Universidad de Buenos Aires

66.00 - Tesis de Ingeniería Electrónica

Redes Determinísticas en Aplicaciones Industriales de IoT

Del Fiore, Julián Martín- 92883

4 de septiembre de 2017

Abstract

Las recientes actividades de normalización aportan una alta calidad de servicio (QoS) y previsibilidad en la Internet de las cosas (IoT), que se están "volviendo industriales". Las aplicaciones críticas como el control de procesos industriales, la red inteligente o la automatización de manejo de vehículos requieren transmisiones determinísticas con propiedades tales como entregas de datos a tiempo y confiabilidad de extremo a extremo cercana al 100 %. Las tecnologías de radio tradicionales basadas en la detección de colisiones y retransmisiones introducen retrasos impredecibles y no pueden garantizar una entrega fiable dentro de un tiempo estrechamente limitado. La solución introducida en esta tesis de estudios propone explotar la diversidad espacial y la redundancia de paquetes para compensar las pérdidas que se producen producto del medio inalámbrico, que justamente es inherentemente con pérdidas. Reducir el número de transmisiones de un nodo aumenta su vida útil. Leapfrog Collaboration es un mecanismo de comunicación que permite oír otras transmisiones (Overhearing) y en el que las mismas se programan en forma paralela en dos rutas. La escucha promiscua entre los caminos permite a los nodos en un camino dado escuchar las transmisiones en el opuesto. El retardo y su fluctuación, el ciclo de trabajo de la red y la tasa de entrega de paquetes de la comunicación se evalúan mediante simulación utilizando Contiki OS.

Contents

1 Motivación

La Internet de las cosas o en inglés Internet of things (IoT) se refiere a las muchas aplicaciones que se pueden encontrar hoy en día que utilizan dispositivos con sistemas embebidos, lo que les permite conectarse a Internet para compartir información. Estos dispositivos son objetos pequeños de baja potencia y capacidad de memoria limitada que se benefician de los servicios de una red mundial. Además, estos objetos usualmente incorporan tecnologías de comunicación inalámbrica, y naturalmente con pérdidas, motivo por el cual a las redes que conforman se las llama LLNs, siglas que provienen del ingles y que significan Low Power Lossy Networks. El crecimiento explosivo de IoT sigue su marcha y es un proceso que al día de hoy continúa cambiando nuestro mundo. Esto en parte se ve facilitado por la caída en el precio de los componentes típicos utilizados en estas tecnologías, lo cual posibilita que haya un constante nacimiento de nuevas aplicaciones innovadoras. La variedad de las mismas es lo suficientemente grande como para permitir pensar que se pueden utilizar prácticamente en cualquier lugar: sistemas de estacionamiento inteligentes, hogares inteligentes, wearables, automatización de la agricultura, gestión de edificios, control de tráfico, telemedicina y Redes de Sensores Inalámbricos (WSNs) son algunos de los mejores ejemplos.

En particular, dentro de IoT existe un concepto emergente llamado Industria 4.0. Este nombre proviene de la evolución de los sistemas de producción. En el siglo XVIII, la producción mecánica fue impulsada principalmente por agua y vapor, conformando lo que se conoce como Industria 1.0. Esto evolucionó en la industria 2.0 en el siglo XX, donde la producción en masa se logró en términos de la división del trabajo, y se vio marcada por el uso de la energía eléctrica. En la década de los setenta, la Industria 3.0 significó el inicio de la aplicación de la electrónica y la informática para automatizar aún más la producción. En la actualidad, existe una nueva evolución del concepto, donde la Industria 4.0 puede ser pensada como sistemas cibernéticos de producción física.

La idea detrás de la Industria 4.0 es reutilizar las tecnologías IoT, pero en las redes de las industrias. Hacia los comienzos, las fábricas tenían redes cableadas, pero esto implicaba un gran costo. Después de un tiempo,

las soluciones inalámbricas comenzaron a implementarse también, y ZigBee, WirelessHART, ISA100.11a y WIA-PA se encuentran entre los estándares de redes de sensores inalámbricos industriales más populares que se utilizan actualmente. Lo que es nuevo en la Industria 4.0 es la aceptación hacia soluciones estandarizadas como las de IoT, en las que IPv6 es ampliamente considerado la mejor manera de lidiar con la cantidad de estos dispositivos. Las razones de esta aceptación son que las soluciones de este tipo facilitarán el despliegue de las fábricas y simplificarán las cadenas de producción. Como resultado, no sólo las industrias se volverán más flexibles y adaptables, sino que también habrá una reducción en los costos de implementación y manutención, y una mayor eficiencia.

En IoT, diferentes aplicaciones requieren diferentes desempeños en la red. El rendimiento, la relación de entrega de paquetes, el retardo y su variación (en inglés jitter) se deben controlar más o menos de manera exacta dependiendo de los requisitos de la aplicación. Sin embargo, como en la Industria 4.0 se controlan los procesos industriales, los requisitos sobre estas nuevas redes se hacen estrictos. Las redes ahora no solo necesitan ser confiables, sino también determinísticas.

Las redes determinísticas son aquellas en las que los paquetes que un nodo envía a su destino están programados para llegar en un cierto punto del tiempo, y después de atravesar toda la red, realmente lo hacen. En otras palabras, una red determinística garantiza que el transporte de la información se llevará a cabo de una manera predefinida (es decir, predecible) y con un retardo constante, sea cual sea la calidad de los enlaces o la congestión de la red.

En el 2016 se publicó el estándar IEEE 802.15.4-2015 para ofrecer Calidad de Servicio (QoS) para aplicaciones determinísticas de tipo industrial. Time-Slotted Channel Hoping (TSCH) está entre los protocolos de control de acceso al medio (MAC) definidos en esta norma, lo que le permite competir con los otros estándares industriales. Además, puede proporcionar entregas a tiempo con una relación de pérdidas de 10^{-5} particularmente debido a la diversidad de canales. Sin embargo, no evita retransmisiones cuando se pierde un paquete de datos, debido a interferencias, colisiones o interrupción de un nodo.

2 ¿Qué es el determinismo?

2.1 Definiciones

La primera pregunta que plantea el concepto de redes determinísticas es qué es el determinismo.

En filosofía, el determinismo es una doctrina que sostiene que todo acontecimiento físico, incluyendo el pensamiento y acciones humanas, está causalmente determinado por la irrompible cadena causa-consecuencia, y por tanto, el estado actual "determina" en algún sentido el futuro.

En matemáticas y física, los sistemas deterministas siguen la misma línea de pensamiento: se denomina sistema determinista a aquel en que el azar no está involucrado en el desarrollo de los futuros estados del sistema. Un modelo determinista producirá siempre la misma salida a partir de las mismas condiciones de partida o el estado inicial. Cada estado futuro del sistema está determinado por el previo en tanto se desprende de cómo queda afectado dadas las variables de entorno y el previsto comportamiento ante los cambios en ese ambiente.

2.2 Determinismo en una Red

El determinismo en una red trae la garantía de que una determinada información será transporta a través de la red en una estrecha ventana de tiempo, y que un proceso periódico será repetido idénticamente cada vez. El determinismo es una propiedad requerida por ejemplo en la red eléctrica para asegurar que interruptores en líneas de alta tensión se puedan activar en milisegundos. Por otro lado, también lo es en el transporte público para asegurar que vehículos autónomos operen de manera segura para sus pasajeros, y en la automatización industrial para el control de lazo. De manera de delinear aún más los conceptos detrás de las redes determinísticas, resulta de utilidad analizar ciertas analogías.

2.2.1 Analogía del tren para el control de lazo del tráfico

El análogo de una pérdida de congestión en el sistema ferroviario, es decir, la colisión de dos trenes utilizando los mismos carriles al mismo tiempo, se evita en el mundo real a través del uso de operaciones completamente programadas que repiten, día tras día, el esquema predeterminado que está determinado en los horarios de trenes. La figura 1 ilustra la complejidad del cálculo de un horario óptimo para múltiples trenes que comparten la misma infraestructura, con el objetivo de minimizar el tiempo de extremo a extremo y con la restricción de evitar colisiones al asegurar que como mucho un tren está presente en cualquier sección de la vía en cualquier punto del tiempo.

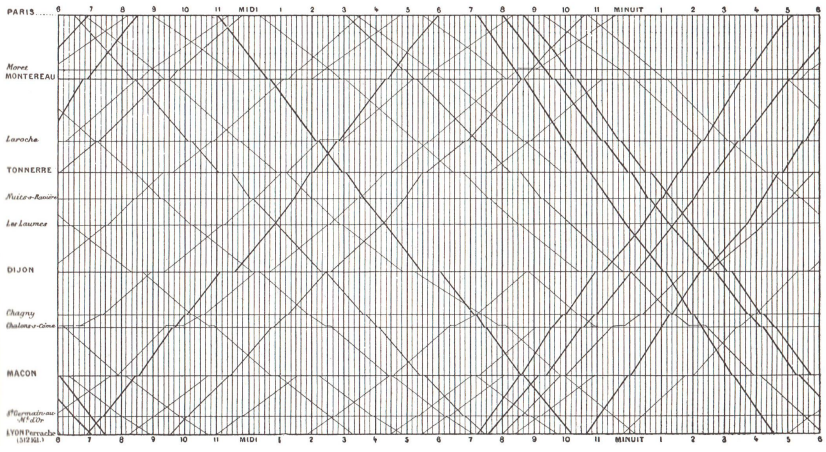


Figura 1: Sistema de trenes y la complejidad de un esquema global.

El problema no resulta para nada trivial, e inclusive en la imagen se observa que existen algunos recorridos que incluyen zonas planas, lo cual significa que, para evitar colisiones, un tren puede detenerse temporalmente en algún lado, generalmente una estación, esperando un tiempo predeterminado para poder luego continuar de manera segura con su trayecto. El hecho de programar estos eventos ayuda a evitar colas y resulta en un proceso mucho más previsible para los pasajeros.

De la misma manera, un flujo determinístico puede experimentar un almacenamiento en nodos intermedios que garanticen que nunca hayan dos paquetes determinísticos programados para ser transmitidos en tiempos que se solapen o, en otras palabras, que el enlace esté siempre disponible al momento de transmisión programado. Debido al retardo fijo extra que se suma en cada

salto, la mayoría de los flujos determinísticos incurren en un retardo mayor.

2.2.2 Analogía con los micros para la conmutación de circuitos determinísticos

El objetivo del uso de carriles reservados para colectivos en ciudades es el evitar que los mismos se vean retrasados por embotellamientos y permitir que el viaje sea realizado más pronto.

Se considera el caso en que un colectivo tarda 30 minutos en viajar desde cierta estación en la calle de la casa de un usuario hasta su oficina en el microcentro. Si los micros pasan a una tasa de 10 minutos, entonces el tiempo de tránsito para un usuario particular dependerá de que tanta suerte tiene ese día, pero siempre será entre 30 y 40 minutos, sin importar de cuánto tráfico haya en los carriles para autos.

Dado que no existen infinitas líneas que lleven a todos los destinos, un usuario puede tener que hacer un cambio de un micro a otro, una o más veces antes de llegar a su destino final. El cambio entre líneas significa un tiempo adicional de transporte para el usuario, pero también significa una enorme reducción en el número de líneas y por ende la complejidad operacional, resultando en un negocio más rentable para la empresa de micros.

Una línea de micro es análoga a los circuitos que pueden ser programados en una red determinística de conmutación de paquetes. Sin importar la carga, garantizan una oportunidad periódica de transmisión con un retardo acotado para comandos asincrónicos y alertas. En dicha analogía, la última realización de una red determinística aparece como la posibilidad del uso de un medio inalámbrico, transportando exactamente la misma aplicación, pero a un costo mucho menor al de una red cableada.

2.2.3 Analogía del lugar de vacaciones para el compartimiento de tiempo

El marketing ha inventado el concepto de las adquisiciones por tiempo compartido, donde un individuo posee una propiedad en un lugar lindo para vacacionar por un tiempo acotado, como ser una semana. En dicha semana,

la persona utiliza el lugar como si fuera suyo: no encontrará gente inesperada en su cama por la noche y, considerando que los demás propietarios son suficientemente decentes, no tendrá idea alguna de la cantidad que en total comparten este mismo lugar en otros momentos a lo largo del año.

En el ejemplo anterior del tren, el determinismo requería que los paquetes de un flujo particular pudiesen ser retenidos en nodos intermedios hasta cierto instante programado en el cual el acceso al medio de transmisión está garantizado. Ello significa que los buffers deben tener la capacidad suficiente como para poder almanecer el paquete durante ese período de tiempo. La analogía con el presente caso reside en que la casa no puede estar ocupada por otro propietario.

Las redes determinísticas requieren unir recursos físicos a los flujos determinísticos. Es decir, con una operación programada, el mismo recurso puede de ser afectado por diferentes flujos en distintos momentos, y la duración de las mismas, y por ende la cantidad de flujos que pueden hacer uso del mismo recurso, se ve directamente afectada por la precisión del sentido de compartir el tiempo en la red. El sistema de para reservar recursos debe ser consciente de las bondades de los dispositivos en términos de la precisión de su reloj y el retardo que insumen en enviar un paquete, así como la cantidad de buffers y colas que disponibles.

Con un modelado apropiado, un flujo determinístico se ve totalmente aislado de cualquier influencia de otro tráfico, sin fugas ni pérdidas, y sin retardo alguno que pudiese ser adjudicado a otro flujo, sin importar la carga de la red.

2.3 Características de una Red Determinística

Idealmente, una red determinística es aquella que asegura que todos los paquetes llegarán a destino y que lo harán en un tiempo dado. En términos del mundo real, una red determinística debe ser una red confiable, libre de variaciones en los retardos de los flujos que la atraviesan, y que puede ser utilizada con confianza para el transporte periódico y asincrónico de información con un retardo acotado. A su vez, resulta importante que la red mantenga

siempre sus propiedades, incluso aún ante cargas que pongan límite los flujos.

Para lograr ello, resulta necesario programar la operación de la red, permitiendo así utilizar los recursos con los que se cuente de manera óptima. Aún así, ello en general supone un aumento en el retardo de los flujos transportados. La red debe estar sincronizada de una manera muy precisa, de manera tal que los recursos requeridos puedan ser reservados para los paquetes individuales dentro de un flujo particular en el momento preciso de llegada.

Gracias al uso de una red determinística, el peor caso de pérdida y de retardo puede ser garantizado consistentemente. Los flujos se ven completamente aislados en sus propios recursos de tiempo compartido y no pueden ser influenciados por otro tráfico. Desde el punto de vista de la aplicación, una conexión determinística de extremo a extremo se asemeja a una conexión punto a punto cableada. Es por ello que este tipo de redes puede ser percibida como la evolución de la calidad de servicio que garantiza a un nivel superador el correcto funcionamiento de servicios basados en redes.

3 Redes 6TiSCH

La evolución tan pronta de la Internet de las Cosas y la intención de aplicarla a procesos industriales utilizando IPv6 para conformar redes determinísticas trajo aparejada la inexistencia de un stack de protocolos que pudiera permitir dicho desempeño. Es en este sentido que la Internet Engineering Task Force (IETF) creó el grupo de trabajo 6TiSCH, con el objetivo de definir un set de protocolos para redes de baja potencia y con pérdidas (LLNs), como las que caracterizan al tipo de dispositivos utilizados en IoT.

En particular, 6TiSCH provee una capa de red multiplexada estadísticamente basada en IPv6 sobre la altamente precedible tecnología IEEE802.15.4-TSCH. De esta manera, hereda las capacidades avanzadas de esta última para combatir el desvanecimiento por múltiples caminos y la interferencia externa, permitiendo que este tipo de redes sean implementadas en entornos industriales de manera de ser aprovechadas para la mejora de operaciones automáticas.

La arquitectura 6TiSCH [1] presenta un stack de referencia que puede ser observado en la figura 2.

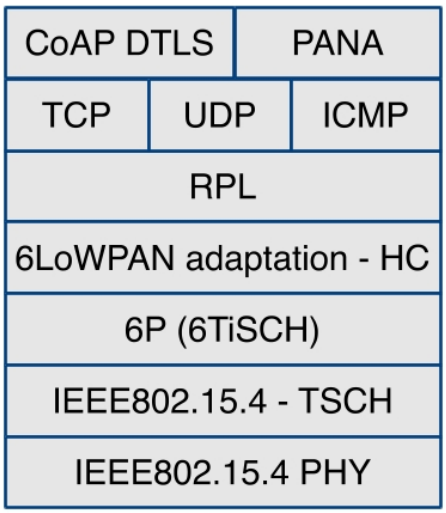


Figura 2: 6TiSCH Protocols Stack

En este stack de protocolos:

- CoAP es el "Constrained Application Protocol" utilizado como protocolo de aplicación de internet definido específicamente para dispositivos

con recursos limitados, como en IoT.

- PANA es el "Protocol for Carrying Authentication for Network Access" y cumple tareas de seguridad.
- TCP, UDP y ICMP son diferentes protocolos que pueden ser utilizados en la capa de transporte. En general se utiliza UDP ya que los nodos no tienen mucha capacidad de almacenamiento.
- RPL es el protocolo de ruteo utilizado en LLNs. El mismo crea Instancias RPL compuestas por Destination Oriented Directed Acyclic Graphs (DODAGs). Los DODAGs poseen un sumidero o una raíz que colecta todo el tráfico de la red. La red crea un gradiente que apunta hacia la raíz adjudicando a cada nodo un Rango en función de lo determinado por una Función Objetivo.
- 6LoWPAN combina la última versión del Protocolo de Internet (IPv6) y Redes de área personal inalámbricas de bajo consumo (LoWPAN). A través de la compresión de Headers permite que los dispositivos más pequeños con capacidad limitada de procesamiento transmitan información de forma inalámbrica usando un protocolo de Internet.
- 6P es el protocolo utilizado en la sub-capa llamada 6top o 6TiSCH Operation sublayer que provee la abastracción de un enlace IP sobre una capa MAC que utiliza Time-Slotted Channel Hopping (TSCH) y que programa los paquetes que son enviados en las celdas TSCH. Para realizar esta tarea, se basa en lo determinado por una Función Programadora (Scheduling Function, en inglés).
- IEEE 802.15.4 - TSCH es el protocolo utilizado en términos de la capa física y la capa de enlace. Sin embargo, dado que los entornos industriales representan un ambiente de trabajo con condiciones mucho más desfavorables, la capa MAC de este estándar fue modificada para incorporar el uso de TSCH. Esto genera que las transmisiones ocurran en el instante y las frecuencias programadas anteriormente.

El desarrollo de la presente tesis requiere la compresión de los protocolos RPL y la capa MAC del estándar IEEE 802.15.4-TSCH, puesto que LeapFrog Collaboration, el mecanismo introducido, se entiende como una propuesta que

coordina ambos en búsqueda de la obtención de una red con cualidades determinísticas. En dicho modo, su funcionamiento podría asemejarse como una solución a lo pretendido por la sub-capa 6top. Sin embargo, el protocolo 6P se encuentra actualmente siendo desarrollado, y por ende ello posibilita el uso de otras propuestas. En última instancia, el mecanismo desarrollado podría considerarse como una nueva Función Programadora, la cual luego deberá adaptarse a los requerimientos de 6P de manera tal de cumplir con el stack estandarizado de una red 6TiSCH. Por otro lado, LeapFrog Collaboration no sólo determina el ordenamiento en una red TSCH, sino que cumple funciones de ruteo al mismo nivel de RPL. Por ende, LeapFrog Collaboration propone una solución integral para el manejo de paquetes en redes 6TiSCH.

4 RPL: IPv6 Routing Protocol for LLNs

RPL [2] es un protocolo de enrutamiento de distancia de vectores diseñado para operar sobre una gran cantidad de mecanismos de enlace de red, entre los cuales se incluyen la capa física y MAC del estándar IEEE 802.15.4 [3]. RPL, como el título de la presente sección lo indica, es un protocolo de ruteo IPv6 pensado para redes de baja potencia y con pérdidas (LLNs). El mismo fue creado por el grupo de trabajo IEFT ROLL (Routing Over Low power and Lossy networks) y se encuentra especificado en el RFC 6550 de dicho organismo.

RPL es un protocolo ampliamente utilizado en redes de baja potencia Power Line Communication (PLC) y en tecnologías inalámbricas de baja potencia, como lo son los casos de las redes Wireless Personal Area Networks (WPANs) y Wireless Sensor Networks (WSNs) [4]. Las mismas están caracterizadas por la escasez de recursos en términos de memoria, batería y potencia de los equipos. Más aún, estas redes presentan enlaces con pérdidas, los cuales en consecuencia se encuentran expuestos a altas tasas de errores de paquetes (PER, por sus siglas en inglés) e incluso a veces se ven interrumpidos. RPL fue específicamente pensado para atender las necesidades de este tipo de redes y por ende su funcionamiento presenta una alta adaptabilidad a las condiciones de la red, proveyendo rutas alternativas en los casos en los que las rutas por default se vuelvan inaccesibles.

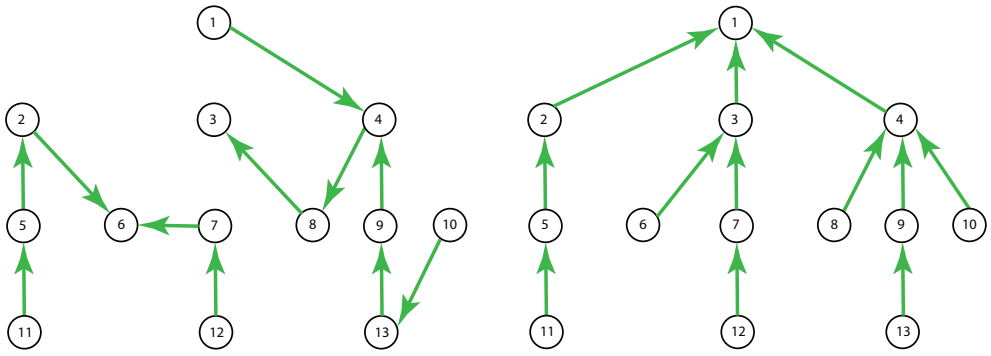
A modo básico, RPL apunta hacia lo que se conoce como redes basadas en colecciones, en las cuales los nodos envían periódicamente mediciones a un punto que actúa de sumidero, pero que también cuenta con la posibilidad de enviar tráfico hacia el resto de los nodos que componen la red. Dada la naturaleza dinámica de la topología de las redes consideradas, RPL provee mensajes de control que se ven diseminados en la red que y que permiten conocer su estructura. Cuando se detectan inconsistencias, entendidas como el resultado de cambios en la topología de la red, la frecuencia con la que estos paquetes se envía aumenta siguiendo el comportamiento marcado por lo que se conoce como temporizador Trickle.

4.1 Topología de RPL

La idea consiste en una organización jerárquica donde los nodos se asocian entre sí en forma de padres-hijos, y donde el tráfico de la red converge hacia un único punto. A continuación se detallan los principales conceptos que le dan forma a la topología de la red.

4.1.1 DODAGs

RPL se basa en el concepto topológico de Directed Acyclic Graph (DAG), definiendo una estructura de tipo árbol que especifica las rutas por default entre los nodos que componen la red. Un DAG se distingue ya que en el no existen loops o caminos cerrados. Sin embargo, RPL toma un subconjunto de ellos, y más concretamente se habla de Destination Oriented DAGs (DODAGs). En estos casos particulares el DAG apunta como destino hacia un nodo especial conocido como sumidero o raíz del DODAG. Un DODAG se ve unívocamente identificado por la combinación de lo que se conoce como una Identificación de Instancia RPL (RPL Instance ID) y una Identificación de DODAG (DODAG ID).



(a) DAG (b) DODAG

Figura 3: Comparación entre un DAG y un DODAG. La principal diferencia radica en que en un DAG pueden presentarse múltiples sumideros, mientras que en un DODAG, todos los caminos fluyen hacia un único.

4.1.2 Raíz del DODAG

Este nodo es el encargado de crear y mantener el DAG. Además, es el punto de colección de la red, es decir, el punto hacia el cual fluyen los paquetes de información. En general actúa como el Low Power and Lossy Border Router

(LBR), conectando al dominio RPL junto con el exterior. Es decir que, en definitiva, es quien conecta una LLN con redes IPv6, o en otras palabras, Internet. Esto puede observarse en la figura 4.

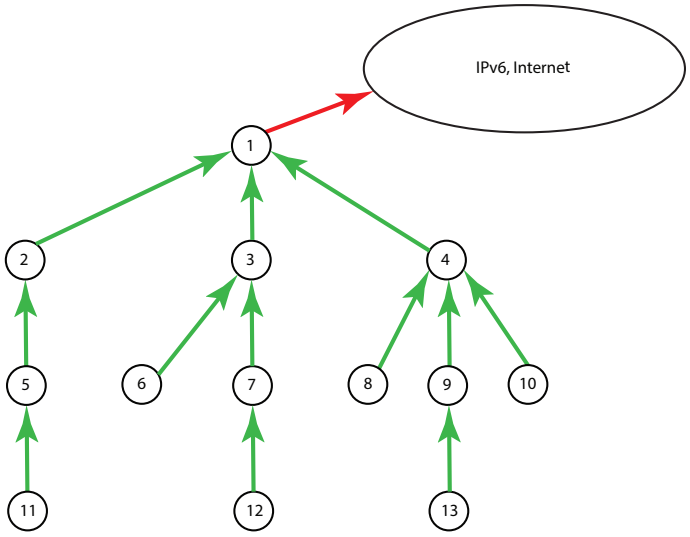


Figura 4: El sumidero de un DODAG es conocido como raíz del DODAG y en general es quien conecta la red a Internet.

4.1.3 Rango

Todos los nodos en un DODAG son asignados un Rango. El mismo se define como la posición individual del nodo relativa a los otros nodos con respecto a la raíz del DODAG. En otras palabras, es simplemente un número entero que representa la localización de un nodo dentro del DODAG.

El Rango aumenta estrictamente en la dirección descendente del DAG (al alejarse de la raíz del DODAG), y por ende disminuye en el sentido ascendente, como puede observarse en la figura 5. De este modo, resulta una manera de medir la distancia hacia la raíz del DODAG.

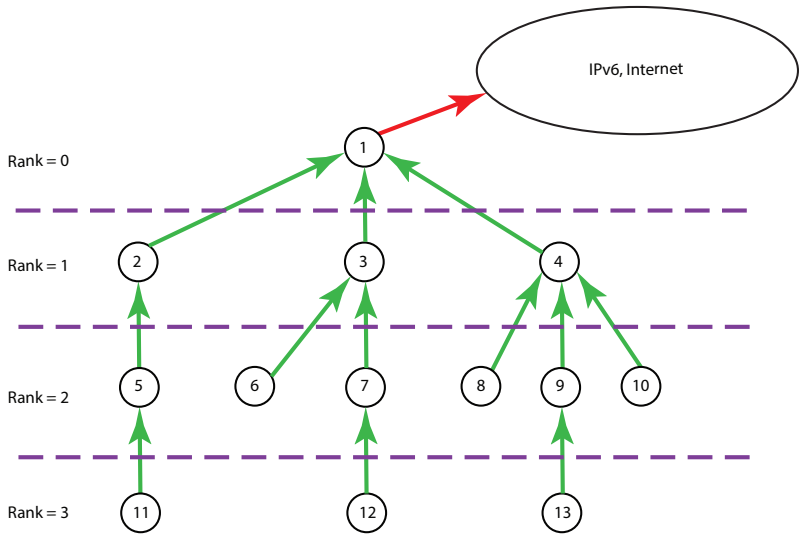


Figura 5: Rango. La raíz del DODAG tiene un Rango de valor nulo, y el resto de los nodos a medida que se alejan de ella van aumentando el suyo. En este caso en particular, el Rango se mide como la cantidad de saltos necesarios para transmitir un paquete a la raíz del DODAG. Si bien en la figura los valores se muestran para cada capa, es tarea de nodo en particular calcular el suyo.

La manera en que el Rango es calculado depende de la Función Objetivo (OF) del DAG. Ésta, entre otros factores, para realizar dicho cálculo puede tener en cuenta no sólo la distancia de un nodo a la raíz del DODAG, sino que también estadísticas en cuanto a las métricas de la calidad de los enlaces.

4.1.4 Instancias RPL

Una red puede consistir en uno o varios DODAGs, que en conjunto conforman lo que se conoce como una Instancia RPL. Esta última está asociada a una RPL Instance ID que lo distingue de otras que pudiese haber. En particular, todos los DODAGs dentro de una misma RPL Instance ID comparten la misma Función Objetivo.

Por otro lado, una red puede tener simultáneamente varias Instancias RPL, pero que son independientes entre sí en cuanto a la lógica: cada una puede implementar una Función Objetivo diferente. Los nodos pueden formar parte de varias Instancias RPL, pero dentro de cada una de ellas, sólo pueden participar en un único DODAG.

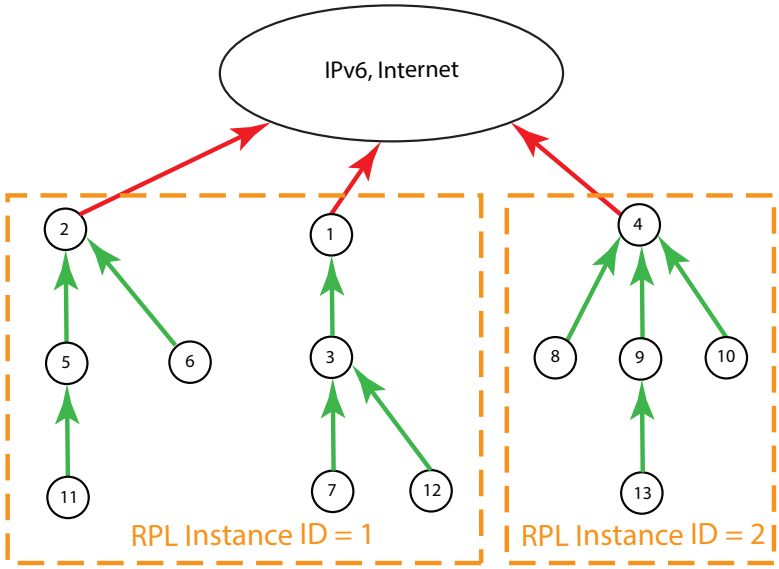


Figura 6: Instancias RPL. A la izquierda se presenta una Instancia RPL que cuenta con dos DODAGs, mientras que a la derecha la Instancia RPL posee un único DODAG.

4.2 Mensajes de Control en RPL

Existen cuatro tipos de mensajes principales soportados por RPL:

- **DODAG Information Object (DIO).** Son mensajes broadcast que se envían y que contienen información que permite a un nodo descubrir una Instancia RPL, aprender la configuración de sus parámetros y seleccionar un set de nodos llamados padres. Estos mensajes se usan para la creación de rutas ascendentes y ayudan a mantener el DODAG.

- **DODAG Information Solicitation (DIS).** Este mensaje puede ser utilizado por un nodo para conocer sus vecinos y requerirles un mensaje DIO. Gracias a ello, luego podrán unirse al DODAG.

