 10% for EOR in the Connected scenarios and below 1% in the Sparse ones. PROPHET achieves a much lower ratio in the Connected scenarios and slightly lower in the Sparse ones.

The numerical results in Table 6-2 shows the exact differences adding all scenarios. In terms of Video Delivery Ratio, EOR clearly outperforms PROPHET in the big picture. In terms of delay, the ECDF shows that for the same number of packets, EOR achieves lower delay than PROPHET. EOR is faster delivering packets. In the numerical results, the average delay obtained by PROPHET in the Sparse scenarios is smaller, but this is a statistical effect. The mean for PROPHET is calculated with fewer samples. Moreover, EOR delivers some packets with high delay, close to 6000 seconds, which increase the overall mean and standard deviation. As we have mentioned in the description of our experiments, the absolute values for the delay are meaningless on their own. They have

to be interpreted together with the mobility of the network, which constraints them. In our mobility scenarios, the value of Intervention Time limits for how long a team is working in the Intervention Area and, therefore, how often someone goes back to visit the Intervention Chief. Thus, delay values move in the same magnitudes of these parameters. The comparison of PROPHET and EOR demonstrate that the latter is faster.

Now, we analyze scenarios individually. We separate results by ferry selection strategy and scenario seed, adding the results from the three repetitions of each. Figure 6-5 and Figure 6-6 represent Delivered Video Packets and average Video Packet Delay for each scenario. Remember that seeds 1 to 5 correspond to connected scenarios and 6 to 10, to sparse scenarios. Figure 6-5 illustrates that EOR delivers more packets in all the connected scenarios. Nevertheless, there are sparse scenarios where PROPHET performs better. A small Incident Area eases communication between the camera and the nodes of firefighters and vehicles. Thus, EOR is likely to use vehicles as ferries. First, they have a higher Node Type (T). Second, routes are likely to be too short to counterbalance it. In the 200x200 meters Incident Area, the camera can often find a 1-hop or 2-hop transmission with a vehicle. Third, firefighters and vehicles in the same team have visited the Incident Chief at the same time, so they have the same value for parameter V. Since they are parked in the area, routes are more reliable and less likely to break, at least until the vehicle leaves the Incident Area. On the contrary, PROPHET selects less reliable transmissions, because vehicles are considered equal to other nodes. On the contrary, these reliable transmissions are less frequently found in a big Incident Area. Vehicles are static and firefighters follow random movement. For that reason, our knowledge-driven strategy is less effective in the sparse scenarios. Figure 6-6 shows that PROPHET obtains better average delay in some scenarios. In some of them, such as seeds 3, 9 and 10, this is due to the statistical effect of more packets being delivered and adding seconds to the average value. Seed 7 generates the only scenario where PROPHET is superior to EOR in both delay and delivery.

Scenario type	**Routing**	**Delivered Video Packets**	**Delivered Video Packet Ratio**	**Mean Delay (seconds)**	**Standard Deviation**
Connected	EOR	341341	0.097	552.0024	465.5865
	PROPHET	145118	0.041	587.5564	716.2637
Sparse	EOR	40718	0.011	2389.3530	2561.5905
	PROPHET	25891	0.007	1211.4365	1786.2271

Table 6-2 Numerical summary of experiments by Scenario Type and Protocol

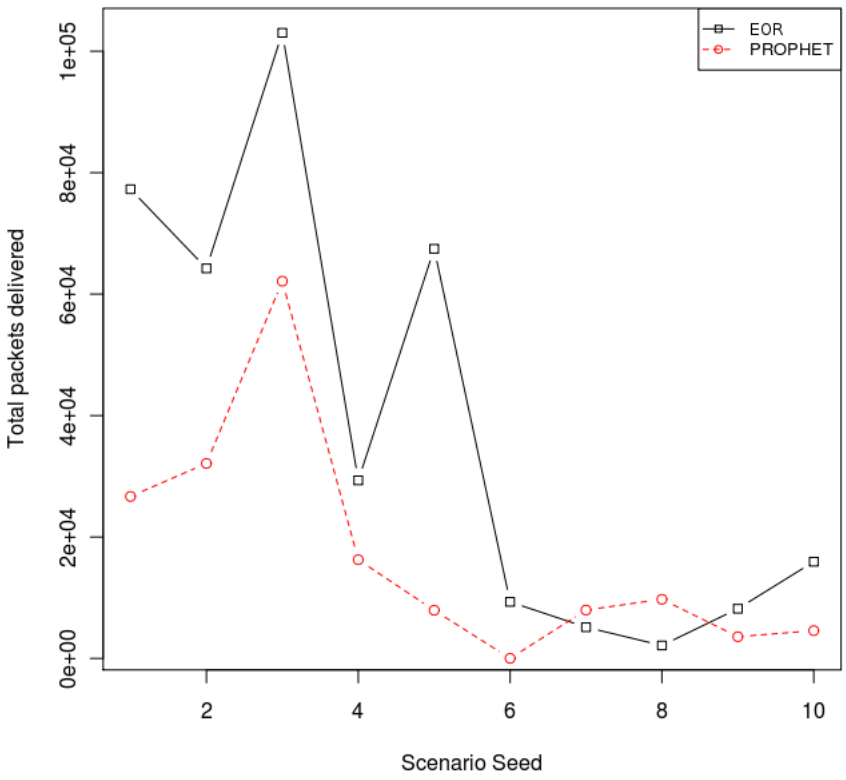

Figure 6-5 Video Packet Delivery

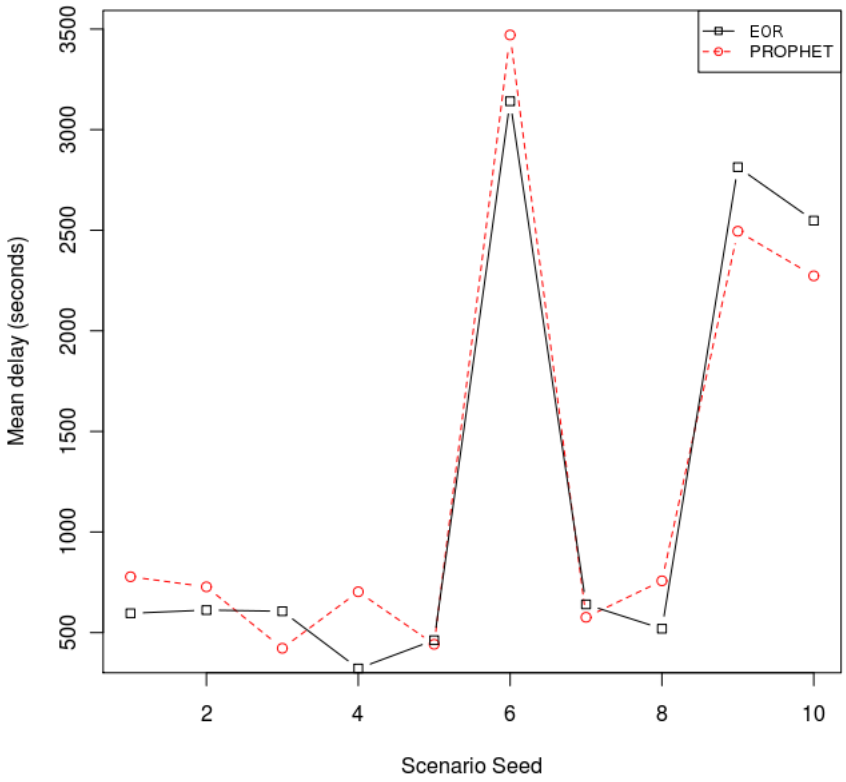

Figure 6-6 Video Packet Delay

6.5 Ferry selection in DTS-overlay

This sections summarizes the evaluation and conclusions from [108]. In this paper, we evaluate four ferry selection strategies in the DTS-overlay architecture [97]. DTS-overlay enhances mechanisms of the WiFi MAC layer to improve video delivery in sparse MANETs. The results are not directly comparable with our previous experiments, as DTS-overlay and MOMENTUM take different approaches. However, they drawn some interesting conclusions about the ferry selection problem and about the strengths and weaknesses of EOR.

6.5.1 Experiment description

DTS-overlay is setup with two configurations: C1 that enables several lower layer adaptations to avoid packet losses and C2 that only adapts to the routing table. We compare four ferry selection strategies. Two of them are EOR and PROPHET. The parameters for PROPHET are the same that in our experiments (Po =0.75, γ=0.98), but the Node Types in EOR change. We assign T=5 for vehicles. Then, a static approach used IP addresses to identified predefined ferries, e.g. vehicles. Thus, the video source looks for those IPs in its routing table and forwards packets to them. The last strategy is Dynamic Selection of Message Carriers (DSMC) [84]. DSMC is a probabilistic approach, as PROPHET is, that does not rely on a priori knowledge. In DSMC, every node maintains a contact probability with every other node that is updated every second as follows:

- If nodes a and b are in contact: $E_{ab} = (1 - \alpha) [E_{ab}]_{old} + \alpha$
- If nodes a and b are not in contact: $E_{ab} = (1 - \alpha) [E_{ab}]_{old}$

The parameter α is the ageing factor in the range [0, 1]. In our experiments, $\alpha = 0.001$.

The goal of the experiments is to compare two probabilistic approaches (DSMC and PROPHET) with static routing and a hybrid solution (EOR) solution. For that purpose, four mobility scenarios are used. First, Random Waypoint scenario had 50 nodes in a 300 m x 1500 m area. Second, a scenario resembles mobility in ER operations, called ER 1, similar to Incident Area Mobility Model. It has 10 nodes moving with Random Walk in a 500 m x 500 m incident area, an Incident Chief at 1750 m and 3 nodes moving back and forth between these two locations (called carriers). The third scenario, called ER 1B, is a variation of this one, where one carrier stopped in the incident area at half of the experiment run (1800 seconds). Finally, we use an Incident Area Mobility Model scenario, called ER 2, using a configuration of the sparse category. These scenarios cover several degrees of uncertainty in the movement. In ER 1, the nodes configured as carriers are the best ferries. In ER 2, a bit of randomness is incorporated, because the network is sparser and nodes move randomly for some time. In Random Waypoint, the scenario is completely random; no assumptions can be done.

6.5.2 Results and discussion

The overall results are uneven. DSMC obtains the best overall packet reception results in the four scenarios. It adapts well to the different levels of uncertainty or randomness. EOR achieves higher delivery than PROPHET in most experiments, but shows little gains over the Static approach. In these scenarios, using EOR is almost equivalent to

using vehicles as ferries, as a consequence of the high Node Type assigned to vehicles (5). EOR does not adapt well when the assumed knowledge is wrong. However, when the assumed knowledge is good, it achieves high delivery rates and lower delay than the other alternatives. All in all, the approaches using "a priori" knowledge show little or no gains over the probabilistic approach DSMC. This would indicate that they are able to automatically infer what is introduced as input in the other approaches. However, it must be also taken into account that some benefits of using "a priori" knowledge are not showed in these experiments. For instance, a node in a vehicle should have less resource limitations than a mobile device, which constitutes an advantage not reflected in this evaluation.

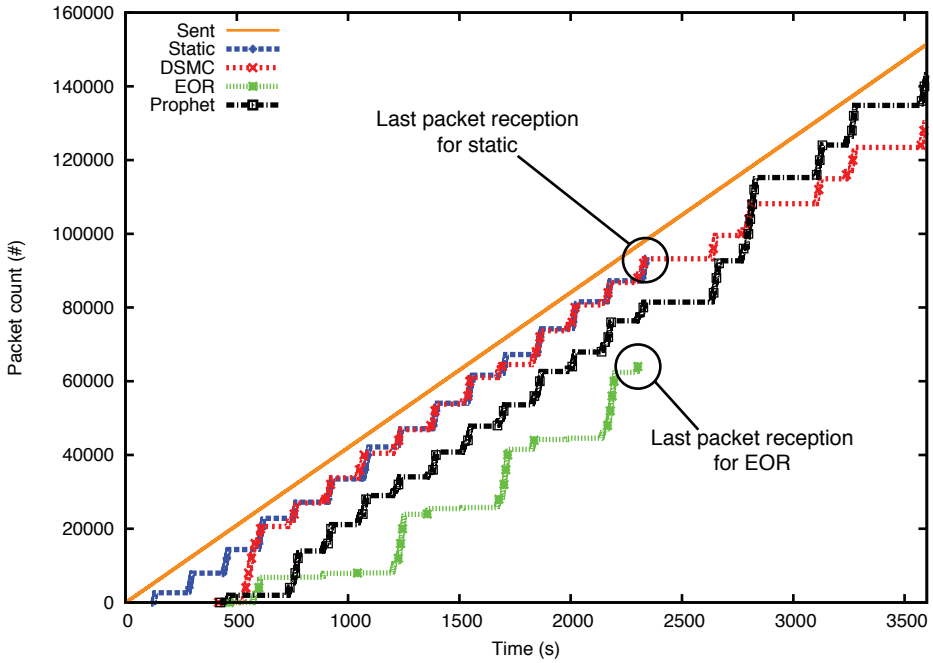

Figure 6-7 Packet reception in a experiment with stopped ferry

An important insight from these experiments is the effect of ageing factors. An ageing factor is incorporated in PROPHET and DSMC, but not in EOR. The scenario ER 1B test the effect of the ageing factor by stopping a ferry halfway the simulation. Figure 6-7 shows packet delivery for all four strategies in a run of this scenario. The way static and EOR work, trusting in the nodes that have been a longer time in the incident area, stops them from delivering packets after some point. However, PROPHET and DSMC detect and overcome this event thanks to their ageing factors. The ageing factor applied to the stopped carrier discards it as a selectable ferry at some point. This is a flaw in the design of EOR.

These experiments also illustrate the importance of considering lower layers and packet losses in the evaluation process. The two configurations of DTS-overlay, C1 and C2, produce very different results. C1 avoids all packet losses, but it does not always obtain the highest packet delivery. In ER 2, C2 delivers more packets for all strategies but PROPHET. Thus, the packet loss prevention mechanisms of DTS-overlay may not always be convenient in sparse networks with fewer chances of packet delivery. An aggressive forwarding strategy may be better in some situations, although it would imply more packet losses and consume more network resources. In addition, ferry selection affects packet losses differently. Depending on the scenario, some strategies suffer more losses than others, but there is not an observable pattern. A similar effect is

produced in the network resource consumption, which is different for all the strategies, scenarios and configurations.

6.6 Conclusions

This chapter studies the problem of ferry selection in ER operations. Our proposed solution, EOR, performs better than PROPHET, the most renowned delay-tolerant routing protocol. It manages to deliver more video packets and to do it with a lower average delay. However, we also discovered important weaknesses that need to be covered in further research. The use of "a priori" information is good as long as this is relevant for the network mobility. EOR performs worse when the mobility patterns used for its design are not present, e.g. in scenario ER 1B. Moreover, the more random the mobility is, the worse EOR performs. This is reflected in the sparse scenarios, where the difference between EOR and PROPHET is lower than in the connected ones. The evaluation over DTS-overlay also reflects this issue. EOR finds it difficult to adapt to different or new mobility patterns. Thus, not only it is important to study possible patterns in ER operations, but also to react to unexpected mobility. Future work may tackle this problem by introducing additional mechanisms, such as ageing factors or a probabilistic component additional to the ferry value. Future solutions could be based in the combination of these enhancements with the presumed potential of "a priori" knowledge of ER operation.

Several reasons cause the low ratio of delivered video in these experiments. First, it is the limits imposed by mobility. The contacts between nodes establish the amount of data they can exchange. If two nodes never get in contact, they cannot exchange information directly. The second reason is the efficiency of the delay-tolerant routing protocol. Mobility may allow nodes to connect using ferries, but only if they are found and used. Finally, packet losses also occur in our experiments when video is being forwarded. Overlay network nodes have unlimited storage, they do not dropt packets, so there are not losses produced by congestion. Some packets are lost due to collisions or other issues related with the shared medium. For instance, the RTS/CTS mechanism does not always perform as expected in MANETs [113]. In addition, multihop routes tend to cause more collisions. Nonetheless, the majority of packet losses detected is due to temporal disruptions in the network routes. The OLSR routing protocol has some delay in detecting broken routes. During this period of time nodes keep a nonexistent route in their routing tables. A packet forwarded over a broken route is likely to be dropt at a node that cannot find the next hop. The precision of OLSR detecting network changes can be improved by increasing the frequency of OLSR packets. However, this increment hinders network performance and increases packet losses, as it is studied in [114]. Another cause of temporal disruptions is the underperformance of ARP in MANETs. ARP resolutions are less stable in a MANET, thus, they must be done with more frequency than in conventional wired or wireless networks. As the authors of [102] suggest: ARP resolutions should be merged with neighbor discovery. However, this is not the case for OLSR and for our testbed. As a consequence, in our experiments, packets forwarded to a node with an unresolved ARP address are silently dropt by lower layers. This event causes a temporal disruption, although the network route exists. DTS-overlay overcomes this problems using lower layer improvements. However, this is not possible in MOMENTUM, which must work over state-of-the-art networking. Therefore, in the next chapter we tackle video forwarding to solve these issues.

Chapter 7

Video forwarding and adaptation

This chapter studies video forwarding and adaptation in sparse MANETs. We enhance the design of the Quality-aware Video Forwarding component with two new mechanisms. Error and Flow Control (EFC) deals with packet losses in video packet forwarding. Dynamic Temporal Scalability (DTS) aims to adapt the quality of the video delivered to the available network throughput. Both solutions are evaluated and the results published in [115] and [116].

7.1 Quality-aware Video Forwarding

Quality-aware Video Forwarding manages video packet forwarding in MOMENTUM. Beyond the functionalities described in the previous chapters, this component tackles two key problems. First, packets are lost during video forwarding between overlay network nodes. Temporal disruptions are the main cause for this loss. We propose a new mechanism, Error and Flow Control (EFC), to cope with them. The second problem is the lack of enough network capacity to the entire deliver video in a DTN. Even if all packet losses were avoided and delay-tolerant routing protocols worked perfectly, the network capacity in a DTN is constrained by the node mobility. Hence, it is possible that throughput is smaller than the video generated in the server. In these situations, the goal of video adaptation is to reduce the quality of the video to adapt it to the available resources. However, adaptation is not trivial in this environment. Dynamic Temporal Scalability (DTS) is a packet scheduling technique to carry out frame rate adaptation. We propose DTS as a suitable mechanism for video adaptation in DTNs.

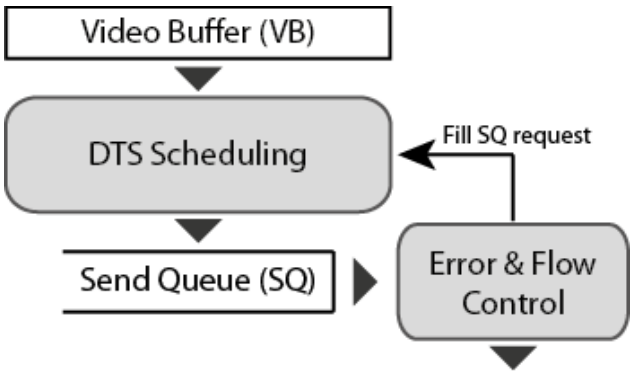

Figure 7-1 Quality-aware Video Forwarding architecture

Both mechanisms, EFC and DTS, collaborate in video packet forwarding with the support of some extra data structures. Figure 7-1 represents the architecture of Quality-aware Video Forwarding. Video Buffer (VB) stores data of a video stream, either received from other nodes, or produced in the node by a video server. Video is stored with associated metadata extracted by the Application Gateway. These metadata include

frame sequence numbers, frame types according to the video codec, e.g., I, P and B frames, and the relation between packet and frame sequence numbers. Send Queue (SQ) stores packets that are ready to be forwarded. DTS is in charge of reading VB and writing packets in SQ. EFC reads SQ and sends the packets to Transport and Multicast Routing, which forwards them to the MANET. As we explain later, EFC uses a transmission window that is filled with the packets in SQ. For that reason, EFC triggers DTS scheduling.

In the following sections, we justify the design of these two mechanisms.

7.2 Error and Flow Control

The experiments described in the previous chapter revealed a low Video Delivery Ratio in several scenarios. This result has three main causes: the constraints of mobility, the inefficiency of the routing policies and packet losses. The first one cannot be solved and solving the second is the task of delay-tolerant routing protocols explored in the previous chapter. Reducing the losses is a problem that can be tackled in video packet forwarding. As we mentioned, most packet losses are produced by temporal network disruptions, e.g. caused by ARP and OLSR incompatibilities. Disruptions hinder video forwarding more than other types of traffic because packets are sent continuously. Thus, a disruption produces the loss of many video packets. MOMENTUM needs to improve video forwarding trying to avoid and correct packet losses occurred during disruptions. The mechanisms in the Transport and Multicast Routing component are not enough to cope with this problem. Flow control in Transport and Multicast Routing component only targets the problem of UDP send buffer overflow. In addition, the acknowledgement and retransmission scheme would be only useful to detect and correct losses, but not to avoid them. We need additional actions in the video forwarding process.

Fixing ARP and OLSR is the most elegant solution in the long term. However, this is in conflict with the requirement of using state-of-the-art networking. Besides, disruptions for other causes may also occur. It would also be possible to use a cross-layer design to detect these events and stop video forwarding when a disruption is detected. For example, the ARP resolution table could be read and the node could stop forwarding when a problem is detected. This is feasible in other proposals, such as DTS-overlay, but not in MOMENTUM. Overlay nodes send video packets to other overlay nodes using conventional network routes. Therefore, intermediate nodes in these routes may forward packets at network level, without the overlay network noticing it. If a disruption occurs in any of these intermediate nodes, the packets will be lost anyway. MOMENTUM needs video forwarding mechanisms that can be implemented between the overlay nodes that are sending and receiving the video, e.g. the source and a ferry.

Error and Flow Control (EFC) detects temporal disruptions when they start, stops video forwarding until they end and retransmits any lost packet. Since cross-layer solutions are not possible, disruptions are only detected after packets are lost. The simplest solution is to forward a packet and wait for an acknowledgement to forward the next one. However, this solution underutilizes the available bandwidth. Transport protocols, such as TCP, employ transmission windows to improve this situation. Hence, EFC uses a simple credit mechanism to establish the transmission window. It works as follows. Initially, the sender asks the receiver for credit to dimension its window. Then, it

forwards as many packets as possible. Each packet that is sent decreases the credit by one. The receiver sends more credit to the sender when packets are received. If the sender runs out of credit, it is interpreted as a disruption and the process is restarted. This mechanism is more efficient than acknowledging each packet, but also implies that some packets are lost during the disruption. The number of packets that are lost depends on the credit granted by the receiver, which determines the size of the window. In EFC, it is measured in packets, and not in bytes like in TCP. This is more coherent with our architecture because MOMENTUM manages RTP video packets.

There are two key design challenges in the transmission window: how much credit must be assigned and when to assign it. The amount of credit to assign could be always the same or adapt. The approach of adapting credit to the situation can be good to get the most of the available bandwidth or to avoid congestion. In a MANET, it can be very complex to leverage these advantages and still be able to detect disruptions. For example, it can be difficult to differentiate between congestion and a disruption. Therefore, EFC uses a fixed amount of credit that the receiver grants to dimension the transmission window. We must take into account that a big window size would slow down disruption detection. On the contrary, a small window size would not utilize bandwidth well. Thus, we carry out experiments with several sizes to study its effect. Then, there is the problem of when the receiver must send credit. If it is sent too often, e.g. for each packet, the overhead will be high. On the contrary, the sender can exhaust the transmission window and stop forwarding. We look for an intermediate solution. In the first credit request, the receiver grants the double of the default credit. Then, the receiver grants credit again when as many packets as the default credit have been received. For example, if the original credit for the sender is 50 packets, the receiver will grant 25 packets of credit after each 25 packets received. Video packet forwarding will stop if more than 25 packets are lost. Otherwise, forwarding will go smoothly.

EFC also detects and corrects packet losses. The receiver uses the packet that grants credit to acknowledge the received packets. Then, it is possible to identify lost packets and retransmit them in the next transmission window if necessary. A packet is considered lost when it has not been acknowledged in any of the two ACKs received after its transmission. Retransmissions are optional and can be triggered on and off in the configuration of MOMENTUM.

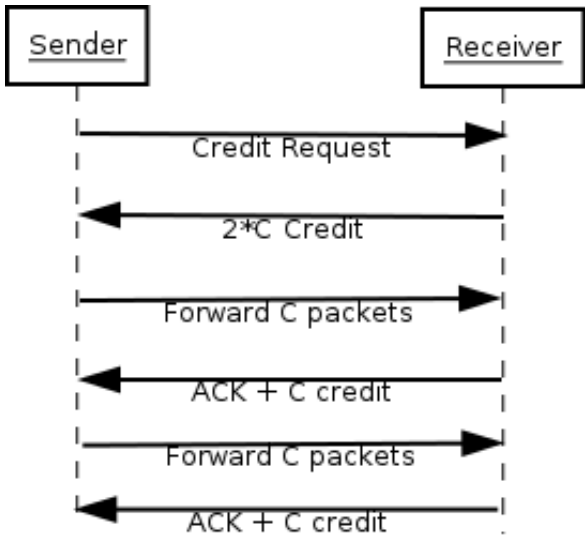

Figure 7-2 EFC sequence diagram

Figure 7-2 summarizes the sequence diagram for EFC, with C as the default credit. First, the sender asks the receiver for credit with a message. The receiver answers granting the double of the default credit, i.e., 2*C packets. The sender initializes the transmission window to 2*C packets. Then, the sender starts forwarding packets. When the receiver has received C packets, it acknowledges them listing their sequence numbers. In the same message, the receiver grants more credit that will increase the transmission window size in the sender. The sender can retransmit lost messages in the next window. The process is repeated until there are not more video packets in the sender, or the transmission window is exhausted. The latter will happen if the credit is not received on time or after more than C packets are lost. In these situations, EFC starts again sending a credit request every second until it receives an answer. If the route is definitely broken, MOMENTUM will eventually detect it and order EFC to stop.

7.3 Dynamic Temporal Scalability

7.3.1 Video adaptation in delay-tolerant networks

As our previous results show, intermittent connectivity produced by mobility hinders video distribution. There are likely situations where it is not possible to transmit all the video that is stored in one node to the next, and, hence, from source to destination. Adapting the video resource consumption to the available network capacity can solve this problem. In this case, the network throughput is in part determined by node mobility. If network mobility is known, like it could be in space Internet applications, the problem is simplified. Network capacity could be estimated and the necessary throughput adjusted beforehand. However, most sparse MANET or DTN application scenarios, such as ER, have nondeterministic mobility.

End-to-end network capacity in a DTN is determined by node encounters. In our Incident Area Mobility Model, meetings of the source, the ferries and the Intervention Chief determine the capacity between source and Intervention Chief. Network capacity is conditioned by (1) how long nodes meet and (2) the available bandwidth between them. When the throughput obtained from the network is insufficient, only a part of the video would be delivered unless it is adapted. The simplest way to avoid this situation is to produce less video data, e.g. reducing its quality. However, the result of nondeterministic node movement is unknown network capacity. Hence, choosing how much the video quality should be reduced is not easy. Furthermore, some applications in ER operations may require a minimum quality, e.g. to identify objects in a video footage you need a minimum size and a relatively high definition. So using low quality videos that underutilize network resources is not always an option. The best solution is to get the maximum video quality with the possible network throughput.

Node mobility is the main difference between this problem and video adaptation in “always connected” networks, because it is impossible to estimate the achievable network throughput to deliver a video. We use Figure 7-3 to illustrate an example of this problem. S is the video source, D is the video destination, and F1 and F2 are ferry nodes. S could estimate its available bandwidth towards F1 and adapt the video stream. Nevertheless, S cannot know the available bandwidth when F1 meets F2 or F2 meets D. In addition, the available bandwidth is only a part of the problem, because nodes meet for a limited time. Normally it is impossible to determine for how long their contact

would last. Therefore, it is also impossible to determine future network capacity and adapt the video accordingly. Furthermore, video packets may be sent over different delay-tolerant routes. Then, the available capacity for each of them may be different. Each delay-tolerant route transport video packets with a different resultant delay and throughput. The movement of nodes is the main cause of these differences. As a result, different video parts are delivered with different available resources, depending on the route they traverse. Therefore, network throughput is not only impossible to estimate under these circumstances, but also it different in different moments.

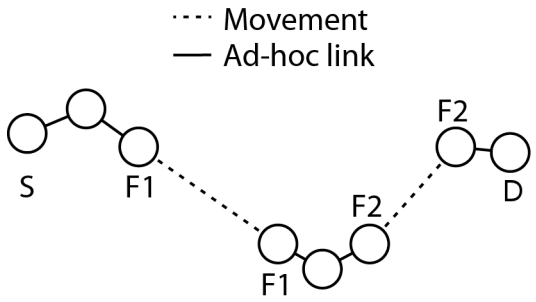

Figure 7-3 Video transmission in an opportunistic network

The consequence of not obtaining enough network throughput is delivering only a part of the video. If video packets are delivered as generated, i.e. in First In First Out (FIFO) order, the video delivered is shorter than the original. The video packets from the are not received, which may be inconvenient for some applications. For example, if the video is used to determine the evolution of a wildfire, the Intervention Chief only has the first part of the footage. Scheduling packets the opposite order, i.e. Last In First Out (LIFO), poses an analogous problem, because the beginning of the video may also contain application relevant information. To avoid receiving only a part of the video, it is necessary to reduce the required bandwidth, which is often done by reducing its quality. Several state of the art adaptive multimedia systems rely on estimating the available bandwidth between the source and the destination of the video. Then, they reduce or increase the video bitrate by adapting its quality, e.g. using transconding [117] or storing several versions of the video with different quality. This is the case of systems based in dash [118] or Scalable Video Coding (SVC) [119]. However, these solutions are not valid in a DTN. They rely on resource estimation. Using layered codecs, such as SVC that uses optional layers to increase video quality, can be useful. They provide a more scalable basis to carry out adaptation and be used by the final system. However, they do not solve the problem as they inherit the aforementioned problems. For example, network throughput may not be enough to deliver the base layer. We need a solution that can scale quality dynamically without depending on throughput estimations. It is also desirable that the solution provides constant quality along the video, so users get a better QoE. In addition, it should be light enough to be carried out by mobile devices.

7.3.2 Temporal scalability

Temporal scalability consists on modifying the video frame rate. SVC leverages this mechanism, but it can be also implemented with other video codecs. Furthermore, it can be implemented in video packet forwarding. For example, dropping the packets that

contain every second frame reduces the frame rate by half. Moreover, previous research, i.e. [120] and [121], claims that adapting the frame rate is often preferred over modifying other video parameters such as frame size (spatial scalability) or frame definition (quality scalability). For these reasons, it is adequate in our application scenario. If only half of the video data could be transmitted end-to-end, it would be useful to apply temporal scalability to deliver a video with half the frame rate, instead of cutting the end or the beginning of it. For this purpose, it is necessary to identify the relationship between frames and packets, which is feasible for many video codecs. The concept is explained in Figure 7-4 comparing it with not adapting the video and scheduling packets FIFO. Using Temporal Scalability (labeled TS) in an ideal situation, video frame rate increases with the available network throughput, while the video length delivered remains the same. The opposite happens if FIFO is used. Frame rate is always as in the original video, but video length depends on available throughput. The video delivered applying temporal scalability is just a video with gaps. A gap is a group of consecutive frames from the original video that are not in the video delivered to the client, as Figure 7-5 shows. If these gaps have the same size along the video delivered, the resulting frame rate is constant. The result is a video with constant quality, the same length as the original, but using the available resources. Nonetheless, the frame rate to adapt to is unknown a priori. Hence, we propose a solution to increase frame rate dynamically without estimating throughput.

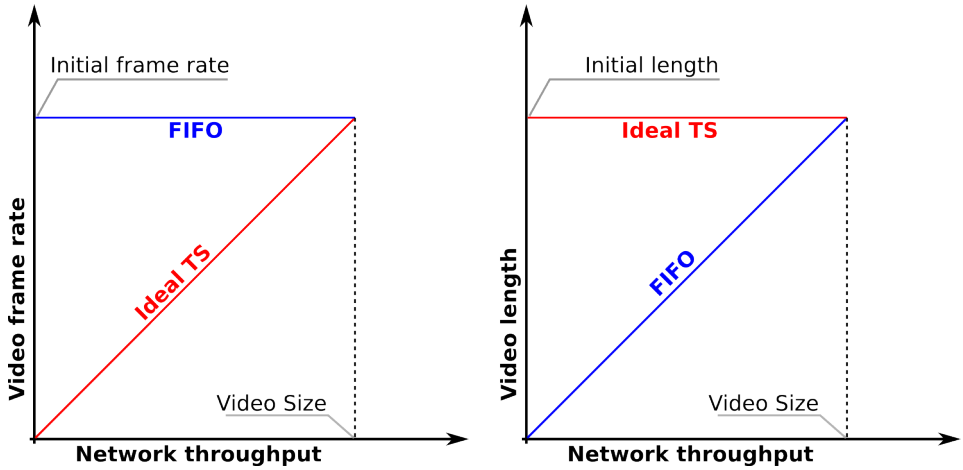

Figure 7-4 Temporal scalability adaptation concept vs. FIFO

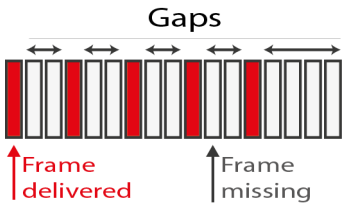

Figure 7-5 Gaps formed by missing frames in the delivered video

Dynamic Temporal Scalability (DTS) schedules video packets to carry out frame rate adaptation. To decide which packets must be forwarded by EFC, it selects frames in such a way that each forwarded frame increases the frame rate of the video delivered. For that purpose, it uses a frame scheduling mechanism that is explained later in this chapter. However, this is not enough to deliver the video with constant frame rate. On

the one hand, more video may be stored in VB, either because it is generated by the server or received from other node. On the other hand, the available network throughput may change over time. Therefore, the available resources are not the same for all the video parts. To balance the resources assigned to all video parts, DTS executes the scheduling sequence in a set of frames of VB limited by a window. Windows over VB are defined to balance the frame rate of all the video parts. Note that this is a window applied over VB and has nothing to do with the EFC transmission window.

7.3.3 Frame selection window

The reason of using a window to select frames from VB is to obtain a constant frame rate in the delivered video while adapting to resource changes and to continuous generation of new video frames. It establishes an upper and a lower boundary to the frames stored to apply frame scheduling. These boundaries are frame sequence numbers extracted from the metadata associated to the video. They define a tumbling window over VB. A window is defined when frame scheduling is triggered by EFC and there are new frames in VB that were not stored in the previous frame scheduling. The aim to establish windows is to forward frames equally in all video parts and, eventually, to deliver a constant video frame rate in the whole video. For that reason, we define the Forwarded Frames Ratio (FFR) as the number of frames forwarded divided by the size of the window. FFR is easy to calculate, based on local information, and it is proportional to the frame rate. The purpose of window selection is to achieve the same FFR for every window defined over VB. Figure 7-6 illustrates an example of this idea. In a given moment (a), a node has 13 frames stored in VB and EFC asks DTS to put packets in SQ. DTS establishes the window W1 and schedules five frames from it. Later, in (b), EFC triggers frame scheduling again. However, now there are nine new frames in VB. In order to apply the same amount of resources to the old and the new frames, the window W2 is selected. Selecting three frames from that window, FFR is the same along the forwarded video. Thus, window W3 would be used for further frame selection if more packets are needed for this forwarding round or if there are no new frames in VB in future rounds.

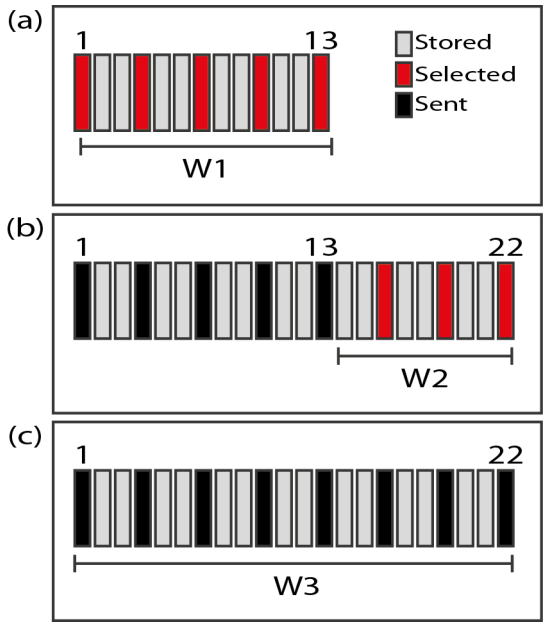

Figure 7-6 Frame window selection example

Each node stores the boundaries of the windows and the value of FFR in them. A new boundary is established when scheduling is triggered and there are new frames in VB since last scheduling. Then, frame scheduling starts over the window with the frames with highest sequence numbers. If this window does not have enough packets to fulfill EFC requirements for SQ, a new window will be established taking as upper boundary the current lower boundary. If a boundary limits two frame sequences that have the same FFR value, it will be removed. The following pseudocode implements this window selection algorithm:

```
SQ: Send Queue
VB: Video Buffer
HF: Highest frame sequence number ever stored in VB.
EFC_TWS: Packets to complete the size of the EFC transmission
window.
{Li}: A set of integers (sequence numbers) ordered so Lk-1 < Lk
n:  the highest index of the set {Li}
W[i, j]: contains every frame sequence number k, so i ≤ k ≤ j
S[i, j]: the number of frames in the interval W[i, j] stored in
VB
FFR[i, j]: is the Forwarded Frames Ratio in the interval W[i, j]

INICIALIZATION()
{
      {Li} = 0                          (i.e. L0 = 0)
      n = 0
}

SCHEDULE(Contact with ferry & VB size > 0 & SQ size < EFC_TWS)
{
      if (HF > Ln)
      {
            Add HF to {Li}
            n = n+1
      }
      k = n
      while (VB size > 0 & SQ size < EFC_TWS)
      {
            Select frame from W[Lk-1, Lk]
            Move frame from VB to SQ
            Update FFR and S
            if ((k > 1) & (FFR[Lk-1, Lk] ≥ FFR[Lk-2, Lk-1]))
            {
                  Remove Lk-1 from {Li}
                  n = n – 1
                  k = k – 1
            }
            if ((k > 1) & S[Lk-1, Lk] == 0)
            {
                  k = k -1
            }
      }
}
```

Apart from VB and SQ, some extra data structures are used. HF is the highest frame sequence number that has ever been stored in VB, even if it was transmitted and it is not in VB anymore. EFC_TWS is the number of packets that EFC requests in SQ to complete the size of its transmission window. The set $\{L_i\}$ contains sequence numbers in ascending order. They are used to establish window boundaries, so they limit frame sequences in VB with different FFR value. New elements are inserted always at the end

of this set, but elements in the middle can be deleted. When an element is deleted, the indexes of the remaining elements are updated. For example, in $\{L_i\} = \{0, 13, 22\}$ $L_1 = 13$ (with i=0, 1, 2), if 13 is deleted, in the new set $\{L_i\} = \{0, 22\}$ $L_1 = 22$. An element is deleted when the two sequences of frames that it separates have the same FFR, as it is exemplified in Figure 7-6. The variable n points to the index of the last and bigger element in $\{L_i\}$. The structure FFR[i, j] stores the value of FFR between the frame with sequence number i and the frame with sequence number j (given i < j). It is calculated by dividing the number of frames forwarded by the length of the interval [i, j], i.e. (j – i)+1. Finally, W[i,j] represents the frame sequence inside the selected window. It contains every frame sequence number k between i and j ($i \leq k \leq j$). Note that these are not only sequence numbers of the frames between i and j stored in VB, but all sequence numbers. Another variable S[i,j] contains how many frames of the interval W[i,j] are stored in VB. The initialization of window selection inserts 0 in the set $\{L_i\}$ and assigns 0 to n.

The Schedule() function is called by EFC when there are frames in the video buffer VB and SQ does not contain enough packets for the transmission window. The first step is to check whether there are new frames in VB with higher sequence number than in the previous call of the function, i.e., HF and the last element of $\{L_i\}$ are compared. $\{L_i\}$ is updated when they are different. Then, for each iteration a window defines a frame sequence using as boundaries the elements of $\{L_i\}$. The window for the first iteration is established between the last two elements (L_{n-1} to L_n). A frame inside the window is selected according to the DTS frame scheduling mechanism. Then, it is moved to SQ. The next instructions update the data structures that are used for window selection in the next iteration. If the FFR of the sequence of frames inside the window surpasses the FFR of the immediately previous sequence defined in $\{L_i\}$, the boundary between them is removed. This expands backwards the current window, because by removing this limit two consecutive frame sequences are merged. Finally, if there are more frames to send in the selected frame sequence, we iterate using the same one. Otherwise, the next frame selection defines a different window. This new interval is the previous frame sequence in $\{L_i\}$. The process is repeated until VB is empty or there are at least EFC_TWS packets in SQ.

7.3.4 Frame scheduling

The DTS frame scheduling selects frames in the frame sequence defined by the window selection over VB. The first scheduled frame is always the first frame of the video, if it is stored in VB. After this or when this frame is not stored in VB, DTS schedules the frame that corresponds to the upper boundary of the current window. Then, frames are selected according to the following sequence:

$$\left\lceil \frac{(2i+1)\cdot N}{2j} \right\rceil,\ 0<(2i+1)\leq j,\ j=1\ldots\frac{N}{2}$$

N is the size of the window and i and j are positive integers (or zero) that are incremented progressively to get the elements of the sequence. When N is an odd number, we take the immediately superior integer as result for the sequence, i.e. the ceiling.

This formula minimizes the gaps between previously forwarded frames. In Figure 7-7, we illustrate an example of it over a window of 13 frames, when five frames are sent.

We consider that it is the beginning of the video, so the first frame is forwarded, next, the last frame of the window and, afterwards, the first three frames as a result of calculating Sequence (1) in {7, 4, 10}. Each forwarded frame reduces the gap and, eventually, produces a constant increment in the frame rate of the forwarded video. In addition, the length in time covered by the original video and the forwarded video is the same. On the contrary, if the order of the frames were maintained, the resulting frame sequence would have the same frame rate as the original in the first frames, but a shorter length, i.e. frames 1 to 5 would be delivered.

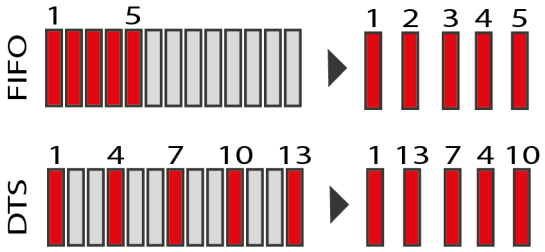

Figure 7-7 Frame scheduling: FIFO and DTS

The sequence numbers of frames to forward are calculated taking into account the boundaries of the window, but not the frames stored in VB. If a frame sequence number is scheduled, but not in VB, it is ignored. For example, in the previous example, if frame 4 were not stored in VB, the selection would be {1, 13, 7, 10, 2}. The purpose is to take into account, as far as possible, frames that were forwarded in earlier rounds or that are transmitted by other ferries.

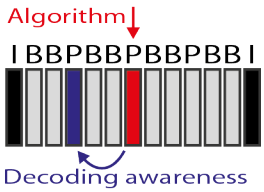

Figure 7-8 Decoding awareness mechanism

The previous frame scheduling is feasible if all the frames can be decoded independently. However, the most used video encoding techniques, such as h.xxx or MPEG families, create sequences with different frame types called Group of Pictures (GOP). Then, decoding dependencies complicate the application of DTS frame scheduling. Usually, the structure of a GOP is repeated throughout the video sequence. For example, in a GOP like "IBBPBBPBBPBB", there are I-frames, which can be decoded independently, P-frames, which depend on previous I or P, and B-frames, which need the previous and the next I or P. The formula used may cause that some of the delivered frames cannot be decoded. DTS frame scheduling is aware of the decoding structure of frames. The selected frame is changed by one that is necessary for decoding it. In other words, when the scheduling algorithm points to a frame in a GOP, the forwarded frame is one of the same GOP following the order: I-frame, P-frames in order or the B-frame (see Figure 7-8). Using this mechanism, the likelihood that all delivered frames can be decoded is increased. The unavoidable disadvantage is that the GOP structure used in the encoding process conditions the frame rate in the delivered video.

7.4 MOMENTUM Prototype

The prototype of MOMENTUM was modified to include EFC and DTS. These mechanisms are implemented in the Quality-aware Video Forwarding component following the described design. The size of the EFC transmission window depends on the credit given by the receiver. The credit value is defined as a parameter configured in MOMENTUM. Besides, the Application Gateway was enhanced to detect the metadata necessary for DTS. The video codec supported in this version is h.264/AVC. It identifies frame types and associates frames and video packets. The current prototype also implements a FIFO video packet scheduler as an alternative to DTS. This scheduler forwards the video packet with lower sequence number first. We use it in our experiments when video adaptation is not relevant or to compare it with DTS. Finally, this version of MOMENTUM also supports static routing, as well as EOR and PROPHET. Static routing is used to evaluate only DTS in a controlled scenario, independent of the efficiency of delay-tolerant routing protocols.

7.5 Evaluation: Packet loss avoidance

7.5.1 Goals

The results from previous chapter show a low video delivery ratio in the chosen scenarios. Losses produced during temporal disruptions are the main cause of this performance. The flow control mechanism in EFC is designed to avoid these losses. Thus, the main goal of these experiments is to demonstrate that EFC prevents packet losses and increases packet delivery. The design of EFC included a fixed amount of credit given by the receiver. This parameter shapes the sender transmission window. Thus, it influences the effectiveness of the mechanism in reducing packet losses and the overhead introduced. Therefore, a secondary goal of these experiments is to study the effect of the credit parameter in the overall system performance.