- **Destination Advertisement Object (DAO).** Son mensajes unicast que se utilizan para propagar la información de ruteo inversa. Es decir, en estos mensajes los nodos visitados en la ruta ascendente van quedando registrados, y ello en última instancia permite la creación de rutas descendentes.

- **Destination Advertisement Object Acknowledgement (DAO-ACK).** Es un mensaje unicast que se envía en respuesta a la recepción

de un mensaje DAO, dependiendo si el mismo es requerido en el mensaje original.

4.3 Tráfico Soportado

RPL soporta tres paradigmas de comunicación diferentes:

1. **Multi-Point-to-Point (MP2P).** Es el tráfico referido a la colección de información: los múltiples nodos de la red envían paquetes hacia la raíz del DODAG. Los paquetes atraviesan la red siguiendo lo que se conoce como rutas ascendentes. RPL fue pensado principalmente para este tipo de tráfico y por ello su funcionamiento se encuentra optimizado en función del mismo.

2. **Point-to-Multi-Point (P2MP).** Representa el tráfico en que va desde la raíz del DODAG hacia el resto de los nodos que componen la red.

3. **Point-to-Point (P2P).** Se da cuando nodos cualesquiera de la red se comunican entre sí. Si los mismos se hallan dentro del rango de transmisión, se comunican directamente entre ellos, y si no se requiere cooperación de la red.

Se destaca que en el caso de tráfico MP2P, para construir las rutas ascendentes sólo resulta necesario el uso de mensajes DIO, y posiblemente DIS también. En cambio, para el tráfico P2P y P2MP, RPL requiere el uso de mensajes DAO y opcionalmente DAO-ACKs también.

4.4 Construcción del DODAG

Para que la red pueda operar deben construirse las rutas ascendentes. Complementariamente, también puede resultar necesaria la construcción de rutas descendentes. A modo genérico se habla de la construcción del DODAG. Dicho proceso se basa en el Descubrimiento de Vecinos (ND), el cual posee dos modos de operación: (1) la transmisión mensajes DIO por parte de la raíz del DODAG hacia los nodos clientes, y (2) el uso de mensajes DAO de los nodos clientes hacia la raíz del DODAG. A continuación se describen ambos procesos, también explicados en [5].

4.4.1 Rutas Ascendentes

Los nodos anuncian su presencia, dan a conocer en qué DODAG se hayan afiliados y su respectivo Rango e informan el set de métricas y restricciones que condicionan a la Función Objetivo, todo ello a través del envío de mensajes DIO [6]. El proceso comienza desde la raíz del DODAG y se va extendiendo a los demás nodos. Se presentan dos situaciones:el nodo receptor desea unirse a la red o ya forma parte de ella.

En el caso de que el nodo receptor quiera unirse al DODAG, el mismo toma la dirección del nodo transmisor y la agrega a la lista de lo que se llama "set de padres". Además, basándose en la información referida a la Función Objetivo, el nodo computa su propio Rango, de manera tal que el mismo sea mayor al del nodo transmisor. Finalizado el proceso, el nuevo nodo partícipe de la red comenzará él también a enviar mensajes DIO, pero en los cuales ahora actualiza la información del Rango, incluyendo el propio.

Por otro lado, si el nodo receptor ya se hallaba unido al DODAG y recibe un mensaje DIO, primero verifica según los criterios establecidos en las políticas de RPL que el mensaje no esté malformado. Un mensaje malformado debe ser descartado, y en el caso contrario procesado. En el caso de que el mensaje sea analizado, el nodo receptor toma el valor Rango1 del Rango del nodo receptor informado en el mensaje, y lo compara con el Rango2 suyo:

- Si el Rango2 es mayor al Rango1, el nodo receptor se halla respecto del nodo transmisor en una posición más lejana a la raíz del DODAG, y por ende prosigue a agregarlo al set de padres. A su vez, el nodo receptor recalcula el valor de su Rango2: si obtiene un valor menor al que tenía antes, lo adopta como su nuevo Rango, y si no mantiene el mismo. Si el Rango se ve actualizado, entonces debe actualizarse no sólo el valor informado en los próximos mensajes DIO, sino que también el set de padres, eliminando de la lista a los nodos que ahora hayan pasado a presentar un Rango menor al nuevo valor.

- En el caso contrario, el nodo receptor presenta una posición más cercana a la raíz del DODAG, y por ende el mensaje recibido no reviste mayor interés.

Es así que los nodos van aprendiendo quienes son sus vecinos y van creando sus respectivos sets de candidatos vecinos. RPL especifica que el mismo es un subconjunto de los nodos que pueden ser alcanzados mediante enlaces locales multicast, pero no aclara la manera en la cual un nodo en particular puede o no pasar a formar parte de dicho subconjunto. Es decir, se lo deja como a definir por la implementación y la Función Objetivo. A pesar de ello, sí se define que dentro de este subconjunto se encuentra el anteriormente mencionado set de padres de un nodo, que se caracteriza porque todos sus miembros poseen un Rango estrictamente inferior al del nodo en cuestión.

Excepto la raíz del DODAG, todos los nodos deben elegir de su set de padres a lo que se llama el padre preferido o padre por default, que es el nodo separado un salto al cual se le enviarán los paquetes de información. Este comportamiento se repite en todos los nodos, y el encadenamiento de transmisiones a los padres por default termina por construir un camino hacia la raíz del DODAG, o lo que se conoce más genéricamente como una ruta ascendente. A su vez, en el caso de que el padre preferido no esté disponible para participar en el ruteo de los paquetes de información hasta la raíz del DODAG, el nodo puede seleccionar alternativamente otro padre dentro del set de padres e intentar a través de él.

Las reglas establecidas anteriormente aseguran que habrá un ordenamiento parcial consistente entre los nodos que conforman el DODAG. Mientras el Rango de cada nodo no cambie, resulta asegurado que la ruta ascendente creada hacia la raíz del DODAG no tendrá bucles, ya que el Rango decrecerá en cada salto a medida que el paquete de información se acerca a la raíz del DODAG.

Por último, cabe destacar que cuando un nodo recién se enciende, el mismo tiene dos opciones:

1. Simplemente quedarse en silencio, esperando recibir un mensaje DIO, como ya descrito.

2. Enviar uno o varios mensajes de control DIS requiriendo a los nodos receptores que envíen mensajes de control DIO, de manera de poder unirse al DODAG.

4.4.2 Rutas Descendentes

Los nodos son conscientes de la necesidad o no de crear rutas descendentes gracias a la información que obtienen en mensajes DIO. Estas rutas son construidas y mantenidas mediante el uso de mensajes DAO y resultan necesarias en los casos en que la red soporte tráfico P2MP o P2P. Se define el modo de operación, el cual condiciona la manera en que la manutención de las rutas descendentes se lleva a cabo. Existen dos modos:

- **Modo Almacenado.** Los nodos envían mensajes DAO a sus padres por default. Ello permite que un nodo sepa que posee uno o múltiples nodos hijos. El nodo receptor luego agrega la información concerniente a su existencia y continúa con la propagación de mensajes DAO, proceso que se repite hasta llegar a la raíz del DODAG. Lo que distingue a este caso es que a medida que los nodos reciben los paquetes de control, los mismos van completando una tabla de ruteo descendente y la misma queda almacenada en ellos.

- **Modo No Almacenado.** A diferencia del caso anterior, los mensajes DAO son enviados a la raíz del DODAG. Los nodos intermedios no almacenan tablas de ruteo descendentes: simplemente agregan su información a la tabla de ruteo inversa antes de continuar con el proceso de transmisión de mensajes DAO.

Para el caso de tráfico P2P la primera opción es que los nodos se hallen separados un único salto, y RPL permite que esta comunicación se realice en forma directa. Si los nodos se hayan a mayor distancia, entonces la manera de seguir dependerá del modo de operación.

Para el Modo No Almacenado el paquete se envía a través de la ruta ascendente. Si la destinación no se halla en dicho dicho camino, entonces el paquete recién comenzará su ruta descendente en la raíz del DODAG, y llegará a destino sólo mediante el uso de un ruteo completo. En cambio, en el Modo Almacenado, el nodo primero podrá chequear en su propia tabla de ruteo inversa si el destinatario se encuentra en su misma sub-DODAG y así podrá enviarle directamente el paquete. Caso contrario, el paquete seguirá el mismo camino que antes, a excepción de que el nodo transmisor y destinatario tengan ambos un ancestro en común. En dicho caso, este nodo podrá

redirigir el paquete hacia el destino final, sin la necesidad de que siga ascendiendo hasta la raíz del DODAG.

Por otro lado, como el tráfico MP2P se genera en la raíz del DODAG, el Modo de Operación lo único que define es la cantidad de información de ruteo que la raíz del DODAG deberá precisar para que el paquete llegue a destino.

4.5 Estabilidad del DODAG

El temporizador Trickle [7] es utilizado para definir de manera adaptativa la tasa de envío de mensajes de control. En él se habla que cuando existe consistencia en la información, entonces se disminuye en forma exponencial la frecuencia con la cual los nodos se comunican, incluso reduciéndose a pocas veces por hora. Sin embargo, cuando se detecta una inconsistencia, los nodos se comunican rápidamente para resolverla. Sin embargo, es tarea de cada protocolo en particular definir qué son consistencias e inconsistencias.

En RPL, si la red es estable, entonces el conocimiento que un nodo posee sobre la topología de la misma se condice continuamente con la información contenida en los mensajes DIO que el nodo recibe. Es ello lo que se define como información consistente y lo que lleva a que se vaya reduciendo más y más la cantidad de mensajes DIO que son enviados. En cambio, una inconsistencia se define por eventos como ser la detección de loops en la red, la unión de un nodo a la misma o simplemente cambios en el Rango de algún nodo que pueden llevar a que la dirección en la que un paquete es enviado no se condiga con lo esperado. En estos casos el tiempo de espera entre mensajes de control se resetea y comienzan a ser enviados en forma más continua.

En conclusión, la reparación de rutas se lleva adelante con una técnica reactiva a los cambios que son detectados, la cual permite solucionar las inconsistencias detectadas en la información y volver a un estado estable de manera más rápida. Por otro lado, si la estabilidad de la red se mantiene en el tiempo, se necesitan menos paquetes de control.

4.6 Función Objetivo

De manera de poder ser útil en un amplio rango del domino de aplicaciones de las LLNs, RPL separa el procesamiento y envío de paquetes del objetivo de optimización de enrutamiento. Algunos de dichos objetivos incluyen minimizar el consumo de energía, minimizar el retardo, o satisfacer restricciones propias de cada aplicación, entre otros.

La Función Objetivo, en conjunto con las métricas de ruteo que se definan en la red y las restricciones de la misma, permite que un nodo pueda llevar adelante la elección del DODAG a la cual se une. Todos los nodos de la red comparten la misma Función Objetivo: la misma es aprendida gracias al uso de mensajes DIO. Dentro de las opciones de configuración de estos mensajes existe el llamado campo Punto de Código Objetivo (OCP) que determina cuál de todas debe utilizarse.

La Función Objetivo determina cómo RPL selecciona los caminos [8], es decir, cómo los nodos eligen y optimizan las rutas dentro de una Instancia RPL. Definida la Función Objetivo, entonces los distintos nodos que componen la red poseen una manera de computar el Rango. A partir de éste, para los nodos resulta posible determinar no sólo su set de padres, sino que su padre preferido también. En definitiva, la Función Objetivo implícitamente es la responsable de cómo se forma el DODAG.

Existen dos Funciones Objetivo:

- **Función Objetivo Cero (OF0)** [9]. Se define como métrica la cantidad de saltos necesarios para llegar a la raíz del DODAG.
- **Función Objetivo Cero de Rango Mínimo con Histéresis (MRHOF)** [10]. Selecciona el camino que minimiza el valor de la métrica considerada, definida según lo indicado en los mensajes DIO. Por ejemplo, de utilizarse el retardo, RPL construye un camino estable que presente el menor retardo entre los otros posibles. Otra métrica que se utiliza es conocida como Expected Transmission Count (ETX) [11]. La ETX de un enlace inalámbrico se estima como el número promedio de transmisiones de paquetes y ACKs que son necesarios para lograr que la

transmisión de un paquete sea exitosa, y es la métrica que se usa por default. El concepto de histéresis está relacionado a que un nodo cambia su padre preferido sólo si el nuevo candidato reduce el costo asociado a su camino comparado con el actual en un valor mayor a cierto umbral.

El poder elegir la Función Objetivo y separarla de la especificación del núcleo del protocolo es lo que le otorga a RPL la posibilidad de adaptarse a diferentes criterios de optimización requeridos por distintas aplicaciones y diseños de redes.

5 IEEE 802.15.4e - TSCH

Los estándares IEEE 802.11 (WiFi) y IEEE 802.15.1 (Bluetooth) poseen una elevada tasa de transferencia de datos, pero no soportan redes con muchos nodos, especialmente con bajo consumo de energía. El estándar IEEE 802.15.4 resulta más acorde que el IEEE 802.15.1 ya que realiza un consumo económico de la batería, posee un rango de comunicación mayor, flexibilidad en el tamaño de la red, entre otras [12].

La capa física del estándar IEEE 802.15.4 fue diseñada para soportar comunicaciones en rangos cortos y ofrece a la industria una solución para aplicaciones de baja tasa de datos, costo y consumo de energía. Su buena performance probada en entornos industriales demostró su utilidad para este tipo de aplicaciones, y por ello no sólo forma parte de los estándares ZigBee, WirelessHART, ISA100.11a, WIA-PA sino que para 6TiSCH también fue la opción adoptada.

Por otro lado, el esquema de la capa MAC de IEEE 802.15.4 presentaba ciertas deficiencias comparado con las propiedades de tiempo real y de confiabilidad requeridas para aplicaciones industriales. Entre los principales inconvenientes, el consumo de energía era elevado y se utilizaba un único canal de transmisión, volviéndolo susceptible a interferencia y desvanecimiento de múltiples caminos.

El comité de IEEE 802.15.4 creó un grupo de trabajo llamado IEEE 802.15.4e en 2008, con el objetivo de mejorar el protocolo de la capa MAC y de

volverlo adecuado para aplicaciones industriales con redes de múltiples saltos y bajo consumo. En 2012 se publica el nuevo estándar IEEE802.15.4e, el cual básicamente rediseña la capa MAC y que soporta tres modos de operación: Time-Slotted Channel Hopping (TSCH), Deterministic and Synchronous Multi-channel Extension (DSME) y Low Latency Deterministic Networks (LLDN).

El grupo de trabajo de 6TiSCH optó por la versión IEEE802.15.4e-TSCH. El uso de TSCH permite conseguir un reducido consumo gracias a la sincronización temporal mientras que el salto de frecuencias aumenta la robustez de la red, brindándole una mayor confiabilidad [13].

Figura 7: Timeslots.

En definitiva, este estándar sí provee capacidades determinísticas al punto de que un paquete que pertenece a cierto flujo puede atravesar la red de nodo a nodo siguiendo un orden de programación muy preciso [1].

5.1 Time Slotted

Como se mencionó, la sincronización temporal permite reducir el consumo de energía y saber con precisión qué está sucediendo en la red a cada instante. A continuación se describen los principales conceptos que caracterizan este comportamiento.

5.1.1 Timeslots

En el nombre TSCH, TS se refiere a Time-Slotted, y ello hace referencia al hecho que el tiempo se halla dividido en ranuras de tiempo, más popularmente llamadas timelosts. Esto puede observarse en la figura 8.

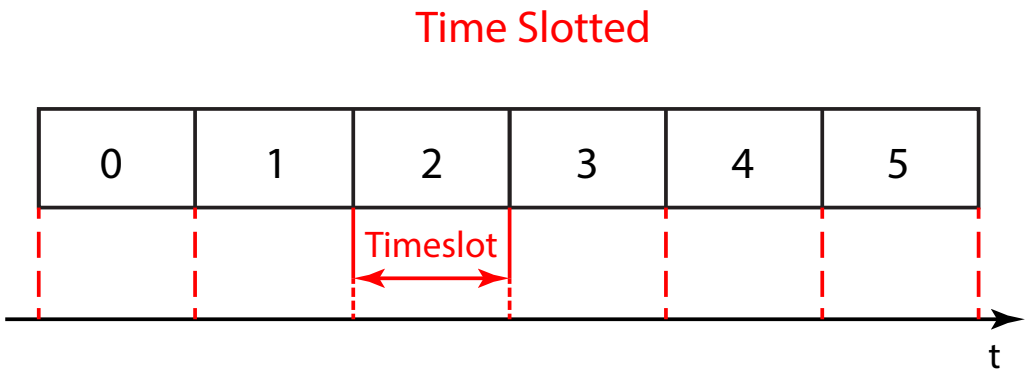


Figura 8: Timeslots.

En cada timeslot debe definirse lo que todos los nodos de la red harán. Existen tres posibilidades:

- Transmitir un paquete difundido únicamente hacia a algún vecino en particular o de difusión ancha para todos ellos,
- Recibir un paquete proveniente de algún vecino dentro del rango de la antena,
- Dormir, o en otras palabras, simplemente apagar su radio.

Una vez que se tienen los distintos timeslots, entonces debe procederse a reservarlos para que en cada uno de ellos se lleve adelante alguna de las tareas recién mencionadas. Sin embargo, es importante notar que el reservar un timeslot para transmitir simplemente crea la oportunidad para que un nodo transmita. Esto significa que el timeslot será únicamente utilizado en los casos en que el nodo que transmite realmente cuente con un paquete que debe ser entregado a su vecino. Si este no fuera el caso, el nodo transmisor simplemente no prenderá su radio para transmitir. En contraste con ello, el nodo receptor no posee manera de saber si el nodo transmisor enviará un paquete o no, es decir, este evento no puede prevenirse, y por ende el nodo receptor no cuenta con un método que le permita evitar prender del todo su radio. Lo que sucederá es que el nodo chequeará si hay un paquete en el medio, y si después de cierto tiempo no recibe nada, entonces apaga su antena. Esto permite ahorrar parcialmente la energía que se desperdiciaría en recibir un paquete que no es enviado.

La duración de un timeslot no se encuentra definida por el estándar, y por ende resulta uno de los parámetros a definir por parte del administrador de la red. Para ello, debe tenerse en cuenta que no tendría sentido definir un tiempo muy extenso, ya que ello no permitiría que se establezcan muchas comunicaciones en un período acotado de tiempo. Por otro lado, debe tenerse cuidado de que la duración del mismo no sea muy pequeña, puesto que esto podría producir que las transmisiones se vean cortadas antes de haber transmitido el paquete entero con éxito. Además, debe tenerse en cuenta que dentro de un timeslot, no es que simplemente un nodo debe transmitir el paquete hacia el destino, sino que también debe recibir un acknowledgement (ACK) del mismo.

Según lo definido en [13], un buen valor de referencia resulta ser 10 ms. El mismo provee suficiente tiempo para que un marco MAC de tamaño máximo sea enviado y que su respectivo ACK asociado sea recibido. Además, trambién provee cierto tiempo para la transición de la radio entre apagado y encendido, para el procesamiento del paquete y para llevar adelante tareas relacionadas con seguridad. Sin embargo, dependiendo de las limitaciones del hardware, este valor de referencia puede no ser suficiente como para permitir que todos estos pasos se lleven adelante, y entonces de manera más general se definen valores típicos que van desde 10 hasta 15 ms, como mencionado en [1].

El beneficio de basar el sistema en una operación de sucesión de timeslots es que, como el principio y fin de cada timeslot está claramente definido, entonces no habrán dos transmisiones que sucedan al mismo tiempo. Esto, se traduce en una reducción de la probabilidad de que resulten colisiones. Además, como los nodos sólo deben despertarse en algunos timeslots, ya sea para transmitir o para recibir, entonces se mantienen en reposo en el resto de ellos, y ello consecuentemente produce un ahorro de energía. Por último, la posibilidad de elegir qué nodo transmite primero y a quién, y el timeslot en particular, permiten diseñar el funcionamiento de la red, de manera de proveerle propiedades determinísticas.

5.1.2 Slotframe

Una colección de timeslots que se repiten en el tiempo forman lo que se conoce como slotframe [14]. De esta manera, la red presenta un comportamiento

cíclico: cada nodo en un timeslot dado se encuentra a sí mismo llevando adelante, o al menos con la posibilidad de realizarlo, aquella misma cosa que hizo en ese mismo timeslot del slotframe anterior.

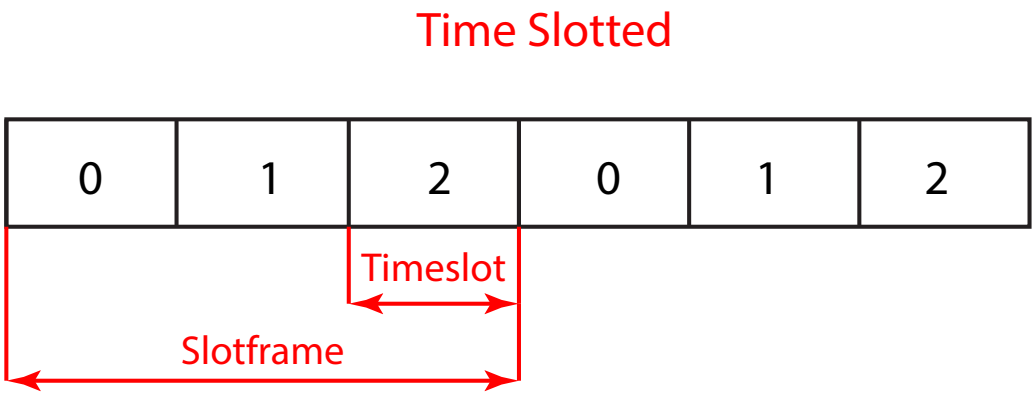


Figura 9: Slotframe.

El tamaño del slotframe es la cantidad de timeslots que el mismo contiene. Dentro de un slotframe, los timeslots se indexan según su SlotOffset, el cual empieza en 0 y da cuenta de la posición del timeslot dentro del slotframe. Así como la duración de los timeslots, el tamaño del slotframe no se encuentra definido por el estándar y es un valor que también debe ser decidido a la hora de configurar la red. En este punto se presenta una relación proporcionalmente directa entre el ancho de banda disponible y el consumo de energía. Si un slotframe es pequeño, entonces en un mismo período de tiempo, los timeslots se repiten de forma más seguida y por ende los nodos se despiertan para transmitir o recibir más seguido también. Ello crea más oportunidades para que más paquetes sean transmitidos, y entonces el ancho de banda disponible se ve automáticamente aumentado, pero no sin suponer también un mayor consumo de energía asociado a la acción de despertar a los nodos una mayor cantidad de veces. En forma opuesta, si sólo una cantidad definida de timeslots resultaran necesarios para componer un slotframe y el tamaño del mismo se hiciera mayor, conteniendo así ciertos timeslots en los que simplemente todos los nodos descansan, la repetición de las transmisiones para un mismo período de tiempo se vería disminuidas y así también el consumo de energía.

En este sentido, una métrica de importancia resulta el llamado ciclo de trabajo de la red. Este valor representa la cantidad de veces que los nodos se despiertan en un slotframe, y por ende tienen una relación directa con el

consumo de energía, factor más aún importante teniendo en cuenta que las redes analizadas trabajan con dispositivos de baja potencia y con pérdidas.

Por último, cabe destacar que un nodo puede poseer múltiples slotframes de distintos tamaños a la misma vez, por lo que cada uno de ellos se ve identificado con un Slotframe ID propio. Basándose en la prioridad de los distintos flujos de paquetes y tráfico que un nodo posee, entonces cada una de estas necesidades podría programarse en un slotframe diferente.

5.1.3 Absolute Slot Number (ASN)

TSCH define un contador de timeslots llamado Número de Ranura Absoluto [15]. Cuando una red es creada, el ASN comienza en cero y luego, a medida que transcurre cada timeslot, aumenta en una unidad. En la figura 8, la numeración contenida en cada timeslot puede ser interpretada como el valor del ASN a medida que transcurre el tiempo.

Un nodo aprende el valor del ASN cuando se une a la red. Dado que todos los nodos se hayan sincronizados, entonces todos saben el valor exacto del contador en cualquier momento. En general, se cumple que:

$$\mathrm{ASN} = \mathrm{m} * \mathrm{SS} + \mathrm{CT}$$

donde m representa la cantidad de veces que el slotframe se ha repetido desde el inicio de las red, SS es el tamaño del slotframe y CT representa el timeslot actual.El ASN está compuesto por 5 bytes, y ello permite que su valor nunca desborde.

Este valor es importante ya que luego es utilizado para determinar la secuencia de frecuencias que sigue el salto de frecuencias y también puede ser utilizado para llevar adelante tareas relacionadas con seguridad.

5.2 Channel Hopping

La capa física de IEEE 802.15.4 define en total 27 canales en 3 bandas de frecuencia distintas [16]. En particular resulta de interés la banda de frecuencia que permite una mayor tasa de bits y que se encuentra en el orden de los 2.4 GHz. Esta banda de frecuencias posee 16 canales, y es la utilizada en

6TiSCH.

Uno de los atractivos de utilizar frecuencias en dicho orden es las mismas no se encuentran licenciadas a nivel mundial, y por ende pueden usarse libremente. Si bien ello es cierto, la banda de frecuencias coincide con las utilizadas por WiFi. Dado que las señales que se utilizan en uno y en otro caso son bastante diferentes, cada sistema interpretará a la señal del otro como interferencia. Además, a ello se suma el desvanecimiento por múltiples caminos, y por ende la capacidad de poder llevar adelante comunicaciones puede verse notoriamente perjudicada.

El salto de canales o channel hopping es una técnica simple que permite combatir ambos problemas de manera efectiva [17]. La idea consiste en utilizar los 16 canales de frecuencia para llevar adelante las transmisiones de paquetes. Para ello se define como parámetro de interés el channel offset, a partir del cual finalmente resulta posible definir la frecuencia en la cual la comunicación será llevada a cabo:

$$\mathrm{F} = \mathcal{F}\big[(\mathrm{ASN} + \mathrm{CO}) \ \% \ \mathrm{nF}\big]$$

donde ASN es el Número de Ranura Absoluto, CO representa el channel offset y nF es el número de frecuencias disponibles, que en este caso son 16. A su vez, $\mathcal{F}$ es una función a definir que crea lo que se conoce como la secuencia de saltos [14]. En sí, simplemente trabaja como una tabla de búsqueda, que según el valor de los otros 3 parámetros, los asocia a alguna frecuencia entre las disponibles en la banda de frecuencias. Si bien nF puede estar dado, la secuencia de saltos puede tener una longitud mayor que ella.

Los nodos además de definir en qué timeslot se comunicarán, ahora también deben definir un valor de channel offset. Con él, los nodos poseen toda la información que se necesita para saber en cual frecuencia se realizará efectivamente la comunicación. Además en el siguiente slotframe, aunque los nodos conserven el channel offset igual, dado que el ASN cambia continuamente, la frecuencia computada será distinta. Para asegurar que todas las frecuencias son recorridas en su totalidad alcanza con definir el tamaño del slotframe como un número primo.

En conclusión, así definido, el sistema implementa un salto de frecuencias: a pesar de que las transmisiones programadas no cambien, las comunicaciones entre pares de nodos siempre suceden a frecuencias distintas.

5.3 Scheduling

Al combinar el uso de timeslots con channel hopping, o dicho de otra manera, la multiplexación por división de tiempo (TDM) y la multiplexación por división de frecuencias (FDM), finalmente se obtiene una matriz llamada la matriz de distribución/uso de canales (CDU), aunque resulta mucho más común referirse a ella como el "schedule".

En esta matriz, su ancho representa el eje temporal dividido en ranuras de tiempo y de período equivalente a la cantidad de éstos que componen un slotframe, y su alto está represntado por la cantidad de canales distintos de frecuencias que se encuentran disponibles. En las secciones anteriores hacía falta definir qué hacía cada nodo en cada timeslot, y ello se mantiene aún, solo que también debe definirse en qué frecuencia. Ello hace que, en el modo más general, en verdad debe definirse lo que hace cada nodo en cada celda de la matriz, las cuales se ven identificadas cada una de ellas por un slotOffset y un channelOffset particular. Una vez que esto se haya decidido, se dice que se tiene el "scheduling" de la red. En la figura 10 puede verse un ejemplo en el cual en cada celda donde aparece la notación "A -> B", la misma hace referencia a que en dicho timeslot y channel offset, el nodo A tiene la posibilidad de transmitir un mensaje al nodo B. Por otro lado, las celdas que se encuentran vacías significa que no hay transmisiones programadas, o que los nodos estarán en modo idle para esos casos.

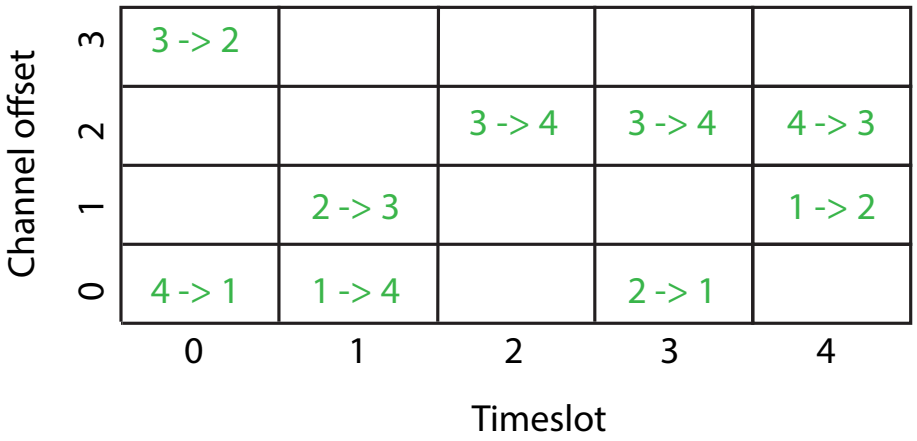


Figura 10: Scheduling.

De esta manera, un scheduling efectivo permite que múltiples comunicaciones se lleven adelante en el mismo instante de tiempo en un mismo dominio de interferencia, pero utilizando diferentes canales. Sin embargo, los nodos vienen equipados con una única antena y por ende sólo pueden transmitir o recibir en un único canal en un punto dado del tiempo. Por ello, en el caso de uso de más de un slotframe, entonces también deberá asignarle un grado de prioridad a cada uno de ellos, de manera de que un nodo sepa cual necesidad atender primero en caso de que dos eventos se solapen.

Los nodos construyen el schedule y asignan a sus vecinos cierta cantidad de celdas, y viceversa. Un nodo puede poseer múltiples celdas equivalentes hacia cierto vecino, en el sentido de que a nivel de la capa MAC, un paquete que se halle presente y necesite ser enviado a dicho vecino, lo será en la primera de las celdas que aparezca luego de que el paquete fue puesto en la cola. La unión de todas las celdas entre dos nodos vecinos se llama “bundle”. En la figura 11 pueden verse los bundles que había presente con anterioridad en el scheduling de la figura 10.

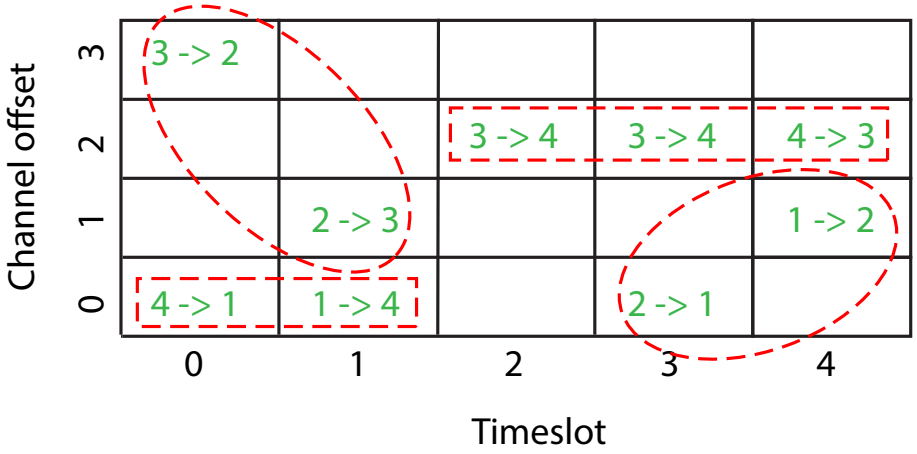


Figura 11: Bundle.

A su vez dado que los slotframes se repiten continuamente en el tiempo, otorgarle a un vecino una cantidad de celdas significa entregarle al mismo un quantum del ancho de banda disponible. En definitiva, el modificar la cantidad de celdas destinadas en un bundle modifica los recursos que se encuentran reservados entre ambos vecinos.

Por otro lado, las celdas de la matriz pueden ser dedicadas o compartidas. Por default, las mismas son dedicadas, refiriéndose a que sólo un nodo puede transmitir a otro nodo allí. Sin embargo el estándar IEEE 802.15.4e también permite que una celda sea identificada como compartida, o en ingles shared. Este es el caso de las celdas utilizadas en los timeslots 2 y 3 de la figura 12.

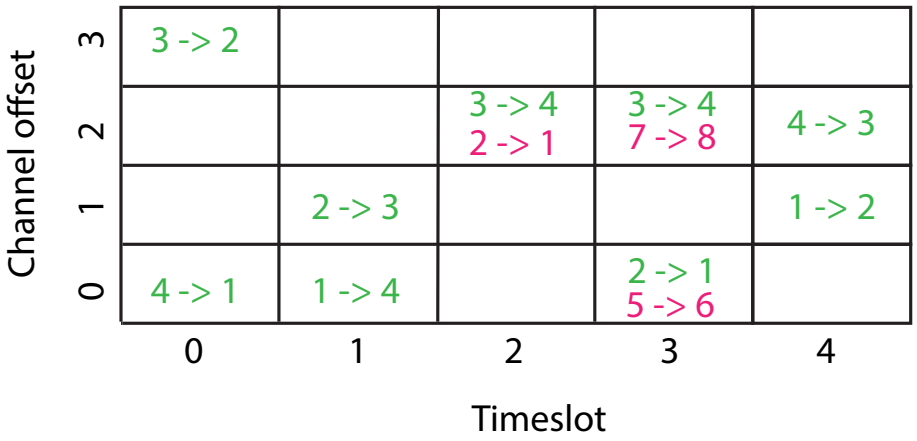


Figura 12: Celdas dedicadas o compartidas.

Al compartir celdas, múltiples nodos poseen la oportunidad de transmitir al mismo tiempo y en la misma frecuencia. Sin embargo, existe un mecanismo de control que provee la precaución de que esto no suceda, de manera de que los paquetes no interfieran entre sí. Las celdas compartidas muchas veces son aquellas en que los nodos no sólo transmiten, sino que también escuchan. En dichos casos, un nodo transmite cuando tiene un paquete en su buffer, y en caso contrario simplemente escucha. Esto permite ahorrar energía dado que se agrupan varios eventos en un mismo timeslot y por ende los nodos no deben encender su antena a diferentes instantes de tiempo. El costo asociado a esta solución es que no siempre se hallará disponible la posibilidad de transmitir un paquete.

5.4 Proceso de Unión

Los nodos que ya forman parte de la red envían periódicamente marcos de "Enhanced Beacons" (EBs) para anunciar su presencia en la red [18]. Los beacons son mensajes con información específica que es utilizada justamente y entre otras cosas, para identificar la red y para sincronizar los dispositivos de la misma.

El estándar IEEE 802.15.4 no sólo no define qué tan seguido los nodos deben enviar EBs, sino que tampoco especifica su contenido [19]. Los nodos deberían transmitir estos paquetes cada cierto período EB_PERIOD a

definir, y cuyo valor permite regular la relación de compromiso existente entre el tiempo de unión y conformación de una red, y el consumo de energía asociado. La información contenida en general incluye el tamaño del timeslot usado en la red, el valor actual del ASN, información de los slotframes y timeslots que el nodo receptor se haya escuchando y una prioridad de unión de 1 byte [13]. Este último campo brinda información que permite decidir a cual nodo escuchar.

Si bien un nodo puede estar configurado para enviar todos sus EBs en una celda definida, gracias al uso de channel hopping, en definitiva en cada slotframe la frecuencia física varía y entonces los EBs resultan enviados en todas las frecuencias.

Cuando un nodo se enciende, el mismo naturalmente se encuentra en un estado de desincronización y sin estar unido a la red. La sincronización inicial se obtiene escuchando estos EBs: dado que éstos son enviados en todas las frecuencias, el nodo que desea unirse puede escuchar en cualquiera hasta recibir un EB. Un nodo puede recibir EBs de múltiples redes y aplicar algún mecanismo que le permita discriminar a cual de ellas unirse. Al recibir un EB, el nuevo nodo de la red obtiene el valor del ASN y otra información complementaria relativa al timing que le permite sincronizarse con la red y en última instancia poder habilitar completamente el uso del modo TSCH. Utilizando la información concerniente a la celda y al slotframe, el nodo va aprendiendo a cómo contactarse con otros nodos de la red.

5.5 Sincronización temporal

Dado que al utilizar TSCH el tiempo se haya dividido en ranuras de tiempo, resulta entonces extremadamente importante que exista una coordinación temporal que permita llevar adelante las comunicaciones en la red. Todos los nodos poseen sus relojes propios y ello les permite tener un seguimiento del tiempo. Sin embargo, existe una deriva distinta entre los distintos relojes, y ello, por más que sea pequeña, dado suficiente tiempo puede provocar una desalineación temporal. Por ello, los nodos vecinos deben sincronizarse periódicamente.

De manera de lograrlo, los nodos comparan su valor con el de algún vecino. Además de ello, los nodos deben compartir el valor de su reloj con el resto de sus vecinos. Dependerá de la configuración de la red a cuál nodo cada nodo tomará como vecino de fuente temporal. Al hacer ello, es importante evitar la formación de bucles, ya que los mismos pueden resultar en la formación de clusters independientes de nodos sincronizados.

Con el uso de TSCH, todos los paquetes que son intercambiados contienen información temporal, sean paquetes de información o ACKs. Ello significa que nodos vecinos pueden resincronizarse entre ambos mediante éstos. En detalle, el estándar IEEE 802.15.4e define dos métodos para permitir que un dispositivo se sincronice a una red TSCH [13]: (i) sincronización basada en ACKs y (ii) sincronización basada en marcos. En ambos casos, el receptor calcula la diferencia entre el tiempo esperado de arribo del marco y su verdadero tiempo de arribo. En el caso (i), el nodo receptor informa esta diferencia al nodo transmisor en un ACK, y es este último el que corrije su reloj en función del del nodo receptor. En cambio, en el caso (ii) el nodo receptor simplemente calcula la diferencia y la utiliza para modificar su propio reloj, sincronizándose así con el nodo transmisor.

Existen distintas políticas de sincronización que pueden llevarse adelante: los nodos pueden utilizar exclusivamente el intercambio de EBs, o pueden utilizar la información en los paquetes de información. También resulta posible utilizar ambos mecanismos a la vez, y depende del administrador de la red definirlo. nization policies are possible.

6 6top

Si bien el estándar IEEE 802.15.4e define el mecanismo mediante el cual un nodo TSCH se comunica, el mismo no define las políticas para construir y mantener la programación que determina el ordenamiento de las comunicaciones y tampoco aclara la manera en que la misma se ajusta a los caminos multi-salto mantenidos por RPL. A su vez, tampoco menciona cómo se deben adaptar los recursos reservados entre dos nodos vecinos para soportar los flujos de tráfico de información, ni aclara si debe haber un tratamiento diferente para mensajes generados en la capa de aplicación o necesitados por 6LoWPAN y RPL para descubrir vecinos. Menos aún da las pautas en cuanto a qué hacer ante cambios en la topología de la red, entre otras cosas.

En definitiva, una red 6TiSCH se encuentra orquestada por una programación de comunicaciones, pero basándose únicamente en el estándar IEEE 802.15.4e, la misma no está definida. En dicho sentido, 6top resulta la subcapa propuesta como control de enlace lógico entre la capa IP y las capa MAC TSCH. Para ello, implementa el protocolo 6P.

El protocolo 6P permite que dos nodos vecinos agreguen, eliminen y relocalicen celdas que utilizan en su scheduling. Conceptualmente, los nodos "negocian" la cantidad de celdas que necesitan para tener una comunicación eficiente y el momento y el canal en el cual la llevarán adelante.

El protocolo 6P propone que los nodos realicen "transacciones 6P" para ponerse de acuerdo en su sheduling y lograr así el arribo a una solución en la cual haya consistencia con sus respectivas programaciones. El disparador de estas negociaciones se determina mediante el uso de una Función Programadora (Scheduling Function, en inglés).

La Función programadora es la entidad de manejo de las celdas TSCH y controla la adjudicación dinámica de las mismas adaptándolas a los requerimientos de un flujo de información cuyo tráfico sea variable [1]. Existen dos Funciones Programadoras actualmente siendo desarrolladas: la Scheduling Function Zero (SF0) [20] y la Autonomous Scheduling Function (ASF) [21]. Mientras que la SF0 propone una solución que adapta la cantidad de

celdas destinadas a cada comunicación en función del tráfico que cada nodo experimenta, la ASF simplemente establece los mecanismos para establecer una comunicación mínima entre nodos. Se destaca que en ninguno de ambos casos la solución apunta a optimizar el retardo ni su variación, la cual los paquetes experimentan extremo a extremo.

Si bien hasta el momento sólo estas dos soluciones se hayan en discusión, existe la posibilidad de definir otras nuevas. Es en este sentido que la propuesta de esta tesis, LeapFrog Collaboration, haya espacio para eventualmente ser adaptado a una red 6TiSCH. Sin embargo, una diferencia importante es que, como se verá más adelante, LeapFrog Collaboration no es un simple draft orientado a que luego una empresa defina de manera más exacta cómo se termine por implementar, sino que apunta a minimizar el retardo y su fluctuación.

7 LeapFrog Collaboration

LeapFrog Collaboration es un mecanismo de ruteo de paquetes. El mismo está pensado para ser utilizado en redes 6TisCH, y por ende se centra en el funcionamiento en conjunto RPL+TSCH, que son los protocolos de red y capa de enlace respectivamente.

La esencia de este método es la duplicación del flujo de datos a través de rutas alternativas en cada salto. En contraste con los métodos best-effort, en LeapFrog Collaboration un nodo no espera hasta que una transmisión falle para reenviar un paquete, sino que lo hace constantemente y, además, siguiendo otro camino distinto.

Para ello, cada nodo define además de su Default Parent (es la forma en que en RPL se llama al nodo al cual se enviará el paquete en el siguiente salto, y que se encuentra más cercano a la raíz de la red), un Alternative Parent al cual también le envía cada uno de los paquetes que poseen información. A su vez, en LeapFrog Collaboration se define el Overhearing, que es un procedimiento que permite que en una única transmisión unicast, más de un nodo pueda recibir al mismo tiempo un paquete. Más aún, se define lo que se conoce como Siblings, donde nodos que se hayan separados una cantidad de saltos equivalentes a la raíz de la red, gracias al Overhearing, comparten entre sí los paquetes de información que tengan, permitiendo así que posteriormente ambos lo puedan reenviar.

En definitiva, todos estos conceptos introducidos crean la posibilidad de que, si por alguna razón un paquete conteniendo información se pierde en el camino original, todavía exista la posibilidad de que la copia del mismo llegue a destino por otro camino. Esto le otorga confiabilidad a la red y le permite mantener un elevado rendimiento al observar la cantidad de paquetes que llegan a destino respecto de los que son enviados. Este parámetro se conoce según su designación en inglés como Packet Delivery Ratio (PDR).

LeapFrog Collaboration crea entre el nodo transmisor y la raíz una sucesión de estructuras de tipo escalera. La idea es crear dos caminos paralelos con una cantidad equivalente de saltos intermedios, y en la que los nodos estén

cercanos entre sí, de manera de permitir una comunicación de los nodos de una rama junto con los de la opuesta. De esta manera, los nodos tienen la posibilidad de enviar los paquetes siguiendo su misma línea, o de cruzarse a la otra. Sin embargo, en LeapFrog Collaboration, se utilizan ambas ramas a la vez: el paquete se envía por la rama original, pero a su vez se duplica a la enfrentada. A través de la lógica implementada, los paquetes quedan confinados dentro de esta estructura, lo cual asegura que el trayecto que seguirá el paquete hacia destino podrá desviarse de un lado a otro, pero siempre avanzará hacia el destino y siguiendo caminos parecidos.

Por otro lado, lo innovador de LeapFrog Collaboration es que no sólo se preocupa por mantener una elevada tasa de paquetes que llegan a la raíz de la red, sino que también se ocupa de que el retardo al cual cada paquete se ve sometido desde que es enviado por el nodo transmisor hasta que efectivamente es recibido por la raíz sea constante. Esta característica es la que lo distingue de los métodos empleados en el estado del arte actual, y que lo alinean con el concepto de redes determinísticas.

Para conseguir un valor estable en el retardo, resulta necesario que los paquetes de información que se van replicando en los distintos caminos alternativos vayan avanzando conjuntamente en la red. Esto permite que al momento de ser entregados a la raíz, entonces todos los nodos separados un salto de la raíz ya posean el paquete que van a transmitir, y por ende se podrá probar secuencialmente con cada uno de ellos.

En conclusión, el esquema propuesto en LeapFrog Collaboration permite prever potenciales errores de enlace en una única ruta al explotar la diversidad de caminos en una red densa. Además, al elegir de manera inteligente el orden en el cual los diferentes nodos transmiten, la variabilidad del retardo al que los paquetes que atraviesan la red se ven sometidos se ve mucho más acotada.

7.1 Alternative Parent (AP)

El Aternative Parent (AP) es la principal componente que caracteriza a LeapFrog Collaboration y mediante la cual la información se envía hacia la raíz a través de múltiples caminos. Ello aumenta considerablemente la con-

fiabilidad de la red, logrando como resultado una performance mucho más cercana a la requerida por una red determinística.

En el funcionamiento básico de RPL, la red posee una topología de tipo árbol donde los distintos nodos intermedios ofrecen una colaboración para llevar el paquete al destino final. Cada nodo transmite el paquete de información a su default parent, hasta llegar a la raíz de la red, que es el destino final. En el ejemplo de la figura 13, el nodo 4 es la raíz, y por ende las rutas que se construyen lo tienen como destino final. El protocolo RPL crea dos ramas: una compuesta por los nodos 1, 2 y 4, y la otra por los nodos 3 y 4.

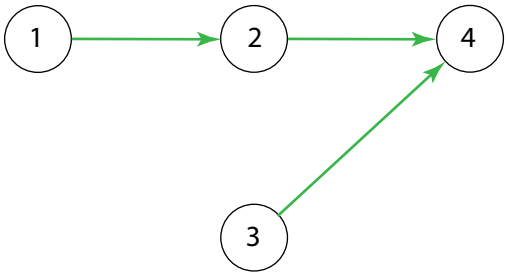


Figura 13: RPL propone una colaboración entre nodos, formando una topología de tipo árbol donde cada nodo transmite el paquete a default parent, hasta llegar a la raíz.

El problema que se suscita es que si en cualquiera de las transmisiones programadas, el paquete no se logra transmitir al siguiente nodo, entonces dado que se utiliza el protocolo TSCH en la capa de acceso al medio, debe esperarse otro slotframe entero para volver a contar con la oportunidad de transmitir el mismo paquete. De esta manera, si bien las retransmisiones permiten mejorar la tasa de entrega de paquetes, las mismas poseen un efecto muy negativo en cuanto a la estabilidad del retardo, el cual varía en forma directamente proporcional a la cantidad de retransmisiones que hayan sido necesarias.

Para mejorar este comportamiento de la red, en LeapFrog Collaboration se propone el uso simultáneo de un camino alternativo. La ventaja de esta propuesta resulta evidente: mientras que con la operación básica de RPL+TSCH perder un paquete significa directamente una retransmisión, con LeapFrog Collaboration el paquete puede perderse en el camino original o alternativo, pero aún así llegar a destino a través del otro camino complementario. Para

ello, cada nodo define además de su Default Parent, a un Alternative Parent al cual también le envía cada uno de sus paquetes, como puede observarse en la figura 14.

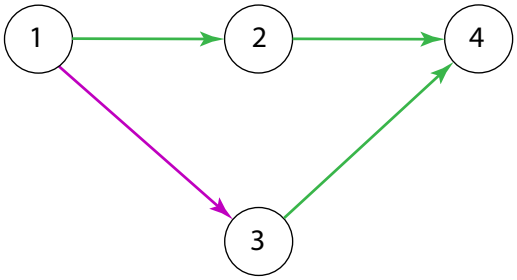


Figura 14: Según el comportamiento básico de RPL+TSCH, el nodo 1 posee al nodo 2 como Default Parent. Sin embargo, gracias a la extensión de LeapFrog Collaboration, el nodo 3 es el Alternative Parent del nodo 1, y por ende una opción más para que el paquete llegue a destino.

Para que un nodo elija a otro como su Alternative Parent, debe darse la situación de que el nodo al cual se llega luego de dos saltos debe ser el mismo tanto por el camino alternativo como por el camino por default. En el ejemplo de la figura 14, tanto el nodo 2 como el nodo 3 poseen al nodo 4 como destinatario, y es por ello que el nodo 1 puede elegir al nodo 3 como Alternative Parent.

Al seguir este comportamiento, la red termina por tener una estructura de tipo escalera: separados dos saltos, el nodo transmisor y destinatario constituyen los escalones, mientras que el Default y Alternative Parent del nodo transmisor, que son los nodos intermedios, constituyen los pilares. De esta manera, los paquetes se ven bifurcados hacia dos caminos distintos, pero que están relacionados entre sí ya que en última instancia convergen hacia un mismo nodo.

En conclusión, el uso del Alternative Parent aumenta las probabilidades de que un paquete llegue a destino. Sin embargo, a pesar de sus bondades, el concepto de Alternative Parent así definido presenta el problema que las condiciones para elegirlo pueden resultar muy restrictivas. Por ello, para evitar casos en que un nodo transmita sólo por su camino Default, entonces debe haber un método más laxo que permita la elección de un Alternative Parent

provisorio. A su vez, existe otra dificultad extra: los nodos deben ser capaces de determinar cual es su respectivo Alternative Parent. Ambos problemas se ven resueltos mediante el uso de paquetes especiales llamados LeapFrog Beacons, los cuales permiten determinar en forma dinámica a qué nodos deben transmitirse los paquetes de información.

Por último, cabe destacar que el uso de caminos alternativos tiene aparejado un costo mayor en términos de consumo de energía, ya que los nodos deben despertarse para recibir o transmitir una mayor cantidad de veces.

7.2 Overhearing

El Overhearing es otra característica incorporada en LeapFrog Collaboration. La misma está orientada a aumentar las posibilidades de que las transmisiones de los paquetes de información sean exitosas, capa a capa de la red. Replicar el uso de dicho mecanismo en las distintas capas termina por aumentar las chances de que los paquetes que el nodo hoja envía hacia la raíz lleguen a destino, y por ende tiene un impacto directo sobre la PDR de la red.

Las redes analizadas están caracterizadas por el uso de tecnología de tipo inalámbrica. Consecuentemente, el medio en el que se propagan las distintas transmisiones es el aire, el cual es naturalmente un medio compartido.

En el ejemplo de la figura 13, si bien el nodo 1 transmite sus paquetes al nodo 2, el nodo 3 se encuentra relativamente cercano a ambos. Si el área en el cual estos tres nodos se encuentran distribuidos resulta tal que para el rango de las antenas la potencia recibida sea suficiente, entonces se dice que los nodos se pueden sobre-escuchar entre sí. Sin embargo, dado que el nodo 2 es el Default Parent del nodo 1, se dice que simplemente el nodo 3 sobre-escucha la transmisión entre los dos primeros. Esto se ve representado mediante la línea punteada.

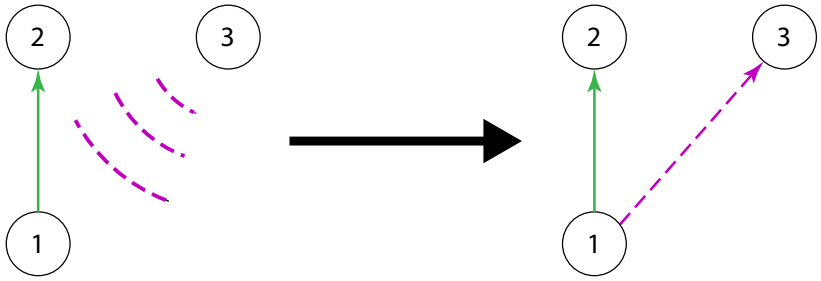


Figura 15: El nodo 3 utiliza Overhearing de manera tal de recibir él también el paquete enviado por el nodo 1 hacia el nodo 2.

Ese es el concepto del Overhearing: no sólo el verdadero destinatario enciende su antena para recibir un paquete, sino que los nodos cercanos a él también lo hacen. Así, los mismos sobre-escuchan la comunicación original y también reciben el paquete enviado. De esta manera, el paquete tiene más chances de progresar entre capas.

Debe notarse que este procedimiento necesita estar coordinado: el nodo 3 debe saber en qué timeslot y en qué canal de frecuencia la comunicación original tendrá lugar, de manera tal de poder sobre-escuchar correctamente. Es decir que, si bien el Overhearing mejora la PDR de la red, también tiene aparejado el compromiso de una mayor dificultad a la hora de diseñar el scheduling de la red, dado que ahora en una misma celda puede haber más que un único nodo receptor. Este inconveniente se ve resuelto también mediante el uso de los LeapFrog Beacons, lo cual da cuenta de la importancia que estos poseen para arribar a un correcto funcionamiento de la red.

7.3 Siblings

Los Siblings son otra pieza importante de la cual LeapFrog Collaboration hace uso. Los mismos son nodos que entre sí se encuentran dentro del rango de alcance de sus respectivas antenas, y que comparten ya sea el Default Parent y/o Alternative Parent. En el ejemplo de la figura 16, el nodo 2 y el nodo 3 comparten como Default Parent al nodo 4, y por ende son Siblings entre sí.

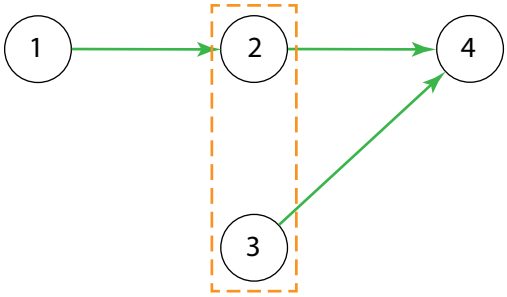


Figura 16: Los nodos 2 y 3 resultan Siblings entre sí para este ejemplo.

A modo más general, estos nodos presentan la cualidad que se hayan en una misma capa o layer de la red, es decir, que están separados una cantidad de saltos equivalentes a la raíz de la red, que en este caso coincide con el nodo 4.

7.4 Alternative Parent + Overhearing + Siblings

Si bien anteriormente se definieron por separado los conceptos de Alternative Parent, Overhearing y Siblings, los mismos fueron concebidos conjuntamente y pensados para ser combinados entre sí. Al acoplarlos, LeapFrog Collaboration termina por tener un funcionamiento mucho más completo, el cual le brinda una robustez superior a la red.

A continuación se analizan los conceptos de a dos, y en última instancia se combinan todos entre sí. Para todos los casos debe tenerse en cuenta que las flechas lisas indican las transmisiones directas (es decir, en las cuales el nodo transmisor pretendía sólo hacerle llegar el paquete al nodo al cual la flecha apunta) y las punteadas representan el proceso de Overhearing. A su vez, las líneas verdes representan los caminos por default, las líneas violeta los caminos alternativos, y las líneas anaranjadas las comunicaciones entre Siblings. Por último, dado que los gráficos son compuestos y muestran las transmisiones en momentos distintos, debe hacerse la excepción para los nodos que sólo posean un Default Parent, puesto que únicamente se transmite una vez el paquete a ellos.

- **Alternative Parent + Overhearing.** En primer lugar, el nodo transmisor realiza una transmisión en modo unicast a su Default Parent. Al mismo tiempo el Alternative Parent realiza el proceso de Overhearing.

Posteriormente, el nodo transmisor transmite una réplica del paquete original al Alternative Parent, y es ahora el Default Parent el cual realiza el Overhearing. Esto puede observarse más claramente en la figura 17, donde el nodo 1 posee como Default Parent al nodo 2 y como Alternative Parent al nodo 3.

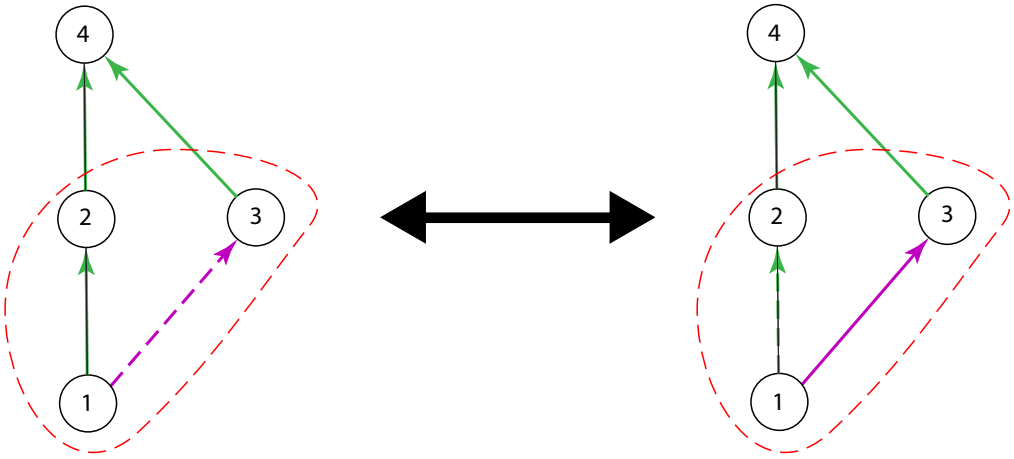


Figura 17: Combinación de Alternative Parent y Overhearing.

Como consecuencia de la combinación de ambos conceptos, el Default y Alternative Parent poseen cada uno 2 oportunidades de recibir el paquete, con lo cual se duplican las chances de que éste avance a la siguiente capa de la red.

- **Siblings + Overhearing.** El objetivo de este proceso es asegurar que todos los Siblings en una misma capa posean el paquete de información, y por ende maximizar la cantidad de veces que se intentará de transmitirlo a la siguiente. Para ello, existe un primer nodo transmisor que realiza una transmisión hacia la siguiente capa de la red, y es en este momento que su Sibling, para conseguir una copia del paquete, realiza Overhearing. Dado que no existe manera de saber cual Sibling tendrá el paquete y cual no, entonces el mismo proceso se repite, solo que ahora el anteriormente considerado Sibling asume el rol de nodo transmisor, y viceversa. En definitiva, esta propuesta permite que en los casos en que un Sibling no hubiese recibido el paquete, ahora este también lo tenga y lo pueda transmitir a la siguiente capa.

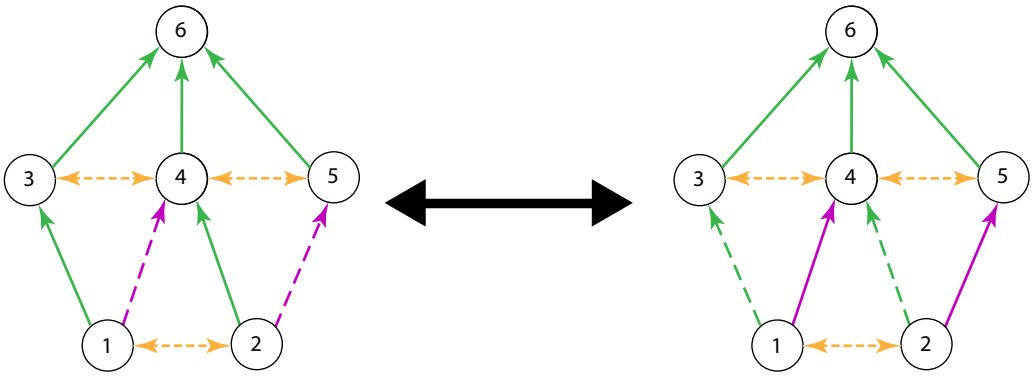


Figura 18: Combinación de Siblings y Overhearing.

En la figura 18, los nodos 1 y 2 comparten los paquetes entre sí y luego, en la siguiente capa de la red, los nodos 3, 4 y 5 repiten el mismo procedimiento entre sí.

- **Alternative Parent + Overhearing + Siblings.** En el caso de que los Siblings compartan ya sea de manera idéntica o intercambiada a sus respectivos Default Parent y Alternative Parent, entonces éstos últimos contarán ahora con hasta 4 posibilidades cada uno de recibir el paquete desde la capa más lejana de la red.