7.5.2 Experiment description

To obtain comparable results, we repeat some of the experiments from the previous chapter with the current prototype. In specific, we have chosen scenarios with seeds 3 and 10 because in these scenarios we obtained the highest delivery ratio of the connected and sparse cases. A relatively high delivery ratio indicates that EOR performed well, so the scenarios have the potential of achieving higher delivery ratios with EFC. We use 10, 30, 60, 120 and Infinite as window sizes. We expect that the best performance of EFC will be in this range of values. The results using Infinite credit simulate forwarding without using EFC. Thus, they should be similar to the results from the previous chapter, although not the same due to the modifications in the MOMENTUM prototype. Each scenario and credit combination is repeated five times. In these experiments MOMENTUM does not use retransmissions. On the one hand, it makes the experiments more comparable to the previous ones. The expected consequence of using retransmission is to increase overhead and reduce (or even eliminate) packet losses. On the other hand, we aim to evaluate the impact of just flow control and the transmission window as preventive measures. Not using retransmissions makes it easier. Finally, these experiments use FIFO packet scheduling and not DTS.

Video adaptation is not relevant for the current goals and can introduce undesirable noise in the results.

7.5.3 Metrics

We measure the following metrics:

Delivered Video Packet Ratio is the number of video packets delivered to the destination (i.e. Incident Chief's node), divided by the packet generated by the video server.

Buffered Video Packet Ratio is the number of video packets that have not been delivered to the destination but are stored in other overlay nodes, divided by the packets generated by the video server.

Lost Video Packet Ratio is the remainder number of packets that are not delivered or buffered, divided by the packets generated by the video server.

Ratio of Packets by Protocol measures the number of packets sent to the MANET by an overlay node sending EFC, EOR and video packets and dividing by their addition. Note that the same packet may be sent several times by the same or a different node, we count all of them, but we do not count forwarding being made at lower layers (e.g. in 2 hop routes).

Ratio of Bytes by Protocol measures the number of bytes sent to the MANET by an overlay node sending EFC, EOR and video packets and dividing by their addition. We multiply the packets from the previous metric by their size in bytes when they are passed to the MANET.

We analyze overhead in packets and bytes. It is important to analyze both since there is a significant difference in size between a video packet, hundreds of kilobytes, and an EFC or EOR packet, below 1 kilobyte.

7.5.4 Results and discussion

Connected scenario

The connected scenario chosen for the evaluation of EFC is the one with seed 3. We run it five times with each credit value. Then, we average the results of these repetitions and normalize ratios to 1. Figure 7-9 summarizes Delivered, Buffered and Lost Video Packet Ratios for each credit value. The figure shows the positive impact of EFC. Delivered and Buffered Video Packet Ratios increase with finite credit. As expected, the smaller the window is, the more reliable the communication. Without retransmissions, the window size determines the maximum number of packets lost in a route break. For that reason, MOMENTUM loses less than 3% of the video packets in average, if the credit is 10. Lost Video Packet Ratio increases almost an 80% with Infinite credit. It is remarkable that the difference between 30 and 60 is very small. Both have similar average Lost Video Packet Ratio. Looking into more detail in each run with 30, four of the runs have a Lost Video Packet Ratio between 5% and 8%, while one is around 30%. We have not detected errors in this run, so there are no reasons to discard it. We cannot find a unique explanation for this high variance in the results, because many factors affect real-time simulation. On the contrary, Lost Packet Ratios

in the runs with 60 are in the range from 6% to 14%. Therefore, it was expected that 30 would provide more reliability than this. These results proof the effectiveness of EFC in detecting temporal disruptions and preventing packet losses.

More reliability is not equal to more video delivered. These results are an evidence of this, since the credit value that achieves a higher Delivered Packet Ratio is 60. A large transmission window implies risky packet forwarding, so more packets may be lost. Nonetheless, it also implies higher delivery if the transmission goes right. A small window is more sensitive detecting temporal disruptions, resulting in less lost packets. A big window will go on forwarding over short disruptions and increase the number of packets that make it through.

Video packets that are neither lost nor delivered before the end of the experiment remain stored in the overlay network nodes. There are two main causes why buffered packets are not delivered. On the one hand, the nodes carrying them have not found the destination or an appropriate ferry. Then, some of these packets could be delivered with a better delay-tolerant routing protocol, but mobility may make it impossible for some others. On the other hand, it could be that overlay nodes had the opportunity, but the network throughput achieved was not enough to forward all the stored packets. For example, a node is connected to the destination during five seconds, but it would take ten to forward all its packets. Depending on the video service targeted, buffered video can be useful to rebuild the video afterwards, e.g. after the ER operation finishes.

Figure 7-9 Connected scenario delivery summary

We have seen that EFC reduces packet losses and increases the amount of video delivered, but it does it with a cost. In Figure 7-10 and Figure 7-11, we analyze the overhead introduced by EFC compared to EOR and video packets. Figure 7-10 shows that EFC and EOR packets are always below the 30% of the total packets. Furthermore, EFC overhead is smaller than EOR overhead. The main reason is that EOR Route Requests are broadcasted to all the nodes in the partition. The relative magnitude of the overhead in bytes is considerably smaller, always below 1.5%. Finally, the amount of credit granted from the receiver is inversely proportional to the overhead introduced, as expected. However, there is a big difference between 10 and the rest of values. As we mentioned before, a small window is more sensitive to disruptions, which implies more

EFC resets and eventually more EFC packets. In these experiments, there is a clear barrier between using 10 and 30 packets of credit. The reason is probably a combination of scenario mobility, the configuration of protocols in MOMENTUM and other external protocols, such as ARP and OLSR. Even in the worst case, overhead is low compared with the volume of video traffic.

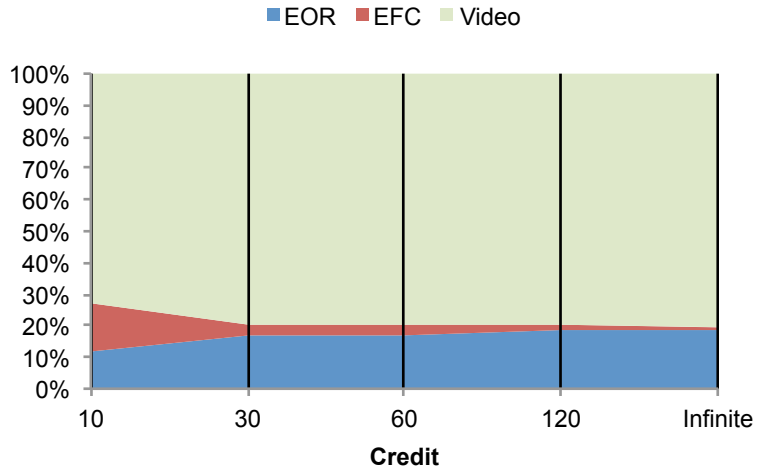

Figure 7-10 Average ratio of packets by protocol

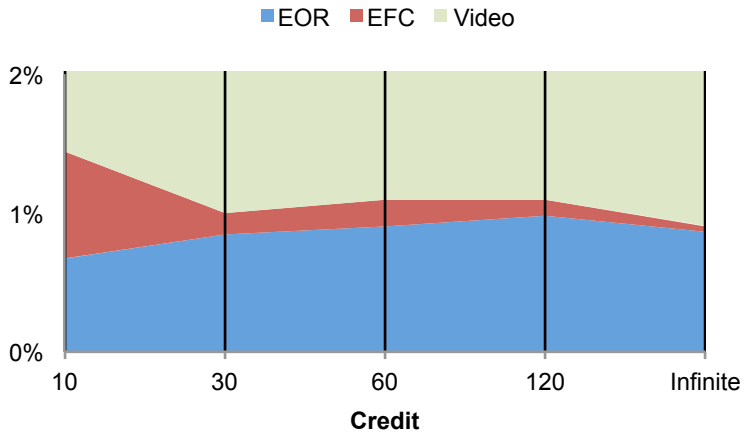

Figure 7-11 Average ratio of bytes by protocol

Sparse scenario

We carried out analogous experiments with the sparse scenario with seed 10. We also obtained analogous results. Figure 7-12 shows Delivered, Buffered and Lost Video Packet Ratios for each credit value. It is noticeable that Delivered Video Packet Ratio with 30, 60 and 120 window sizes rises to a level similar or even superior than for the connected scenario. On the contrary, the experiment with an infinite window reveals very low delivery. This means that EFC has a bigger impact in scenarios where contact opportunities are scarce and topology changes are frequent. A reason may be that a sparse scenario is likely to produce more partitions and produce more ARP and OLSR failures. EFC is effective preventing packet losses when this happens. When credit is set

to 10, delivery is lower for the same reason we explained before. Forwarding is more conservative and loses transmission opportunities. Figure 7-13 and Figure 7-14 show overhead in packets and bytes. The trend of the overhead is similar to the Connected Scenario. A difference is the reduction of EOR overhead. The scenario has a larger area, so network partitions are smaller. Thus, EOR generates less broadcasts and less traffic as it could be expected. In general, these results reinforce our conclusions from the Connected Scenario.

Figure 7-12 Sparse scenario video packet summary

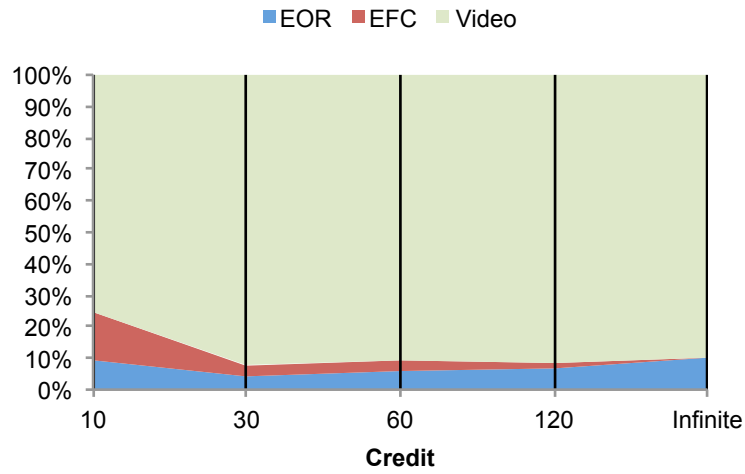

Figure 7-13 Average ratio of packets by protocol

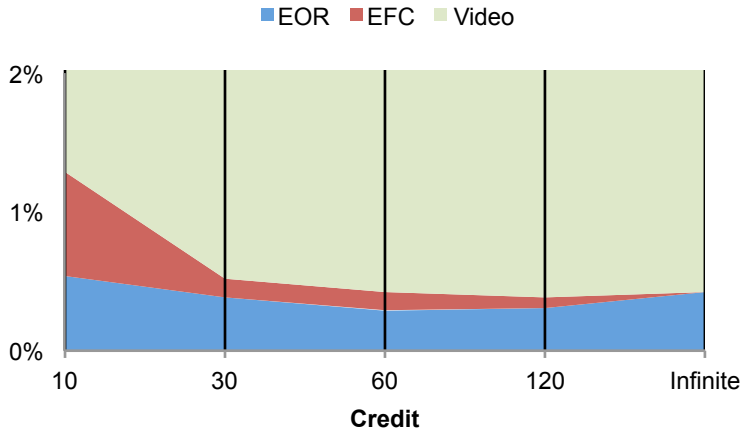

Figure 7-14 Average ratio of bytes by protocol

7.6 Evaluation: Video adaptation

7.6.1 Goals

The goal of these experiments is to demonstrate the properties of DTS as adaptation technique to deliver video. We look into its efficiency using network resources. We study the amount of video that is delivered to the user and how useful it is for decoding it. Then, we look into the properties of the video frames sequence delivered to the destination. The delivered video should fulfill our Quality of Service and Quality of Experience Requirements. In specific, video delivered with DTS should have a length similar to the length of the original video. However, it may never be equal due to the delay of the ferries when live video is transmitted. The quality of the delivered video should be constant in order to provide a better quality of experience to the users. Finally, the quality of the delivered video must scale with the available resources.

7.6.2 Experiment description

The testbed configuration used is the same than the one used in the previous chapter (ns-3, OLSR, etc.). However, we use different mobility scenarios. We have designed a DTN scenario of three nodes that lasts 600 seconds. It is a simplification of our Incident Area Mobility Model. It contains the basic properties of the expected mobility in an ER operation, but it is simplified to isolate DTS performance. A video source, the firefighter in the incident area, and a video destination, the Incident Chief, are static and out of communication range. A ferry node moves between them, stopping close to them. The ferry is at the destination and moves towards the source with a speed of 25 m/s, when the experiment starts. Then, it goes back to the destination. This movement is repeated cyclically. We have built 25 variations of the scenario by combining five different stop times in the destination and the source. These are 1, 10, 30, 50 and 100 seconds. The goal is to simulate different conditions on available resources. Each of the scenarios is repeated five times. Routing is static: packets go from the source to the ferry and from the ferry to the destination. The publicly available video Big Buck Bunny is sent from the source node. The video has 14317 frames and a length slightly lower than 600 seconds with a 24 fps frame rate. It has been encoded using h.264 and a

frame size of 1080p. The average bitrate is 1.5 Mbps and it repeats the GOP structure: "IBBPBBPBBPBB". In the RTP stream generated as input for our system, the maximum packet payload size is 1024 bytes. Bigger frames are fragmented in several packets, but smaller frames are not grouped, they use a packet for their own.

For these experiments, EFC is configured to give 50 packets of credit. This value is in between the two values with better performance in our previous experiments (30 and 60), so it is expected to have a good performance as well. Nonetheless, EFC uses retransmissions to make packet forwarding reliable. For each mobility scenario, we compare the performance of DTS with FIFO, because it is the default approach to packet forwarding. In our implementation, FIFO moves the 50 packets with lowest sequence numbers from VB to SQ. FIFO carries out this operation when requested by EFC.

7.6.3 Metrics

We count the video packets generated by the video source and delivered to the destination to calculate the frames delivered, decoded and fragmented. Conventional video decoders are not prepared to work with incomplete video sequences, like the ones received using DTS. For that reason, we analyze packet traces to calculate our metrics considering the worst-case situation. A frame is delivered if all the packets that carry it are delivered. If some packets, but not all of them, are received, we consider it a fragmented frame. Due to the dependencies between frames generated by video codecs, it can be impossible to decode a delivered frame without others. We consider that a frame can be decoded by the user's video application when all the frames that it depends on can also be decoded. For example, a delivered I-frame can always be decoded, but a P-frame needs previous I and P frames in the same GOP. Finally, all decoded frames form a frame sequence that can be played out by the user. This sequence is compared with the original frame sequence produced by the source. A frame is missing for the user if it was not delivered or cannot be decoded. Missing frames create gaps, as showed in Figure 7-5. The size of a gap is the number of consecutive missing frames. From these basic concepts, we build the following metrics.

Normalized Throughput is calculated by dividing the number of video bytes delivered to the destination and the bytes generated by the video source. This should be similar for FIFO and DTS and relates the available network resources and the video resource requirements. The closer it is to 1, the less video adaptation will be needed.

Fragmentation Ratio is the number of fragmented frames divided by the sum of fragmented and delivered frames. The goal is to minimize this ratio.

Decoding Ratio is the number of decoded frames divided by the number of delivered frames. The goal is to maximize this number.

Gap Size Distribution represents the statistical distribution of the size of gaps in each received sequence. The statistics of this distribution provide objective insights of the quality in the delivered video, because the only video quality property affected by DTS is frame rate. Since subjective QoE tests cannot yet be carried out in our current setup, we use this distribution as an approximation. The gap sizes are proportional to the frame rate. In the following equation we represent this relationship.

$$\overline{FR}_{delivered} \sim \frac{FR_{original}}{1+\overline{Gap\ Size}}$$

The average frame rate of the delivered video can be expressed as a function of the original frame rate of the video and the mean of the gap sizes. The standard deviation represents how the frame rate varies along a video sequence. A zero standard deviation means that the quality is constant. Given a low standard deviation, quality scalability is represented by a mean gap size scaling down (up) when available resources increase (decrease). Note that FIFO leaves only one gap at the end, so this statistical analysis is only meaningful with DTS. Finally, a gap is also a period of time while the user has no information; long gaps are undesired in the ER application domain. The Gap Size Distribution is useful to detect them.

7.6.4 Results and discussion

First of all, we compare normalized throughputs of DTS and FIFO in our experiments. Figure 7-15 represents the Normalized Throughput in each of the scenario runs.

The throughput is similar for FIFO and DTS when they suffer the same ferry stops times. Therefore, frame scheduling does not affect the performance of packet forwarding carried out by EFC and lower layers. Nevertheless, the traffic pattern generated by them is different. FIFO schedules packets as they are created so in the 50 packets sent by EFC big and small packets are included. However, DTS sends I-frames first, which are fragmented into several packets of the maximum size. Thus, DTS sends big packets more frequently. Therefore, FIFO carries out more EFC rounds to obtain the same throughput. This does not affect the video throughput.

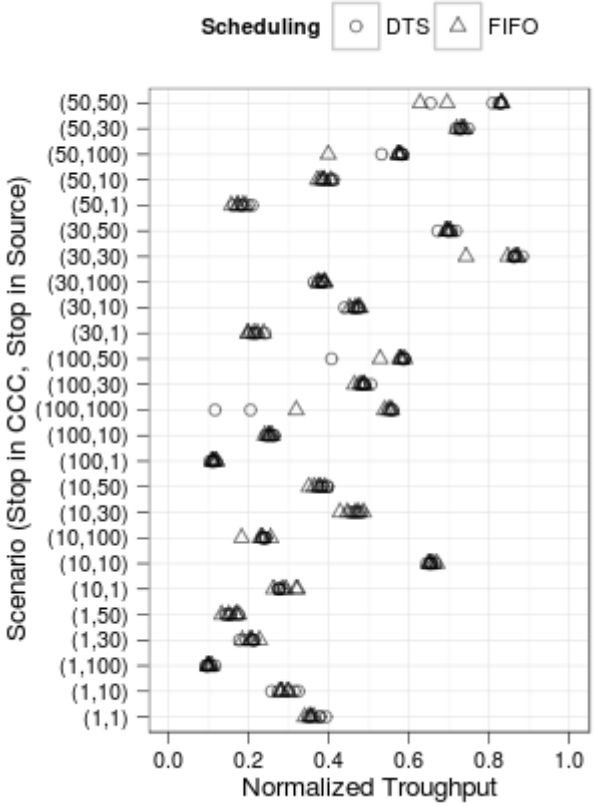

Figure 7-15 Video throughput obtained for DTS and FIFO

These results show that the movement of the ferry determines the available capacity. The best performance is obtained with similar stop times at the source and the destination. Very short contact times (1 second) affect negatively. When the routing protocol is able to discover the route, the node is almost leaving, so the actual opportunity for transmission is too short. Contacts that are relatively long compared to the scenario duration (e.g. 100 and 600 seconds) produce few ferry trips between the source and the destination. These results show the high dependency between mobility and network capacity and, therefore, the importance to make the most out of forwarding opportunities by sending the most relevant information for the user.

We now compare the efficiency in the use of the network capacity to deliver video that can be decoded. For that purpose, we analyze Fragmentation and Decoding Ratios. Figure 7-16 shows Fragmentation Ratio compared with the Normalized Throughput. Each point represents a scenario run and the lines represent the average of the samples. FIFO produces a very low ratio of fragmented frames, lower than 0.25%. This result is coherent with the expected outcome. Only the last frame delivered would have chances of being fragmented, if there were no packet losses. DTS produces a slightly higher fragmentation ratio, but it is also very low. It is below 5% in the worst sample and its average is below 1%. Furthermore, fragmentation decreases notably with increasing throughput. DTS tries to prevent fragmentation of frames by moving all packets of a scheduled frame from VB to SQ. This is reflected in the low ratio obtained, but there is room for improvement. The reason behind fragmented frames is that some packets may remain in ferry buffers without being delivered to the CCC. For example, the connection between the source and the ferry is lost before all the packets of a frame can be forwarded. Then, the ferry forwards these packets to the CCC. In the next connection with the source, the ferry will receive the remainder packets because they are already in the sending queue SQ. However, if this frame is not scheduled again in following contacts between ferry and CCC, it will be fragmented. Although the results show a minimal effect in the overall performance, solutions to this are part of future work.

Decoding Ratio also impacts efficiency. A frame that is delivered but cannot be decoded by the user application is a waste of resources. We represent the Decoding Ratio against the Normalized Throughput for each experiment in Figure 7-17. FIFO decodes almost 100% of the delivered frames. DTS is designed to take into account decoding dependencies, although frames are not transmitted in order. For that reason, the decoding ratio is also high. The user can decode more than 99% of the frames delivered in all scenario runs but one. Reducing the number of fragmented frames will improve the ratio of decoded frames as well.

Now, we analyze the properties of the frame sequence that is delivered to the user. For that purpose, we look into the gap sizes in the frame sequences delivered in the experiments. First, we analyze how close these frame sequences are to live video. The frame with the highest sequence number delivered to the CCC can be used for this purpose. Figure 7-18 represents the highest of the sequence numbers among all decoded frames for DTS and FIFO against the Normalized Throughput. The horizontal line marks the sequence number of the last frame produced in the source, i.e., 14317. The behavior of FIFO is as expected: the higher is throughput, the longer the delivered frame sequence. The effect of this behavior with a low throughput is a big gap at the end of the video. However, the highest frame delivered by DTS has a lower dependency with throughput. Given that the first frame of the video is always delivered, the behavior of DTS is closer to live video than FIFO. The length of the video delivered,

measured as the distance between the lowest and highest frame, is closer to the length of the original video.