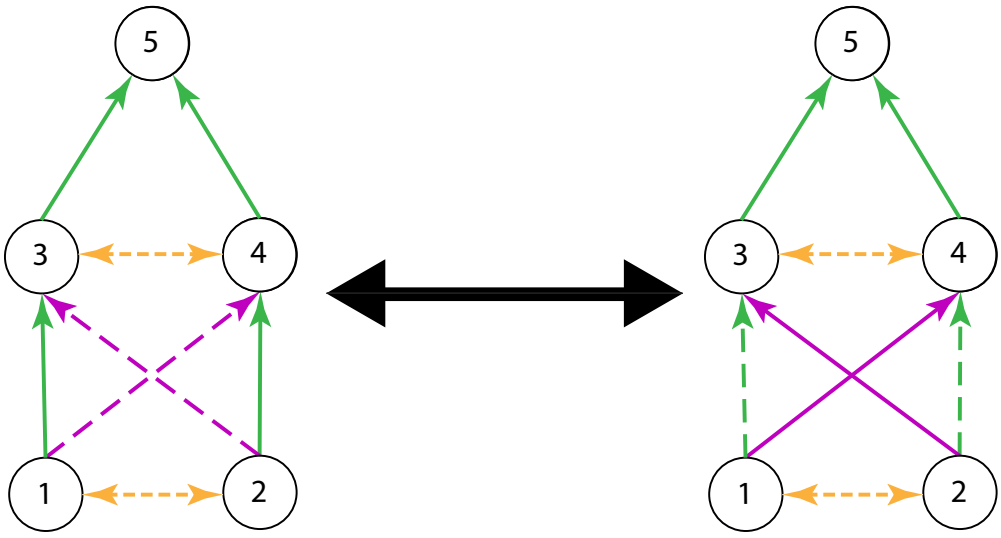


Figura 19: Combinación de Alternative Parent, Overhearing y Siblings.

En la figura 19, los nodos 3 y 4 son los Default y Alternative Parents de los nodos 1 y 2 respectivamente. Por ello reciben el paquete 4 veces

cada uno: dos debido a transmisiones directas y dos debido al proceso de Overhearing.

7.5 Retransmisiones

Según lo expresado en la sección anterior, la combinación del Alternative Parent, Overhearing y Siblings permiten aumentar hasta 4 veces las posibilidades de que un paquete pueda avanzar de capa acercándose hacia la raíz del DODAG. Sin embargo, de manera de volver más robusta la red, la posibilidad de realizar retransmisiones en caso de una transmisión fallida también fue incorporada.

En el caso de LeapFrog Collaboration, los nodos sólo intentan una vez de retransmitir el paquete, y si dicho proceso falla, simplemente descartan el paquete, asumiendo como resultado una comunicación sin éxito. Agregar una posibilidad de transmisión cuando originalmente sólo se intentaba de transmitir una única vez significa directamente la duplicación de oportunidades de transmitir un paquete, por lo que ahora no serán 4 las chances de que el mismo progrese de capa, sino que 8.

Las retransmisiones son importantes ya que su incorporación tiene una incidencia directa sobre la PDR de la red, parámetro de vital importante. Por otro lado, también poseen un costo asociado de tener que reservar un timeslot para ellas. Esto incide no sólo en el consumo de energía sino que en la latencia de extremo a extremo también.

7.6 LeapFrog Beacons

Para entender de qué se tratan los LeapFrog Beacons, primero hace falta comprender cuál es la información relativa al nuevo protocolo que cada nodo almacena constantemente. La misma se puede resumir en:

- LeapFrog Default Parent (LDP): Es el Default Parent (DP) que brinda el protocolo RPL.
- LeapFrog Grand Parent (LGP): Es el LDP del LDP del nodo en cuestión.
- LeapFrog Alternative Parent (LAP): Es el nuevo padre que se define y al cual también se le envia una copia de los paquetes de información.

- Possible Parents (PPS): Es una lista de los nodos que tienen menor rango que el nodo en cuestión. Por ende, los mismos se encuentran más cercanos a la raíz de la red, y bien podrían ser el Default Parent del nodo, si este cambiara. Esta información se construye en base a los mensajes DIO que el nodo recibe.

- LeapFrog Layer (LL): La cantidad de saltos que separen a un nodo de la raíz define la capa en la cual el nodo se encuentra. Este valor resulta útil para que el nodo sepa luego si los otros nodos que se están comunicando con él están más lejos o más cerca a la raíz que él mismo.

Gracias a todos estos datos, los nodos son capaces de generar los distintos caminos alternativos para los paquetes de información, y de decidir cuándo tienen que realizar o no el proceso de Overhearing. Además, también pueden identificar sus Siblings a escuchar mediante el Overhearing, y así continuar construyendo más posibilidades para que los paquetes enviados finalmente puedan llegar a destino. En definitiva, todos estos datos resultan esenciales ya que complementan la información brindada por el protocolo RPL, y son por ello el corazón de LeapFrog Collaboration.

Sin embargo, al momento de inicialización de la red, los nodos no cuentan con esta información, necesitan conseguirla de alguna manera, y es allí donde entran en juego los LeapFrog Beacons. Los mismos son paquetes enviados por todos los nodos de la red, donde cada uno comparte la propia información almacenada. Los mismos se conforman de distintos campos, como se puede observar en detalle en la figura 20.

Figura 20: LeapFrog Beacon

El campo LFH hace referencia al header que se utiliza en este tipo de paquetes y que permite distinguirlo de los demás que son enviados en la red. El valor del mismo es Oxf1. A su vez, hacia el final del paquete, el campo MN incluye un número de secuencia, pero que simplemente se envía como referencia y que no cumple ninguna función en particular. El resto de los

campos está compuesto por los valores previamente definidos.

Los LeapFrog Beacons se envían como paquetes broadcast, hacia todos los vecinos que rodeen al nodo. Además, estos paquetes se envían en forma periódica de forma tal de que los nodos cuenten siempre con los datos del último estado de las variables. En particular, el período para el envío de estos paquetes es de 30 s. Los nodos sólo plantean la necesidad de chequear los datos relativos al nuevo protocolo cuando reciben paquetes de este tipo. El nodo que recibe un LeapFrog Beacon, le extrae los distintos campos, y a su vez actualiza la información de su DP dada por RPL. Todos estos datos los contrasta con su información propia. De este análisis puede resultar que haga falta actualizar los valores, o que simplemente el paquete no revista importancia.

Existen varios pasos en la lógica que un nodo que recibe un LeapFrog Beacon debe seguir, y que le permiten determinar qué es necesario hacer en cada situación. A continuación se describen los controles que se realizan:

- **Obtención de un LDP.** El DP puede cambiar en cualquier momento ya que está definido por RPL. Cuando esto sucede, el LDP deberá ser actualizado también, copiando su valor.
- **Obtención de un LGP.** Esto se da cuando el nodo que transmitió el LeapFrog Beacon es el LDP del nodo que lo recibió. Aquí, el nodo 1 es el nodo receptor, y el nodo 2 es el transmisor. Deberá contrastarse la información en el campo LDP del Beacon junto con el LGP del nodo receptor, y actualizar su valor en caso que haya habido algún cambio. Este proceso puede observarse mejor en la figura 21.

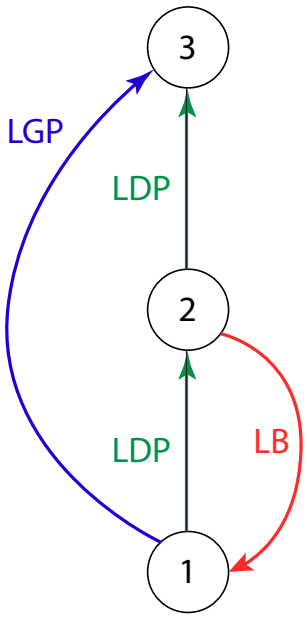


Figura 21: Obtención de un LDP

El nodo 2 transmite un LeapFrog Beacon, que si bien es un mensaje broadcast y por ende el nodo 3 también lo recibiría, se destaca sólo la recepción por parte del nodo 1. Como el nodo 2 es el LDP del nodo 1, este último constata a través de la información contenida en el paquete que el LDP del primero es el nodo 3. Por ende, de ser necesario, deberá actualizar su valor.

Como método alternativo, si el nodo transmisor no es el LDP del nodo receptor (evitando así nuevamente que se suscite la situación en la cual el LDP y el LAP del nodo sean iguales), entonces si entre sus Possible Parents se encuentra incluido el LGP del nodo receptor, entonces el nodo transmisor puede ser considerado como el LAP. En la figura 22 puede verse este caso en detalle.

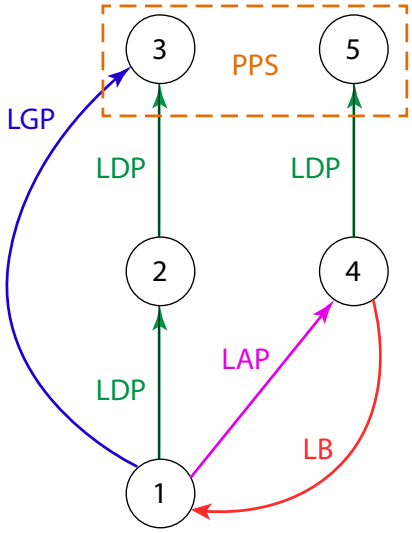


Figura 22: Obtención alternativa de un LAP

A diferencia del caso de la figura 22, el nodo 4 ahora posee como LDP al nodo 5, que no coincide con el LGP del nodo 1. Sin embargo, éste se entre encuentra incluido entre sus PPS, y por ello el nodo 4 puede ser considerado nuevamente como un potencial LAP.

- **Obtención de un LAP.** El el caso que el nodo receptor posea un LGP, mientras que el nodo transmisor un LDP, y estos coincidan, entonces el nodo transmisor podrá ser elegido como LAP, a no ser que el mismo ya sea el LDP del nodo receptor, como puede verse en la figura 23.

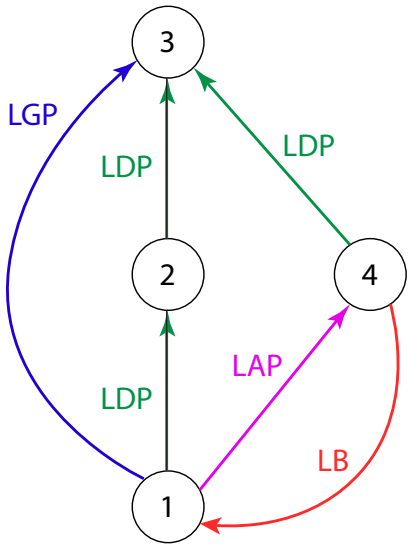


Figura 23: Obtención de un LAP

Cuando el nodo 4 transmite su LeapFrog Beacon, el nodo 1 lo recibe. Su LDP es el nodo 2, y el LDP de éste el nodo 3. Por ende, el nodo 3 es el LGP del nodo 1. Como el LDP del nodo 4 es el nodo 3, y coincide con el LGP del nodo 1, entonces el nodo 4 puede ser elegido como el LAP del nodo 1. Sin embargo, si el nodo que enviase el LeapFrog Beacon fuese el nodo 2 en vez del 4, el caso analizado sería equivalente al de la figura 23, y no tendría sentido que el nodo 1 lo eligiese también como su LAP dado que él ya es su LGP.

- **Obtención de la LL.** Cuando un nodo recibe un paquete, chequea si él es el LDP o el LAP del nodo transmisor . En cualquiera de ambos casos, su LL deberá ser establecida como la LL del nodo transmisor aumentada en una unidad. Además, si el nodo receptor es el LDP o LAP, entonces deberá chequear si el nodo transmisor posee un LAP o LDP, respectivamente. En caso afirmativo, deberá escuchar a través del Overhearing las transmisiones que el nodo transmisor pudiera tener con su padre complementario.

- **Definición de Siblings.** Si el nodo receptor y el nodo transmisor poseen el mismo LDP, entonces los mismos son Siblings. Como método alternativo, si ambos nodos no comparten el LDP, entonces si entre los Possible Parents del nodo transmisor se encuentra incluido el LDP del nodo receptor, se deduce que el nodo transmisor puede ser considerado como un Sibling. En cualquiera de ambos casos, el nodo receptor deberá configurarse para realizar Overhearing de las comunicaciones que su Sibling pueda realizar con sus respectivos LDP y LAP.

A través de toda esta secuencia, todos los nodos de la red logran determinar sus respectivos LDP, LGP y LAP, y así construir los caminos alternativos que permiten no sólo aumentar las posibilidades de que un paquete llegue a destino y no se pierda, sino también de que lo haga con un retardo similar.

7.7 TSCH Scheduling

El scheduling es una de las partes más importantes de LeapFrog Collaboration, ya que en función de qué tan bien esté diseñado, tendrá un impacto directo sobre el retardo y la variación del mismo en la red. Antes de comenzar a diseñarlo, primero deben entenderse las necesidades que cada una de las características de LeapFrog Collaboration agrega sobre la manera en que los nodos pueden comunicarse, y las restricciones que eso impone.

A modo de ejemplo, a continuación se detalla cómo debe modificarse el scheduling "básico" de la figura 24 (asociado a la red que allí se presenta) de manera tal de poder implementar el funcionamiento de LeapFrog Collaboration sobre la misma.

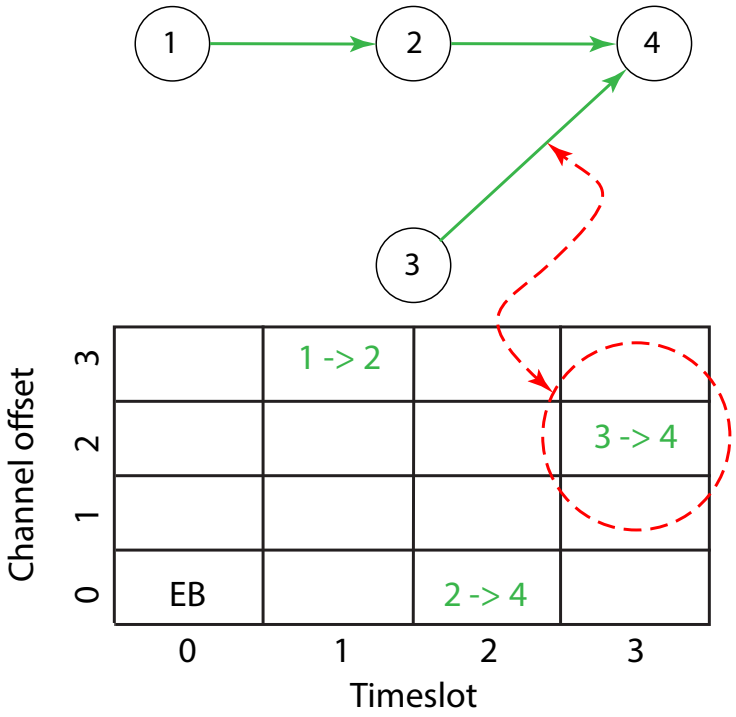


Figura 24: Red de 4 nodos junto a su scheduling asociado.

Si bien este ejemplo es sobre una red simple, resulta más simple analizarlo en ella. Luego, extrapolar cómo llevar adelante el proceso en un caso con más nodos resulta directo y no requiere mayor esfuerzo para hacerse.

7.7.1 Operaciones sobre el Scheduling

- **Overhearing.** En el timeslot reservado para la transmisión al default parent, ahora no sólo éste deberá encender su antena, sino que el nodo que efectúe Overhearing lo hará también. Esto puede observarse en la figura 25.

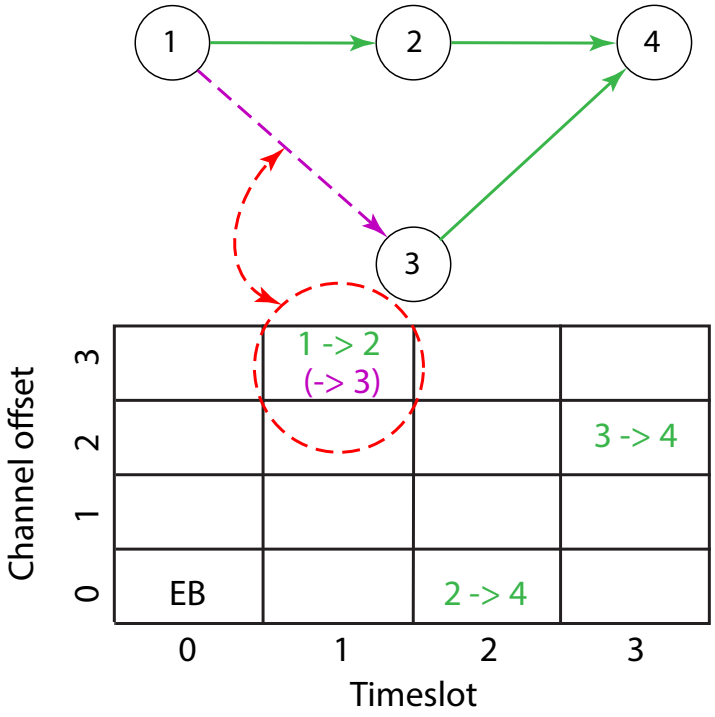


Figura 25: Overhearing sobre el timeslot original.

La línea violeta indica un camino alternativo, y, además, el hecho de que sea particularmente punteada hace referencia a que se está realizando Overhearing. La notación "(-> A)" significa que en dicho timeslot y channel offset, el nodo A es quien realiza Overhearing. No se aclara quien es el nodo transmisor ya que en verdad el paquete no poseía la intención de ser enviado a dicho nodo. Sin embargo, el mismo puede identificarse observando cuál era la transmisión original en dicha celda, en este caso el nodo 1. Este tipo de notación se mantiene a lo largo de toda la presente sección.

- **Alternative Parent.** Un nodo transmite a su padre por default, y por ende tendría un único timeslot asociado para dicha acción. Sin

embargo, agregando el padre alternativo, el nodo también requiere otro timeslot para poder comunicarse con él. Esto puede observarse en la figura 26.

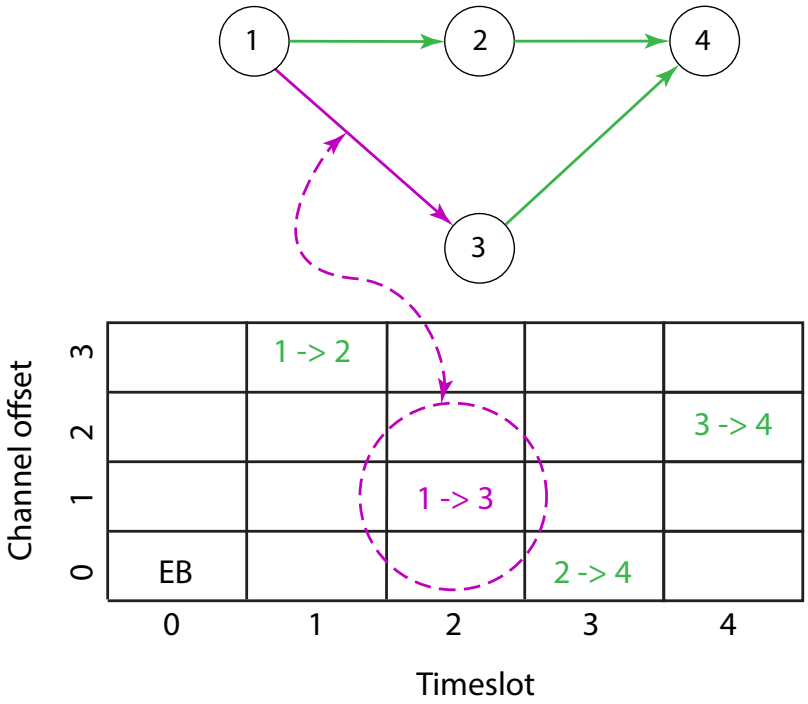


Figura 26: Timeslot para transmisión al Alternative Parent.

Es importante notar que, en principio, nada obliga a que ambas transmisiones sean realizadas con el mismo channel offset, aunque podrían serlo. Como consecuencia de cada transmisión directa a los dos tipos de padres, cada nodo ocupará al menos 2 timeslots para realizar la transmisión de un paquete a la próxima capa de la red.

- **Retransmisiones.** En LeapFrog Collaboration las retransmisiones se realizan todas dentro de un mismo slotframe. La idea detrás de esto es que, entonces, así puede asegurarse que, si un paquete llega a destino, entonces lo hará dentro del mismo slotframe en el cual fue enviado. En particular, por el momento sólo se considera una única retransmisión. Ello lleva a que la cantidad de timeslots necesarios para transmitir un paquete a la siguente capa ahora sean 4 por cada nodo, y no 2. Además, el slotframe crece a un tamaño cercano al doble, como puede observarse en la figura 27.

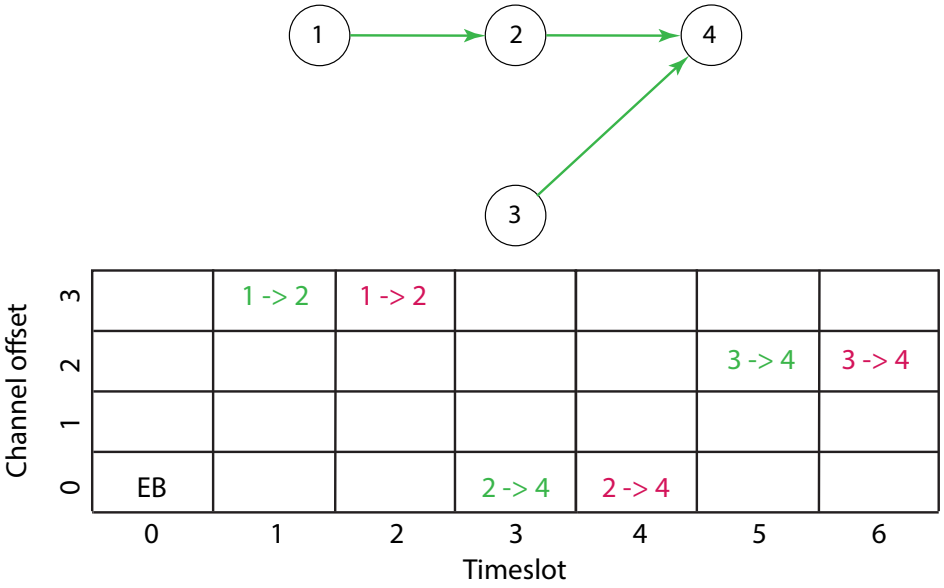


Figura 27: Un timeslot adicional por cada retransmisión.

En este ejemplo, las retransmisiones suceden todas en el mismo channel offset, pero ello no es una condición necesaria.

Una vez entendidos los procesos anteriores, es importante notar el resultado que tendrá la combinación de los mismos, ya que justamente todos estos conceptos serán utilizado simultáneamente.

El primer caso resulta de la combinación del Alternative Parent junto con el proceso de Overhearing, como puede observarse en la figura 28.

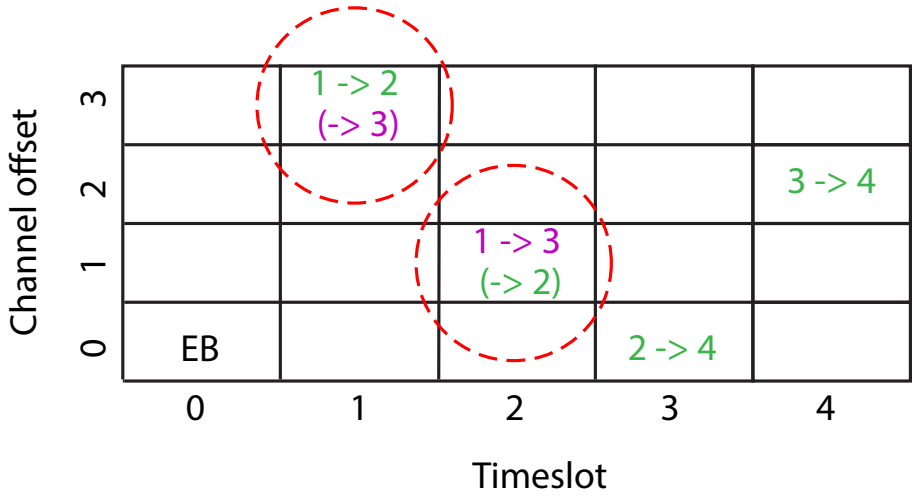


Figura 28: Modificación del scheduling teniendo en cuenta el Alternative Parent y Overhearing.

El Alternative Parent genera la necesidad de un timeslot adicional, de manera de que el paquete le sea transmitido directamente. Pero, además, ahora se incluye en el timeslot original y en el agregado la posibilidad de realizar Overhearing.

A su vez, luego también puede incorporarse el concepto de las retransmisiones dentro del mismo slotframe, lo cual básicamente lleva a que el tamaño del mismo se vea aumentado, como muestra la figura 29.

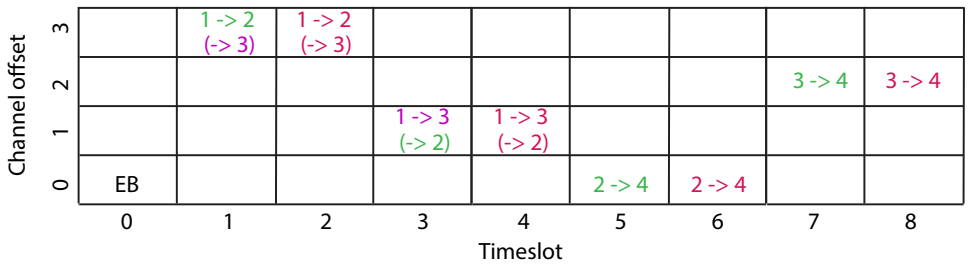


Figura 29: Modificación del scheduling teniendo en cuenta el Alternative Parent, Overhearing y las restransmisiones.

Por último queda tener en cuenta el concepto de Siblings, que anteriormente no se había tenido en cuenta, particularmente porque los mismos se aprovechan mediante el uso del Overhearing, y por ende no requieren ninguna

consideración extra para comprender cómo se modifica el scheduling. En la red analizada, los nodos 2 y 3 son siblings entre sí, y por ende pueden también obtener el paquete uno del otro, como puede verse en la figura 30.

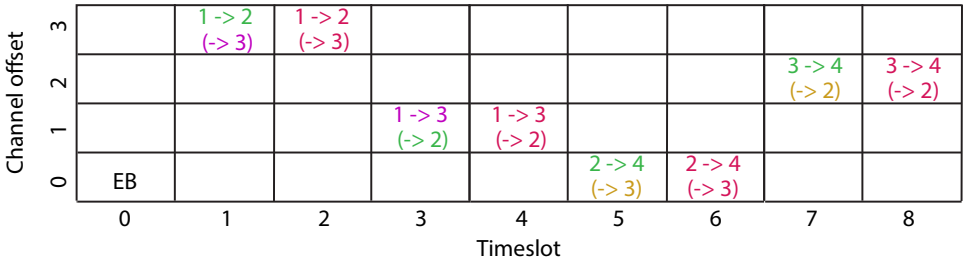


Figura 30: Modificación del scheduling teniendo en cuenta todas las características de LeapFrog Collaboration.

En este caso, la recepeción por parte de un Sibling de un paquete de información se ve representada mediante el color dorado, y siguiendo la notación utilizada antes para simbolizar el uso de Overhearing.

7.7.2 Ordenamiento de las transmisiones

Hasta el momento han sido expuestas todos los detalles que surgen de la introducción de nuevos conceptos y que determinan el ancho mínimo que tendrá el scheduling y la cantidad de transmisiones que serán necesarias. Sin embargo, no se ha tenido en cuenta todavía la incidencia de en qué orden se realizan las comunicaciones: ello afecta directamente el retardo y su estabilidad, es decir su variabilidad, resultantes.

En LeapFrog Collaboration se produce una división del slotframe en 3 partes. La primera de ellos es el “camino directo”, y los paquetes de información avanzan sobre esta ruta ascendente hacia el destino en la manera más rápida posible. Luego, el slotframe posee la sección de “otras transmisiones”, donde los nodos realizan las transmisiones de los LeapFrog Beacons y además construyen las rutas descendentes. Por último, la sección final es únicamente utilizada para los Enhanced Beacons.

7.7.2.1 Camino Directo

La manera propuesta de resolver este problema en LeapFrog Collaboration es el ordenamiento de las transmisiones empezando por darle la oportunidad a los nodos más lejanos a la raíz del DODAG, y siguiendo luego con aquellos que se encuentren más cercanos a la misma. Más precisamente, primero los nodos de mayor Rango transmiten sus paquetes, y luego los nodos intermedios de acuerdo a su distancia a la raíz del DODAG.

Como resultado, todos los nodos acceden al medio inalámbrico de manera secuencial, uno detrás del otro y, por ende, los paquetes de información pueden arribar al destino dentro de un único slotframe. Es por ello también que resulta necesario incluir las retransmisiones dentro del slotframe original, de manera de que si algo falla, se intenta de corregirlo inmediatamente y no requiriendo un mayor tiempo de espera que en definitiva afecta de manera muy seria el valor del jitter.

En conclusión, esta propuesta acata los requerimientos de redes determinísticas para la Industria 4.0, las cuales necesitan de una performance que consiga un jitter ultra reducido. Volviendo al ejemplo de la figura 30, vemos que ello ya había sido tenido en cuenta a la hora de armar el scheduling, pero no era trivial, sino el resultado de un diseño inteligente.

En dicho ejemplo, el nodo 1 es el nodo más alejado, y quien en cierto modo finaliza el DODAG. Como RPL crea una estructura de tipo árbol, donde el destino final de los paquetes es la raíz del DODAG, o en castellano la raíz del DODAG, entonces a los nodos más alejados se los llama nodo hoja o Leaf Node. En el diseño de LeapFrog Collaboration, los Leaf Nodes son los único nodos que crean información, la cual luego simplemente es acarreada por los nodos intermedios hacia el destino. El Leaf Node es quien transmite primero, lo cual se observa en los timeslots 1 a 4. A continuación, son los nodos de la capa siguiente los que transmiten, entre los timeslos 5 a 8.

Considerando ahora los ejemplos de la figura 31, nuevamente para construirse el camino directo se utiliza el mismo concepto que antes, transmitiendo desde el nodo más alejado a la raíz del DODAG hacia el nodo más cercano.

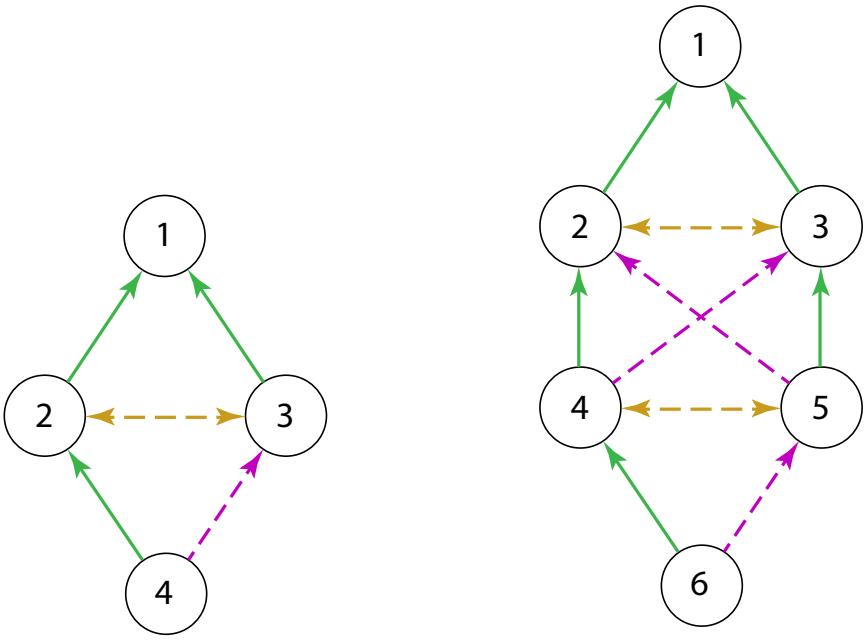


Figura 31: Redes con distinta cantidad de nodos.

En vez de utilizar de guía el scheduling, resulta más claro utilizar una tabla que indique qué hace cada nodo en cada timeslot. Ello puede observarse por ejemplo en la figura 32 que muestra el caso para la red con 4 nodos.

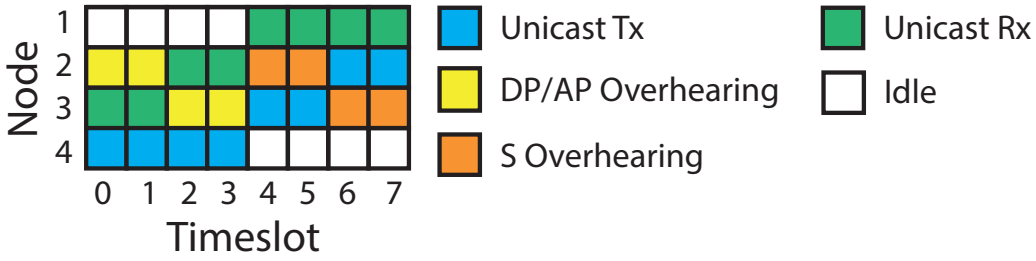


Figura 32: Tabla de transmisiones para la red de 4 nodos.

El color azul indica qué nodo realiza la transmisión en el timeslot dado. En verde se muestra el nodo que recibe el paquete, que hipotéticamente sería el Default Parent del nodo transmisor. Sin embargo, la aleatoriedad existente en principio no permitiría asegurar ello, dado que el nodo podría ser el Alternative Parent y, como consecuencia, en amarillo se aclara que el nodo que realiza el Overhearing podría ser cualquiera de ambos. Por último, en naranja se simboliza el Overhearing entre Siblings y en blanco son casos en que los nodos no llevan adelante ninguna tarea en particular, y por ende se hayan en reposo.

Para el caso con 6 nodos, se tiene el ordenamiento de la figura 33, y que resulta un tanto más complejo que el caso anterior.

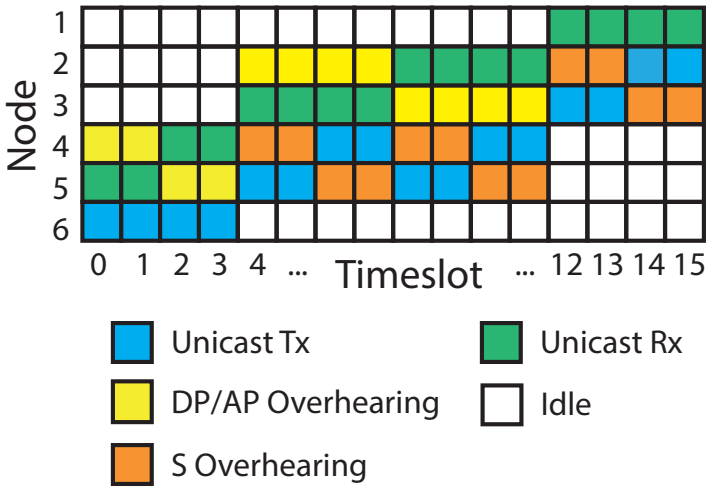


Figura 33: Tabla de transmisiones para la red de 6 nodos.

El nodo 6, que es el Leaf Node transmite primero en los timeslots 0 a 3. Luego, en la siguiente capa se hallan los nodos 4 y 5, que realizan sus respectivas transmisiones también a sus Default Parent y Alternative Parent entre los timeslots 4 y 11. Esta situación es nueva comparado con los casos anteriores ya que ahora los nodos de esta segunda capa poseen dos destinatarios posibles también. En LeapFrog Collaboration, primero transmite uno de los nodos de la capa y luego el siguiente, primero a uno de los destinatarios y luego al restante. Es decir, los nodos 5 y 4 transmiten al nodo 3, y luego lo hacen al nodo 2. Finalmente, las transmisiones de los nodos 2 y 3 son análogas a las del caso con 4 nodos, solo que se dan en los timeslots 12 a 15.

Si bien el esquema de transmisiones para una red de 6 nodos es más difícil de conformar, en verdad resulta simplemente de una expansión de lo hecho anteriormente para el caso de 4 nodos. Simplemente basta notar que, tomando los timeslots 0 a 7 del segundo caso, y renombrando sus nodos 3, 4, 5 y 6 como 1, 2, 3 y 4 respectivamente, se tiene el mismo scheduling de caso con 4 nodos.

7.7.2.2 Otras Transmisiones

La sección del scheduling "otras transmisiones" está reservada para todo tipo de tráfico que no sea concerniente a paquetes de información que fluyen hacia la raíz del DODAG ni a paquetes para anunciar la presencia de la red. En otras palabras, el tipo de tráfico esperado los timeslots destinados para ella es el de paquetes de control y tráfico P2MP.

En primer lugar, se debe crear espacio para la transmisión de mensajes DIO y LeapFrog Beacons. Estos mensajes se transmiten en forma Broadcast, y por ende todo el conjunto de los nodos deben escuchar al nodo transmisor.

Por otro lado, las rutas descendentes se crean siguiendo el mismo pensamiento que para las rutas ascendentes, solo que en el sentido contrario: ahora la raíz del DODAG será el primer nodo con la posibilidad de transmitir un paquete, mientras que el Leaf Node será el último receptor.

7.7.2.3 Enhanced Beacons

En LeapFrog Collaboration se optó por la utilización de un timeslot para paquetes que anuncian la red por cada nodo que existe en la misma. Ello facilitaba la confección de una primera solución en la cual no hubiese nodos compartiendo una celda para enviar o recibir paquetes. Si bien esta solución previene posibles colisiones entre paquetes que puedan ser transmitidos en simultáneao por distintos nodos, significan un consumo mucho mayor de energía ya que los nodos deben despertarse reiteradas veces para escuchar por Enhanced Beacons.

7.7.2.4 Schedule Final

Combinando las tres secciones, se obtiene el scheduling que LeapFrog Collaboration utiliza, como puede observarse en la figura 34 para el caso de 4 nodos.

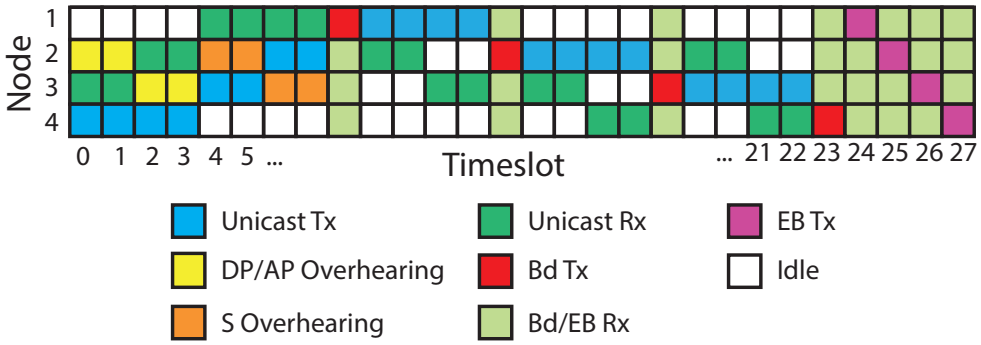


Figura 34: Scheduling de la red de 4 nodos.

Por otro lado, para el caso de 6 nodos, el scheduling se halla en la figura 35.

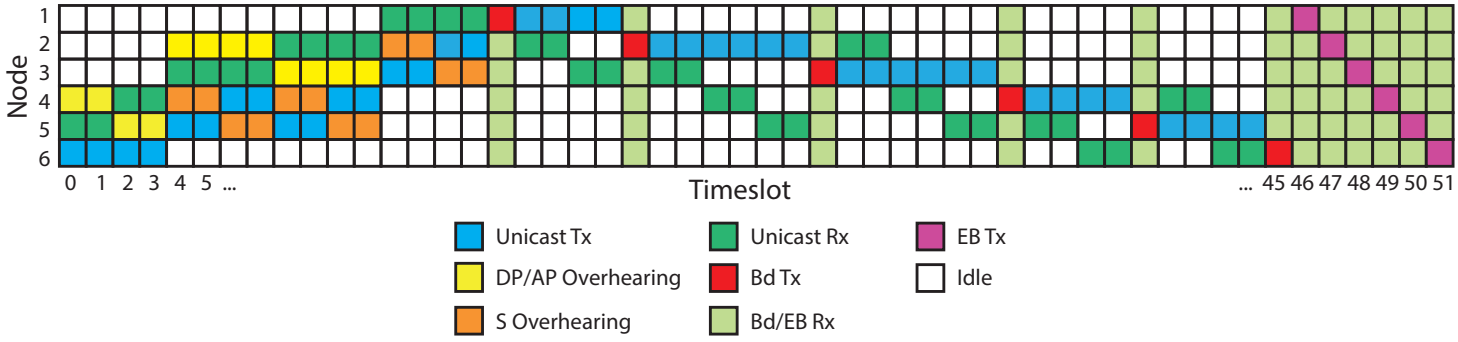


Figura 35: Scheduling de la red de 6 nodos.

Es importante notar que, a modo de simplificación, la solución propuesta en LeapFrog Collaboration utiliza únicamente el channel offset 0 para realizar todas las transmisiones y por ello las tablas de transmisiones en definitiva representan también el scheduling utilizado para la red. El fijar el channel offset en 0 no significa no utilizar la robustez provista por el channel hopping, dado que al progresar el tiempo, el valor del ASN siempre cambia y por ende, la frecuencia física sobre la cual se realiza la transmisión. Sin embargo, el no utilizar el resto de los canales significa una oportunidad perdida para paralelizar transmisiones, y ello es algo que deberá tenerse en futuras ampliaciones de la performance de LeapFrog Collaboration.

Por último, cabe notar que los gráficos muestran hasta el último timeslot en el cual efectivamente existe algún nodo activo, pero lo cierto es que todos

los slotframes poseen un ancho total de 101 timeslots. Es decir que el resto de los timeslots no se muestran ya que en ellos simplemente todos los nodos estarían en modo Idle o de reposo.

7.7.3 Análisis Matemático para el Camino Directo

En esta sección primero se presenta un análisis de la performance retardo-jitter y la relación de compromiso que existe entre ambos. Ello permite entender el tipo de decisiones que fue necesarias tomar para llevar adelante la propuesta de LeapFrog Collaboration. Luego, se provee un análisis matemático para el cálculo de varias de las métricas que luego se constatarán mediante simulaciones en LeapFrog Collaboration, incluyendo la PDR, y límites superiores del retardo extremo a extremo y jitter.

7.7.3.1 Relación de Compromiso Retardo-Jitter

Además de utilizar Overhearing y explotar el uso de múltiples caminos, LeapFrog Collaboration incorpora el uso de una retransmisión para cada enlace, como estudiado anteriormente. La combinación de todos sus mecanismos, concluyen en que un paquete posea al menos 8 oportunidades de progresar hacia un nodo con Rango menor dentro de la red, en el mismo slotframe.

Fundamentalmente, el número de oportunidades únicas que un paquete tiene de progresar en la red hacia una capa superior resulta controlado por una relación de compromiso. En efecto, aumentar dicho número resulta beneficioso para disminuir el jitter resultante de extremo a extremo, ya que se aumentan las probabilidades de que un paquete llegue a destino dentro del mismo marco. Sin embargo, mientras mayor es la cantidad de timeslots dedicados a simplemente otorgar una oportunidad de retransmisión, más largo resulta el slotframe. Como resultado, el uso de estos timeslots adicionales para retransmisiones impactan de manera negativa sobre el retardo, aumentándolo. De allí que existe una relación de compromiso entre el retardo y el jitter.

Se considera una vecindad de nodos con un receptor r y N transmisores: $n_1, n_2, \ldots, n_N$. El número de timeslots consecutivos por nodo por slotframe se denota como k. Por ejemplo, $k = 1$ se corresponde con un slotframe de

tamaño N, con un timeslot por nodo ($n_1 \to r, n_2 \to r, \ldots$). De forma similar, $k = 2$ se corresponde con un slotframe de tamaño $2N$, con 2 timeslots consecutivos por cada nodo ($n_1 \to r, n_1 \to r, n_2 \to r, n_2 \to r, \ldots$).

Se modela cada transmisión como un ensayo Bernoulli independiente con la misma probabilidad de éxito p para cada intento. Por simplicidad, se asumen infinitas retransmisiones. La probabilidad de que un paquete sea transmitido con éxito en el intento i está dada por:

$$q_i = (1-p)^i p \ . \tag{1}$$

Sin pérdida de generalidad, se puede focalizar el análisis en el nodo N y asumiendo que el paquete a transmitir es generado en el comienzo del slotframe. Esta suposición constituye el caso de peor escenario para el retardo. Si el paquete logra ser transmitido en el intento i, entonces el retardo viene dado por:

$$d_i = kN\left\lfloor \frac{i}{k} \right\rfloor + i \ \texttt{mod} \ k + k(N-1) \ . \tag{2}$$

Aquí $k(N-1)$ representa el retardo de la transmisión de los N-1 nodos anteriores dentro del último slotframe y i `mod` k es el retardo producto de la transmisión del nodo N en ese mismo slotframe. Por otro lado $kN\left\lfloor \frac{i}{k} \right\rfloor$ es el retardo de la cantidad de slotframes completos en los cuales no se tuvo éxito para transmitir el paquete.

El retardo medio viene dado por:

$$D = \sum_{i=0}^{\infty} q_i d_i \ , \tag{3}$$

Y el jitter puede ser modelado como la desviación estándar del mismo:

$$J = \sqrt{\sum_{i=0}^{\infty} q_i (d_i - D)^2} \ . \tag{4}$$

La figura 36 expone la relación de compromiso retardo-jitter para el caso de $N = 4$ y para distintos valores de calidad de enlace.

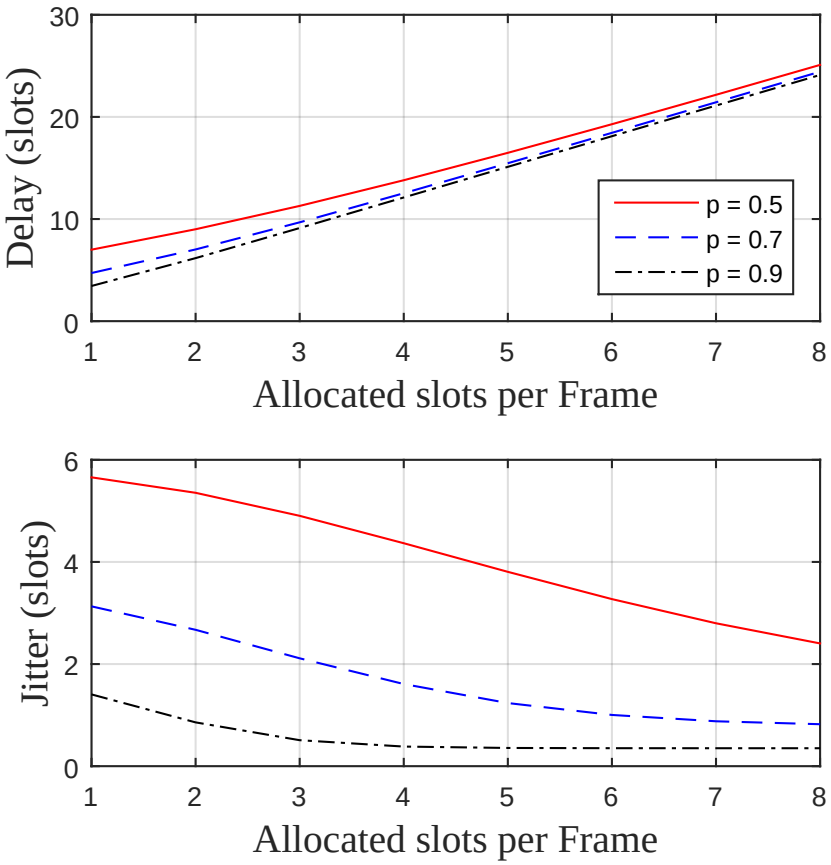


Figura 36: Relación de compromiso retardo-jitter en una topología con 4 nodos.

Se puede observar que a medida que aumenta el número de timeslots por nodo por slotframe, el jitter disminuye, aunque con una mejora decreciente. Por otro lado, el retardo incrementa rápidamente, casi siguiendo una relación lineal.

LeapFrog Collaboration balancea la relación de compromiso retardo-jitter asignando $n^2 \times m$ timeslots para transmitir un paquete dado a un nodo con un Rango mayor. Aquí, $n = 2$ se corresponde con el número de Padres (Default y Alternative) y, por otro lado, $m = 2$ hace referencia al número máximo de transmisiones a un único padre.

7.7.3.2 Performance en el peor escenario

A continuación se presentan las expresiones analíticas que permiten calcular la PDR de extremo a extremo así como el máximo retardo extremo a extremo,

y su jitter.

Se asume que un paquete se genera en el Leaf Node al comienzo de cierto slotframe en una red con R Rangos. El número de padres (incluyendo los alternativos) se denota n. El número máximo de transmisiones a un único padre dentro de un mismo slotframe se asocia al valor de m.

De extremo a extremo, se pueden analizar el caso en 3 pasos. El primero de ellos se corresponde con el Leaf Node, i, transmitiendo a sus padres, que se hallan localizados en el Rango $R-1$. La probabilidad de que un paquete falle en llegar a su padre, j, en el Rango $R-1$, viene dada por q_{R-1}, que se calcula según:

$$q_{R-1}^{(j)} = p_{ij}^{mn} \ , \tag{5}$$

donde p_{ij} es la probabilidad de error de enlace para el nodo i transmitiendo al padre j. Este resultado surge de tener en cuenta que cada uno de los padres del Leaf Node poseen en total mn oportunidades de recibir el paquete: m de ellas debido a transmisiones directas y las otras $m(n-1)$ restantes debido al proceso de Overhearing realizado en las transmisiones a los otros posibles padres. Debe notarse que dado que se está calculando la probabilidad de error de enlace, en el análisis ninguna transmisión es exitosa, y por ende todos los intentos de comunicación, es decir cada transmisión y todas sus retransmisiones, deben ser contempladas. Por otro lado, por más que un nodo j esté realizando Overhearing sobre la comunicación con un nodo k, la calidad del enlace que lo vincula con el nodo transmisor i continúa siendo p_{ij} y no es p_{ik}.

El segundo paso se obtiene considerando la situación en que un paquete falla en progresar a un Rango intermedio $h \in [1, R-2]$. La probabilidad de que ello suceda depende de si logró alcanzar el Rango previo:

$$q_h^{(j)} = \prod_{i=1}^{n} \left(q_{h+1}^{(i)} + (1 - q_{h+1}^{(i)}) \cdot p_{ij}^{mn} \right) \ . \tag{6}$$

A diferencia del paso anterior, un nodo en un Rango intermedio puede recibir el paquete de hasta n fuentes. Las chances de que el paquete no llegue al nodo depende de que, por cada una de dichas fuentes, el paquete no lo haya hecho. Para cada una de ellas, esto se asocia a que directamente el nodo transmisor no haya recibido el paquete que venía de un Rango inferior o a

que el nodo transmisor cuente con el paquete pero no tenga éxito al intentar de transmitirlo al nodo receptor. Aquí, nuevamente debido al Overhearing, un nodo posee hasta mn chances de recibir el paquete.

El tercer paso se corresponde con el último salto, el cual significa el arribo del paquete de información a la raíz del DODAG. La probabilidad de que el paquete falle en llegar a la raíz del DODAG, de Rango 0, es:

$$q_0 = \prod_{i=1}^{n} \left(q_1^{(i)} + (1 - q_1^{(i)}) \cdot p_{ij}^m \right) . \tag{7}$$

El análisis en este paso es equivalente al del paso anterior, solo que con una pequeña diferencia. Si bien continúan habiendo n fuentes que intentarán de transmitir el paquete, ahora debe tenerse en cuenta que, llegando a la raíz del DODAG, los nodos no poseen n padres alternativos, sino que justamente el sumidero es el único destinatario. Como consecuencia, por cada nodo transmisor el paquete cuenta con sólo m chances de ser transmitido a la raíz del DODAG.

Asumiendo que el paquete no vuelve a ser retransmitido en un slotframe futuro, la PDR de extremo a extremo está dada por:

$$\text{PDR} = 1 - q_0 \tag{8}$$

Cabe destacar que la Ecuación 6 no tiene en cuenta el Overhearing realizado por los Siblings: por ende, el modelo provee una cota inferior de la PDR extremo a extremo.

Dado que un paquete logra ser transmitido con éxito a la raíz del DODAG dentro de un slotframe, el retardo extremo a extremo en el peor de los casos está definido por:

$$D_{max} = 2 \cdot n \cdot m + (R - 2) \cdot n^2 \cdot m \ , \tag{9}$$

donde:

- El primer $n \cdot m$ son las m transmisiones que el Leaf node realiza secuencialmente a cada uno de sus n padres.
- El segundo $n \cdot m$ son las m transmisiones que los n nodos de Rango 1 realizan únicamente hacia la raíz del DODAG.

- El término $(R-2) \cdot n^2 \cdot m$ es porque un existen $R-2$ saltos intermedios en los que n fuentes poseen n destinatarios posibles a los que le envía el paquete hasta m veces.

Es importante notar que para el cálculo del retardo, los términos asociados a las transmisiones que no sean hacia la raíz del DODAG se encuentran fijos. Es decir, dado que los eventos están todos programados, por más que un paquete haya logrado progresar con $l < m$ transmisiones, las restantes $l - m$ celdas del scheduling están fijas y formarán parte del retardo resultante siempre. En cambio, para el último paso, el paquete se considera que llegó a destino en el momento que se obtiene una transmisión exitosa y por ello el único término que resulta variable es la cantidad de transmisiones que los n nodos de Rango 1 realizan únicamente hacia la raíz del DODAG. En el peor de los casos, las mn transmisiones son necesarias. En el mejor de los casos, simplemente 1. Es dicha variabilidad justamente la que determina el jitter resultante para el peor de los casos:

$$J_{max} = n \cdot m - 1 \ . \tag{10}$$

Por último, D_{max} and J_{max} en este análisis están dados en cantidad de timeslots, ya que en dicho modo constituye un análisis independiente de las limitaciones de las tecnologías utilizadas. Sin embargo, dado que ambos son medidos en unidades de tiempo, para contemplar los resultados para cada caso particular, los mismos deben ser multiplicados por la duración de cada timeslot.

8 Simulaciones

En la presente sección se introduce COOJA, el software utilizado para la realización de todas las simulaciones realizadas en esta tesis. Además se define cual fue la configuración y los parámetros utilizados en RPL y IEEE 802.15.4e - TSCH. Por otro lado, también se determina el escenario analizado en las simulaciones, en términos de la topología de la red estudiada y las variaciones que la misma sufre a lo largo de cada simulación.

8.1 COOJA

De manera tal de evaluar la performance de LeapFrog Collaboration se llevaron adelante múltiples simulaciones. Las mismas fueron llevadas adelante utilizando el software COOJA. El mismo es un simulador de redes distribuido como parte del sistema operativo Contiki OS[1].

Contiki es un sistema operativo desarrollado para sistemas con restricciones de memoria, con foco en dispositivos de baja potencia e inalámbricos, como los encontrados en la Internet de las cosas. El núcleo básico y la mayor parte de las funciones principales fueron y continúan siendo desarrolladas por Adam Dunkels. Sin embargo Contiki es de código abierto y por ende ha recibido numerosas contribuciones de distintos sectores de investigación a lo largo de los años.

Contiki está diseñado para correr sobre hardware que presenta severas restricciones de memoria, potencia, poder de procesamiento y ancho de banda de comunicación. Un típico sistema de Contiki posee memoria en el orden de los kilobytes, potencia que no suele superar el orden de los miliwatts, velocidad de procesamiento medida en megahertz y ancho de banda en el orden de los cientos de kilobits/segundo. Dichos sistemas incluyen muchos tipos de sistemas embebidos.

De manera de funcionar eficientemente en estos dispositivos con pequeña memoria, Contiki usa un modelo de programación basado en protothreads. Un protothread es una abstracción que comparte características de múltiples

[1] http://www.contiki-os.org/start.html

hilos y de programación por eventos, lo cual le permite conseguir un pequeño costo en términos de memoria. El kernel invoca el protorhread de un proceso en respuesta a un evento interno o externo. Ejemplos de eventos internos pueden ser el desborde de un temporizador o mensajes que provengan de otros procesos. En cambio, un evento externo se puede asociar a un evento como el disparo de un sensor o la llevada de un paquete proveniendo de un vecino.

Como se mencionó, Contiki incluye un simulador de redes llamado COOJA. La ventaja de utilizar este software es, según la plataforma elegida, que el mismo permite simular hardware. De dicha manera, las limitaciones del mismo también pueden ser tenidas en cuenta a la hora del diseño. En particular, para todas las simulaciones se utilizaron los Z1 motes[2]. Para ello, resultó necesario utilizar el compilador TI MSP430x.

COOJA posee la interfaz gráfica que se observa en la figura 37.

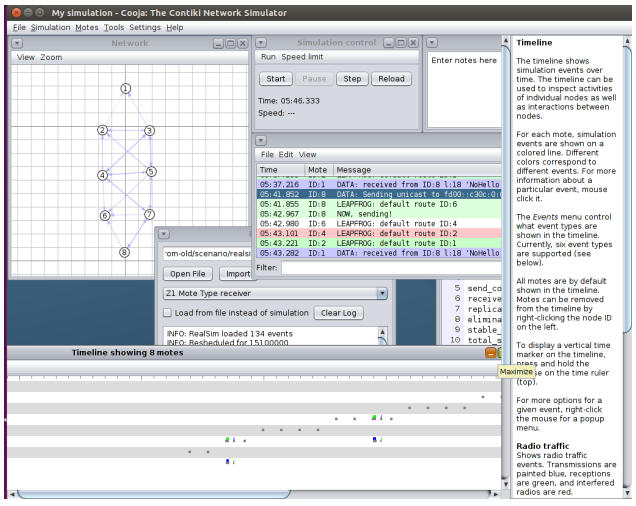


Figura 37: La GUI de COOJA.

El software permite ver la topología de la red diseñada y modificarla manualmente. A su vez, para incluir un nodo nuevo, se debe definir qué tipo de nodo será y compilar el programa que cada uno de ellos correrá. Por otro lado, también se puede ver una sección que muestra los mensajes que cada nodo imprime y que son útiles para luego analizar los resultados obtenidos. Además, se incluye una línea de tiempo, en la cual se puede ver gráficamente

[2]http://zolertia.io/z1

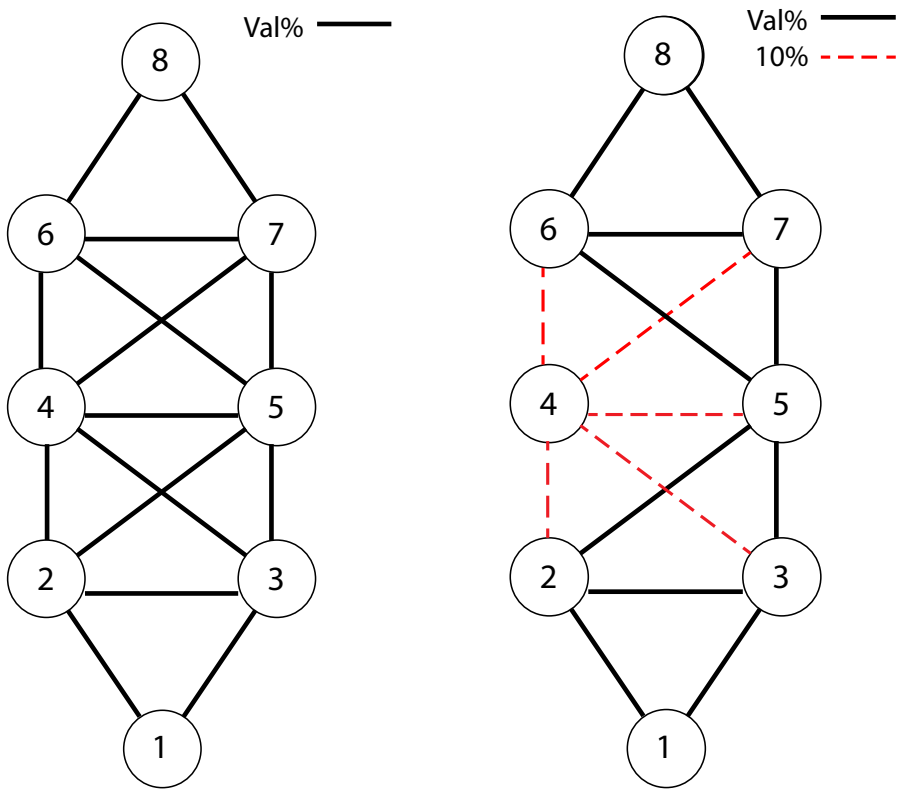
los eventos que suceden en la red: qué nodo transmite, recibe, enciende su antena o simplemente se haya en reposo. Por otro lado, utilizando una plugin llamado RealSim, resulta posible controlar la calidad de los enlaces entre los nodos en todo momento.

Si bien la interfaz gráfica resulta muy útil a la hora de crear una red y para verificar visualmente el correcto funcionamiento de la misma, para correr múltiples simulaciones no es útil ya que los resultados obtenidos no son almacenados en ningún archivo. Para ello resulta necesario realizar las simulaciones a través de comandos de línea. En dichos casos, COOJA crea dos archivos de texto donde pueden verse los resultados. En uno de ellos simplemente describe cómo se desarrollo la simulación, y en el otro se obtienen efectivamente los distintos outputs de los nodos, basados en un script de javascript que debe ser cargado de antemano. El posterior parseo de estos archivos es lo que permite obtener los resultados de las simulaciones.

8.2 RealSim

Como se mencionó, sobre COOJA se habilitó la extensión del plugin llamado RealSim. La misma permite realizar simulaciones más reales en las cuales se puede determinar exactamente el nivel de la calidad de enlace entre los distintos nodos que componen una red. De esta manera, diferentes situaciones fueron simuladas a lo largo de cada simulación, en la que alternativamente distintos nodos fallaban.

En sí, las simulaciones tomaban en cuenta la topología de la red para la creación de los vínculos entre nodos. Nodos separados más de un salto se consideraban lo suficientemente alejados y tenían nulas chances de transmitirse un paquete directamente entre ellos. Las simulaciones se dividían en distintos escenarios que se daban uno detrás del otro. En principio los enlaces entre nodos eran todos iguales y de cierto valor. Luego, este estado se sucedía por un estado de error, en el que algún nodo entraba en modo de falla y por lo cual todos sus enlaces pasaban a tener una calidad drásticamente pequeña. Ejemplos de ambos escenarios pueden verse en la figura 38.