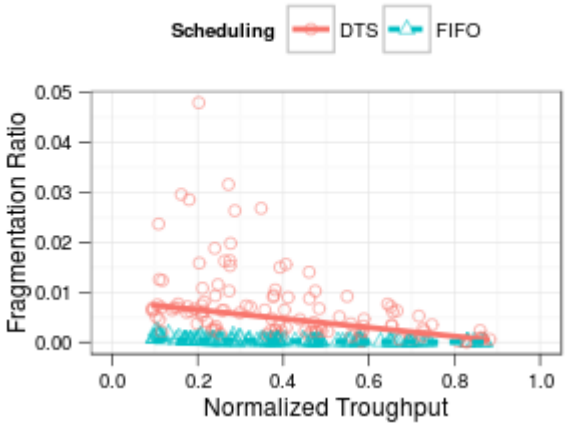

Figure 7-16 Fragmentation ratio against normalized throughput

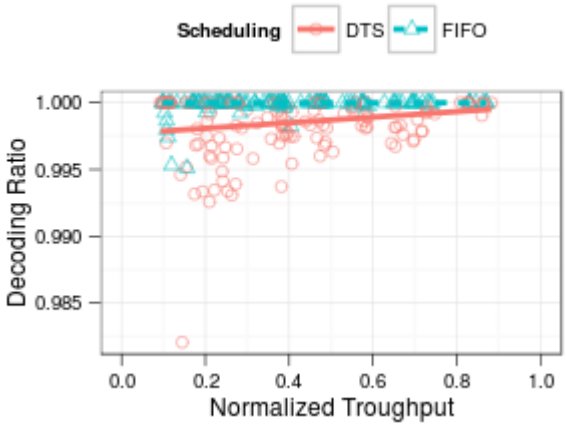

Figure 7-17 Decoding ratio against normalized throughput

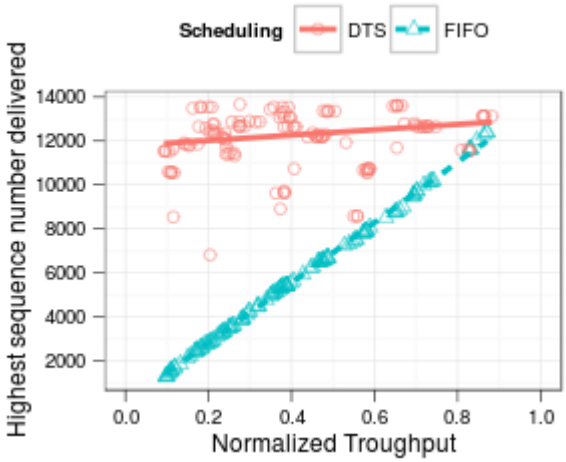

Figure 7-18 Sequence number of the last frame delivered

The increased length of video delivered with DTS comes at the cost of leaving gaps in the middle of the video. Gaps of different sizes vary the frame rate of the video, which is an undesirable effect. In addition, big gaps not only produce a big variation in quality, but also hide information to the user. For that reason, DTS aims for many small gaps. Figure 7-19 represents the probability density function calculated from the gap sizes of the DTS experiments. Most of the gaps are concentrated close to low values. If we extrapolate this to the quality perceived by a user viewing the video sequence, he will watch a video with lower frame rate than the original, but times without any frame will be short. This also represents that the variations between gap sizes are small; hence, the frame rate perceived by the user is close to constant. Since the frame rate is the parameter that DTS is adapting, a low standard deviation in gap sizes and small gap sizes produce low variations in the quality, hence, a good user experience. Nonetheless, further subjective user tests will be carried out to verify this.

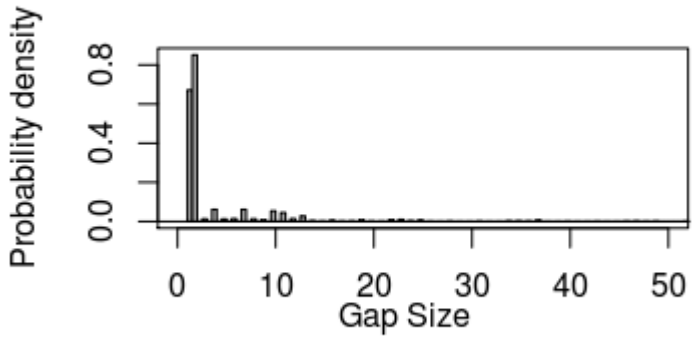

Figure 7-19 Probability density function of Gap Size distribution of DTS

Finally, we evaluate the scalability of DTS with available resources. Figure 7-20 represents the mean of the gap size distribution for each scenario compared with the normalized throughput. We observe that the trend is similar to a (Normalized Throughput)$^{-1}$ function. Furthermore, the mean is always small and may be assumed for a user. For example, a 48 frames gap is equivalent to 2 seconds without video.

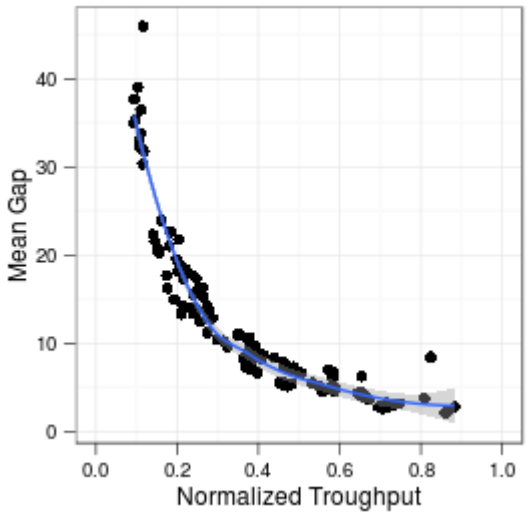

Figure 7-20 Mean gap size compared with the normalized throughput

7.7 Conclusions

This chapter explores video forwarding in MOMENTUM. We have demonstrated that a simple flow control mechanism can reduce packet loss ratio drastically, i.e. from 80% to below 10%. This evaluation has proved our initial hypothesis to cope with temporal disruptions from the overlay network. As expected, even if routing and forwarding worked well, mobility would limit video delivery. In most scenarios, being able to deliver all video packets is an exception and not a reality. Furthermore, many real scenarios present unpredictable mobility, which makes it difficult to guess future packet delivery. DTS leverages the structure of the stream to schedule frames and provides dynamic adaptation to available resources. Moreover, resource estimation is not necessary. The experiments confirm that the efficiency of DTS using network resources is close to FIFO. However, the video frame rate is adapted to available resources. The length of the video delivered to the user is closer to the original video. In addition, the variations of the frame rate in the delivered frame sequence are small, which improves user experience. Finally, DTS scales with the available network capacity. It provides a better quality when the network capacity increases. If applied in ER operations, a video delivered by DTS with enough network capacity is as useful as one delivered using FIFO. However, when the network capacity is insufficient, DTS provides a frame sequence that covers more span of the video recorded. Even if the frame rate of this sequence is very low, users could use it to detect interesting parts that they want to watch with higher quality and request them from the overlay network.

Chapter 8

Conclusions and Future Work

This Chapter describes the main conclusions from this thesis. We summarize the work done, analyze the most important insights and provide a critical analysis of them. We also propose future work in the topics investigated.

8.1 Conclusions

The main goal of this work is to investigate techniques for video transport in sparse MANETs used in ER operations. MANETs are an alternative to support communication if network infrastructure is unavailable, which makes them attractive for ER operations. Video transport in this context poses several challenges that remain unsolved, despite being an appealing application for ER personnel. In this thesis, we present original solutions to some of these challenges. First, we looked into the support of different connectivity scenarios through one system, because video source and destination can be in the same or in different partitions in a sparse MANET. These two situations must trigger different transport mechanisms, i.e., MANET streaming and delay-tolerant transport. MANET streaming is addressed in many works, but DTN video transport has been basically neglected. However, our study of mobility in ER operations shows that it is likely to occur. Hence, we focus on routing, video packet forwarding and video adaptation for delay-tolerant video transport. To investigate our solutions, we have followed a set of requirements that emerge from the ER application domain and the use of existent networking and video technologies. As a result, our proposals are easy to implement in a real deployment with off-the-shelf mobile devices.

We propose an overlay network, MOMENTUM, which is designed to transport video over sparse MANETs. The essential functionalities to cope with it are identified and grouped in components. Thus, each component assumes a set of tasks in the overlay network. This approach is useful to divide the problem and study components independently, as we have done. This thesis provides a basic design for some components, while others are studied with more detail. Thus, although it is not complete, MOMENTUM provides a good approach to build a solution based on state-of-the-art technologies. The flexibility of component-based design can be used to plugin existing solutions to particular problems. For example, adaptation to the network can be improved with proposals such as CliSuite [122] or the Bundle Protocol [22] included as part of the Transport and Multicast Routing component. One basic assumption for MOMENTUM is that it runs in all nodes of the MANET. This eased the design of the components, but it also constitutes a limitation, because it is necessary to manage all nodes in the MANET to guarantee that the overlay network works. If the network consists only of mobile devices carried by ER personnel that form the MANET, this assumption is realistic. However, if other devices, such as devices carried by victims, were included in the network, the design of some mechanisms should be revisited.

MOMENTUM fulfills the requirements of integration with existing technologies. It is compatible with off-the-shelf video applications. One interesting implication of fulfilling this requirement is that it allows the use of exiting devices, e.g. cameras. We achieve this compatibility using the Application Gateway component. We provide basic proof of this in our prototypes, supporting RTSP/RTCP/RTP servers and clients. Although standard clients are supported, they do not provide the best experience to users. For that reason, we also investigate in Appendix C client applications tailored for sparse MANETs and ER operations. MOMENTUM is also compatible with off-the-shelf networking. It uses UDP to send and receive packets. Hence, the default TCP/IP stack can be used for MANET nodes. This is a fundamental difference with other approaches, such as DtsOverlay [97], which also aims to video transport in sparse MANETs, but modifies the MAC layer.

Experiments over MANETs with different mobility and density have demonstrated that MOMENTUM is also able to adapt to MANETs and DTNs. OLSR is used as routing protocol for the MANET. MOMENTUM uses the information of OLSR to determine if a MANET route exists between video source and destination. This is used to trigger streaming over the MANET or delay-tolerant transport. This is a novel contribution over solutions like the Bundle protocol [22], which is tailored for delay-tolerant transport, or the traditional client-server streaming, which only works when end-to-end connectivity exists. The overlay network breaks the end-to-end principle by using intermediate nodes to support video delivery. This is necessary to adapt to DTNs. Ferry nodes are used in delay-tolerant transport. For streaming, we propose to introduce intermediate nodes working as relays. The aim is to increase reliability over long network routes, although we have not studied this in our experiments. Using intermediate nodes adds a small overhead in the form of delay and extra packets. Furthermore, intermediate nodes introduce additional failure points to the system. If a relay or a ferry fails, it affects video delivery. Nevertheless, they are crucial in many situations. For that reason, we focus on the use of intermediate nodes in a DTN: first, how to find them using delay-tolerant routing and, then, how to forward video between them, the source and the destination.

Delay-tolerant routing is the key to enable video delivery when the network is partitioned. We approach this problem by studying mobility in ER operations. The study of mobility scenarios either created by models or gathered from GPS devices is not trivial. Researchers find out difficult to analyze them without adequate tools. There is a lack of applications that integrate easy scenario generation, visualization and analysis for MANET research. For this purpose, we create MASS, which is described in Appendix B The knowledge acquired from mobility analysis is used to design a delay-tolerant routing protocol, called EOR. EOR aims to high video packet deliver and low delay, as well as low resource consumption to meet the given requirements. EOR is a reactive protocol integrated in MOMENTUM. EOR selects ferries considering mobility patterns in ER operations. Most delay-tolerant routing protocols use either static or probabilistic approaches. EOR combines both. It uses static parameters, such as the type of node, and dynamic metrics, such as the time elapsed since the contact between nodes. We show that, when both protocols run inside MOMENTUM, the overall performance of EOR is better than the performance of PROPHET [112] in sparse MANETs that resemble ER operations. In most experiments, EOR obtains higher packet delivery and lower delay. Therefore, the use of static parameters extracted from mobility models is useful. However, the criteria that EOR uses to select ferries are less successful when compared with other protocols inside DtsOverlay and over scenarios with different

mobility. EOR is too specific to MOMENTUM and to the mobility patterns found in our research. It performs badly when the assumptions on mobility and the architecture where it runs are wrong. Therefore, it is important to customize delay-tolerant routing protocols to the expected mobility. Nonetheless, it is also necessary to prepare them with mechanisms to react when mobility is not as expected.

In our experiments, we detect that most video packets are lost during temporal disruptions that are not detected by MOMENTUM. They occur, for example, when OLSR declares that a route exists even if it is unavailable because ARP resolutions fail. We propose to prevent and correct packet losses with a video packet forwarding mechanism based on a transmission window and group acknowledgements, called EFC. Its evaluation shows that it prevents packet losses. One key advantage of EFC over other approaches is its simplicity. EFC introduces a small overhead with the messages needed to dimension the window, but this is insignificant compared to the cost of handling video packets. Nonetheless, the packet loss ratios and delivery ratios obtained are comparable to other solutions, such as DtsOverlay. Furthermore, EFC is compatible with them. The current design of EFC implies that the receiver always gives the same amount of credit to the sender. Hence, the maximum size of the transmission window is always the same. This can lead to inefficient use of the available bandwidth, because different situations may require different transmission window sizes. The need for a mechanism like EFC demonstrates that state-of-the-art networking protocols and MANET routing protocols do not perform flawless. Since we aimed to use them, we have to deal with their problems. However, these mechanisms would be unnecessary if the protocol stack underneath the overlay network worked correctly in multihop networks. In this case, either ARP does not fail with resolutions or OLSR declares only available routes.

When video is delivered using a DTN, it is likely that the capacity of the network will not be sufficient. To tackle this problem, we study the problem of video adaptation in DTNs. Resource estimation is not possible when nodes move freely without predefined patterns. Besides, video adaptation should be possible in intermediate nodes and not be resource consuming. To the best of our knowledge, this problem has never been studied before. We propose DTS as a novel technique that adapts frame rate to the network throughput. It leverages the division of video streams in frames to achieve this. Thus, the performance of DTS is limited by the video encoding, i.e., the length of the GOP. We demonstrate that using DTS does not affect network throughput significantly, because in our experiments it obtains values similar to not using it, i.e., using FIFO. The advantage of using DTS is that it adapts the frame rate to the available network throughput, whereas FIFO cuts the length of the delivered video. However, a negative consequence is that in its current design not all the packets received result in video that can be decoded.

Low resource consumption is a goal for the design of our solutions. The overhead introduced by overlay network mechanisms, e.g. EOR or EFC, is insignificant in comparison with the demands of video traffic. In particular, EOR and EFC packets are only sent when necessary. Thus, they do not imply a constant consumption of network resources. Furthermore, we keep only one copy of each video packet in the overlay network. Although this may limit video delivery, it is the best alternative resource wise. Finally, DTS is likely to impose a lighter CPU load than using video transcoding.

8.2 Future work

This thesis contributes to the solution of relevant problems in the transport of video over sparse MANETs. However, this topic is far from being completely explored. On the one hand, MOMENTUM may be improved in several ways. There are some open problems that we have not studied deeply and it is worth to solve them. On the other hand, there are hints for future work research topic. In this section, we propose some worth studying issues.

Overlay network membership management is one of them. For the current design, we assume that all nodes are part of the overlay. This simplifies the design of the overlay network and provides maximum resource availability. However, this may not be always feasible, nor interesting. In some situations, it may be convenient to extend the MANET with nodes that are not under the control of ER personnel; for example, a security camera previously deployed in the area or the mobile phone of a victim. It is unlikely that these devices would have MOMENTUM installed. Therefore, it is worth to, first, study the consequences of adding them to the MANET and, then, redefine the overlay network mechanisms to support their presence. In addition, we have considered all network nodes equal and with enough available resources. The truth in a real deployment is very different. Devices are likely to be heterogeneous and have different levels of battery or available storage. This is not considered in our current proposals, but we are aware that this can highly influence their performance. For example, choosing a ferry node with a low battery level may not be the best choice. Considering that there are nodes not part of the overlay and that nodes in the overlay have limited resources may be studied together, because both problems impose a limitation on resource availability. Related work, such as SMON [100], can be useful in this task.

Signaling is another important part of video transport. MOMENTUM uses an extension of RTSP, which provides the minimum functionality required. Nonetheless, signaling in a real deployment should be more tailored to the environment. One key aspect of a signaling protocol is that it should express in the network the requests made by users to the client applications. In addition, it must support disconnections, disruptions as well as the participation of intermediate nodes. In the DT-Stream, we have done some preliminary research in this issue, [93] and [92], but none of these protocols have been tested in MOMENTUM. Furthermore, research on signaling is complemented with research in client applications, like [94] or ERPlayer in Appendix C

Security related issues are also important in a real deployment, but we have not considered them in the current version of MOMENTUM. The challenges of security in MANETs are well-known [123], even in the ER application domain [124]. An interesting challenge is how to limit the access to video streams. While any node may be used to transport video, not all of them should be allowed access to it. Video stream decoding can be limited to a node or a group of nodes that have the rights to watch it.

The proliferation of GPS devices makes mobility traces more accessible. This has a huge impact on the study of delay-tolerant routing. Our proposal, EOR, uses knowledge extracted from mobility in ER operations. Nonetheless, only a few scenarios from ER operations could be obtained. New research possibilities open up when there are enough mobility traces. Appendix A contains an analysis of GPS traces of the vehicles from an emergency service. Network science and Social Network Analysis techniques are used to understand mobility and the relationship between network nodes. This knowledge is

appealing for delay-tolerant routing. Nodes can be trained to recognize connectivity patterns and to leverage them for video transport. In this task, machine learning algorithms could be used in several ways. On the one hand, mobility traces could be used to find the best routing strategies, which could be afterwards implemented in nodes. The successes and failures of these strategies could be analyzed to fine-tune them. On the other hand, machine learning could be implemented in the nodes, so they guess these strategies online. This could be done individually by each node or collaboratively. If mobility presents patterns, these ideas could lead to find fast and reliable delay-tolerant routes.

We have proposed DTS as a solution for video adaptation in DTNs. Its results are promising, but it needs further research. DTS adapts the video frame rate to the network throughput when video packets are carried by one ferry. Now, it is necessary to investigate what happens when multiple ferries consecutive and in parallel are used and see if the video delivered preserves the same properties. In addition, DTS performance depends on video coding. It is necessary to investigate different GOP structures and also different codecs. It could be interesting to use DTS with layered codecs, such as MDC or SVC.

Video streaming over sparse MANETs is still some steps away from being a reality. Although the path is not completed, this thesis makes necessary contributions to walk it. The architecture of MOMENTUM can provide all the essential functionalities for video streaming in connected and partitioned MANETs. It also integrates nicely with off-the-shelf devices, networking and technologies, which could shorten the time to achieve a completely functional system. Moreover, the idea of using “a priori” knowledge from ER scenarios for routing has shown part of its potential. However, the study of delay-tolerant routing protocols also suggests that real mobility traces are essential to make further advances in this area. A new concept introduced by this thesis is video adaptation over delay-tolerant networks. The problem of not been able to estimate future available resources to carry out adaptation is present in DTNs, but it may be generalized to other environments where the network behaves in unpredictable ways and video quality has to be maximized. Finally, our proposals use the context of ER operations, but many of them can apply to other application domains as well. Hence we believe that this thesis provides valuable knowledge for future research and system development in this topic.

Chapter 9

Conclusiones

Este capítulo describe las principales conclusiones de esta tesis. En primer lugar, se resume el trabajo realizado, analizando las aportaciones más importantes. También, se hace un análisis crítico de las mismas.

El objetivo principal del trabajo realizado es investigar técnicas para el transporte de vídeo en MANETs (Mobile Ad-hoc Networks) poco densas empleadas en emergencias. Las MANETs son una buena alternativa de comunicación cuando la infraestructura de red convencional no está disponible. Por esta razón, su uso es interesante en emergencias. El transporte de vídeo en estas circunstancias presenta numerosos retos aún no resueltos, en esta tesis se proponen soluciones originales a algunos de ellos.

Puesto que en una MANET los dispositivos de origen y destino del vídeo pueden estar en la misma partición de red o en particiones distintas; se busca un sistema que soporte escenarios con tipos de conectividad. Estas dos situaciones deben utilizar mecanismos distintos para el transporte de vídeo, en concreto, streaming sobre MANETs o transporte sobre DTNs (Delay-Tolerant Networks). El vídeo streaming sobre MANETs es una aplicación que ha sido objeto de varias investigaciones en el pasado; sin embargo, el transporte de vídeo sobre DTNs es un campo en el que apenas hay aportaciones. Esto es así pese a que los estudios realizados sobre movilidad indican que el paradigma DTN existiría en una emergencia. Por ello gran parte de esta tesis se dedica a estudiar los mecanismos de routing, forwarding y adaptación de vídeo DTNs.

Para proponer soluciones a estos problemas, se ha seguido un conjunto de requisitos que surgen de los escenarios de emergencia y de la motivación por construir soluciones compatibles con las tecnologías de red y de vídeo actuales. La consecuencia de esto son propuestas fácilmente aplicables en un entorno real con dispositivos móviles disponibles en el mercado actualmente.

La solución propuesta en esta tesis consiste en una overlay network (red superpuesta), MOMENTUM, diseñada para transportar vídeo en MANETs poco densas. Sus funcionalidades esenciales están agrupadas en componentes, de tal forma que cada componente asume una serie de tareas relacionadas con problemas concretos. Este diseño ha sido útil para dividir el problema y estudiar los componentes de forma independiente. En esta tesis se ha presentado un diseño básico de todos los componentes de MOMENTUM; profundizando en aquellos relacionados con los problemas a estudiar. Por tanto, aunque no sea un sistema completo, MOMENTUM propone una solución factible basada en tecnologías estándar. Además, la flexibilidad del diseño por componentes permite incorporar soluciones a problemas particulares. Por ejemplo, la adaptación a la red puede ser mejorada incluyendo en el componente Transport and Multicast Routing las propuestas de CliSuite [122] o el Bundle Protocol [22].

Asimismo, MOMENTUM asume que todos los nodos de la MANET forman parte de la overlay network. Esta premisa ha facilitado el diseño de mecanismos, que de otro modo serían excesivamente complejos. Sin embargo, también constituye una limitación. Si la red consistiese únicamente en dispositivos de los servicios de emergencia, sería realista pensar que todos forman parte de la overlay network. Si no fuese así, el diseño de los mecanismos propuestos debería variar.

MOMENTUM cumple los requisitos establecidos de integración con tecnologías existentes. Es compatible con aplicaciones de vídeo estándar. Esto implica que podría usarse con dispositivos existentes, como cámaras de vídeo. El componente Application Gateway está pensado para proporcionar esta compatibilidad. En este sentido, esta tesis ha demostrado como integrar MOMENTUM con aplicaciones estándar RTSP/RTP/RTCP. Sin embargo, a pesar de que sea posible utilizar reproductores de vídeo estándar; la experiencia de los usuarios no es la idónea. Por ello, el Apéndice C describe una propuesta y un prototipo de un reproductor específico para este tipo de escenarios. MOMENTUM también es compatible con tecnologías de red estándar. Usa UDP para enviar y recibir paquetes. Por tanto, los nodos de la MANET pueden utilizar la pila de protocolos TCP/IP. Esto constituye una diferencia fundamental con otras soluciones, como DtsOverlay [97], que también tienen el objetivo de enviar vídeo sobre la MANET, pero que modifica la capa MAC.

Los experimentos sobre MANETs con distinta movilidad y densidad de nodos han demostrado que MOMENTUM puede adaptarse a los entornos MANET y DTN. Para conseguir esta adaptación, se ha usado OLSR como protocolo de routing en la MANET. MOMENTUM usa su información para determinar si es necesario realizar streaming sobre la MANET o transporte tolerante a retardos. Esto es una contribución novedosa con respecto a soluciones como el Bundle Protocol [22], que está exclusivamente orientado a transporte tolerante a retardos; o con respecto a las arquitecturas cliente-servidor, que sólo funcionan cuando existe una ruta de red entre los nodos origen y destino. La overlay network rompe el principio de comunicación end-to-end utilizando nodos intermedios. Por un lado, esto es necesario para adaptarse a las DTNs. Los nodos de tipo ferry se utilizan para conseguir transporte tolerante a retardos. Por otro lado, los nodos intermedios de tipo relay pueden aumentar la fiabilidad en el streaming de vídeo sobre rutas de varios saltos en la MANET. Añadir nodos intermedios añade un pequeño coste adicional en la comunicación, por ejemplo algunos mensajes extra o mayor retardo. Además, los nodos intermedios pueden introducir nuevos puntos de fallo en el sistema. Si un ferry o un relay fallan, afecta la recepción de vídeo. De todas formas, su uso es esencial en muchas situaciones, especialmente para el transporte tolerante a retardos. Por esa razón, se ha investigado el papel de los nodos intermedios en DTN (nodos ferry): primero, cómo encontrarlos usando técnicas de routing tolerante a retardos y, después, como reenviar paquetes de vídeo entre nodos ferry, el origen y el destino.