(a) Escenario "def" (b) Escenario "id:4"

Figura 38: Escenarios en una misma simulación.

En el ejemplo de la figura, todos los nodos poseían enlaces de valor "Val%" para el escenario "def". Este último fue llamado así ya que representaba el caso por default. A su vez, muchos de estos enlaces conservaban su valor en los escenario de fallas. Las distintas simulaciones realizadas variaban el valor de Val%: se consideraron los casos para valores 100%, 90%, 80%, 70%, 60% y 50%. Por otro lado, el estado de error, representado por el escenario "id:4" en la figura, recibe dicho nombre ya que en él es el nodo 4 el que justamente entra en modo de fallas. En dicho caso, todos sus enlaces que lo conectan con otros nodos bajan su calidad y adoptan un valor de 10%. A modo más genérico, los estados de error se identifican como "IdX", donde X se refiere al ID del nodo cuyos enlaces fallan.

8.3 Setup General

En la tabla 1 se presentan las primeras definiciones en cuento a las características de las simulaciones realizadas.

Tabla 1: Topología y Hardware.

Topología	**Valor**
Topología	Multi-hop
Número de nodos	8 (incluyendo la raíz del DODAG)
Número de fuentes	1 fuente
Espaciamiento entre nodos	10 m (en promedio)
General	**Valor**
Tráfico generado	1 pkt/15 sec
Payload	17 bytes
Hardware	**Valor**
Tipo de nodo	Z1 motes
Modelo de antena	CC2420
Propagación de radio	2.4 GHz
Potencia de transmisión	0 dBm

La topología de la red puede verse específicamente en la figura 39. En ella los nodos se encuentran distribuidos en dos rectas paralelas en un área de $40 \cdot 10$ m^2. A su vez, el nodo 8 es la fuente de información (o el Leaf Node) que envía paquetes cuyo payload es de 17 bytes cada 15 s. Dichos paquetes tienen por destino al nodo 1, que es la raíz del DODAG y el punto de colección de la red que conectaría a la misma con Internet. El resto de los nodos simplemente ayudan a transmitir el paquete a destino, pero no generan tráfico por sí mismos. Los nodos sólo "ven" los nodos a los cuales están conectados con las líneas punteadas, y la calidad de los enlaces se controlan mediante RealSim, como se explicó en la sección anterior.

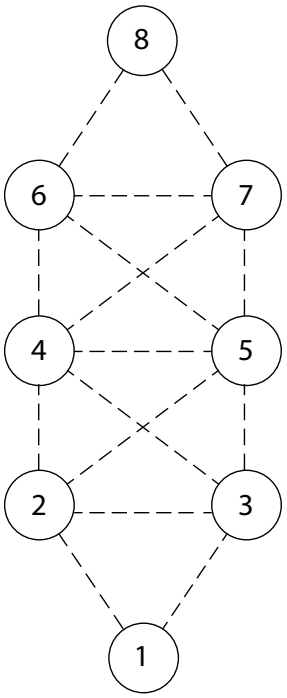


Figura 39: Topología analizada en las simulaciones.

En cuanto al hardware, se utilizaron los Z1 motes para simular, cuya antena en COOJA es el modelo CC2420 y a través de la cual las transmisiones ocurrían en la banda de los 2.4 GHz. Por último, la potencia de transmisión era 0 dBm.

Por otro lado, en COOJA resultó necesario establecer a RPL como método de ruteo, y el uso de TSCH en la capa MAC. Las respectivas configuraciones se observan en las tablas 2 y 3 respectivamente.

Tabla 2: Configuración de RPL.

RPL	**Value**
Instancias RPL	1
DODAGs	1
Función Objetivo	MRHOF
Nétrica para el Rango	ETX
Modo de Operación	Almacenado

La red consistía de una única Instancia RPL equivalente al único DODAG formado, cuya raíz era el nodo 1. La Función Objetivo establecida fue la MRHOF, que utiliza como métrica la ETX. La misma pondera los enlaces

en función de la cantidad de paquetes que son transmitidos con éxito. Por último, el Modo de Operación fue definido como Almacenado, lo cual significa que la creación de rutas descendentes y el envío adicional de paquetes DAO y DAO-ACK en la red.

Tabla 3: Configuración de TSCH.

TSCH	**Value**
Tamaño Slotframe	101 timeslots
Duración Timeslot	15 ms
$\mathcal{F}$ - Cant. Canales	4
$\mathcal{F}$ - Long. Sec.	4
EB_PERIOD	4 s

En cuanto a la capa de enlace, para el uso de TSCH, se definió un slotframe de 101 timeslots, los cuales duraban cada uno 15 ms. Este último valor fue definido siguiendo las recomendaciones establecidas para el tipo de nodo Z1. Además, la función $\mathcal{F}$, que determina la secuencia de saltos, se escogió siguiendo una implementación dada en COOJA, que propone el uso de 4 canales de frecuencia para conformar una secuencia de dicha longitud. Las frecuencias escogidas están en la banda de los 2.4 GHz, en concordancia con lo establecido en la tabla 1. Por último, los Enhanced Beacons son enviados por cada nodo que se halle ya unido a la red cada 4 s.

Además, dada la topología de la red, el scheduling quedaba definido como el de la figura 40.

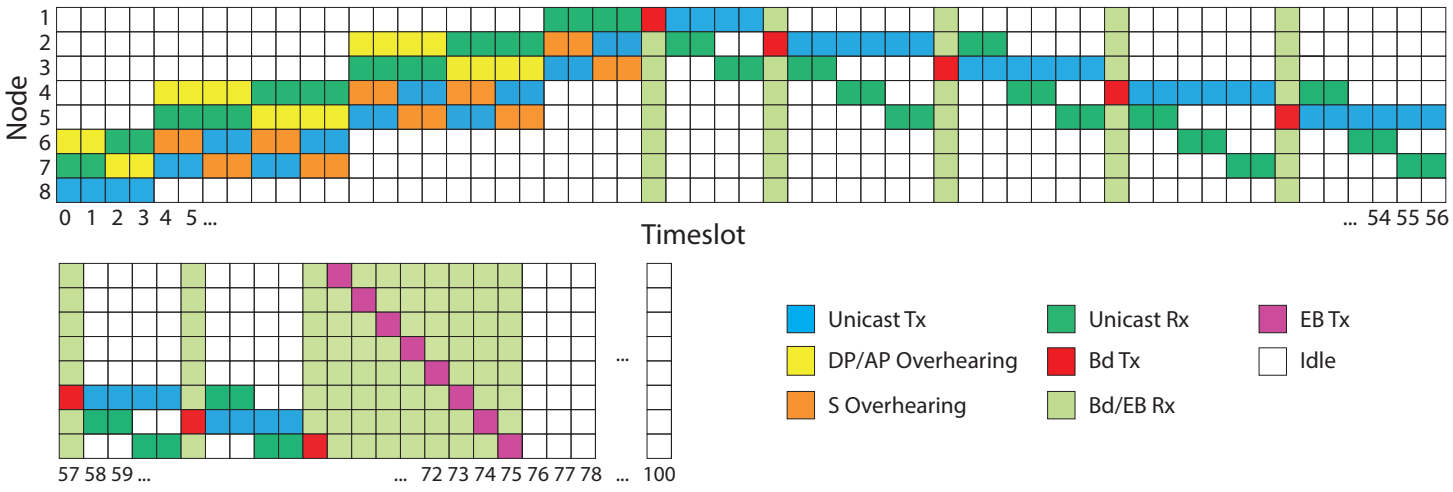


Figura 40: Scheduling de la red de 8 nodos.

Donde, teniendo en cuenta el análisis matemático realizado con anterioridad, y recordando que la duración de cada timeslot es de 15 ms, en la tabla 4 se presentan como quedan definidos ciertos valores de interés.

Tabla 4: Métricas del Scheduling

Scheduling	**Value**
n (# padres)	2
m (# transmisiones)	2
D_{max}	24 timeslots - 360 ms
D_{min}	21 timeslots - 315 ms
J_{max}	3 timeslots - 75 ms

En conclusión, para esta topología de la red, el scheduling propuesto tendrá un retardo máximo de 360 ms y un retardo mínimo de 315 ms, lo cual determina directamente que el valor máximo del jitter, que es la diferencia entre ambos, sea de 75 ms. Cabe destacar que, sin embargo, el funcionamiento de LeapFrog Collaboration está pensado para que todos los nodos en cada capa de la red posean el paquete a entregar a la raíz del DODAG, y ello hace que sea esperable que la mayor cantidad de paquetes sea entregada con un retardo mínimo.

8.4 Simulaciones Consideradas

El desempeño de LeapFrog Collaboration implementado sobre COOJA fue comparado con la implementación de RPL+TSCH-RTx donde x representa la cantidad máxima de retransmisiones que un nodo puede llegar a realizar en su misión de transmitir un paquete a otro nodo. Estas retransmisiones fueron variadas entre 0, 2, 4, 6 y 8, donde cada una de ellas se da en un slotframe diferente. A su vez, en todos los casos se simuló utilizando el scheduling diseñado para LeapFrog Collaboration.

Por otro lado, los primeros 5 minutos de todas las simulaciones fueron dedicados a la conformación de la red, y no se enviaron paquetes con información. Es decir, sólo hubo tráfico de paquetes TSCH y RPL, que permitieron que los nodos se unieran a la red, sincronizando sus relojes, y determinaran sus respectivos Default Parents. En el caso particular de LeapFrog Collaboration, este tiempo también fue utilizado por los nodos para determinar sus respec-

tivos Alternative Parents y definir el proceso de Overhearing. A partir de dicho momento, el nodo 8 de la figura 39 comenzó a enviar tráfico (paquetes de información) también, siguiendo las características indicadas en la tabla 1.

8.4.1 Caso de Prueba: una falla por nodo

Este caso sirvió de referencia para saber si la implementación de LeapFrog Collaboration producía resultados en torno a lo esperado. Los distintos escenarios que se sucedieron en la simulación seguían la secuencia: “def”, “id:6”, “def”, “id:4”, “def”, “id:3”, “def”, “id:5”, “def”, “id:2”, “def”, “id:7”. Cada escenario tenía una duración de dos minutos y medio.

8.4.2 Caso Final: dos fallas por nodo

El caso de prueba permitía tener una idea estimativa de la performance de LeapFrog Collaboration, pero dado que no tantos paquetes eran enviados, ello limitaba la precisión de los resultados. Por ello, se optó por la creación de una nueva simulación final. En la misma se duplicaron los escenarios, ampliando la secuencia original: “def”, “id:6”, ”def”, “id:4”, “def”, “id:3”, “def”, “id:5”, “def”, “id:2”, “def”, “id:7”, “def”, “id:6”, “def”, “id:7”, “def”, “id:2”, “def”, “id:5”, “def”, “id:3”, “def”, “id:4”. Cada escenario tuvo una duración de cinco minutos.

8.4.3 Resultados Esperados

Según el scheduling de la figura 40, el retardo y jitter quedaban fijados y fueron determinados en la tabla 4. En cuanto al ciclo de trabajo, a priori resultaba esperable un mayor consumo por parte de LeapFrog Collaboration comparado con las implementaciones de RPL+TSCH-RTx, a coste de una mejor performance en términos de PDR.

En cuanto al valor exacto de la PDR esperada, si bien en la sección del cálculo matemático se hallaron algunas expresiones, las mismas consideraban que las distintas p_{ij}, la probabilidad de error de enlace para el nodo i transmitiendo al padre j, eran constantes. Del mismo modo, las distintas $1 - p_{ij}$, la probabilidad de éxito de enlace para el nodo i transmitiendo al padre j, también lo eran. Sin embargo, en las simulaciones consideradas, en

los escenarios "def" todos los nodos poseían enlaces de cierta calidad p, que eventualmente podía pasar a tener un valor de p_{bad} si el nodo i o j entraba en modo de fallas, es decir en un escenario "id:i" o "id:j".

El calculo detallado de la PDR teórica se halla en la sección 13, y sus resultados se resumen en la tabla 5.

Tabla 5: PDR teórica según el valor de la calidad de los enlaces.

p	**PDR**
100%	0.99997
90%	0.99923
80%	0.99553
70%	0.98483
60%	0.96072
50%	0.91198

9 Resultados

9.1 Caso de Prueba: una falla por nodo

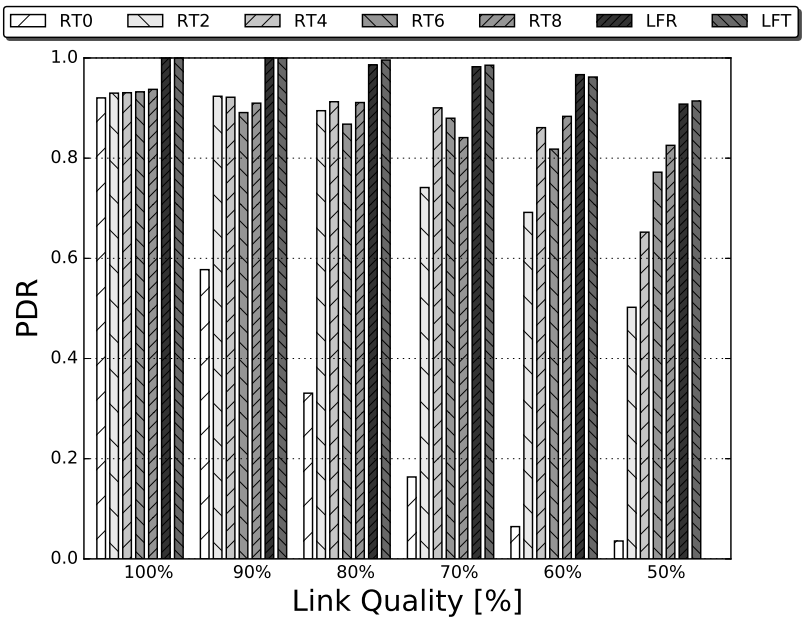


Figura 41: PDR para TSCH-RTx, LFC y LFC teórico.

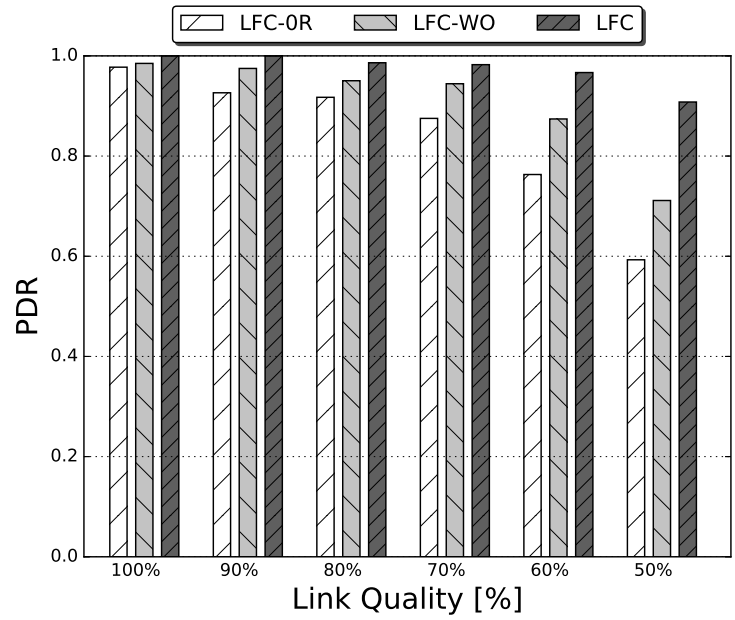


Figura 42: PDR para LFC full, sin retransmitir (0R) y sin Overhearing (WO).

9.2 Caso Final: dos fallas por nodo

9.2.1 PDR

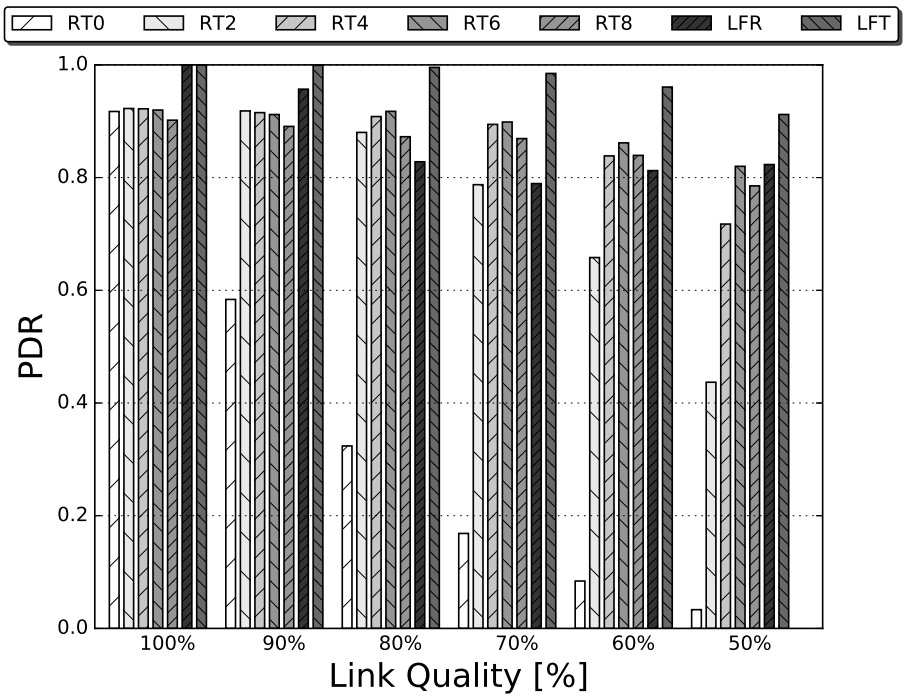


Figura 43: PDR para TSCH-RTx, LFC y LFC teórico.

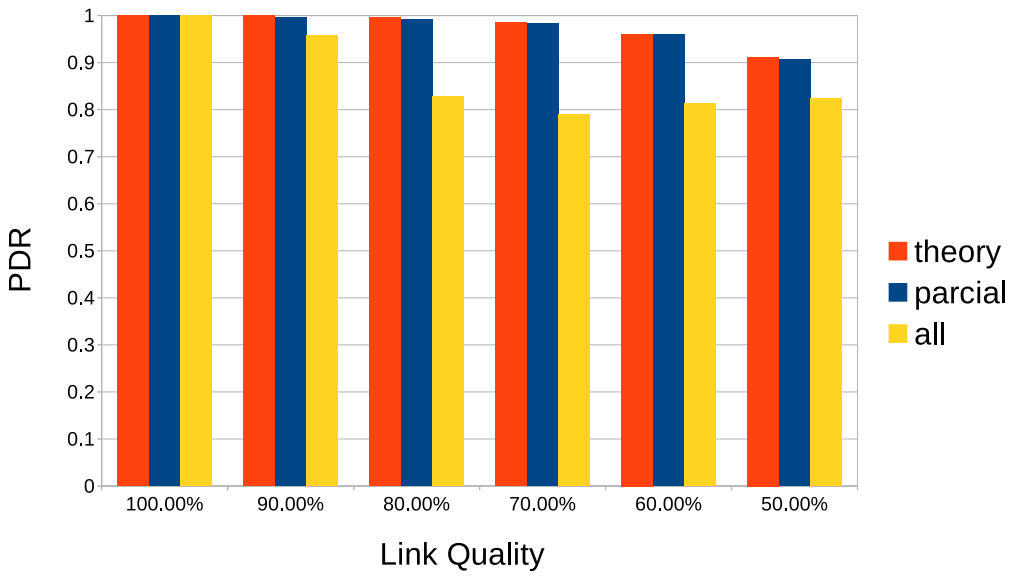


Figura 44: PDR para LFC teórico, parcial y total.

9.2.2 Delay-Jitter

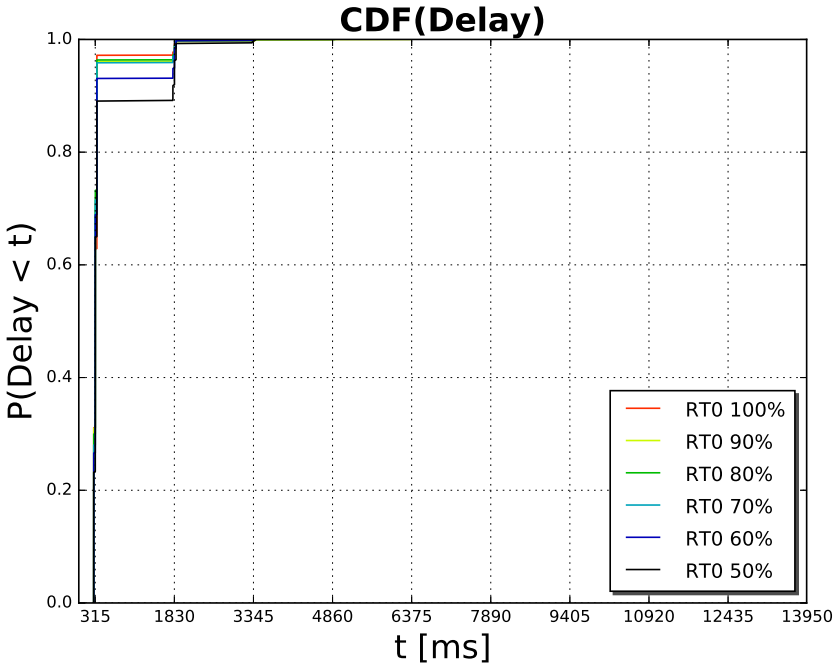


Figura 45: Función Acumulativa Estimada para TSCH con 0 retransmisiones.

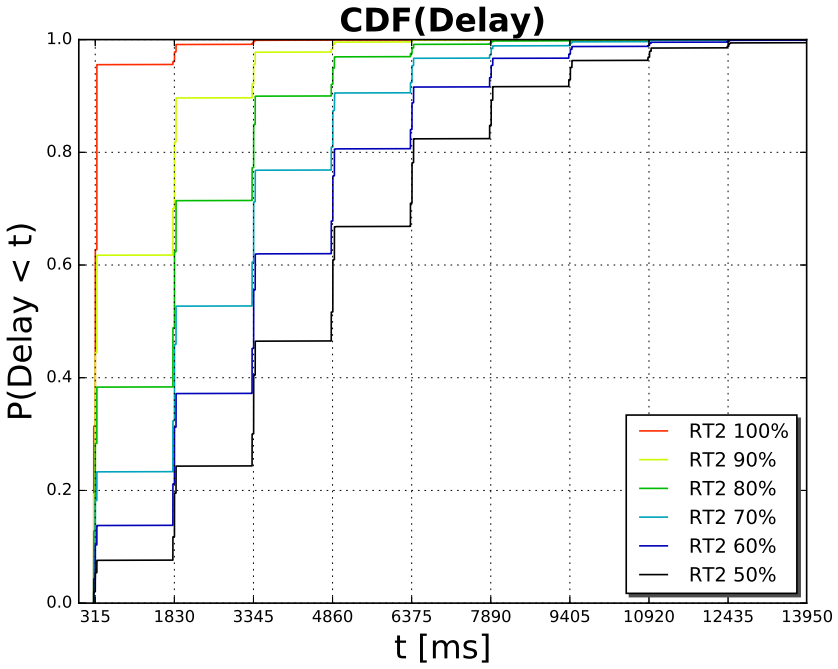


Figura 46: Función Acumulativa Estimada para TSCH con 2 retransmisiones.

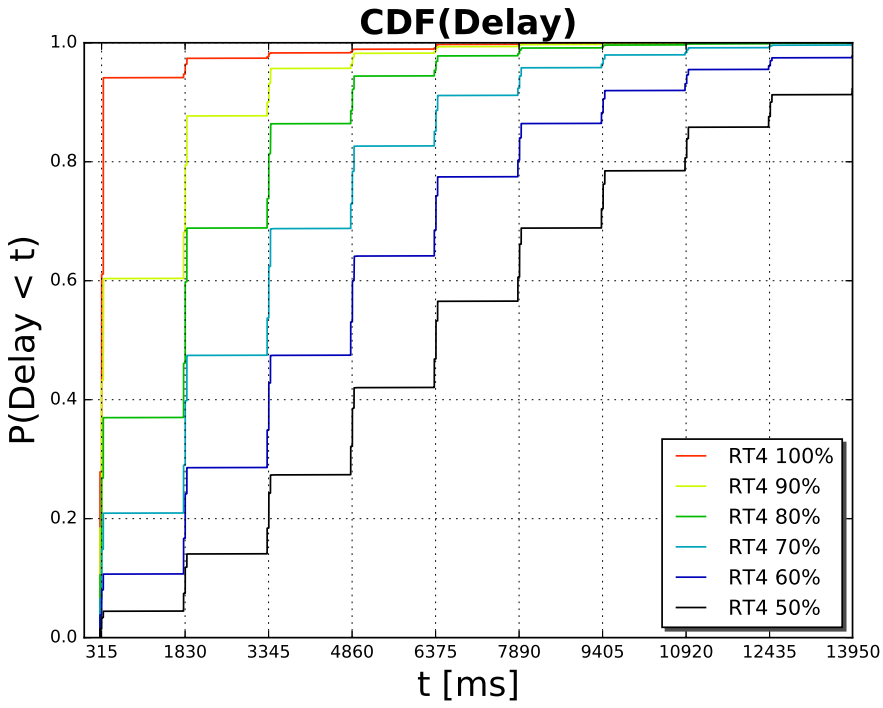


Figura 47: Función Acumulativa Estimada para TSCH con 4 retransmisiones.

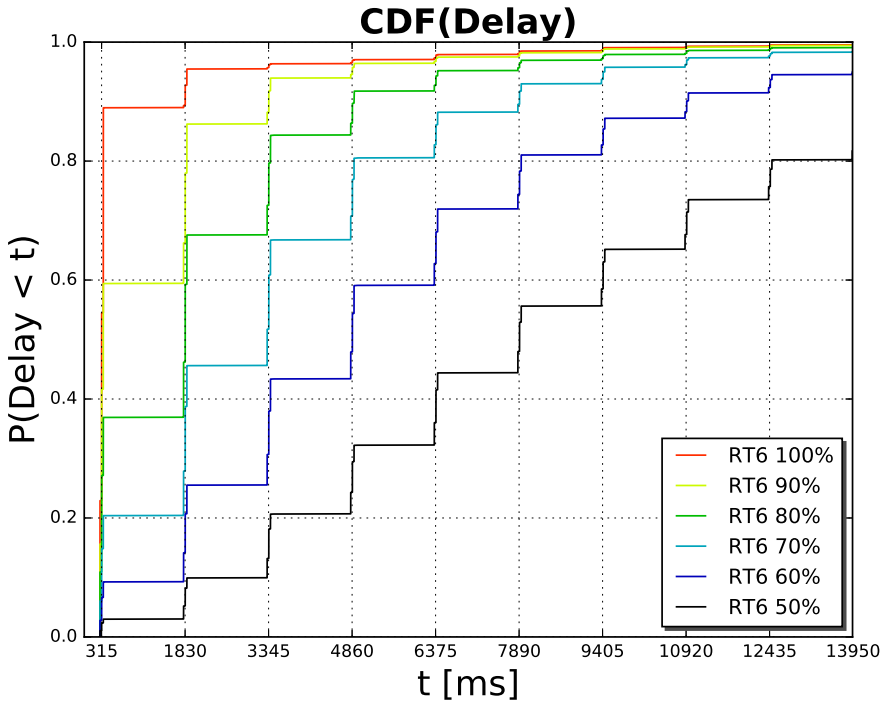


Figura 48: Función Acumulativa Estimada para TSCH con 6 retransmisiones.

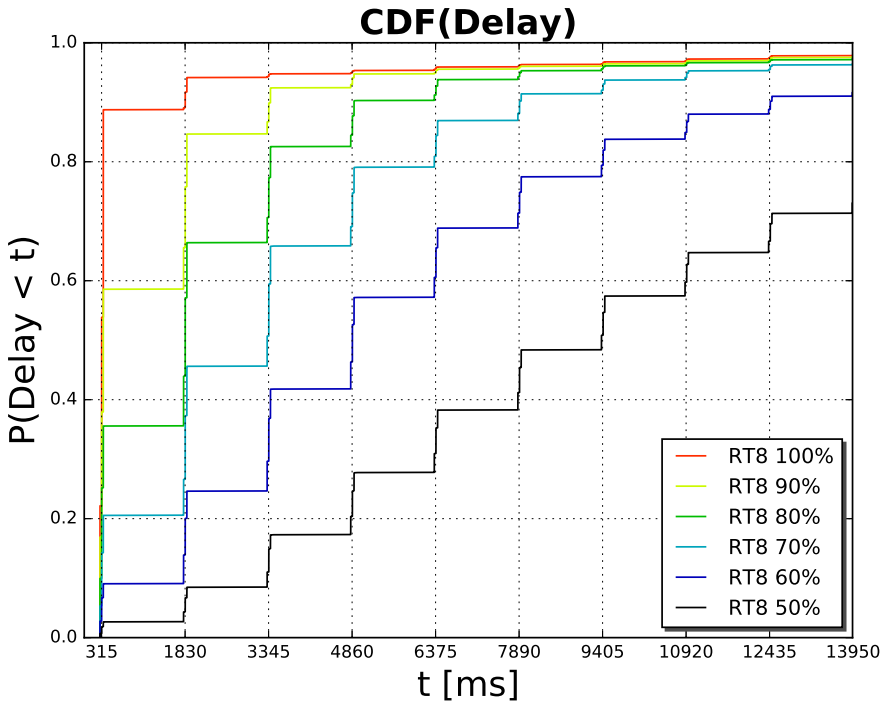


Figura 49: Función Acumulativa Estimada para TSCH con 8 retransmisiones.

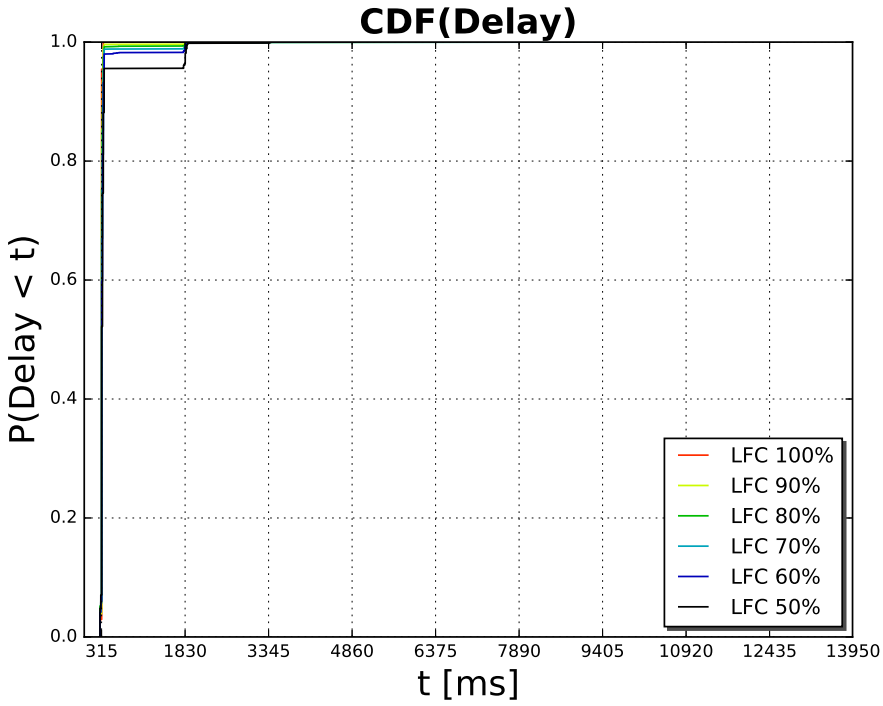


Figura 50: Función Acumulativa Estimada para LeapFrog Collaboration.