El routing tolerante a retardos es clave para la transmisión de vídeo cuando existen particiones en la red. El primer paso para estudiar este problema ha sido analizar la movilidad en escenarios de emergencia; tarea compleja, tanto si los escenarios son producto de modelos, como si son trazas provenientes de dispositivos GPS. No existen aplicaciones adecuadas para realizar esta tarea; por lo que, en el Apéndice B se propone una herramienta, llamada MASS, con este fin.

El conocimiento adquirido mediante el análisis de escenarios ha sido utilizado para diseñar un protocolo de routing tolerante a retardos, llamado EOR. El objetivo de este

protocolo es maximizar el número de paquetes de vídeo entregados, buscando un retardo mínimo y un bajo consumo de recursos en la red. EOR en un protocolo reactivo y se integra dentro de MOMENTUM. Selecciona nodos ferry considerando los patrones de movimiento observados en emergencias. Mientras que la mayoría de los protocolos de routing tolerante a retardos utilizan una aproximación estática o probabilística; EOR combina ambas. Por un lado, usa parámetros estáticos, como el tipo de nodo. Por el otro, calcula métricas dinámicas, como el tiempo transcurrido entre contactos. Los resultados demuestran que EOR es capaz de mejorar a PROPHET [112], ejecutándose ambos en MOMENTUM y en escenarios inspirados en emergencias. En la mayoría de los experimentos realizados, EOR entrega más vídeo y con un menor retardo. Por tanto, se observa un beneficio en el uso de parámetros estáticos frente a una aproximación meramente probabilística. Sin embargo, los criterios utilizados por EOR para elegir nodos ferry presentan problemas en ciertas circunstancias. Así ha sido reflejado en os experimentos realizados incorporando EOR en DtsOverlay. El diseño de EOR está demasiado ligado a los principios de MOMENTUM y a los patrones de movilidad surgidos de nuestra investigación. La consecuencia es que EOR obtiene un rendimiento peor que otros protocolos cuando no se cumplen las premisas para las que fue diseñado. Por esta razón, es importante que el diseño de los protocolos de routing tolerantes a retardos se ajuste a la movilidad esperada en el escenario en el que se quieran utilizar; pero que también incorporen mecanismos capaces de corregir desviaciones en los comportamientos esperados.

En los experimentos realizados con EOR y PROPHET, se ha detectado que la mayoría de los paquetes de vídeo se pierden durante desconexiones que no son detectadas por MOMENTUM. Estas pérdidas ocurren principalmente cuando OLSR declara una ruta, pero el protocolo ARP falla a la hora de resolver las direcciones. Para prevenir y corregir la perdida de paquetes debido a este problema, esta tesis propone un mecanismo, llamado EFC, basado en una ventana de transmisión y asentimientos grupales. EFC consigue una importante reducción del número de paquetes perdidos. Una gran ventaja de esta propuesta sobre otras es su sencillez. A pesar de que EFC aumenta el consumo de recursos, lo hace de manera despreciable en comparación con el tráfico de vídeo. Asimismo, los resultados obtenidos son comparables a las de otras soluciones, por ejemplo las de DtsOvelay, pero sin necesidad de modificar otros protocolos red. Además, los mecanismos de EFC pueden ser complementarios a estas otras soluciones propuestas. Un problema existente en el diseño actual de EFC es que utiliza una cantidad fija de crédito concedido por el receptor para que el transmisor aumente su ventana. Este diseño no es óptimo; ya que el crédito concedido debería variar según la fiabilidad de la red, para que EFC utilice el ancho de banda de forma más eficiente. También es importante señalar, que EFC es una solución a problemas de sincronización o rendimiento de los protocolos estándar. Un mecanismo de este estilo no sería necesario en un entorno en el que los protocolos estén específicamente diseñados y bien integrados para soportar MANETs.

Cuando el vídeo se transmite sobre una DTN, es muy probable que la capacidad de la red sea insuficiente para entregar todo el vídeo. Para abordar este problema se ha estudiado la adaptación de vídeo en estas circunstancias. Realizar una estimación precisa de los recursos de la red es muy complicado, o imposible, porque los nodos de la red se mueven libremente. A este problema se une el hecho de que, idealmente, los nodos intermedios deben ser capaces de realizar el proceso de adaptación. Con estas limitaciones en mente, se ha propuesto una nueva técnica, DTS, que adapta el número de frames por segundo del vídeo recibido a la capacidad disponible en la red. El

rendimiento de DTS está limitado por las técnicas de codificación; ya que se aprovecha de la división del vídeo stream en frames que el codificador realiza. En esta tesis se ha comparado DTS con la manera estándar de enviar el vídeo: con los frames ordenados según son generados (FIFO). La evaluación realizada demuestra que el uso de DTS tiene un impacto muy bajo en la cantidad de vídeo que se recibe. Su gran ventaja es que el vídeo recibido tiene una longitud mayor que utilizando FIFO, pero el número de frames por segundo es proporcional a la capacidad de la red. Una consecuencia negativa del uso de DTS es que no todos los paquetes recibidos pueden ser decodificados. Aunque el número es pequeño, se trabaja en futuros diseños que corrijan este problema.

Mantener un bajo consumo de recursos ha sido un objetivo durante el diseño de todos los mecanismos. El coste introducido por MOMENTUM es insignificante comparado con el tráfico de vídeo. Además, sólo se consumen recursos cuando se envía vídeo. Otra medida en este sentido ha sido mantener sólo una copia de cada paquete de vídeo en toda la MANET. Aunque esto puede limitar las posibilidades de que el vídeo sea recibido, es la mejor alternativa desde el punto de vista de los recursos. Por último, dada la sencillez en el diseño de DTS, puede afirmarse que su consumo de recursos será menor que el de otras técnicas de adaptación que, por ejemplo, usen recodificación.

El vídeo streaming sobre redes MANET poco densas no es aún una realidad consolidada. Sin embargo, aunque el camino no esté completo, esta tesis realiza contribuciones necesarias para andarlo. La arquitectura de MOMENTUM proporciona las funcionalidades esenciales para el video streaming en MANETs conectadas y particionadas. Además, se integra con los dispositivos y tecnologías existentes, lo que podría acortar el tiempo necesario para implementar una solución en entornos reales. Asimismo, la idea de utilizar el conocimiento “a priori” de los escenarios en los protocolos de routing ha mostrado parte de su potencial. También se ha comprobado como en el campo de los protocolos tolerantes a retardos es esencial utilizar movilidad de entornos reales para poder realizar avances significativos. Por otro lado, esta tesis también ha introducido el concepto de adaptación de vídeo en DTNs. La técnica propuesta podría ser usada en otros entornos donde la estimación de recursos no es posible, pero es necesario maximizar la calidad del vídeo. Por último, la tesis se enmarca en el contexto de las emergencias, pero se espera que el conocimiento aportado sea válido en otros entornos, como por ejemplo redes formadas por teléfonos móviles convencionales. Por todo ello, se espera las aportaciones realizadas en esta tesis puedan servir investigaciones futuras.

Appendix A

Mobility in emergencies

This appendix describes the analysis of mobility traces form 112 Asturias / Bomberos de Asturias. These traces belong mainly to vehicles, what makes difficult to extract scenarios of ER operations. However, we acquired knowledge from this analysis that is interesting in the context of this thesis. We have used Network Science to analyze one year of real mobility traces. The results show implications for the design of the network from the physical to the application layer. An important finding is that the network is often sparse and partitioned, but that delay-tolerant routes connecting these partitions exist. Moreover, there are patterns in the connection between nodes that can ease the discovery of these routes and the deployment of delay-tolerant services.

A.1 Introduction

Communication is a critical tool for emergency services. It is used for coordination, information gathering and alerting population. However, communication infrastructure is not present everywhere and it is likely to be destroyed or overloaded during natural disasters. For that reason, Mobile Ad-hoc Networks (MANETs) emerge as an attractive alternative, especially when infrastructure is unavailable. MANETs are deployed using wireless protocols, such as 802.11. Devices, also known as nodes, establish network links with others in their communications range. They act as hosts, but also forward traffic on behalf of others. They use opportunities given by their location to communicate with other nodes. Thus, these networks are also called opportunistic networks. Understanding node mobility is crucial to engineer solutions for disaster relief. The research community is aware of this fact and has worked on it. Mobility models have been proposed, for example in the area of tactical networks [70], which includes emergencies. However, researchers also confront the scarcity of real mobility traces to support these models. Fortunately, the trend is changing and more mobility data is becoming publicly available in different application domains. However, this process is being slower in disaster relief, because emergency services are logically reluctant to open this information to the public for security and privacy reasons. On the other hand, many emergency services are registering it in Geographical Information Systems for their private use. It is in the interest of both the emergency services and the research community to make it available for analysis.

This paper describes the analysis of one year of mobility traces from an emergency service. We have worked together with the regional fire department of Asturias (Spain) to analyze their mobility without compromising security and privacy. The goal is to understand how a hypothetical opportunistic network would work. We have designed an analysis method based on Network Science [125], which is used to deal with complex networks. Network Science principles have been successfully applied to real mobility for protocol design (Hui et al., 2011). However, as far as we know, nobody has tackled

the analysis of such a big dataset of mobility in the context of emergencies. Our results reveal several insights that can be used for protocol design. The most relevant outcome is that the network would support delay-tolerant services, but it would be unlikely to perform well with real time services. In addition, we have discovered patterns in how nodes contact and how these patterns may be used for delay-tolerant routing.

The remainder of the paper is organized as follows. Next, we describe our method to process and analyze mobility traces in order to extract relevant information. Then, we present the most relevant results from applying our method to analyze mobility. After, we discuss the implications of the most relevant outcomes. Finally, we formulate some conclusions and potential future work.

A.2 Mobility analysis method

The original data source is a database containing traces from GPS devices embedded in a few firefighter personal radios and all their vehicles, which include cars, trucks and helicopters. First, we extracted one year of these traces. They contain positions of 228 devices. To have a better understanding of disaster relief mobility, we extracted the mobility from the biggest incident occurred in that year. It was a wildfire that burned 7.35 Km^2 and lasted 5 days. Wildfires are very common incidents, so it is a representative case study. This subset contains positions of 95 devices. For brevity, we use ES to refer to all the mobility traces and WF to refer to the wildfire subset.

The second step was to build network links from mobility traces. For that purpose, we made several assumptions. First, we assume that there is one network node associated with each GPS device. Second, we assume that the position of a network node is the last position registered by the GPS device. When a new position is registered, a node moves instantaneously to it. This is realistic if positions are registered often and avoids making other assumptions about how nodes move from one registered position to the next. The last assumption is that nodes have a communication range of 50 meters. This distance is in the order of the range obtained by 802.11 protocols. Thus, two nodes closer than 50 meters have a network link between them. Then, we processed mobility traces from their beginning to their end with a resolution of 1 second to discover when links between nodes are established and broken. Once we knew the status of links in every second, we built two lists of contacts, one for ES and one for WF. A contact is an uninterrupted period of time while the link between two nodes is established. Thus, a contact is defined with two nodes, the time when they connect and the time they disconnect. Each list can contain several contacts for each pair of nodes. Using the contact lists, we measured the duration of each individual contact, the aggregated duration of all the contacts between two nodes and the times elapsed between consecutive contacts, i.e. link disruptions. However, further analysis is necessary to understand the network better.

The third step was to build network topology from the contact lists, but topology is dynamic and can be analyzed from different point of views. On the one hand, network topology can be represented with the status of network links in a given second. This is interesting to analyze the network behavior in real time. On the other hand, network topology can be also analyzed using a time interval longer than 1 second. For example, all the contacts in an hour can be used to build a single network in which a link between two nodes exists if there is at least one contact during this hour. This provides the

perspective of the network from the point of view of delay-tolerant networking. As we will demonstrate, this analysis leads to more interesting results in our case. To build different representations of the network topology according to a time interval, the lists of contacts are divided into not overlapping sublists of the length of the interval. Each sublist of contacts is then used to build a graph that represents the network topology. We select time intervals to generate the following situations:

ES: Contains one network topology using all the contacts in the year of the ES contact list.

ES-86400: Contains 366 network topologies, one for each day (86400 seconds) or portion of day.

ES-3600: Contains one network for hour (3600 seconds) or portion of hour in the year.

WF: Contains one network using all the contacts in the 5 days of the WF contact list.

WF-3600: Contains one network for each hour or portion of hour in the 5 days of the WF contact list.

WF-60: Contains one network for each minute or portion of minute in the 5 days of the WF contact list.

These time intervals have been selected to cover different possible delay-tolerant situations. For example, WF-3600 is useful to analyze services in the wildfire that supports one hour of delay. Network Science metrics are used to analyze the topologies of these case studies. For each of them, we look into partitions, routes, the clustering coefficient and different centrality metrics.

A.3 Results

A.3.1 Contacts

This subsection presents results from analyzing the contacts lists of WF and ES. First, we focus on describing contact duration, aggregated contact duration and disruption duration. Figure 1 contains the Probability Density Function of these metrics. The x-axis is the duration in seconds and in a logarithmic scale. To ease readability, we also added markers in the axis that are equivalent to long periods of time in seconds.

In total, there are more than 2 million contacts between nodes in ES and more than 15 thousand in WF. These contacts take place between 14,904 pairs of nodes in ES, which is approximately a 28% of the total possible pairs, and 919 pairs (10%) in WF. The distribution of **contact duration** look similar for both, as it can be observed in Figure 1. Of course, ES lasts for 1 year and WF only for 5 days; therefore, ES presents some long contacts that are impossible in WF. For that reason, the mean contact duration is 1.08 hours for ES and 34.2 minutes for WF. However, their medians are 120 seconds in ES and 150 seconds in WF. In addition, 75% of the samples for ES are smaller than 571 seconds in ES and 450 seconds in WF. The similarity between both distributions is unexpected being WF a significantly shorter subset of ES.

Durations of contacts between nodes can be added in an **aggregated contact duration** for each pair of nodes. These values represent the total time that two nodes are

connected in 1 year for ES or 5 days for WF. As it is showed in Figure 1, the range of values is wide in both cases. There are pairs of nodes that connect for a longer time than others. Eventually, nodes that connect longer can exchange more data. Thus, the aggregated contact duration is closely related to the potential network capacity that could be obtained from the opportunistic network. The mean aggregated contact duration is 7.66 days for ES and 10.95 hours for WF.

A disruption is the period of time between two consecutive contacts of the same pair of nodes. **Disruption duration** is an interesting metric from the point of view of network delay. Long disruptions increase the delay of packet delivery if there are not alternative paths between the nodes. On the contrary, if disruptions are known to be shorter than the admissible delay of a packet, the best delivery policy will be waiting for the next contact instead of looking for alternative paths. Again, the distributions for ES and WF (Figure 1) look surprisingly similar. The medians are also similar, 60 seconds in WF and 89 seconds in ES. However, the means are very different 59.58 minutes in WF and 15.42 hours in ES. This is due to the fact that ES has a much longer tail and with disruptions longer than 350 days (more than 3 million seconds). These extremely long disruptions heavily increase the mean value for ES and also reveal the fact that some nodes contact infrequently.

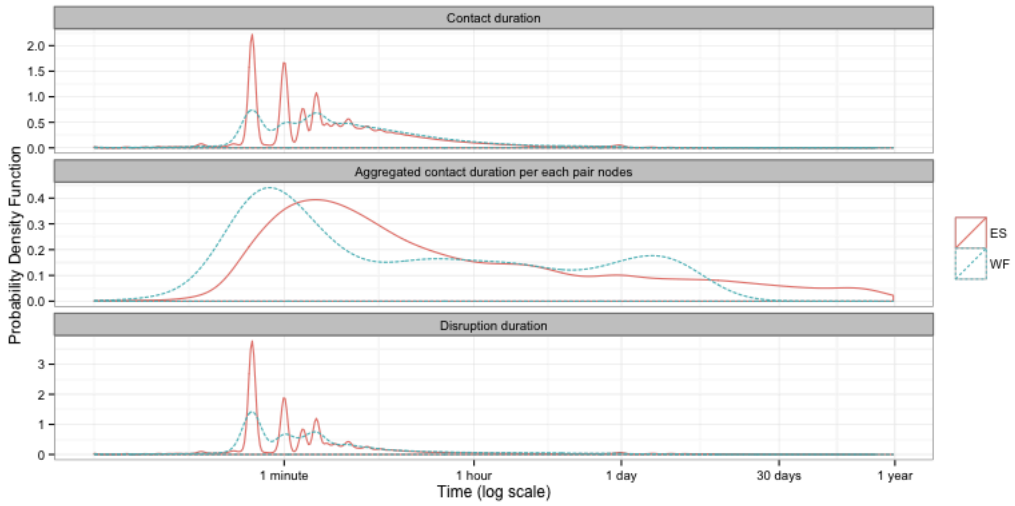

Figure A-1 PDF of contact duration, aggregated contact duration and disruption duration

The **frequency of contact** for a pair of nodes can be calculated dividing the aggregated contact duration by the total time analyzed. For example, if one minute of mobility is analyzed and two nodes are connected for 30 seconds, the frequency of contact is 0.5. This is a good indicator of possible contact patterns, e.g. the frequency of contacts between two nodes is the same every day. If patterns existed, it would be useful to apply them in delay-tolerant routing policies. For that reason, we tried to find out if it would be possible to predict future frequency of contact from past frequency of contact. Lets define a variable *t* that takes as value all the seconds in our traces. Then, for each *t* we calculate the frequency of contact before *t* (fcb_t) and the frequency of contact after *t* (fca_t). So, for a pair of nodes fcb_t is the aggregated contact duration before *t* divided by the seconds elapsed from the beginning of the traces to *t*. On the other hand, fca_t is the aggregated contact duration after *t* divided by the seconds left from *t* to the end of the traces. We now suppose that every time a contact ends, the nodes predict that fca_t is

equal to fcb_t. This is a simple prediction that can be implemented in real systems. Since we know fca_t, the error of making these predictions in our traces is fca_t - fcb_t. Figure 2 shows the Probability Density Function for the prediction errors in ES and WF. The means of the errors are -0.03 in ES and 0.05 in WF and their standard deviations are 0.28 in ES and 0.26 in WF. Thus, most errors are in values around 0, so they are small errors that may be assumable in a real system. These results indicate that repeating patterns may be present in the contacts.

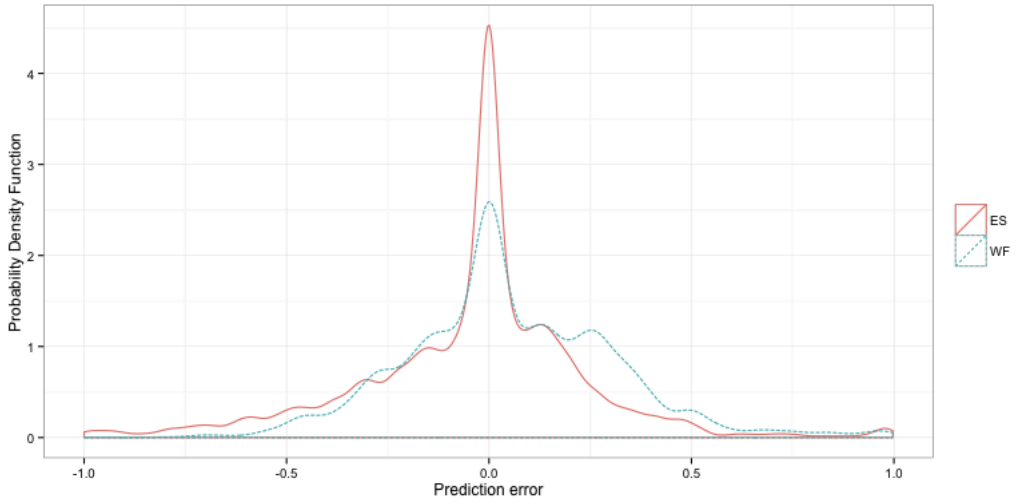

Figure A-2 Frequency of contact prediction error

A.3.2 Network topology

This section analyzes network topologies obtained from the contact lists using network science metrics. First, we present results from analyzing network partitions. Second, we describe shortest paths between nodes, which are relevant for network routing. Then, we discuss about more complex properties of the network structure, i.e. we look into the clustering coefficient and centrality metrics. We have summarized the average value of these metrics in Table A-1. To obtain each value, metrics are averaged for all the values obtained for each time interval analyzed.

	ES	WF	ES-86400	ES-3600	WF-3600	WF-60
Number of partitions	1	1	4.64	10.25	9.16	2.5
Size of partitions (nodes)	228	95	33	6.08	4.86	2.72
Shortest path length (hops)	1.45	1.99	3.33	2.38	2.08	1.42
Clustering Coefficient	0.70	0.43	0.40	0.28	0.42	0.046
Degree Centrality	125.22	18.56	6.78	2.51	2.30	1.30

Betweenness Centrality	51.40	48.30	154.49	7.80	3.60	0.47
Closeness Centrality	0.003	0.005	0.00006	0.00005	0.00007	0.08

Table A-1 – Summary of network metrics averages

Network partitions are groups of isolated nodes that can only communicate among them. Partitions are important from the networking point of view, because they give an idea if the nodes in the network are connected and how. Two nodes are not able to exchange information if they are in different partitions. We have measured the number and size of partitions in the network topologies built from each time interval study. The general trend is that the number of partitions decreases with the length of the interval used to build the networks. This is sound with the expectations, because, in a longer period of time, nodes are more likely to contact others, build new links and eventually merge in fewer (and bigger) partitions. However, this does not hold for WF-60 that presents a low average of partitions. The explanation is in the definition of partition and the number of nodes in each partition, i.e. partition size. Opposite to the number of partitions, the average partition size increases with the time interval used. In WF-60, the average scenario is a few small partitions, although partitions up to 15 nodes have also been found. Since partitions have a minimum size of 2 nodes, this implies that there are a lot of isolated nodes totally disconnected from others. This implies that in real time the network is sparse.

The **shortest path** between two nodes in a network is a path that connects them traversing fewer nodes and links. Obviously, according to this definition there may be more than one shortest path, but this is not relevant for our current analysis that focuses on the length of them. In a communications network, data packets undergo one retransmission for each node and link they traverse. Hence, a resource saving routing strategy could select shortest paths as the most resource efficient ones. In our analysis, shortest path lengths are similar in average. Long routes are infrequent. The longest route found is 16 hops for ES-86400, while for WF-3600 is 9 hops. There is an interesting effect in the average length of paths (see Table 1) when the analysis time interval grows. Using 60 seconds, there are few links between nodes, which is a logical context for short paths. As the interval grows, nodes link with others that were not in their partition and some paths also grow. In our analysis, this happens in ES-3600, WF-3600 and ES-86400. However, when the interval grows even more, the likeliness of linking to a node already in the partition increases. Therefore, new links occur inside the partition, which creates shorter paths, as it happens in ES and WF.

The **clustering coefficient**, or transitivity, of a network is the probability that two nodes connected to a third one (its neighbors) also have a connection between they two. It is a measure of the connectivity between neighbors: a high clustering coefficient indicates that the neighbors of a node are very likely to be connected between them. This metric gives a basic idea about network density and about possible clusters that are highly interconnected. Clusters are interesting because they can be leveraged in the design of distributed systems running over the network. In addition, the clustering coefficient also gives an idea of how strongly connected is the network, which is important for network resilience as we discuss in the next section. The clustering coefficient increases with the time interval used for the network building. For ES, the value is noticeably high: in the

70% of the situations, two nodes connected to another will be also connected between them.

Centrality metrics measure the importance or popularity of nodes in the network. There are several centrality metrics, but we consider three that are interesting to analyze ad-hoc networks according to [126]: degree, betweenness and closeness. **Degree centrality** is the number of links of a node. The average degree increases with the analyzed time interval. As it happens with the clustering coefficient, the average degree found in ES is significantly higher. This means that some nodes in the network do not connect in a daily basis, but in longer periods, e.g. once a week. These links are not observed in the daily analysis (ES-86400), but they are present in ES. Popular nodes are referred to as hubs. In our observations, there are hubs with a relatively high degree, e.g. 221 in ES. This is easily observed in Figure 3 that shows the Cumulative Density Function (CDF) of the normalized degrees found in our networks. Since ES and WF have different number of nodes, we have normalized the degree to the numbers of nodes in each network to get comparable values. **Betweenness centrality** is the number of shortest paths that traverse a node. This is important to identify bottlenecks in the network. A node with high betweenness is likely to forward more packets on behalf of others and, as a consequence, may become congested. The observed average betweenness is relatively low in most cases. **Closeness centrality** is the inverse of the distance from a node to the rest of the nodes in the network. This metric is relevant for information dissemination in a network. If we wanted to send a message to all nodes in a partition, the most efficient way would be to send it from the node with the highest closeness. The most interesting result is for WF and ES, where the average of the observations are relatively high. This means that most nodes are close to each other in the long term and that there would be many candidates to disseminate information.

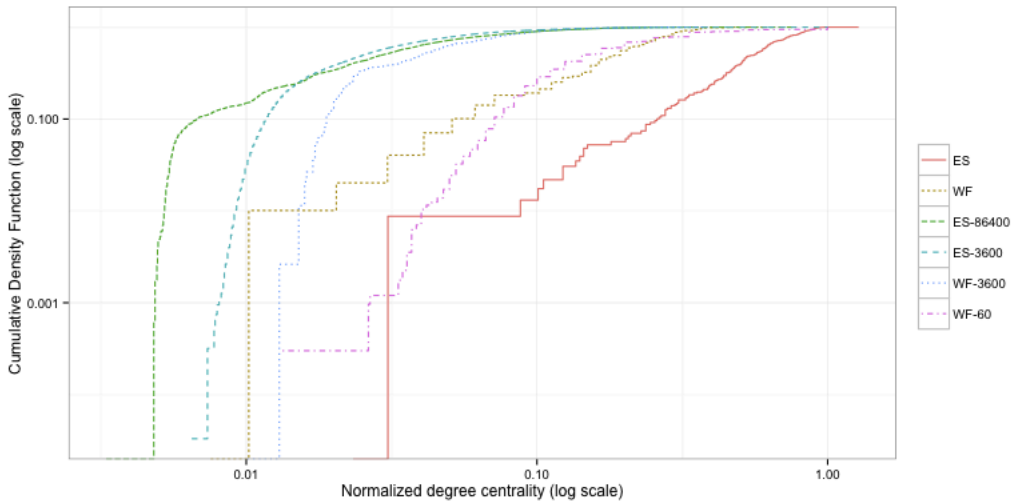

Figure A-3 Normalized degree Cumulative Density Function (CDF) with logarithmic axis

A.4 Discussion

Based on the previous results, this section discusses implications for the design and deployment of an opportunistic network from the physical to the application layer. Relevant to the lower layers is the stability shown by contacts. Fast connections and

disconnections of nodes are infrequent. Therefore, there is time enough for state-of-the-art link layer protocols, e.g. 802.11, to manage links between nodes. If contacts were too short, these protocols would have to support very fast link establishment and soon link break detection. Shared medium is also problematic in wireless ad-hoc networks. There are two main issues: collisions produced by too many nodes sharing the medium and the well-known hidden node problem. These issues are only relevant if nodes compete for the medium at the same time, so results from WF-60 are the most relevant. The analysis of partitions and routes revealed that in real time the network would be sparse, with many isolated nodes, 2 or 3 nodes partitions and short multihop routes. Therefore, the network is unlikely to suffer heavily from any shared medium problem.

Although the network is sparse in real time, nodes connect when looking at longer time intervals. This demonstrates a common assumption when designing systems for MANETs in emergencies: the network is sparse and partitioned, but there are nodes that can be used as data ferries or data mules. The store-carry-forward paradigm [19] can then be applied to design delay-tolerant services. However, the problem is how to find ferry nodes. Delay-tolerant routing protocols aim to solve this task, being PROPHET [112] the most representative. PROPHET uses past contacts to predict future contacts in a similar way we have built our prediction. According to our results, PROPHET-like protocols would be adequate for this application domain. To increase packet delivery probability and overcome prediction errors, several ferries may be used. This is possible in this network because the analysis showed that the network is well connected and several routes between nodes are likely. On the contrary, a sparse network is not adequate for real time services. Nonetheless, a MANET routing protocol is necessary to leverage multihop routes in real time. These protocols are sometimes criticized due to their bad scalability. However, this would not be a problem in this case, because multihop routes are likely to be short and the number of nodes in the partition low, which implies a small routing table. Short routes positively affect communications reliability as well. It is well known that the probability of losing a packet increases with the length of the route used to send it.

The centrality metrics indicate that hubs are common in the network and that nodes are highly interconnected, especially in ES and WF. There are many popular nodes, which is positive for network resilience. If a node failed, alternative nodes would be able to rebuild connectivity. These results are also relevant for network congestion as bottlenecks are unlikely according to observed betweenness. Finally, a problem in MANETs is the underperformance of TCP. Basically, TCP assumes disruptions as congestion [7]. Thus, if disruptions are frequent, TCP obtains a very low throughput. In our analysis we have observed short contacts, but also very long ones, in which it should be possible to use TCP. Therefore, TCP use may be reconsidered in some situations.

A.5 Conclusions

Our mobility traces have revealed important properties of a hypothetical opportunistic network for an emergency service. We have also showed the importance of analyzing the network topology using different time intervals, as each of them reveals different properties. Mobility showed that delay-tolerant routes exist, although the network is sparse in real time. This closes the door for deploying critical services in real time, but opens it to the design of delay-tolerant services for both disaster relief and every day use

of the emergency services. Moreover, the results obtained in the prediction of contacts are an optimistic scenario to evolve delay-tolerant routing protocols. Network topology properties, like the existence of hubs or the low average betweenness, are useful to define priorities in the design of novel protocols at different layers. For this purpose, it is also important to understand that the network is heterogeneous, e.g. there are very short and very long contacts. Therefore, solutions can be fine-tuned to the most likely situation, but they must also adapt to less common scenarios. In addition, ES and WF, have similar characteristics in many of the observations. This was unexpected, especially in contact duration distributions, because WF is a small subset of ES. This raises some interesting questions about the similarities between specific disaster relief mobility and everyday mobility. In future work, it would be interesting to extract more disaster scenarios and check if these similarities are present and find the reasons behind them. Although we study a specific case, emergency services of similar size and with similar tasks may also have similar properties. Future work in this area must be encouraged to obtain a wider view of mobility in disaster relief, because, as we have demonstrated in this paper, this type of analysis provides highly valuable insights for researcher and engineers.

Appendix B

MASS: Scenario editor

Mobile ad-hoc network research tackles many different environments. In all of them, node mobility is a determining factor, because credibility of the experiments in this subject is strongly related to the realism of the scenario. For that reason, a lot of effort has gone into designing realistic mobility models and gathering real world mobility traces. However, these scenarios are scarce and difficult to examine. With the purpose of building more complete benchmarks, researchers may want to modify them or obtain derived situations. The MASS editor is designed to ease the laborious task of editing MANET scenarios. In this paper, we present the first version, showing its potential. In addition, a markup language MASS-ML is defined with the purpose of standardizing MANET descriptions.

B.1 Introduction

Mobile ad-hoc networks (MANETs) are a hot topic in communications research. The development of wireless technologies and mobile devices allows new application environments for these networks. A MANET consists of a set of mobile devices (i.e. nodes) that communicate with each other without additional infrastructure. Therefore, there is a wide range of applications such as emergencies, vehicular communication (VANET), disaster relief, the battlefield or more common situations like improvised business meetings. For that reason, many research groups are carrying out studies on this topic. They aim either to design applications of general use, for all situations, or to study specific scenarios, such as emergencies. In both cases, evaluation of these solutions is a complicated and laborious task. Real deployments are not common; therefore most of the process relies on simulation or emulation tools, such as ns-2, ns-3 or OMNET++. The realism of this type of evaluation depends on many parameters, such as node mobility, network models, device models and so on.

Node mobility seriously affects the performance of any application deployed over an ad-hoc network. For instance, in a dense network, with many nodes in a relatively small area, mobility determines network route stability. In a sparse network, mobile nodes make possible the communication between separate network partitions. Almost all applications deployed over MANETs must take into account node mobility in order to achieve a good output or even to perform at all. As a rule of thumb, when speaking about mobility and network applications, one size does not fit all. On the one hand, researchers may test a general solution over a great number of situations finding out where their performance is better. However, this only seems to be possible when considering very simple applications. On the other hand, they may choose to focus on specific scenarios and design solutions tailored for them. For the latter approach, it is very important to accurately define node mobility in a realistic way. Synthetic mobility models or, rarely, traces from real world situations may be used to define the position of

the nodes during the evaluation. Mobility is often described in plain text files making it difficult for researchers to modify or even understand what the real network looks like.

In this paper, an editor for Mobile Ad-hoc network ScenarioS, also called MASS editor, is presented together with a markup language (MASS-ML) to define MANET mobility scenarios. MASS offers a graphical interface to create and edit MANET scenarios. Therefore, allowing researchers to obtain a better understanding of scenarios and adapt them to their needs. One interesting application can be the generation of derived scenarios from a real situation. For instance, in a rescue, what would have happened if the ambulance had arrived two/five/ten minutes later? Using MASS to modify the original scenario, researchers may build a benchmark to analyze the performance of their proposal under variations of a real situation. This would otherwise require a great effort on the scenario generation, limiting energy and time for the system evaluation.

The remainder of the paper is organized as follows. In Section B.2, we expose the main problems when dealing with mobility scenarios, the motivation and goals for creating MASS. Sections B.3 and B.4 describe the markup languages for the scenarios and the editor. Section B.5 proposes a workflow for editing scenarios. A use case of MASS is presented in Section B.6. Finally, Section B.7 concludes the paper and suggests future works on this tool.

B.2 Motivation And Goals

This section describes the process that a researcher may follow in the evaluation of a MANET from the mobility point of view. This has been the main motivation to develop a tool like MASS.

As we have previously stated, most of the topics related with MANET research greatly depend on the mobility of the nodes. Since there are only a few MANET deployments, mainly because of the subject's novelty, experimentation has to make some assumptions including mobility. The first MANET studies mostly considered random mobility scenarios [106]. These scenarios were generated through parameterized algorithms, which follow a mathematical model. The Random Waypoint mobility model is the most popular. The first proposal tended to attract nodes to the center of the scenario [127], which led to the appearance of other models [66] which claimed to solve these issues, such as Random Walk. However, these models are not realistic in other situations, such as people who move in groups [128] or cars travelling along avenues [68]. Thus, new proposals focused on mobility under specific circumstances, for instance disaster relief [129]. Together with these, new tools emerged as an aid for the research community. That is the case of the scenario generator BonnMotion[11] or the IMPORTANT framework [64]. The former implements some mobility models, being also capable to write them in different formats. The latter defines a set of mobility related metrics that are relevant to evaluate routing protocols over a mobile scenario.

Despite the interest of these artificial models, they may be very different to real situations. Our experience [104] shows that randomness in the scenarios leads to unpredictable results. It is extremely difficult to design and evaluate the performance of a communication service using random behavior of nodes. A good way to increase the

[11] http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/

credibility of our results is to use scenarios from traces captured in real world circumstances. This is made feasible by the low price of GPS devices and the increasing interest of this research. Apart from describing past events, traces may be useful to study future situations. For instance, firemen follow some protocols that repeat in different disasters. Nevertheless, scenarios based on traces may not be enough to evaluate new solutions, because they are often scarce and do not cover all the desired use cases. More variation is needed. Thus, trace based mobility models are being studied [130] as one option. However, it is also possible to slightly modify real scenarios manually to cover a set of related situations.

It may be that researchers want to modify the scenarios obtained by any of these means. On the one hand, scenarios from mobility models are not realistic enough and may be too simple. To achieve more credibility or to evaluate different circumstances, they may be manually modified by, for instance, adding nodes or changing trajectories. On the other hand, real world traces are not easy to manage. They normally have to be filtered and graphically represented to obtain further knowledge about them. For instance, it may be that only a period of time is interesting for the scenario, that some traces are mistaken or that it is desirable to remove or add nodes. Whatever manipulation of the original traces is sought, it requires a great effort. There exist many different formats for GPS traces that depend mostly on the device used. Moreover, almost every network simulator/emulator uses a different format to represent the mobile ad-hoc network. This ranges from plain text to C programs, passing through tcl scripts. In brief, there are a great variety of formats that are frequently cryptic for the untrained eye.

For all these reasons, researchers face a lot of problems building realistic evaluation scenarios. Thus, a graphical tool would be of great help. With the right interface, it would save time and effort in this process. Its users would be able to modify scenarios without thoroughly studying the specific formats. This tool should also allow the creation of simple scenarios, with a few nodes, which may be useful in initial experiments or for educational purposes. Finally, the scenarios managed by this tool would be represented in a format independent of the simulation platform used that should be human readable and easy to import/export from/to other formats. With this in mind, we have designed the tool presented in this paper (MASS). Given that such a tool is a very ambitious project, we have implemented a first version that includes the basic features.

The goals pursued with the current version are:

- To provide a comfortable scenario edition environment with the following functionalities:
    - Creation of simple scenarios by adding nodes, configuring their communication range and defining their trajectory.
    - Edition of existing scenarios with the possibility of changing node properties and path, adding and removing new nodes.
    - Scenario representation and animation to carry out visual analysis of the network.
- To define a simple format (MASS-ML) for these scenarios combining mobility and networking features.

Finally, to simplify the initial implementation, some assumptions have been carried out which are also generally undertaken in other tools, such as network simulators or scenario generators. However, it is the purpose of this tool to become as general as possible in future releases, in the sense of considering all types of MANET scenarios. These assumptions are:

- Movement of nodes is a composition of straight uniform speed movements. In the near future, uniform acceleration movements may also be interesting, because of the information received from GPS devices (position and current speed).
- Nodes are represented in a two-dimensional space, although three dimensions are considered for the scenario definition.
- The radiation pattern of nodes is always spherical (a circle in two dimensions).

B.3 MASS Workflow

Figure B-1 represents the MASS workflow. A concise definition of a MANET scenario must consider both mobility of the nodes and networking features such as the communication range of the devices. This information may come from many sources, such as mobility generators, real traces, network models or network device descriptions. Then, this data is represented in MASS-ML. For instance, GPS traces can be easily processed using scripting languages, while adding the networking properties for each node.

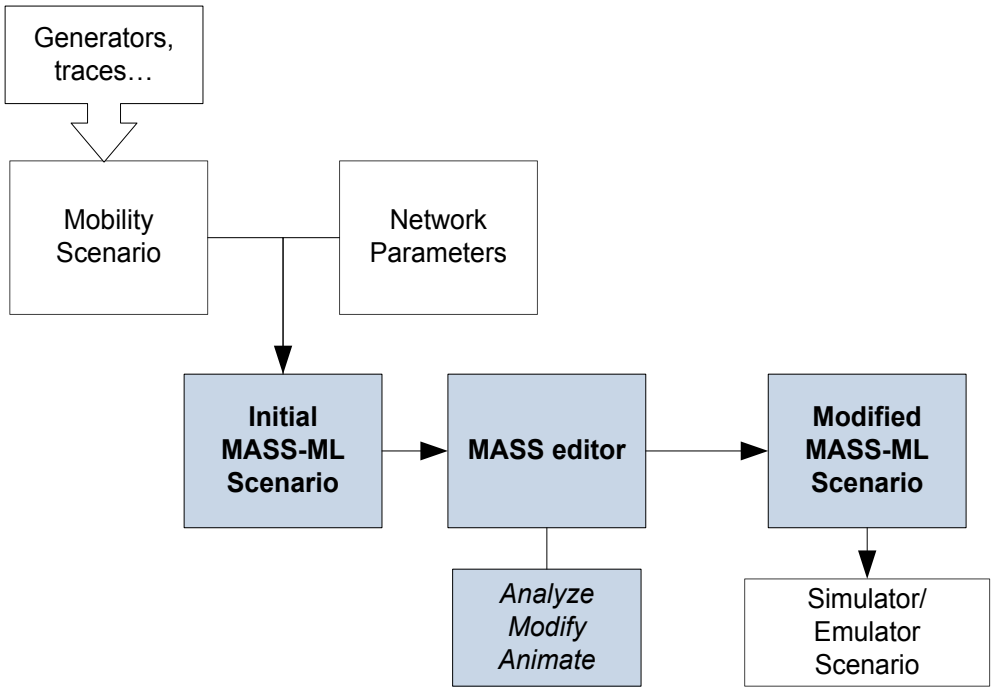

Figure B-1 MASS Workflow

All the scenarios expressed in MASS-ML can be opened with the edition tool. Then, they can be just visualized or modified by adding new nodes, changing their paths or their ranges. Once they have been satisfactorily processed, they can be saved as a new

MASS-ML scenario. The final step for researchers is to transform this scenario into the format of the tool used for simulation/emulation. Again scripting languages provide fast ways of doing this.

B.4 Scenario Definition: MASS-ML

The first important contribution of this paper is the definition of an XML based format to represent MANET scenarios. On the one hand, several formats exist to represent mobility scenarios coming from GPS traces. Some of them are just plain text based, like simple CSV files, while others make use of XML, such as GPX or KML. Often, the former group is simpler and the latter allows a better description. On the other hand, there are formats to represent networks. They are normally very complex and dependent on the simulator that uses them. For instance, ns-2 MANET scenarios are represented in TCL scripts, which establish the position and the links of a node at any given time. Thus, to the best of our knowledge, there is a great diversity of different formats, none of which are specifically focused on mobile ad-hoc networks.

MASS-ML (MASS markup language) is able to express MANET scenarios in a simple fashion. It is based on XML, which facilitates the task of using it in other programs and makes it human readable. The objective is to gather all the data needed to perfectly describe a MANET in a single file. Therefore, it includes general information about the scenario where the MANET is deployed and specific information about each node. A MASS-ML document not only includes all the movements of each node, but also specifies its networking properties, radiation pattern and range in the current version. In addition, possible events for each node are considered; switching on and off.

According to the assumptions made for this first version, there are limitations on the scenario definition. However, the language structure is designed to support extensions. For instance, new movements could be expressed by adding new parameters to the adequate tag. It may also be appealing for researchers to include more event types for the nodes, such as declaring battery level or resource consumption.

The remainder of the section is dedicated to describe the first version of the language: MASS-ML v1. This is an example of a MASS-ML document:

```
<?xml version="1.0" ?>

<scenario   t="Tue Feb 1 2005 12:00:00 AM"
            X="800" Y="500" Z="100"
            background="map.png">

   <node id="0" icon=”node.png” pattern="circular” range="75" >

     <switchOn  t="3000" x="10" y="20" z="0" />
     <moveTo start="5000" end="7500" x="30" y="45" z="0" />
      …
     <switchOff t="220000" />
  </node>
…
</scenario>
```

The first version of this markup language considers the following tags and properties:

- **scenario**: defines the properties of the scenario.

- *t*: is the point in the time when the scenario starts.
- *X, Y, Z*: are the maximum dimensions.
- *background*: indicates a background image to visualize the scenario.

- **node**: contains all the properties, events and movements of a node.
  - *id*: is the unique identifier of the node.
  - *pattern*: is the radiation pattern of the node. Currently, only circular patterns are considered, but this attribute may contain a reference to an SVG (Scalable Vector Graphics) definition.
  - *range*: is the radius of the circular radiation pattern in meters.
  - *icon*: indicates an image to represent this node in the scenario.
- **switchOn**: represents that the node is switched on at the time (t) and in the position (x,y,z).
- **switchOff**: represents that the node is switched off at the time (t).
- **moveTo**: indicates that the node moves from the current node position to (x,y,z) between the instants start and end. In this version, all movements have constant speed, although this can be extended with the initial speed and acceleration parameters.

Note that in the MASS editor all distances are measured in meters and all times are milliseconds.

B.5 MASS Editor

There are few proposals that provide graphical representation of mobility or networking scenarios. Most often, only those built-in into simulators allow scenario modification, e.g. OPNET. As occurs with formats, to the best of our knowledge, there is no tool that supports MANET scenario graphical edition. The first version of the MASS editor provides core functionality for this purpose. Adobe Flex, generating it as Air application, has been selected for the implementation, because it provides a powerful development environment for graphical tools. Moreover, it is possible to run it over any operating system supporting Air.

MASS is able to open and save MASS-ML files and represent them graphically. An area proportional to the scenario size is represented. Its background, if present, can also be displayed. These properties can be easily modified. In addition, visual aids, such as zoom or a grid, are included.

Each node is shown as a circle, or an icon, with a surrounding circle proportional to its range. Nodes can be modified or removed and new ones can also be added. The trajectory of a node is represented by a set of checkpoints connected by lines. Each checkpoint is associated to a position in the scenario and to two instants in time. These values are the arrival and departing time of the node to that checkpoint. MASS allows the edition of these values. Furthermore, it is possible to drag and drop checkpoints

through the scenario, changing their position. This feature eases the quick modification of node trajectory. Checkpoints can also be added or removed.