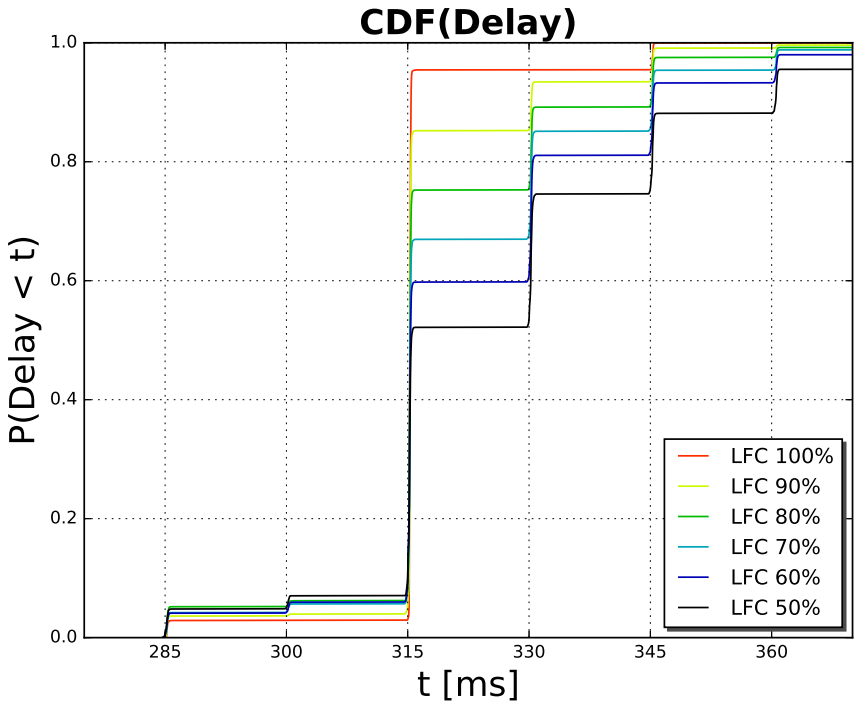


Figura 51: Zoom Función Acumulativa Estimada para LeapFrog Collaboration.

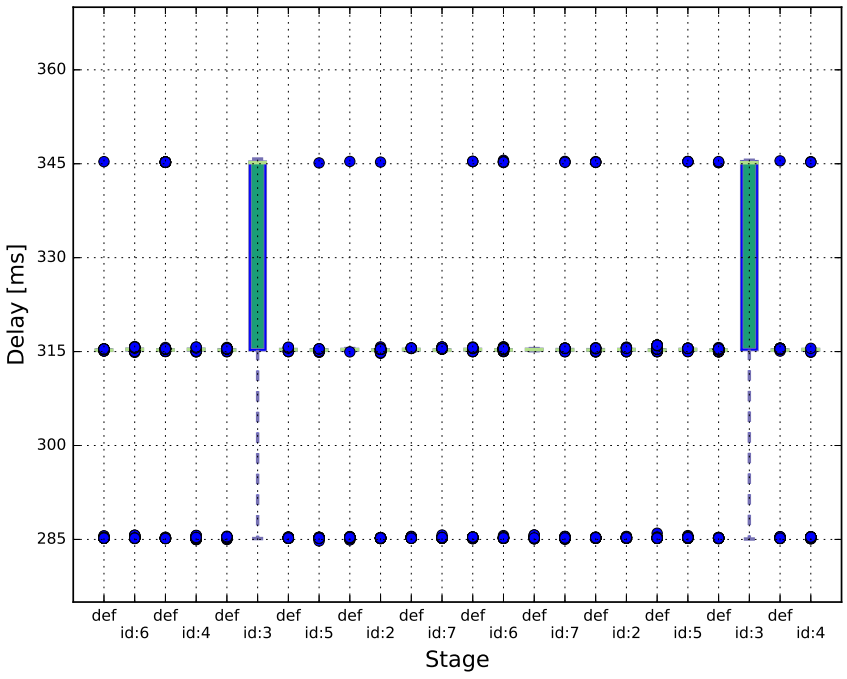


Figura 52: Detalle retardo LFC caso enlaces 100%

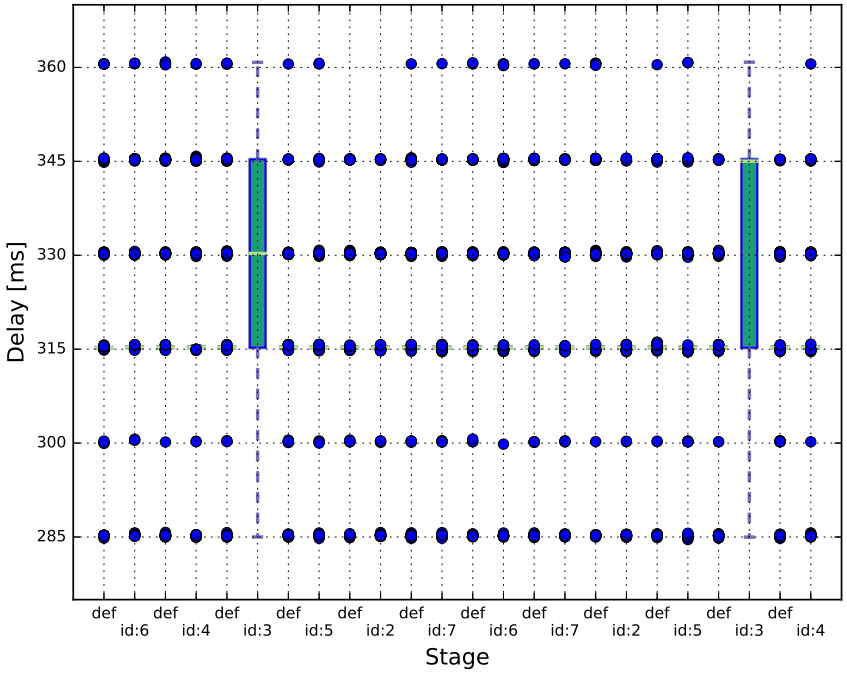


Figura 53: Detalle retardo LFC caso enlaces 90%

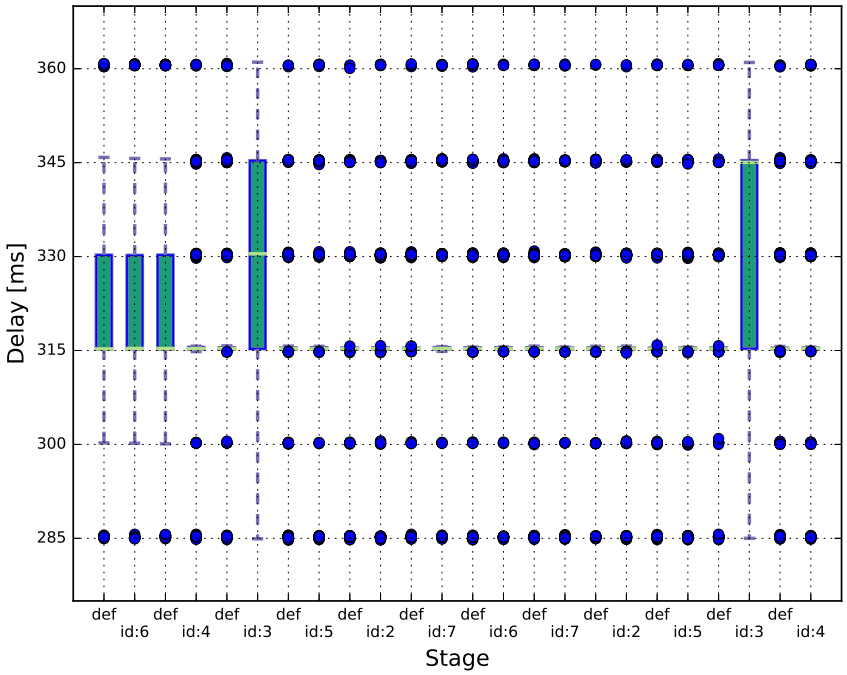


Figura 54: Detalle retardo LFC caso enlaces 80%

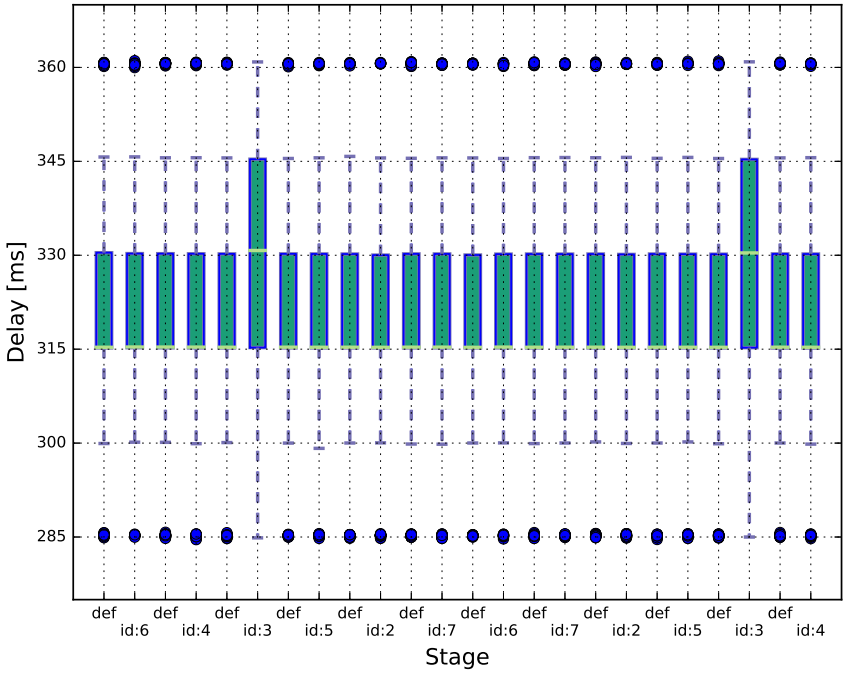


Figura 55: Detalle retardo LFC caso enlaces 70%

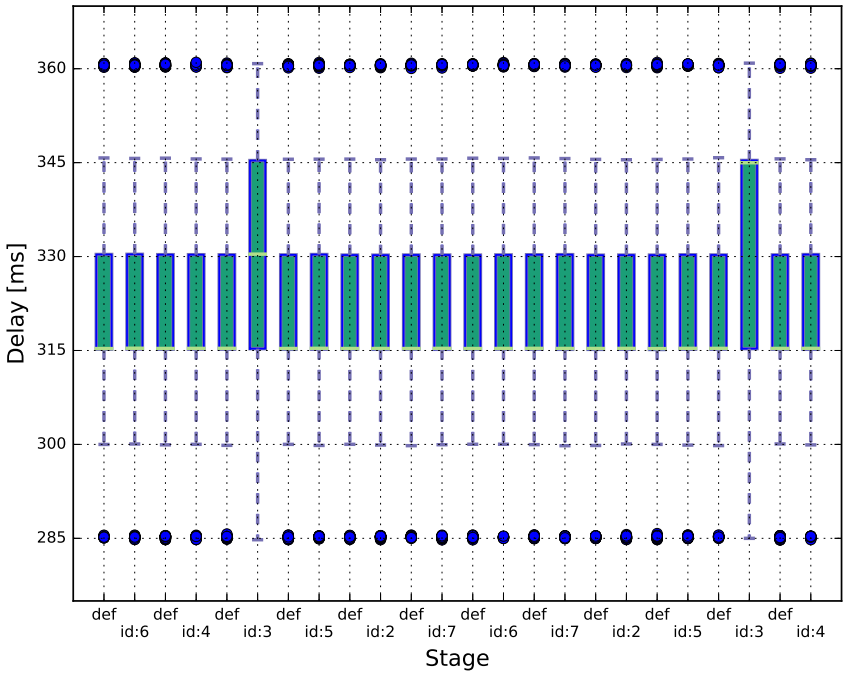


Figura 56: Detalle retardo LFC caso enlaces 60%

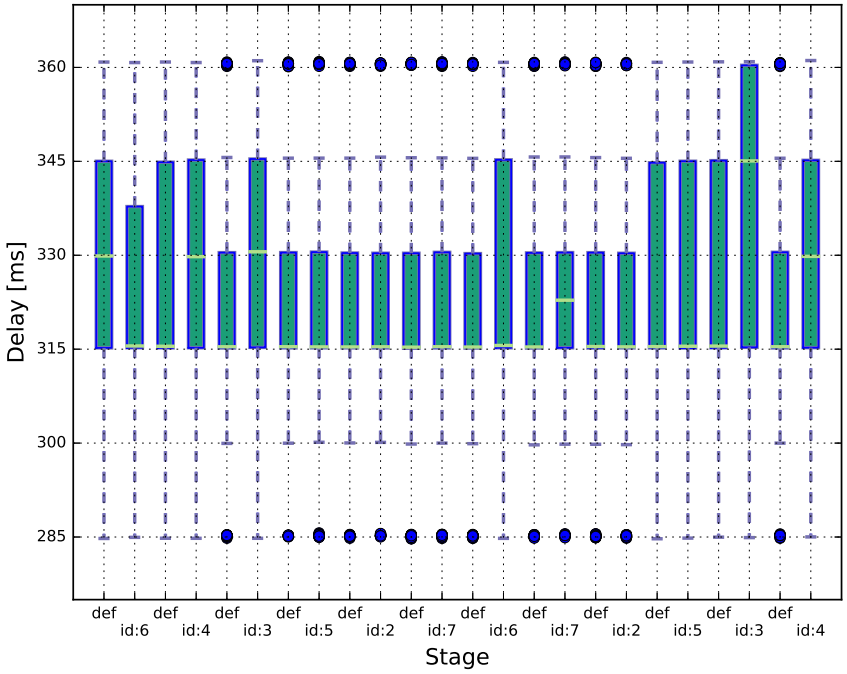


Figura 57: Detalle retardo LFC caso enlaces 50%

9.2.3 Ciclo de Trabajo

Tabla 6: Ciclo de Trabajo para el caso de enlaces al 100%.

Configuración	Ciclo de Trabajo
RT0	4.52%
RT2	4.58%
RT4	4.61%
RT6	4.68%
RT8	4.75%
LFC	5.25%

9.2.4 Otros Resultados

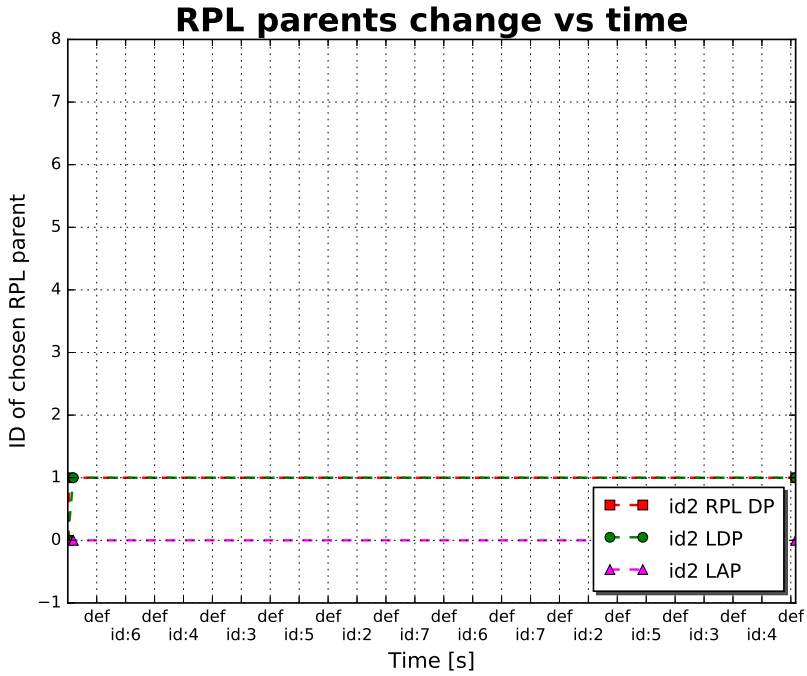


Figura 58: Evolución del RPL parent, LDP y LAP para el nodo 2.

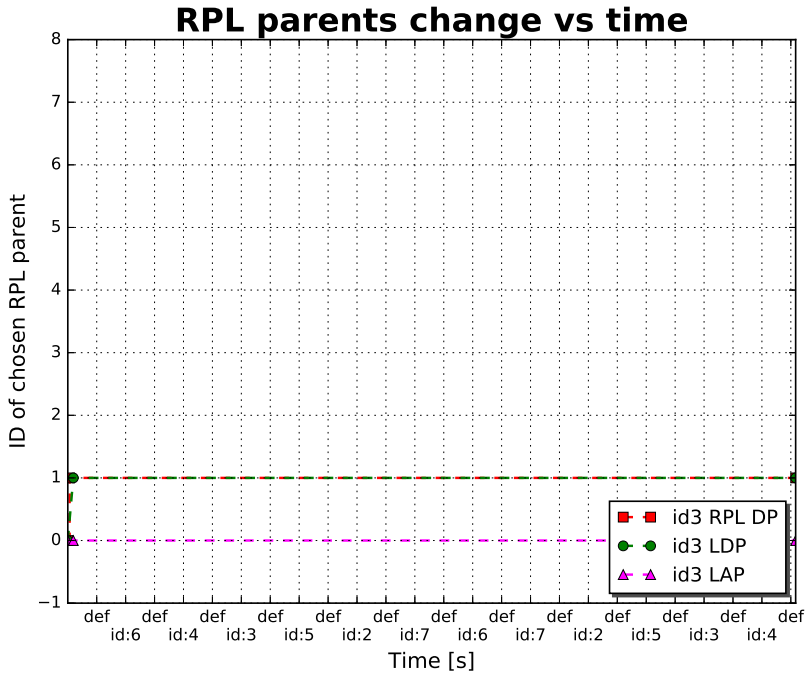


Figura 59: Evolución del RPL parent, LDP y LAP para el nodo 3.

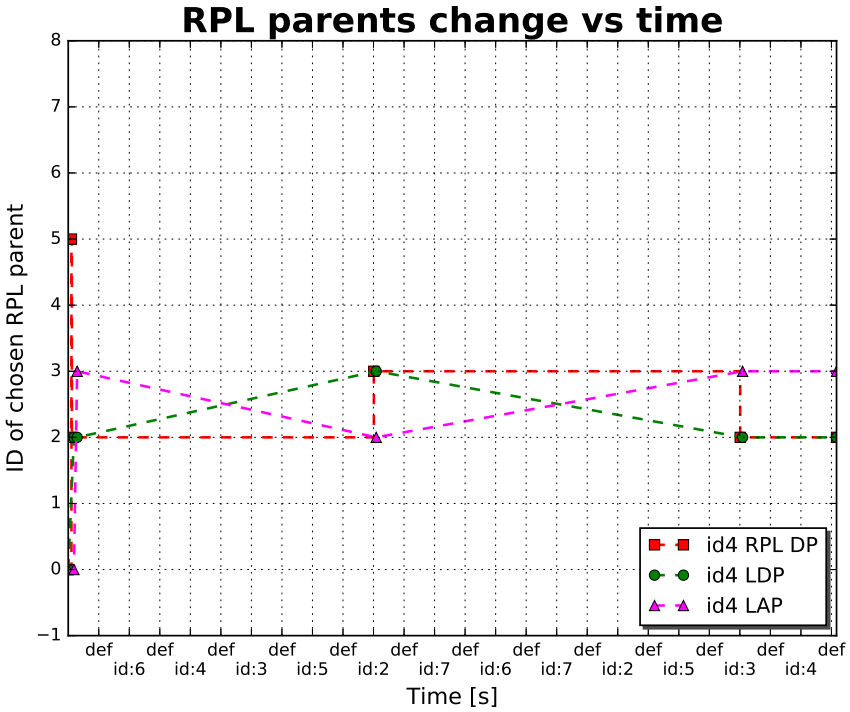


Figura 60: Evolución del RPL parent, LDP y LAP para el nodo 4.

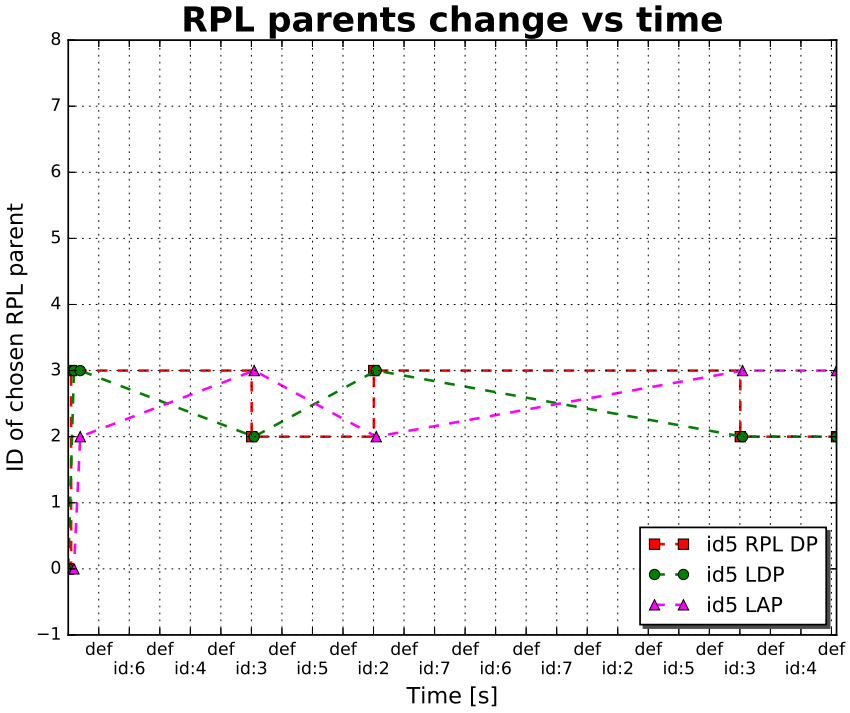


Figura 61: Evolución del RPL parent, LDP y LAP para el nodo 5.

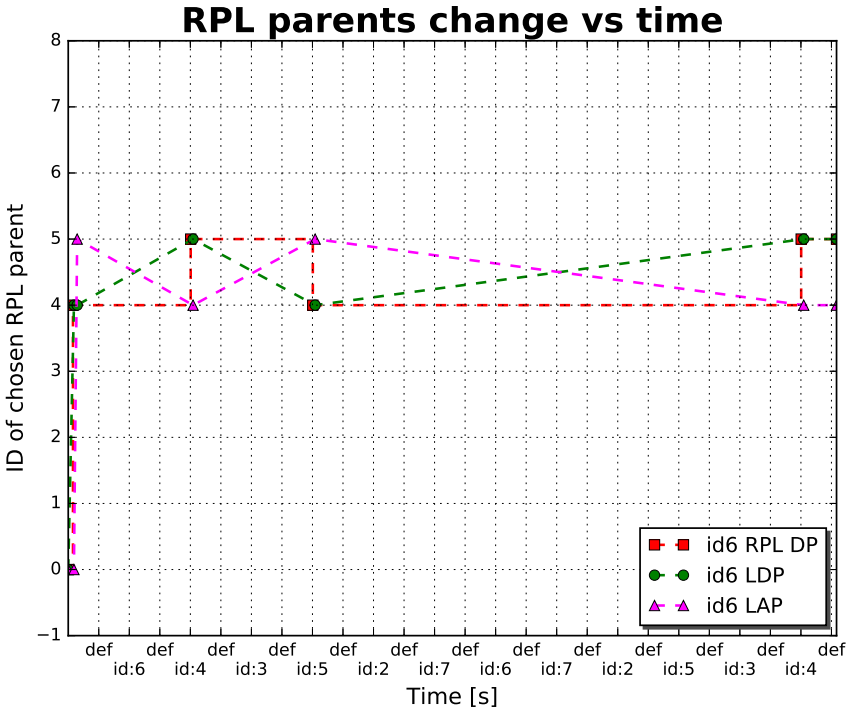


Figura 62: Evolución del RPL parent, LDP y LAP para el nodo 6.

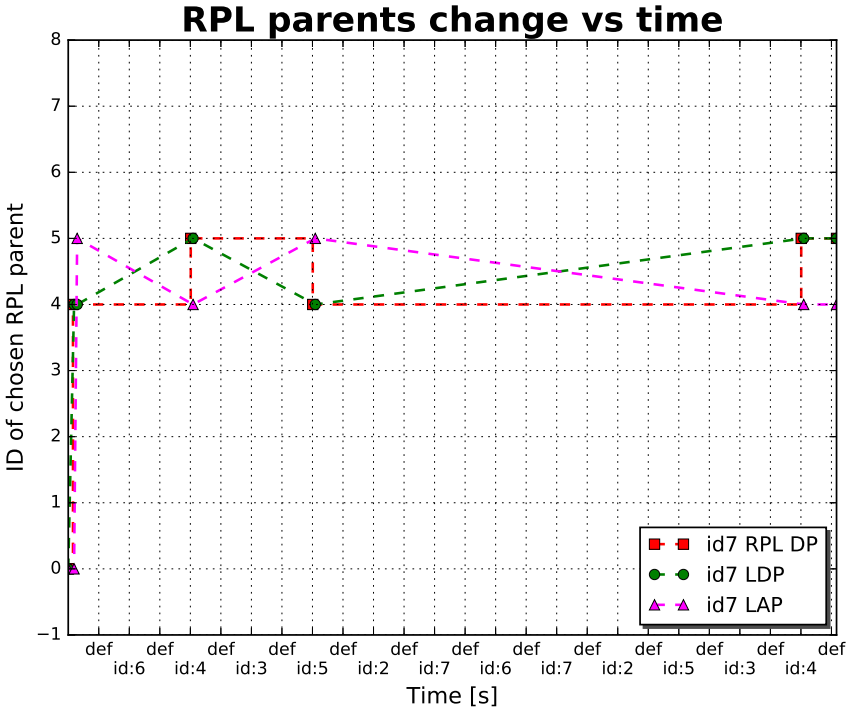


Figura 63: Evolución del RPL parent, LDP y LAP para el nodo 7.

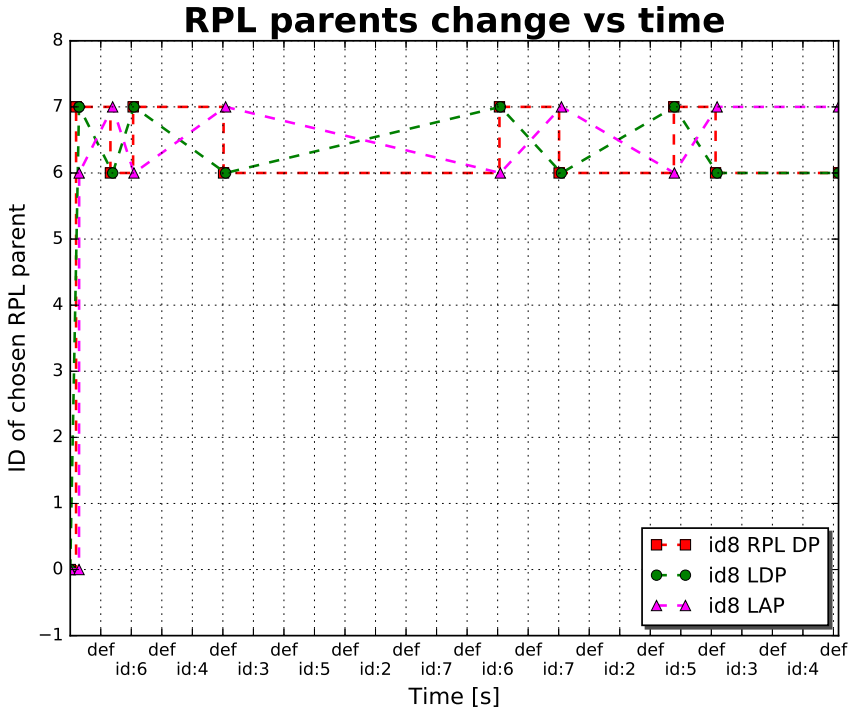


Figura 64: Evolución del RPL parent, LDP y LAP para el nodo 8.

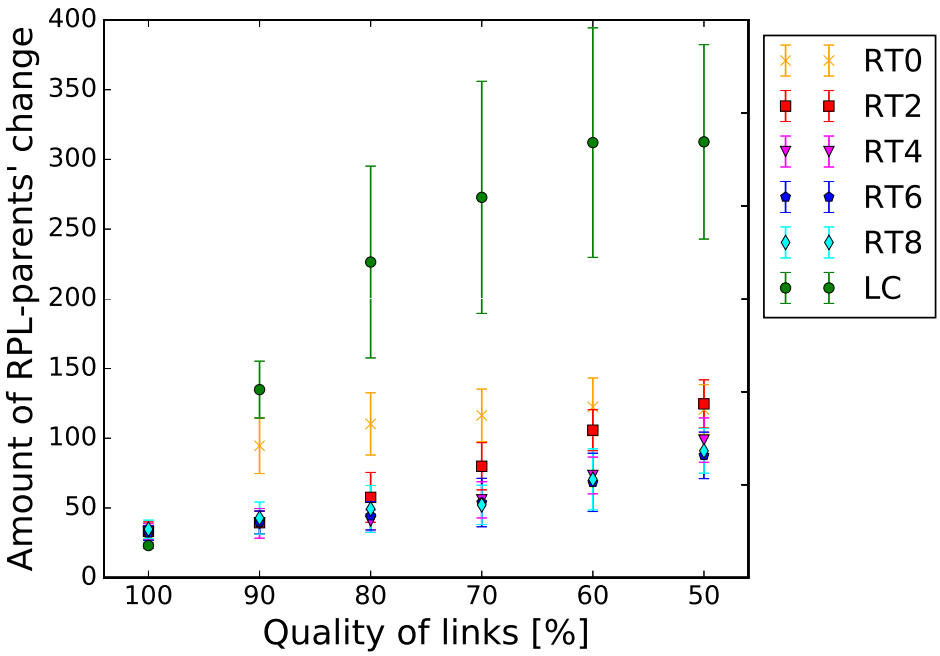


Figura 65: Cantidad de cambios de RPL Parents en función de la calidad de los enlaces.