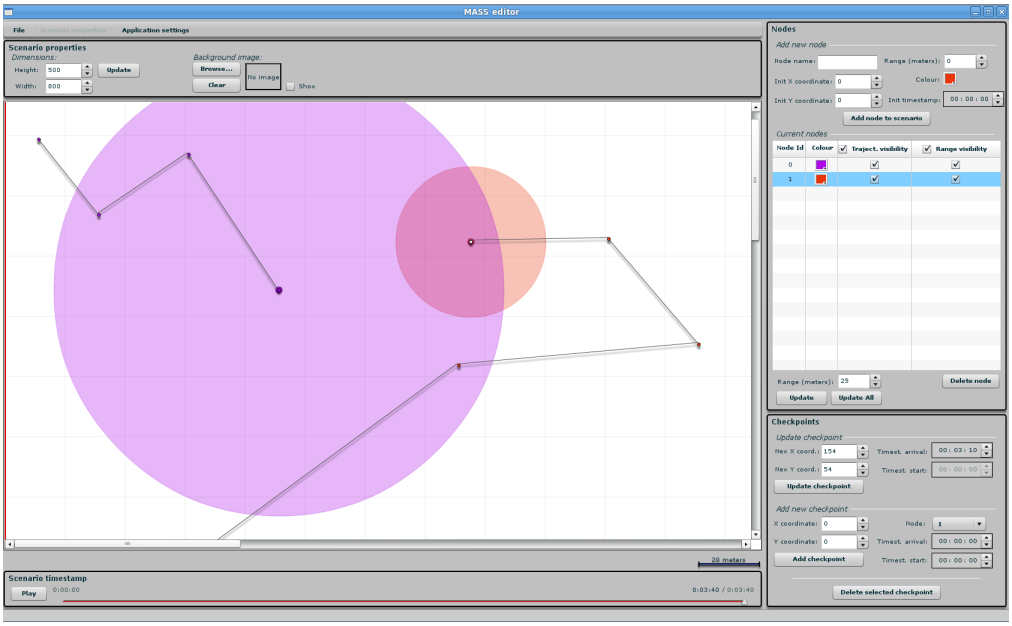

Figure B-2 MASS Editor Interface

The visualization of the whole scenario gives a first impression of the MANET, for instance, is it dense or sparse? Are there partitions? However, researchers may desire a more detailed visual analysis. For that reason, the view of the scenario can be customized to fit the user's requirements. The color or icon of nodes can be modified. Checkpoints, ranges of each node can be shown or hidden by request, as well as the general background. Moreover, information about the position of the nodes and checkpoints is shown when the mouse rolls over them. All these visual aids provide a comfortable use of the tool and a better understanding of the scenarios. For example, movement patterns of nodes in a scenario may be easily detected by examining them individually.

The user interface is designed to provide a friendly environment. The interface, as shown in Figure B-2, is divided into four areas. The top area is dedicated to general purpose controls. It has a File tab to load, save and clear the scenario. The Application tab contains visualization tools, such as the zoom. Finally, the Scenario tab contains controls to define the properties of the scenario: size and background. The area on the right is dedicated to configure the nodes and checkpoints presented in the central area. Mobility or networking can be defined here, as well as aesthetic properties (colors, sizes, etc.). Finally, the bottom contains the timeline and controls to animate the scenario.

B.6 Case Study

In this case study, we show how MASS may be used to analyze a real scenario and extend it to create hypothetical situations for a complete benchmark. We use GPS traces from a chemical accident trial carried out in Asturias, Spain in 2009. The original traces show the movement of the firemen's vehicles from 9am to 2pm. For simplicity, we have

considered only the action that took place in the surroundings of the chemical incident during the minutes with most activity.

The first step is to transform the data from the original format (a spreadsheet) to a more manageable format (a csv file) containing: node identifier, latitude, longitude and timestamp. Secondly, the file has to be processed in order to convert it to the right units for MASS: meters and milliseconds. Sometimes it is also desirable to filter the number of checkpoints given by the GPS devices, especially in long scenarios. Otherwise, it may be difficult to manage the probably large amount of checkpoints represented for each node. A good tactic is to remove checkpoints when the node is stopped at one location. For this use case, we have filtered all the traces that are at less than 5 meters from the previous one. Finally, the MASS-ML scenario is generated reading the CSV file with a Shell Script.

Figure B-3 shows the result once opened with MASS. As we can see there are four nodes moving in a disaster area. Each of them has a different trajectory, although we can observe that they tend to be located inside two different zones. They are clearly identified by hiding two of the nodes. One is the command center for the firemen and the other is the place where the incident took place. Hence, this kind of representation is very useful to understand the scenario. In addition, we can animate the scenario to see that direct communication between the command center and the firemen deployed in the incident is not possible. They form two different partitions of the MANET.

Our goal may be to look for communication possibilities between these two zones. Thus, we can use MASS to analyze or design new possibilities. In Figure B-4a, we see that there is a node moving between the zones. Using it as a ferry may be one possibility. We now want to evaluate hypothetical behaviors of this node. Therefore, we may modify its path, as in Figure B-4b, remove checkpoints or delay it. Then, we may see how this affects the MANET. Furthermore, we may try other alternatives, such as locating a static node acting as repeater between both partitions. The position of this node, its communication range and other requirements may be guessed using the graphical representation of the tool.

B.7 Conclusions

The tool presented in this paper allows the easy creation of new scenarios. Moreover, it is adequate to build derived scenarios from an existing one. Thus, MASS helps researchers obtain a better understanding of the situations under study. In addition, it can be used to generate new scenarios and to generate complete benchmarks based on a specific use case. More realistic scenarios can be produced, because they are easier to build, and, hence, credibility of the evaluations is improved. We have also defined a markup language (MASS-ML) to describe the scenarios. Researchers may use it to convert scenarios from/to other formats and share MANET definitions with others easily. Finally, the friendly environment and the possibility to include icons for the nodes and a background for the scenario facilitate the preparation of demos of real situations, such as emergencies showing fire trucks and firemen on a map.

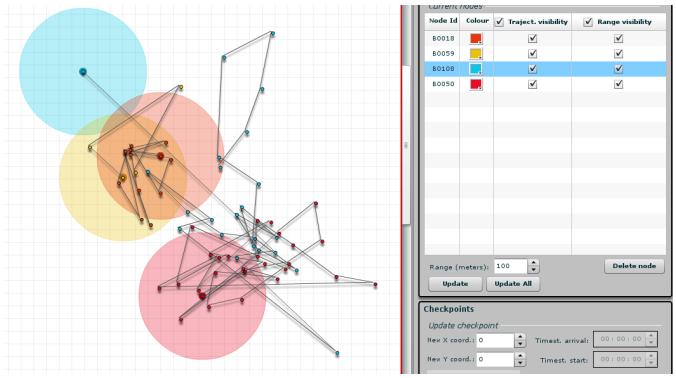

Figure B-3 Scenario with 4 nodes

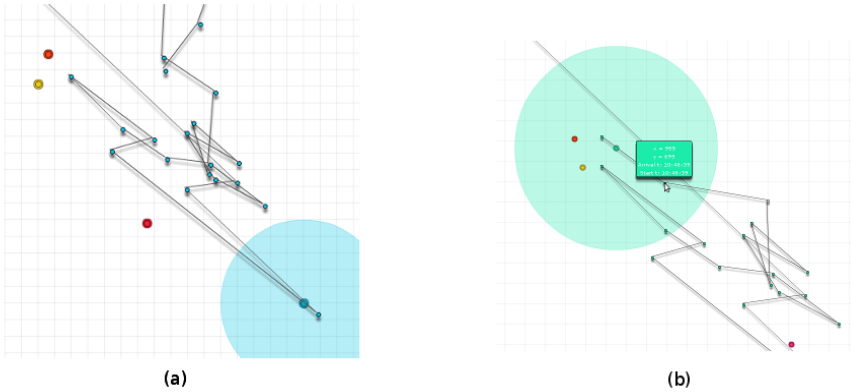

Figure B-4 (a) Original and (b) modified paths of a ferry node

Appendix C

ERPlayer: video player for ER operations

This appendix describes a video player application conceived to reproduce the video packets delivered by a sparse MANET in ER operations. First, we argue why a specific video player is needed. Then, we describe the main ideas behind its design. Finally, we describe a proof-of-concept prototype that applies these ideas. More details can be found in [131].

C.1 Introduction

MOMENTUM supports off-the-self RTSP/RTP video clients. However, conventional video players expect video packets in order and with relatively small delay and jitter. They use buffers to accommodate to changes in network conditions. So, they can reproduce video smoothly and without interruptions. However, the way video packets are delivered by MOMENTUM in a sparse MANET is different from streaming in the Internet. For example, Figure C-1 shows video packets delivered in one of our experiments. They are received discontinuously and, sometimes, in bursts with a higher bitrate than the video stream bitrate. Packets are only delivered with the original bitrate if they are streamed from source to destination over a multihop route. Off-the-shelf video players produce a bad user experience under these conditions, because late packets are not decoded and the reproduction freezes when packets do not arrive. Enlarging the client video buffer is not a good solution on its own. A bigger buffer increases startup delay and it would be the solution for some of the late packets, but this is unlikely to be enough in a DTN. We believe that the user experience improves with a video player that supports both streaming and delay-tolerant video transport. In addition, ER operations are situations that may require extra functionalities from a video application.

For these reasons, we propose a tailored video player, called ERPlayer. The Incident Chief or other personnel in a Command and Control Center can use it integrated with other tools, such as Geographical Information Systems or health monitoring applications. The envisioned design for this application is showed in the next section.

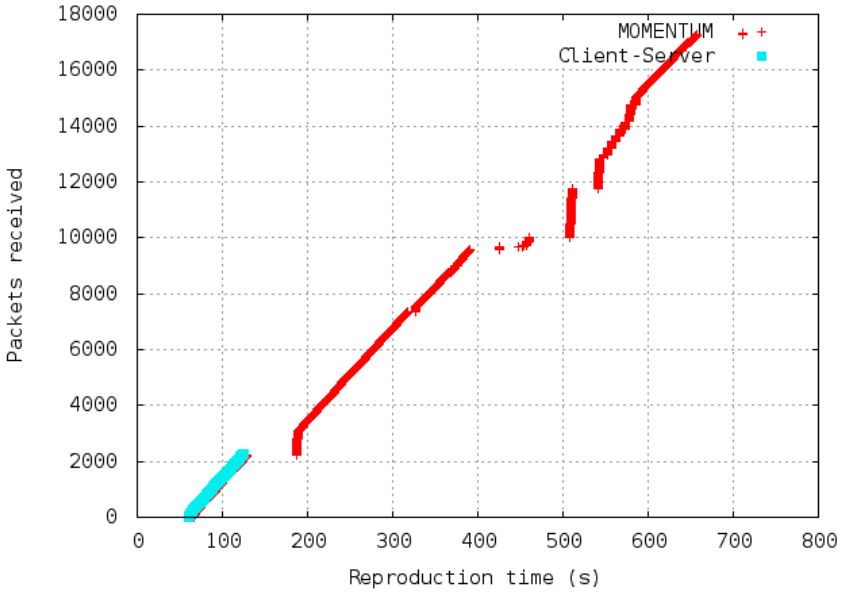

Figure C-1 Packet delivery in a scenario with disconnections (4 nodes, 7 m/s)

C.2 Design

This section briefly identifies the main functionalities of ERPlayer and proposes a user interface to integrate them. ERPlayer supports the way video packets are delivered in a sparse MANET. Using MOMENTUM some packets arrive out of order and with big delays. In addition, using adaptation mechanisms like DTS, video packets can increase the video quality (e.g. the frame rate) of parts of the video previously received. The video player must get the best possible user experience from the video packets received, independently on their order and delay. Thus, we propose to store all video packets delivered. This packets are used to show video received with the best possible quality. In addition, users can access to all received video parts. ERPlayer uses the reproduction timeline to indicate the user what parts are available and also the quality of each part. This mechanism allows quick visualization and access to the received video. Therefore, when video packets are received, ERPlayer analyzes them and uses either to create a new video part or to improve the quality of a previously delivered part. This is shown to the user through the timeline.

Users can be interested in getting statistics about video delivery and configuring video delivery options, such as prioritization of some video streams over others. Moreover, cameras in an ER operation may have advanced functionalities. For example, some may be surveillance cameras that can be moved. In addition, metadata can be associated to each camera, such as its location. Both advanced functionalities and metadata are integrated in the player too. Finally, ERPlayer must support receiving video from several sources at the same time.

These functionalities should be implemented in a user friendly interface. Figure C-2 shows a sketch of the interface we propose. The screen is divided in several areas. The main part is the video player. Only one video screen is shown in the figure, although the application could support showing more than one camera at the same time. Below the player, there is a timeline that shows with a quick look the video that it is available for the user. Different colors are used to identify the available quality. For instance, green

represents high quality, orange, medium and red, low. A triangle indicates the current reproduction time. The user can drag it to reproduce any available video segment. Play, record and stop controls, camera status and other stats are shown below. On the left, a menu lists all the cameras that are available in the network. A status line for each camera informs the user about new events, such as video received. Finally, the user can change the session parameters of the active camera. For example, he or she can establish different priorities or levels of quality for watching and for recording. MOMENTUM can use these preferences to define transport policies for the video streams. For example, if recording quality of a video is high, reliable transport is always used to transport video packets. Then, packets are always stored in the overlay network and can be recovered for offline analysis after the ER operation. We envision this multimedia application as part of a complete control panel. Ideally, officers would use a single application to access available information about the current operation. It would merge data coming from many different sources, such as sensors, geographical information systems and so on.

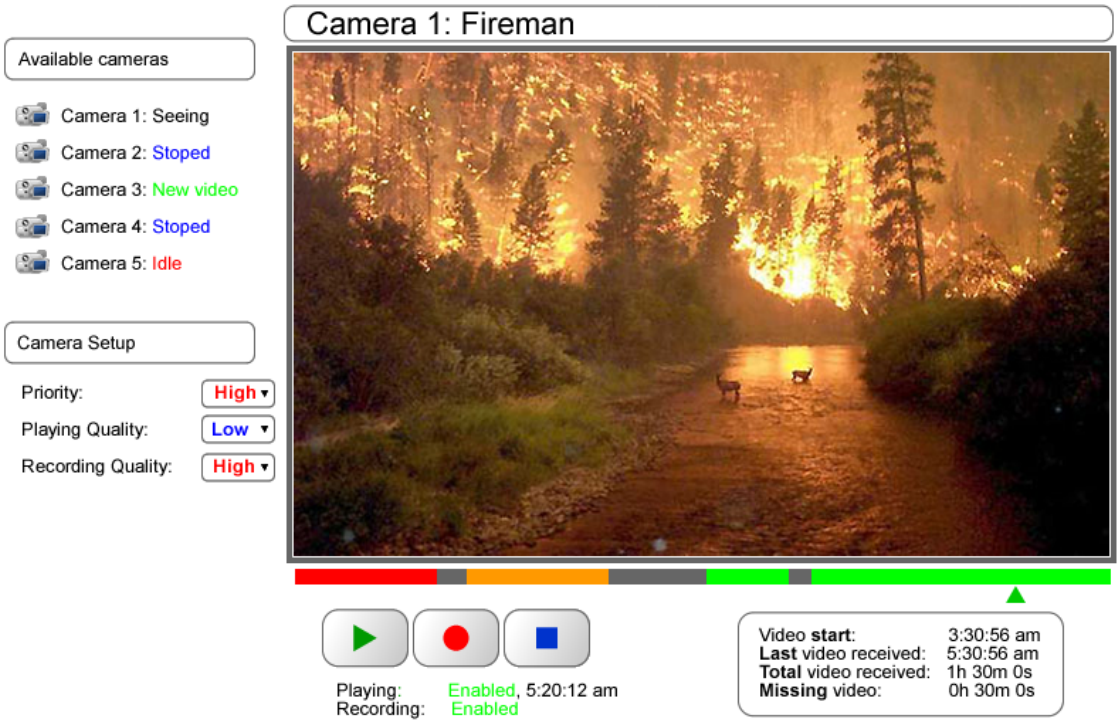

Figure C-2 Video player interface design concept

C.3Proof-of-concept prototype

We have implemented a proof-of-concept prototype of ERPlayer, see Figure C-3. It is implemented using Adobe Flex technology. The prototype focuses on supporting video packets delivered delayed and disordered. For this purpose, it implements a first version of the timeline described in our design. This implies two interesting implementation challenges. The first one is how to identify and reproduce the available video parts from the video packets received. The second one is how to measure the quality of these parts.

The first challenged is solved by creating short video files of fixed length, e.g. 5 seconds, from the received packets. This solution resembles the system used by HTTP Adaptive Streaming systems. When the user wants to reproduce the video, video files are opened and smoothly reproduced. Hence, users do not perceived that the video is split in several files. To generate these files, ERPlayer analyze the metadata associated with the video stream (i.e. timestamps and sequence numbers). Then, ERPlayer identifies the quality of each segment. We use a simple formula based on the frames available to score each file. We consider that I, P and B frames contribute in a different way to the final score. Scores are mapped to three quality levels that are represented with green, yellow and red in the timeline. When packets received belong to an already existent file, it is regenerated with the old and new packets. The color used to represent this video file in the timeline changes, if the quality was improved.

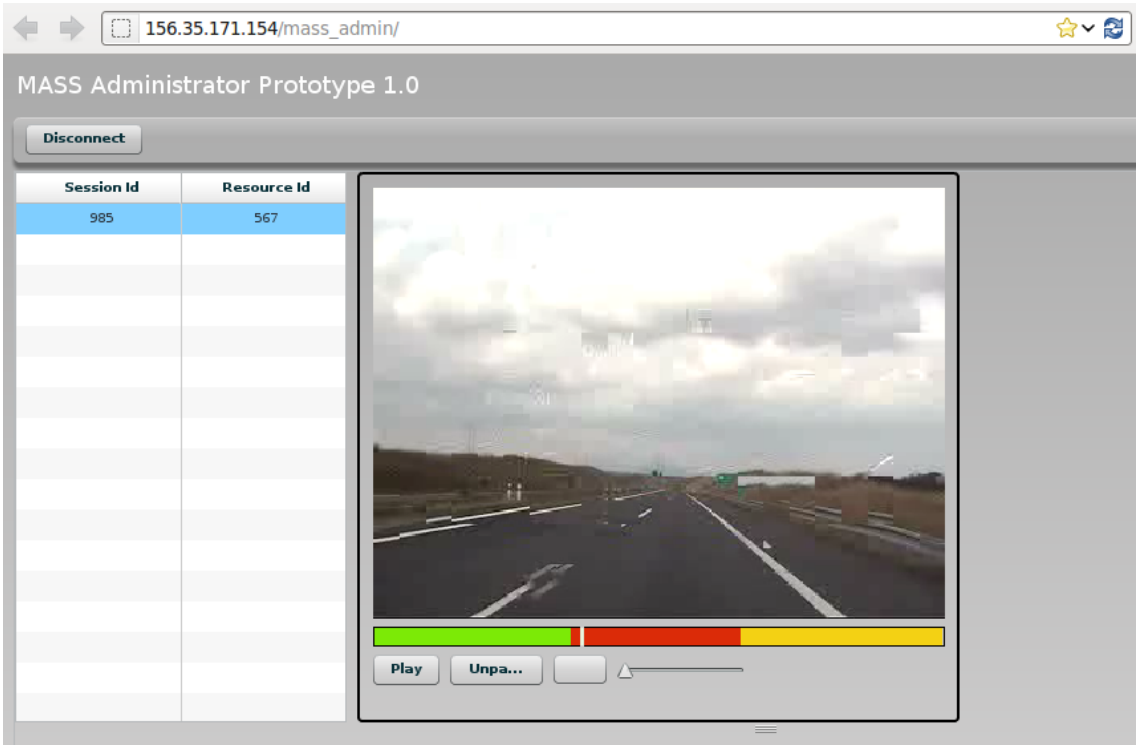

Figure C-3 Video player interface implementation

The video player was tested in a controlled environment. The aim was to test many different scenarios for the timeline by sending packets to ERPlayer in predetermined order. The results are satisfactory, as it could generate the timelines according to the packets received. Nonetheless, further evaluation with final users would be interesting to determine future work on it.

References

[1] W. Zhao and M. H. Ammar, "Message ferrying: proactive routing in highly-partitioned wireless ad hoc networks," in *The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems,* 2003, pp. 308–314.

[2] G. Dodig-Crnkovic, "Scientific methods in computer science," *Proceedings of the Conference for the Promotion of Research in IT at New Universities and at University Colleges in Sweden,* 2002, pp. 126–130.

[3] M. Álvarez Pascual, "Ánalisis de prestaciones de dispositivos móviles para un servicio de vídeo streaming sobre redes móviles ad-hoc en entornos de emergencia," M.S. Thesis, University of Oviedo, Spain, 2013.

[4] I. Fuentes Fernández and I. Álvarez Arenas, "Evaluación de la transmisión de vídeo sobre redes MANET parcialmente emuladas," B.S. Thesis, University of Oviedo, 2009.

[5] M. Frodigh, P. Johansson, and P. Larsson, "Wireless ad hoc networking: the art of networking without a network," *Ericsson Review*, vol. No.4, pp. 248–263, 2000.

[6] S. Xu and T. Saadawi, "Does the IEEE 802.11 MAC Protocol Work Well in Multihop Wireless Ad Hoc Networks?," IEEE *Communications Magazine*, vol. 39, no. 6, pp. 130–137, Jun. 2001.

[7] G. Holland and N. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, 1999, pp. 219–230.

[8] T. D. Dyer and R. V. Boppana, "A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks," in *Proceedings of the 2Nd ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing*, 2001, pp. 56–66.

[9] X. Chen, H. Zhai, J. Wang, and Y. Fang, "TCP performance over mobile ad hoc networks," *Canadian Journal of Electrical and Computer Engineering*, vol. 29, no. 1/2, pp. 129–134, Jan. 2004.

[10] J. P. Monks, P. Sinha, and V. Bharghavan, "Limitations of TCP-ELFN for ad hoc networks," in *IEEE International Workshop on Mobile Multimedia Communications, MoMuC*, 2000.

[11] G. Anastasi, E. Ancillotti, M. Conti, and A. Passarella, "TPA: a transport protocol for ad hoc networks," in *10th IEEE Symposium on Computers and Communications,* 2005, pp. 51–56.

[12] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh, and R. Sivakumar, "ATP: a reliable transport protocol for ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 6, pp. 588–603, Nov. 2005.

[13] Z. J. Haas, M. R. Pearlman, and P. Samar, “The Zone Routing Protocol (ZRP) for Ad Hoc Networks,” Internet Engineering Tasks Force (IETF), 2002.

[14] Y. Ko and N. H. Vaidya, “Location-Aided Routing (LAR) in mobile ad hoc networks,” in *International Conference on Mobile Computing and Networking,*1998.

[15] C. Perkins, E. Royer, and S. Das, “RFC 3561 Ad hoc On-Demand Distance Vector (AODV) Routing,” Internet Engineering Tasks Force (IETF), 2003.

[16] T. Clausen and P. Jacquet, “RFC 3626: Optimized Link State Routing Protocol (OLSR),” Internet Engineering Tasks Force (IETF), 2003.

[17] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[18] J. H. Saltzer, D. P. Reed, and D. D. Clark, “End-to-end Arguments in System Design,” *ACM Transactions on Computer Systems*, vol. 2, no. 4, pp. 277–288, Nov. 1984.

[19] S. Merugu, M. H. (Mostafa H. Ammar, and E. W. Zegura, “Routing in Space and Time in Networks with Predictable Mobility,” Technical Report GIT-CC-04-07, Georgia Institute of Technology, 2004.

[20] B. Burns, O. Brock, and B. N. Levine, “MORA routing and capacity building in disruption-tolerant networks,” *Ad Hoc Networks*, vol. 6, no. 4, pp. 600–620, Jun. 2008.

[21] V. Cerf, K. Fall, K. L. Scott, S. C. Burleigh, L. Torgerson, A. J. Hooke, H. S. Weiss, and R. C. Durst, “RFC 4838: Delay-Tolerant Networking Architecture,” Internet Engineering Task Force (IETF), 2007.

[22] K. L. Scott and S. Burleigh, “RFC 5050: Bundle Protocol Specification,” Internet Engineering Task Force (IETF), 2007.

[23] K. Fall, W. Hong, and S. Madden, “Custody Transfer for Reliable Delivery in Delay Tolerant Networks,” Technical Report, Intel Research, Berkeley, 2003.

[24] L. Wood, W. Ivancic, W. M. Eddy, D. Stewart, J. Northam, C. Jackson, and A. D. S. Curiel, “Use of the Delay-Tolerant Networking Bundle Protocol from Space,” *In Proceedings of the 59th International Astronautical Congress*, 2008, vol. 5, pp. 3123–3133.

[25] L. Wood, W. M. Eddy, and P. Holliday, “A bundle of problems,” in *2009 IEEE Aerospace conference*, 2009, pp. 1–17.

[26] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss, “Delay-tolerant networking: an approach to interplanetary Internet,” *IEEE Communications Magazine*, vol. 41, no. 6, pp. 128–136, Jun. 2003.

[27] A. Pentland, R. Fletcher, and A. Hasson, “DakNet: rethinking connectivity in developing nations,” IEEE *Computer*, vol. 37, no. 1, pp. 78–83, 2004.

[28] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, “MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks,” in *IEEE International Conference on Computer Communications*, 2006, vol. 6, pp. 1–11.

[29] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, “Energy-efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet,” in *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2002, pp. 96–107.

[30] L. Pelusi, A. Passarella, and M. Conti, “Opportunistic networking: data forwarding in disconnected mobile ad hoc networks,” *IEEE Communications Magazine*, vol. 44, no. 11, pp. 134–141, Nov. 2006.

[31] H. Schulzrinne, A. Rao, and R. Lanphier, “RFC 2326: RTSP: real time streaming protocol,” Internet Engineering Task Force (IETF), 1998.

[32] M. Handley, C. Perkins, and V. Jacobson, “RFC 4566: SDP: session description protocol,” Internet Engineering Task Force (IETF), 2006.

[33] H. Schulzrinne, G. Fokus, S. Casner, R. Frederick, and V. Jacobson, “RFC 1889: RTP: A transport protocol for real-time applications,” Internet Engineering Task Force (IETF), 1996.

[34] T. Stockhammer, “Dynamic Adaptive Streaming over HTTP –: Standards and Design Principles,” in *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, 2011, pp. 133–144.

[35] S. Wenger and T. Stockhammer, “RFC 6184: RTP payload format for H. 264 video,” Internet Engineering Task Force (IETF), 2005.

[36] T. Friedman, R. Caceres, and A. Clark, “RFC 3611: RTP control protocol extended reports (RTCP XR),” Internet Engineering Task Force (IETF), 2003.

[37] L. Berc, W. Fenner, R. Frederick, S. McCanne, and P. Stewart, “RFC 2435: RTP payload format for JPEG-compressed video,” Internet Engineering Task Force (IETF), 1996.

[38] International Standards Organization (ISO), “Generic coding of moving pictures and associated audio information: Video,” ISO/IEC 13818-2:2000 - Information technology, 2009.

[39] K. Rijkse, “H.263: video coding for low-bit-rate communication,” *IEEE Communications Magazine*, vol. 34, no. 12, pp. 42–45, Dec. 1996.

[40] International Telecommunication Union (ITU), “H.264 : Advanced video coding for generic audiovisual services,” ITU-T Recommendation, Nov-2007.

[41] V. K. Goyal, “Multiple Description Coding: Compression Meets the Network,” *Signal Processing Magazine*, vol. 18, no. 5, pp. 74–93, 2001.

[42] H. Schwarz, D. Marpe, and T. Wiegand, “Overview of the scalable video coding extension of the H. 264/AVC standard,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 17, no. 9, pp. 1103–1120, 2007.

[43] S. Kristiansen, M. Lindeberg, D. Rodriguez-Fernandez, and T. Plagemann, “On the Forwarding Capability of Mobile Handhelds for Video Streaming over MANETs,” in *Proceedings of the Second ACM SIGCOMM Workshop on Networking, Systems, and Applications on Mobile Handhelds*, 2010, pp. 33–38.

[44] M. Lindeberg, S. Kristiansen, T. Plagemann, and V. Goebel, “Challenges and techniques for video streaming over mobile ad hoc networks,” *Multimedia Systems*, vol. 17, no. 1, pp. 51–82, Feb. 2011.

[45] A. Asif, U. T. Nguyen, and G. Xu, “Scalable Video Multicast over MANETs,” in *IEEE 8th Workshop on Multimedia Signal Processing*, 2006, pp. 403–408.

[46] C.-O. Chow and H. Ishii, “Enhancing real-time video streaming over mobile ad hoc networks using multipoint-to-point communication,” *Computer Communications*, vol. 30, no. 8, pp. 1754–1764, Jun. 2007.

[47] D. B. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad Hoc Wireless Networks,” in *Mobile Computing*, vol. 353, T. Imielinski and H. F. Korth, Eds. 1996, pp. 153–181.

[48] S. Mao, S. Lin, S. S. Panwar, Y. Wang, and E. Celebi, “Video transport over ad hoc networks: multistream coding with multipath transport,” *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 10, pp. 1721–1737, Dec. 2003.

[49] S. Mao, D. Bushmitch, S. Narayanan, and S. S. Panwar, “MRTP: a multiflow realtime transport protocol for ad hoc networks,” in *Vehicular Technology Conference,* 2003, vol. 4, pp. 2629–2634 Vol.4.

[50] S. Kompella, S. Mao, Y. T. Hou, and H. D. Sherali, “Cross-layer optimized multipath routing for video communications in wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 831–840, May 2007.

[51] S. Mao, Y. T. Hou, H. D. Sherali, and S. F. Midkiff, “Multimedia-Centric Routing for Multiple Description Video in Wireless Mesh Networks,” *IEEE Network*, vol. 22, no. 1, pp. 19–24, Jan. 2008.

[52] E. Setton, T. Yoo, X. Zhu, A. Goldsmith, and B. Girod, “Cross-layer design of ad hoc networks for real-time video streaming,” *IEEE Wireless Communications*, vol. 12, no. 4, pp. 59–65, Aug. 2005.

[53] J. Seo, E. Cho, and S.-J. Yoo, “Protocol Design for Adaptive Video Transmission over MANET,” in *Embedded and Ubiquitous Computing*, Springer, 2006, pp. 234–243.

[54] G. D. Delgado, V. C. Frias, and M. A. Igartua, “ViStA-XL: A Cross-Layer Design for Video-Streaming over Ad hoc Networks,” in *3rd International Symposium on Wireless Communication Systems*, 2006, pp. 243–247.

[55] Z. Fu, X. Meng, and S. Lu, “A transport protocol for supporting multimedia streaming in mobile ad hoc networks,” *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 10, pp. 1615–1626, Dec. 2003.

[56] E. Macías, A. Suárez, J. Martín, and V. Sunderam, “Using OLSR for Streaming Video in 802.11 Ad Hoc Networks to Save Bandwidth,” *IAENG International Journal of Computer Science*, vol. 33, no. 1, p. 101, Mar. 2007.

[57] E. Gökturk, “MICA: A Minimalistic, Component-Based Approach to Realization of Network Simulators and Emulators,” Ph.D. Dissertation, University of Oslo, 2007.

[58] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, “Advances in network simulation,” *IEEE Computer*, vol. 33, no. 5, pp. 59–67, May 2000.

[59] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, “Network simulations with the ns-3 simulator,” *SIGCOMM demonstration*, 2008.

[60] M. Pužar and T. Plagemann, “NEMAN: A Network Emulator for Mobile Ad-Hoc Networks,” in *International Conference on Telecommunications*, 2005.

[61] A. Keränen, J. Ott, and T. Kärkkäinen, “The ONE Simulator for DTN Protocol Evaluation,” in *SIMUTools ’09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, 2009.

[62] S. Kristiansen, T. Plagemann, and V. Goebel, “Modeling Communication Software Execution for Accurate Simulation of Distributed Systems,” in *Proceedings of the 2013 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, 2013, pp. 67–78.

[63] S. Kristiansen, T. Plagemann, and V. Goebel, “Extending network simulators with communication software execution models,” in *Fifth International Conference on Communication Systems and Networks (COMSNETS)*, 2013, pp. 1–10.

[64] F. Bai, N. Sadagopan, and A. Helmy, “The IMPORTANT framework for analyzing the Impact of Mobility on Performance Of RouTing protocols for Adhoc NeTworks,” *Ad Hoc Networks*, vol. 1, no. 4, pp. 383–403, Nov. 2003.

[65] D. Kotz, T. Henderson, and I. Abyzov, “CRAWDAD data set dartmouth/campus (v. 2004-12-18),” 2004.

[66] T. Camp, J. Boleng, and V. Davies, “A Survey of Mobility Models for Ad Hoc Network Research,” *Wireless Communication and Mobile Computing (WCMC)*, vol. 2, no. 5, pp. 483–502, 2002.

[67] X. Hong, M. Gerla, G. Pei, and C.-C. Chiang, “A group mobility model for ad hoc wireless networks,” in *MSWiM ’99: Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, 1999.

[68] D. R. Choffnes and F. E. Bustamante, “An Integrated Mobility and Traffic Model for Vehicular Wireless Networks,” in *Proceedings of the 2Nd ACM International Workshop on Vehicular Ad Hoc Networks*, 2005, pp. 69–78.

[69] J. Harri, F. Filali, and C. Bonnet, “Mobility models for vehicular ad hoc networks: a survey and taxonomy,” *IEEE Communications Surveys Tutorials*, vol. 11, no. 4, pp. 19–41, 2009.

[70] N. Aschenbruck, E. Gerhards-Padilla, and P. Martini, “A survey on mobility models for performance analysis in tactical mobile networks,” *Journal of Telecommunications and Information Technology*, vol. nr 2, pp. 54–61, 2008.

[71] N. Aschenbruck, E. Gerhards-Padilla, and P. Martini, “Modeling mobility in disaster area scenarios,” *Performance Evaluation*, vol. 66, no. 12, pp. 773–790, Dec. 2009.

[72] N. Aschenbruck, A. Munjal, and T. Camp, “Trace-based mobility modeling for multi-hop wireless networks,” *Computer Communications*, vol. 34, no. 6, pp. 704–714, May 2011.

[73] 112 Asturias, *Plan Territorial de Protección Civil del Principado de Asturias (PLATERPA)*. Protección Civil 112 Asturias, 2005.

[74] J. Norman, *Fire officer’s handbook of tactics*. Tulsa, Okla.: PennWell, 2005.

[75] J. Gorman, “The MIDAS Project: Interworking and Data Sharing,” in *Interworking 2006*, 2006.

[76] T. Plagemann, J. E. Andersson, V. Goebel, C. Griwodz, P. al Halvorsen, A. Skogsfjord, and J. Walpole, *Ad-hoc InfoWare: Middleware Services for Information Sharing in Ad-hoc Networks. http://folk.uio.no/infoware/* (Last Visited June 2014), 2003.

[77] M. Portmann and A. A. Pirzada, “Wireless Mesh Networks for Public Safety and Crisis Management Applications,” *IEEE Internet Computing*, vol. 12, no. 1, pp. 18–25, Jan. 2008.

[78] D. Rodríguez-Fernández, I. Martinez-Yelmo, E. Munthe-Kaas, and T. Plagemann, “Always Best (dis-)Connected: Challenges to Interconnect Highly Heterogeneous Networks,” in *Wired/Wireless Internet Communications*, Springer, 2011, pp. 410–421.

[79] A. A. Pirzada, M. Portmann, R. Wishart, and J. Indulska, “SafeMesh: A wireless mesh network routing protocol for incident area communications,” *Pervasive and Mobile Computing*, Elsevier, vol. 5, no. 2, pp. 201–221, Apr. 2009.

[80] M. H. Pinson, S. Wolf, and R. B. Stafford, “Video Performance Requirements for Tactical Video Applications,” in *IEEE Conference on Technologies for Homeland Security*, 2007, pp. 85–90.

[81] Public Safety Communications, “Tactical and surveillance video quality experiments,” Deparment of Homeland Security, 2007.

[82] Public Safety Communications, “Video quality tests for object recognition applications,” Deparment of Homeland Security, 2007.

[83] D. Rodriguez-Fernandez, S. Kristiansen, M. G. B. Lindeberg, O. V. Drugan, S. Cabrero, T. P. Plagemann, V. H. Goebel, E. Munthe-Kaas, X. G. Pañeda, and K. Omang, “Delay Tolerant Streaming in Rescue Scenarios: Requirements Analysis and Resulting Industrial Issues - Version 2,”, Technical Report, University of Oslo 2012.

[84] J. E. Haavet, “Dynamic Selection of Message Carriers in a Sparse and Disruptive MANET,” M.S. Thesis, University of Oslo, 2011.

[85] A. C. Santiago, “Design, Implementation and Evaluation of an Utility Based Algorithm for Prioritization of Packet Carriers in Delay-Tolerant Networks,” M.S. Thesis, University of Oviedo, 2011.

[86] J. S. Rodriguez, “An experimental analysis of overlay configurations to support video streaming over sparse MANETs,” M.S. Thesis, University of Oviedo, 2010.

[87] S. Kristiansen, “A Methodology to Model the Execution of Communication Software for Accurate Simulation of Distributed Systems,” Ph.D. Dissertation, University of Oslo, 2013.

[88] S. Kristiansen and T. Plagemann, “Accuracy and scalability of ns-2’s distributed emulation extension*,” *SIMULATION*, vol. 87, no. 1–2, pp. 45–65, Jan. 2011.

[89] M. Santirso, “Basic Interactive Extensible Network Visualization Tool: Bienvisto,” M.S. Thesis, University of Oviedo, 2011.

[90] S. Cabrero, X. G. Pañeda, D. Melendi, and R. Garcia, “MASS: Editor for mobile ad-hoc network scenarios,” in *2010 IEEE GLOBECOM Workshops (GC Wkshps)*, 2010, pp. 1679–1683.

[91] C. Cima Granda, “Módulo de cálculo de métricas de movilidad en MANETs para MASS,” B.S. Thesis, University of Oviedo, 2011.

[92] R. Alonso Martín, “Signalling Protocol ofr Multimedia Streaming Applications in Delay Tolerant Networks,” M.S. Thesis, University of Oviedo, 2012.

[93] L. O. Dybsjord, “DTSS - Signalling for Media Streaming and Delay Tolerant Network,” M.S. Thesis, University of Oslo, 2010.

[94] S. Anes González, “DT-Stream application,” M.S. Thesis, University of Oviedo, 2011.

[95] Select Bipartisan Committee to Investigate the Preparation for and Response to Hurricane Katrina, *A Failure of Initiative: Final Report of the Select Bipartisan Committee To Investigate the Preparation for and Response to Hurricane Katrina,* Government Printing Office, 2006.

[96] S.-A. Lenas, S. C. Burleigh, and V. Tsaoussidis, “Reliable Data Streaming over Delay Tolerant Networks,” in *Wired/Wireless Internet Communication*, Springer, 2012, pp. 358–365.

[97] M. Lindeberg, S. Kristiansen, V. Goebel, and T. Plagemann, “MAC Layer Support for Delay Tolerant Video Transport in Disruptive MANETs,” in *Proceedings of the 10th international IFIP TC 6 conference on Networking*, 2011, pp. 106–119.

[98] M. Li, D. Agrawal, D. Ganesan, and A. Venkataramani, “Block-switched Networks: A New Paradigm for Wireless Transport,” in *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*, 2009, pp. 423–436.

[99] K. S. Skjelsvik, “A Distributed Event Notification Service for Sparse Mobile Ad-hoc Networks*,”* Ph.D. Dissertation, University of Oslo, 2008.

[100] R. Pant, A. Tunpan, P. Mekbungwan, R. Virochpoka, and K. Kanchanasut, “DTN Overlay on OLSR Network,” in *Proceedings of the Sixth Asian Internet Engineering Conference*, 2010, pp. 56–63.

[101] R. Wang, X. Wu, T. Wang, X. Liu, and L. Zhou, “TCP Convergence Layer-Based Operation of DTN for Long-Delay Cislunar Communications,” *IEEE Systems Journal*, vol. 4, no. 3, pp. 385–395, Sep. 2010.

[102] C. Carter, S. Yi, and R. Kravets, “ARP Considered Harmful: Manycast Transactions in Ad Hoc Networks,” in *IEEE Wireless Communications and Networking,* 2003, pp 1801-1803.

[103] S. Cabrero, X. G. Pañeda, D. Melendi Palacio, and R. García Fernández, “En busca de un protocolo de transporte multimedia para redes móviles ad-hoc poco densas,” in *Conferencia de Ingeniería Telemática*, 2009.

[104] S. Cabrero, X. G. Pañeda, T. Plagemann, V. Goebel, and M. Siekkinen, “Overlay solution for multimedia data over sparse MANETs,” in *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*, 2009, pp. 1056–1061.

[105] J. Lee and S. Park, “Optimum UDP Packet Sizes in Ad Hoc Networks,” *IEICE TRANSACTIONS on Communications*, vol. E88-B, no. 2, pp. 815–820, Feb. 2005.

[106] S. Kurkowski, T. Camp, and M. Colagrosso, “MANET simulation studies: the incredibles,” *SIGMOBILE Mobile Computing & Communications Review*, vol. 9, no. 4, pp. 50–61, Oct. 2005.

[107] S. Cabrero, X. G. Pañeda, T. Plagemann, D. Melendi, and R. García, “An Overlay Routing Protocol for Video over sparse MANETs in Emergencies,” *Cadernos de Informática*, vol. 6, no. 1, pp. 195–202, 2011.

[108] M. Lindeberg, J. E. Haavet, S. C. Barros, V. Goebel, and T. Plagemann, “Message lost or message taken - on message ferry selection in DTNs -,” in *IFIP Wireless Days*, 2012, pp. 1 –7.

[109] Z. Zhang, “Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges,” *IEEE Communications Surveys Tutorials*, vol. 8, no. 1, pp. 24–37, 2006.

[110] A. Vahdat and D. Becker, “Epidemic Routing for Partially-Connected Ad Hoc Networks,” Duke Tech Report CS-2000-06, 2000.

[111] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, “Impact of Human Mobility on Opportunistic Forwarding Algorithms,” *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 606–620, 2007.

[112] A. Lindgren, A. Doria, and O. Schelén, “Probabilistic routing in intermittently connected networks,” *SIGMOBILE Mobile Computing & Communications Review*, vol. 7, pp. 19–20, Jul. 2003.

[113] K. Xu, M. Gerla, and S. Bae, “How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks,” in *IEEE Global Telecommunications Conference,* 2002, vol. 1, pp. 72–76 vol.1.

[114] M. Voorhaen and C. Blondia, “Analyzing the Impact of Neighbor Sensing on the Performance of the OLSR protocol,” in *International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2006, pp. 1–6.

[115] S. Cabrero, X. García Pañeda, T. Plagemann, D. Melendi, and R. García, “Towards reliable video transmission over sparse MANETs in emergencies,” *Informática na educação: teoria & prática*, vol. 14, no. 1, Nov. 2011.

[116] S. Cabrero, X. G. Paneda, R. Garcia, D. Melendi, and T. Plagemann, “Dynamic Temporal Scalability: Video adaptation in sparse Mobile Ad-Hoc Networks,” in *IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications*, 2012, pp. 349–356.

[117] A. Fraga, L. Pozueco, X. Garcia Pañeda, R. García, D. Melendi, and S. Cabrero, “A non-intrusive estimation for high-quality Internet TV services,” *Multimed Tools & Applications*, pp. 569-588, vol. 54 Jul. 2010.

[118] L. Pozueco, X. G. Pañeda, R. García, D. Melendi, S. Cabrero, and G. D. Orueta, “Adaptation engine for a streaming service based on MPEG-DASH,” *Multimed Tools & Applications*, pp. 1–20, May 2014.

[119] L. Pozueco, X. G. Pañeda, R. García, D. Melendi, and S. Cabrero, “Adaptable system based on Scalable Video Coding for high-quality video service,” *Computers & Electrical Engineering*, vol. 39, no. 3, pp. 775–789, Apr. 2013.

[120] J.-S. Lee, F. De Simone, and T. Ebrahimi, “Subjective Quality Evaluation via Paired Comparison: Application to Scalable Video Coding,” *IEEE Transactions on Multimedia*, vol. 13, no. 5, pp. 882–893, 2011.

[121] J.-S. Lee, F. De Simone, T. Ebrahimi, N. Ramzan, and E. Izquierdo, “Quality assessment of multidimensional video scalability,” *IEEE Communications Magazine*, vol. 50, no. 4, pp. 38–46, 2012.

[122] M. Lindeberg, V. Goebel, and T. Plagemann, “CLiSuite: simplifying the development of cross-layer adaptive applications,” in *Proceedings of the 7th Workshop on Middleware for Next Generation Internet Computing*, 2012, pp. 2:1–2:6.

[123] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, “Security in mobile ad hoc networks: challenges and solutions,” *IEEE Wireless Communications*, vol. 11, no. 1, pp. 38–47, Feb. 2004.

[124] M. Pužar, T. Plagemann, and Y. Roudier, “Security and privacy issues in middleware for emergency and rescue applications,” in *Second International Conference on Pervasive Computing Technologies for Healthcare*, 2008, pp. 89–92.

[125] A.-L. Barabási, *Linked: the new science of networks*. Cambridge: Perseus, 2002.

[126] D. Katsaros, N. Dimokas, and L. Tassiulas, “Social network analysis concepts in the design of wireless Ad Hoc network protocols,” *IEEE Network*, vol. 24, no. 6, pp. 23–29, 2010.

[127] J. Yoon, M. Liu, and B. Noble, “Random waypoint considered harmful,” in *INFOCOM 2003. Conference of the IEEE Computer and Communications*, 2003, vol. 2, pp. 1312–1321 vol.2.

[128] G. Pei, M. Gerla, X. Hong, and C.-C. Chiang, “A wireless hierarchical routing protocol with group mobility,” in *IEEE Wireless Communications and Networking Conference,* 1999, pp. 1538–1542 vol.3.

[129] N. Aschenbruck, E. Gerhards-Padilla, M. Gerharz, M. Frank, and P. Martini, “Modelling Mobility in Disaster Area Scenarios,” in *Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, 2007, pp. 4–12.

[130] M. Kim and D. Kotz, “Extracting a mobility model from real user traces,” in *In INFOCOM 2006 Conference of the IEEE Computer and Communications,* 2006.

[131] J. Martínez Alfonso, “Diseño e implementación de una pasarela de vídeo en una red móvil ad-hoc para entornos de emergencias,” M.S. Thesis, University of Oviedo, 2013.