10 Análisis de los Resultados

A continuación se presentan las conclusiones a las cuales se puede arribar a través del análisis de las figuras 41 a 65. Se analiza la PDR para el caso de prueba y también el resto de las métricas de interés para el caso final.

10.1 Caso de Prueba: una falla por nodo

En la figura 41 se observa que LeapFrog Collaboration (LFR) presenta una perfomance que supera en todos los casos a la obtenida por TSCH con x retransmisiones (RTx), cualquiera fuese el valor de los considerados para x. A su vez, los resultados teóricos (LFT) se hallaron en total sintonía con los resultados obtenidos, lo cual en parte da cuenta de que el modelizado del problema fue correctamente llevado a cabo.

Por otro lado, de manera de exponer más claramente las bondades aportadas por cada una de las características de LeapFrog Collaboration (LFC), se lo simuló nuevamente, en primer lugar desactivando la opción de Overhearing (LFC-WO) y en segundo término sin el uso de las retransmisiones (LFC-0R).

En la figura 42 se observan los resultados complementarios comparativos para LeapFrog Collaboration, nuevamente en términos de PDR. En líneas generales, el uso del Alternative Parent demostró ser la cualidad que más permite mantener una elevada PDR. Esto se verifica teniendo en cuenta que las 3 curvas presentaron resultados muy buenos, excepto en los últimos dos casos donde la situación es adversa y los enlaces presentaban valores de 60% y 50% de probabilidad de éxito respectivamente.

Si bien en estos últimos casos es donde más se evidencia el aporte de las características complementarias que conforman la versión completa de LeapFrog Collaboration, debe tenerse en cuenta que la suma de aproximadamente un 6% en el valor de la PDR que éstos producen en los casos de enlaces buenos, también resultan considerables: mientras más cerca se está de un resultado ideal, más difícil se vuelve encontrar una solución superadora. El Overhearing y las retransmisiones cumplen con ello, y ayudan a que la performance de LeapFrog Collaboration, comparada con la implementación de RPL+TSCH-

RTx, se condiga más con el de una red determinística.

En definitiva, los resultados preliminares en el caso de prueba permitieron corroborar la robustez en términos de PDR de la propuesta de esta tesis.

10.2 Caso Final: dos fallas por nodo

Dado que el caso de dos fallas por nodo representa el caso final a analizar, se incluyen los resultados no sólo teniendo en cuenta la PDR, sino que también en función del retardo-jitter y el ciclo de trabajo.

10.2.1 PDR

En la figura 43 se observa que LeapFrog Collaboration (LFR) presentó una perfomance muy diferente al caso de prueba. En el caso anterior la similitud entre los resultados teóricos (LFT) con los reales obtenidos había sido directa. Sin embargo, en las nuevas simulaciones únicamente para el caso de enlaces al 100% los resultados se encontraron tan finamente de acuerdo entre sí. El caso de enlaces al 90% también tuvo mejores resultados que para TSCH con x retransmisiones (RTx), pero marcando una diferencia con lo esperado, que se fue acrecentando para el resto de las simulaciones con decreciente calidad de enlaces.

El análisis de los resultados permitió obtener una explicación para este fenómeno: del total de las simulaciones realizadas para las distintas calidades de enlaces había un porcentaje cercano al 30% cuyos resultados eran por debajo de lo esperado, pero el restante 70% seguían la tendencia teórica. Esto puede observarse claramente en la figura 44, donde se grafica el resultado teórico, el resultado obtenido en las simulaciones teniendo en cuenta sólo las simulaciones "exitosas" y el resultado según la totalidad de las simulaciones realizadas. Resulta notorio cómo en los casos de éxito, el funcionamiento de LeapFrog Collaboration fue acorde a lo esperado.

Se procedió a repetir las simulaciones con resultados fuera de lo esperado, pero utilizando la interfaz gráfica de COOJA, que provee más herramientas en el análisis de funcionamiento de un código. De allí surgió rápidamente cual era la fuente de error: el nodo 8 que generaba el tráfico, a nivel UDP creaba

la trama, a nivel de RPL elegía las rutas en función de sus padres por default y alternativo, pero al momento de efectivamente realizar la transmisión a nivel de la capa MAC y física, no se producía. Sin embargo, como en todo el proceso los distintos outputs que se utilizan para obtener los resultados continuaban siendo generados, se pensaba en forma errónea que los paquetes se habían perdido en el proceso, cuando en verdad no habían sido enviados.

Teniendo esto en cuenta, se decide descartar las simulaciones con malos resultados a la hora de juzgar el funcionamiento de LeapFrog Collaboration: éstos se producían por una falla a un nivel de programación global y no debido a un mal desempeño de la propuesta de esta tesis. Es decir, es la versión implementada sobre COOJA la que presenta el problema, pero no la idea detrás de LeapFrog Collaboration en sí misma.

En cuanto al motivo que desencadena el problema, un factor a tener en cuenta es que las simulaciones en el caso final son 4 veces más largas que en el caso de prueba. Ello siembra más posibilidades a que, si existe cierta condición de error que puede darse en la implementación, entonces la misma efectivamente suceda. Sin embargo, resulta desconcertante, quizás el hecho de que para los casos con enlaces al 100% y al 90%, los resultados hayan diferido en menos de un 5% con respecto a los esperados, pero en el resto de los casos por al menos el doble de dicha cifra. Por otro lado, la versión de COOJA utilizada fue la 2.6, existiendo otras más recientes en las que ciertos errores pueden haber sido corregidos, y que por ende pueden continuar influenciando los resultados de la implementación de LeapFrog Collaboration. A pesar de todas estas consideraciones, al momento de realizar esta tesis aún no se llegó a determinar y corregir fehacientemente el problema.

En definitiva, aceptando que los resultados parciales de la figura 44 son los que representan verdaderamente el funcionamiento como idea teórica de LeapFrog Collaboration, entonces se puede afirmar nuevamente que, comparado con RPL+TSCH con retransmisiones, la propuesta de esta tesis permite obtener una red mucho más fiable en cuanto a la entrega de paquetes de extremo a extremo.

10.2.2 Retardo-Jitter

Las figuras 45 a 50 muestran el retardo experimentado por los distintos paquetes que efectivamente llegan a destino para TSCH con 0, 2, 4, 6, y 8 retransmisiones y LeapFrog Collaboration respectivamente al variar la calidad de los enlaces.

Comparando las figuras 45 a 49 se puede ver que, a medida que se aumenta la cantidad máxima de retransmisiones posibles, para el caso de RPL+TSCH, son cada vez más y más los paquetes que acumulan probabilidad para mayores retardos de extremo a extremo. A su vez, observando cada gráfico en detalle, esta tendencia se repite al ir considerando uno por uno los resultados obtenidos para la calidad decreciente de enlaces.

Sólo RPL+TSCH sin retransmisiones presenta un resultado comparable en cuanto al retardo teórico esperado por LeapFrog Collaboration. Esto sucede dado que, sin retransmisiones, los paquetes que arriban sólo lo hacen si en la única chance que tuvieron de atravesar cada uno de los enlaces, entonces lo lograron. Sin embargo, la PDR que esa configuración consigue es muy pobre, como puede apreciarse en la figura 43, y por ello no constituye un ejemplo de referencia.

En el caso de LeapFrog Collaboration, la figura 50 demuestra que, con esta propuesta, el retardo resulta más predecible y acotado. A pesar de ello, el retardo máximo resultó mayor a los 360 ms predichos en la sección de resultados esperados: algunos paquetes llegaron con retardo de uno o dos slotframes adicionales, acumulando cierta probabilidad en dicha región de la figura.

El estudio en detalle de dicho fenómeno permitió determinar en donde radicaba la causa de esto: si bien el nodo hoja generaba tráfico y enviaba los paquetes, en algunos casos, los nodos intermedios contaban con paquetes de control de RPL, específicamente paquetes DAO, previamente almacenados en el buffer y por ende les daban prioridad. Ello cortaba la cadena de transmisiones del paquete de información, el cual entonces se veía obligado a esperar a la siguiente oportunidad de transmisión, que no se daba sino hasta el siguiente slotframe. El encadenamiento en cascada de este efecto para los

diferentes nodos permite a su vez entender que tal vez el retardo adicional no fuese únicamente de un slotframe, sino de más. Este efecto, para el peor casos acumulaba un 5% de probabilidad.

A su vez, la figura 51 permite observar con detalle la franja de retardo entre 285 ms y 360 ms. Este zoom expone otra situación no contemplada previamente: existen paquetes que llegan con un retardo menor al considerado retardo mínimo de 315 ms, y que acumulan hasta un 7% de probabilidad. Esta diferencia surge porque previamente se consideró que el retardo sólo podía variar en función del tiempo de arribo del paquete. Es decir, según cual de las transmisiones hacia la raíz del DODAG fuese exitosa. Éstas son las últimas transmisiones que se hayan presente en el camino directo del scheduling. Sin embargo, existe otra fuente de variación: el nodo hoja no necesariamente envía el paquete por primera vez siempre en el mismo timeslot, y por ende el tiempo de partida también se vuelve variable.

El hecho de que el nodo hoja no envíe el paquete siempre en el mismo timeslot es consecuencia de que la información de RPL y de los LeapFrog Beacons puede estar desfasada hasta por 30 segundos. Ello sucede dado que estos paquetes de control introducidos con esta propuesta de tesis son enviados justamente cada esa cantidad de tiempo. Por ende, eso posibilita a que un nodo pueda adoptar momentáneamente un padre alternativo igual al padre por default. Más en detalle, esto se produce cuando el protocolo RPL decide cambiar de padre por default, eligiendo al que ya era el padre alternativo. El funcionamiento de RPL es independiente de LeapFrog Collaboration e ignora la existencia de un padre alternativo. Por ello, RPL no acciona ninguna alarma adicional advirtiendo esta situación, la cual no se solucionará sino hasta el arribo de un LeapFrog Beacon.

El resultado de este defecto, particularizándolo al nodo 8, es que el mismo a veces envía el paquete sólo al nodo 7 o sólo al nodo 6. Así, para el scheduling diseñado (figura 40), el tiempo de partida del paquete será el timeslot 0 o 2 según el caso. Esto, recordando que los timeslots tienen una duración de 15 ms cada uno, explica de manera concreta por qué el retardo mínimo observado fue exactamente 30 ms menor al esperado.

Por otro lado, en la figura 51 resulta notorio como todos los efectos se ven más marcados a medida que la probabilidad de éxito de los enlaces decrece. En primer lugar, aumenta la cantidad de paquetes que llegan destino a través del nodo 2 (345 ms a 360 ms) y no a través del nodo 3 (315 ms a 330 ms). Además cada vez son más los paquetes que llegan con un retardo de 330 ms o 360 ms, lo cual da cuenta del aumento del uso de las retransmisiones. En segundo lugar, son también mayores las cantidades de paquetes que llegan con un retardo menor y mayor a los calculados teóricamente. En este caso, la explicación es que, al haber peores enlaces, la red es más inestable. Esto trae aparejado el uso de más paquetes de control de RPL y una actualización más seguida de los padres por default, creando respectivamente más oportunidades a que un paquete se vea retrasado a otro slotframe y a que la tasa de envío de paquetes LeapFrog Beacons no sea suficiente.

Las figuras 52 a 57 dan cuenta de cómo varía el retardo considerando los distintos escenarios que se suceden dentro de una misma simulación. Cada gráfica contempla todas las simulaciones realizadas para cada caso de calidad de enlace. En ellas se muestran el valor de la mediana, los desvíos y los outliers del retardo al cual un paquete se ve sometido al atravesar la red de extremo a extremo.

En el caso de enlaces con probabilidad de éxito del 100%, ningún paquete llega con la necesidad de utilizar una retransmisión y entonces los valores de 300, 330 y 360 ms se hallan vacíos. El valor de la mediana del retardo se halla en 315 ms, momento en el cual el nodo 3 transmite al nodo 1 por primera vez. Sin embargo, es interesante notar cómo en los escenarios en que dicho nodo falla, entonces el valor de la mediana del retardo asciende a 345 ms, que se condice con la primera transmisión del nodo 2 hacia el 1. Además, sólo en estos dos escenarios la banda del desvío resulta visible, mientras que en los restantes escenarios se funden con el valor de la mediana. De aquí en adelante, al bajar progresivamente la calidad de los enlaces, los paquetes comienzan a llegar con el uso de retransmisiones y aumentan los outliers. Además, las bandas de desvío crecen y el valor de la mediana varía más de escenario a escenario. Todos estos efectos se ven expuestos en mayor cantidad en el caso de la figura 57, cuyos enlaces poseen una probabilidad de éxito de transmisión al 50%.

En conclusión, la mayor cantidad de paquetes arriba a destino con un retardo de 315 ms, que es el retardo mínimo teórico. Si bien entre este valor y 360 ms, el retardo máximo teórico, se acumula la mayor probabilidad para el valor del retardo, existen paquetes que llegan con menor o mayor retardo respectivamente, debido a inestabilidades en la red. Esto resulta en un jitter mayor y un retardo más difícil de predecir.

10.2.3 Ciclo de trabajo

En cuanto al ciclo de trabajo, únicamente se consideró el caso de enlaces al 100% puesto que en dichos casos las simulaciones de LeapFrog Collaboration funcionaron siempre hasta el final presentando resultados acorde a lo esperado. La tabla 6 demuestra que efectivamente el consumo de energía al utilizar un padre alternativo y overhearing resulta mayor comparado con el uso de RPL+TSCH. Este resultado era esperable puesto que, justamente, LeapFrog Collaboration explota la diversidad a través del uso de caminos múltiples, lo cual requiere un mayor número de transmisiones.

10.2.4 Otros Resultados

Las figuras 58 a 64 muestran cómo los distintos nodos cambian sus padres por default y alternativo a lo largo de una simulación al utilizar la propuesta de LeapFrog Collaboration. Estos resultados son los obtenidos para una única simulación con enlaces al 100%, y por ende deben considerarse únicamente una referencia para comprobar el correcto funcionamiento de la lógica implementada en LeapFrog Collaboration.

Las curvas rojas representan el padre por default establecido por RPL. El funcionamiento de RPL es independiente de LeapFrog Collaboration y en cualquier momento puede cambiar de padre preferido. Por otro lado, las curvas verdes y violetas hacen referencia al padre por default LeapFrog y al padre alternativo LeapFrog, respectivamente. Esta es la información que viaja en los LeapFrog Beacons y a través de la cual los distintos nodos pueden ir actualizando y recalculando las distintas características de LeapFrog Collaboration. Lo importante es notar que los cambios de RPL en general producen cambios en la lógica de LeapFrog Collaboration. Estas actualizaciones se producen

cuando un LeapFrog Beacon es recibido, y por ende con hasta 30 segundos de retardo. Ello explica por qué los círculos verdes se hayan tan cercanos a los cuadrados rojos. Por otro lado, en general el padre alternativo asume el viejo valor del padre por default, mientras que el padre por default pasa a ser el viejo padre alternativo.

Como se observa en las figuras 58 y 59, los nodos 2 y 3 asumen como padre por default al nodo 1, y jamás definen un padre alternativo, lo cual se representa por el valor 0 asumido por dicho parámetro. Éste último es la raíz del DODAG, y por ende no tiene padre por default ni padre alternativo, motivo por el cual no se incluye una gráfica para él. Entre los nodos 4 y 7, los cambios sólo se producen cuando el padre por default de RPL es el nodo que falla, lo cual se observa entre las figuras 60 y 63. Por otro lado, el nodo 8 es el único nodo que presentó algunos cambios en algunos instantes de tiempo no predecibles, tanto al inicio de la simulación como hacia el final. De todos modos, los mismos no causaron mayores problemas a la hora de ser interpretados por la lógica de LeapFrog Collaboration.

Por último, cabe destacar que se escogió el caso de enlaces al 100% ya que en estos casos, la red es más estable y las curvas resultan legibles. Sin embargo, como explicado con anterioridad, al disminuir la calidad de los enlaces, la red se vuelve más inestable y los cambios se producen más frecuentemente, dificultando el entendimiento. Una forma de cuantificar dicha inestabilidad es observar la cantidad de veces totales que los nodos cambian de padre por default. La figura 65 muestra cómo evoluciona dicho parámetro al disminuir la probabilidad de éxito de transmisión. Comparado con RPL+TSCH con retransmisiones, LeapFrog Collaboration presenta la menor cantidad de cambios de padres para el caso de enlaces al 100%. Sin embargo, esta tendencia se revierte luego al disminuir la calidad de los enlaces: si bien inclusive para RPL+TSCH existe un aumento de cambios de padres, esto se ve ampliamente exacerbado para LeapFrog Collaboration, que presenta más del doble de cambios. Este comportamiento introduce más paquetes de control de RPL y posibilita más a que el retardo sufra los efectos negativos explicados en detalle anteriormente.

El motivo de que LeapFrog Collaboration produzca mayores cambios de

padres puede entenderse por el hecho de que los nodos no sólo transmiten a su padre por default, sino que lo hacen a su padre alternativo también. Esto permite tener a los nodos estadísticas que se renuevan constantemente en cuanto a la probabilidad de éxito de una transmisión, y por ende un nodo posee mejores herramientas para determinar cual es el mejor camino que un paquete podría seguir. Es decir, la métrica ETX utilizada por la Función Objetivo adoptada para RPL cambia más seguido, y eso hace que el Rango con que un nodo ve sus posibles padres varíe también. Esto, en última instancia, termina por generar cambios seguidos en cuanto al padre adoptado.

En resumidas cuentas, los resultados presentados en esta sección permiten asegurar que la lógica implementada para determinar los distintos padres de LeapFrog Collaboration funciona según lo esperado. Los cambios en LeapFrog Collaboration se producen con un retardo de hasta 30 segundos con respecto a los cambios de RPL. Por otro lado, al disminuir la calidad de los enlaces, la cantidad de cambios de padres aumenta hasta 6 veces debido a las estadísticas más cambiantes en cuanto a la performance de los enlaces. Ello se traduce en una red más inestable, con más paquetes de control y un retardo más difícil de acotar.

11 Conclusiones

LeapFrog Collaboration es un mecanismo de ruteo de paquetes que se basa en RPL, pero que amplía su funcionamiento al utilizar un padre alternativo. Ello permite explotar la diversidad de caminos y aumenta las posibilidades de que un paquete llegue a destino. Además, los nodos sobre-escuchan otras comunicaciones y ello nuevamente implica más chances de que la red sea robusta en términos de entrega de paquetes. Más aún, también incorpora la posibilidad de realizar una retransmisión en caso de que un nodo transmisor no reciba un ACK por parte del nodo receptor.

La PDR obtenida en las simulaciones supera a la alcanzada por RPL+TSCH. Sin embargo, LeapFrog Collaboration también requiere que los nodos realicen más transmisiones o que reciban una mayor cantidad de paquetes. Esto se traduce en un mayor consumo de energía.

Por otro lado, LeapFrog Collaboration también propone minimizar el retardo de extremo a extremo y sus variaciones. De manera de llevar esto a cabo, propone un cronograma de transmisiones en el cual los nodos van transmitiendo por capas, desde la más alejada hacia la más cercana a la raíz del DODAG. La ventaja de esta propuesta es que, entonces, los paquetes llegan a destino en un único slotframe, permitiendo acotar el retardo mínimo y máximo respectivamente de manera precisa.

Los resultados de las simulaciones evidencian que LeapFrog Collaboration mejora la estabilidad del delay al compararlo con los resultados de RPL+TSCH.

En conlusión, comparado con las soluciones implementadas hoy en día, LeapFrog Collaboration mejora simultáneamente la PDR y la estabilidad del delay. Ello lo vuelve una opción más adecuada a la hora de implementar soluciones que pretendan obtener una red determinística.

Sin embargo, si bien los resultados obtenidos fueron prometedores y dan cuenta del potencial de LeapFrog Collaboration, aún así existen ciertos comportamientos que no se ajustaron totalmente a lo esperado: i) la PDR sólo fue

buena considerando una parcialidad de las simulaciones y ii) el delay mínimo y máximo no cumplieron con las cotas teóricas establecidas con anterioridad. Esto permite pensar en mejoras que pueden ser logradas en cuanto al funcionamiento de la imprmentación de LeapFrog Colaboration y forman parte del trabajo a futuro propuesto en la sección siguiente.

12 Trabajo a Futuro

La discrepancia entre los resultados teóricos y los obtenidos para LeapFrog Collaboration puede verse reducida llevando adelante una pequeña cantidad de mejoras en la implementación en COOJA de dicho mecanismo. Las mismas se detallan a continuación.

- **Mejorar la PDR.** En primer lugar, resulta necesario encontrar el motivo que produce que el nodo que envía información, deje de hacerlo. Una vez corregido ello, el total de las simulaciones mostrarán resultados con una PDR que se ajusta a lo esperado, y podrán obtenerse más conclusiones en cuanto al ciclo de trabajo de la red.

- **Mejorar la estabilidad del retardo.** De manera de obtener un jitter más cercano a cero, debe mejorarse asegurar la cota máxima y mínima del retardo de manera más certera. Para ello, se deben llevar adelante dos acciones.

    1. Para mitigar completamente la posibilidad de que algunos paquetes lleguen con un retardo superior al retardo máximo de 360 ms, se debe generar un control que otorgue prioridad a los paquetes de información antes que a los de control. Así, los paquetes de información serán siempre la primera opción de paquetes a enviar, y por ende el retardo no ascenderá nunca del máximo teórico.
    2. En el afán de asegurar un retardo mínimo en 315 ms, cada vez que hay un cambio en cuanto al padre determinado por RPL, dicho evento debe activar una actualización de la información de RPL en el nodo y disparar el envío asincrónico de un LeapFrog Beacon para poner al tanto a los restantes nodos del cambio. Esto eliminaría el desfasaje entre la información de RPL y de LeapFrog Collaboration que en muchos casos termina por generar que algunos paquetes lleguen con retardo inferior al teóricamente calculado. A su vez, se puede contemplar la posibilidad de directamente incluir la información de los LeapFrog Beacons en los mensajes DIO.

- **Mejorar la estabilidad de la red.** Al disminuir la calidad de los enlaces, los nodos cambian mucho más seguido de padre por default.

De manera de volver la red más estable y minimizar este efecto, puede considerarse aumentar el nivel de histéresis. De esta manera, si bien se continúa teniendo una estadística más cambiante, sólo cuando los cambios sean significativos RPL adoptará un nuevo padre por default. Esto disminuye los paquetes de control necesitados en la red y la posibilidad de que un nodo tenga un padre alternativo igual al padre por default.

13 Apéndice: Cálculo de la PDR teórica

En cuanto al valor exacto de la PDR esperada, si bien en la sección del cálculo matemático se hallaron algunas expresiones, las mismas consideraban que las distintas p_{ij}, la probabilidad de error de enlace para el nodo i transmitiendo al padre j, eran constantes. Del mismo modo, las distintas $1 - p_{ij}$, la probabilidad de éxito de enlace para el nodo i transmitiendo al padre j, también lo eran. Sin embargo, en las simulaciones consideradas, en los escenarios "def" todos los nodos poseían enlaces de cierta calidad p, que eventualmente podía pasar a tener un valor de p_{bad} si el nodo i o j entraba en modo de fallas, es decir en un escenario "id:i" o "id:j".

13.1 Análisis Matemático

Una manera de incluir este efecto sobre los resultados es definir las p_{ij} emulando la calidad media de los enlaces en las simulaciones pesándolas según la cantidad de tiempo que cada enlace en particular se mantiene funcionando correctamente y en modo de fallas. Para ello, resulta necesario además considerar que:

- En los enlaces internos de la red, cada enlace particular $i - j$ pasa a tener una calidad p_{bad} ya sea cuando el nodo i o j entra en modo de fallas. Por ende, cada uno de estos enlaces falla durante un tiempo $T_{int,bad}$ donde:

$$T_{int,bad} = \sum_{n=i,j} T_n \ , \tag{11}$$

  donde T_n es el tiempo total que falla el nodo n a lo largo de la simulación, que se calcula como:

$$T_n = CF_n \cdot T_{id:n} \tag{12}$$

  donde $T_{id:n}$ se asume constante y representa el tiempo durante el cual el nodo n falla cada vez que falla, y CF_n es la cantidad de veces que dicho nodo n falla.

  De manera de simplificar el análisis, asumimos que el tiempo $T_{id:n}$ es constante y el mismo a todos los nodos, de valor T_{id}. A su vez, la

cantidad de fallas CF_n también se considera equivalente para todos los nodo, de valor CF. Así resulta:

$$T_{int,bad} = 2 \cdot CF \cdot T_{id} \ . \tag{13}$$

- En los enlaces en los extremos de la red, dado que en las simulaciones se considera que el nodo que genera el tráfico y la raíz de la DODAG nunca fallan, cada enlace falla sólo una vez. Por ende, asumiendo las mismas simplificaciones que antes, el tiempo de falla de estos enlaces es $T_{bor,bad}$ de valor:

$$T_{bor,bad} = 2 \cdot CF \cdot T_{id} \ . \tag{14}$$

De todos modos, lo que termina importando es la proporción de tiempo que un enlace posee enlaces buenos o malos respecto del tiempo total. Asumiendo que la duración de los escenarios "def" es igual a la de los escenarios "id:x" para cualquier nodo x, cuyo valor era T_{id}, y que hay tantos escenarios escenarios de unos como de otros, se tiene un tiempo total de simulación

$$T_{sim} = 2 \cdot CF \cdot CN \cdot T_{id} \ , \tag{15}$$

donde CN es la cantidad de nodos que fallan en la red. En definitiva, se tienen las tasas $r_{i,bad}$ y $r_{b,bad}$:

$$r_{i,bad} = CN^{-1} \ , \tag{16}$$

$$r_{b,bad} = (2 \cdot CN)^{-1} \ . \tag{17}$$

Por otro lado, todos los enlaces poseen la misma calidad p mientras los nodos que los unen no fallen y se reducen a un valor p_{bad} cuando cualquiera de ellos lo hace. La probabilidad de que una transmisión sea exitosa o que fracase se puede calcular respectivamente según:

$$p_{x,ok} = p \cdot (1 - r_{x,bad}) + p_{bad} \cdot r_{x,bad} \ , \tag{18}$$

$$p_{x,e} = 1 - p_{x,ok} \ , \tag{19}$$

donde x se ajusta a i o b según se trate del caso de enlaces internos o de los bordes de la red.

Además, analizando particularmente la red propuesta, y teniendo en cuenta el funcionamiento de LeapFrog Collaboration, en la tabla 7 se resumen los valores de las variables necesarias para calcular la PDR.

Tabla 7: Variables necesarias para el cálculo de la PDR.

Parámetros	**Value**
n (# padres)	2
m (# transmisiones)	2
R (Máximo valor del Rango)	4
CN	6
p	De 100% a 50%
p_{bad}	10%
$r_{i,bad}$	1/6
$r_{b,bad}$	1/12

Finalmente, el valor de p_{ij} en las ecuaciones (5) y (7) adoptará el valor $p_{b,e}$, mientras que en la ecuación (6) será $p_{i,e}$. Ambos se obtienen utilizando los datos de la tabla 7 para evaluar la ecuación (19).

Reuniendo toda esta información, se tiene:

$$
\begin{aligned}
q_3 &= p_{b,e}^4 \\
q_2 &= \left(q_3 + (1 - q_3) \cdot p_{i,e}^4\right)^2 \\
q_1 &= \left(q_2 + (1 - q_2) \cdot p_{i,e}^4\right)^2 \\
q_0 &= \left(q_1^{(i)} + (1 - q_1^{(i)}) \cdot p_{b,e}^2\right)^2
\end{aligned}
$$

para finalmente obtener PDR $= 1 - q_0$. La PDR teórica para los distintos valores de p simulados se observa en la tabla 8.

Tabla 8: PDR teórica según el valor de la calidad de los enlaces.

p	**PDR**
100%	0.99997
90%	0.99923
80%	0.99553
70%	0.98483
60%	0.96072
50%	0.91198

Los valores de la tabla 8 fueron obtenidos utilizando un script en Octave, el cual se incluye a continuación a modo de referencia.

13.2 Script para evaluar

A partir del siguiente código, resulta posible obtener los resultados de la PDR esperada para LeapFrog Collaboration en el caso particular de la tabla 7.

```
% Parámetros Necesarios
p = [1, 0.9, 0.8, 0.7, 0.6, 0.5];
p_bad = 0.1;
CN = 6;

% Caso bordes internos
rb = 1/CN;
rg = 1 - rb;

p_i_e = 1 - (rg .* p + rb * p_bad);

% Caso bordes externos
rb = 1/(2*CN);
rg = 1 - rb;

p_b_e = 1 - (rg .* p + rb * p_bad);

% Cálculo de la PDR teórica
q3 = p_b_e.^4;
q2 = (q4 + (1-q4).*p_i_e.^4).^2;
q1 = (q3 + (1-q3).*p_i_e.^4).^2;
q0 = (q2 + (1-q2).*p_b_e.^2).^2;

PDR = (1-q0)*100
```

14 Apéndice: Cronograma de actividades

La tesis se desarrollará en la prestigiosa escuela francesa de ingeniería IMT Atlantique (ex Télécom Bretagne). Allí, el tesista se encuentra actualmente realizando una pasantía con fines académicos de 7hs diarias de trabajo en el laboratorio de investigación de redes que dicha casa de estudios posee en Rennes, Francia.

Los tutores en Francia son el profesor adjunto Georgios Z. Papadopoulos y el profesor Nicolás Montavont, ambos ya habituados a recibir y trabajar con estudiantes de la Facultad de Ingeniería de la Universidad de Buenos Aires. Además, la tesis estará tutoreada en Argentina por el Ing. Alberto Dams, quien también cuenta con una basta experiencia en la coordinación con estudiantes realizando sus estudios en el exterior.

El cronograma de actividades se puede resumir en:

- **Acuerdo de intercambio.** Gestión para la obtención de la posición en el laboratorio por parte del tesista: entrega de certificado analítico, CV y posteriores entrevistas. Finalmente, firma del contrato de pasantía por parte del tesista, el secretario de intercambios académicos Ing. Marcelo Utard, y algún miembro de la Comisión Curricular del Departamento de Electrónica.

- **Estudio previo.** Dado que se trata de un tema nuevo para el tesista, el mismo deberá comenzar de cero, leyendo toda la bibliografía que se pueda para interiorizarse en el área. En particular, deberá tenerse una idea de redes tipo 6TiSCH y su stack de protocolos. Resultan de mayor interés RPL en la capa de ruteo, 6P como capa de adaptación y principalmente TSCH en la capa de enlace. La bibliografía recomendada para ello se encuentra en las referencias y abarca desde [1] hasta [11].

- **Integración al grupo de trabajo.** Arribo a Francia. Reconocimiento de las instalaciones de la universidad y el campus de la misma. Entender la dinámica de trabajo de los tutores.

- **Estudio del Estado del Arte.** Repaso de temas leídos con anterioridad como introducción al área. Compresión de conceptos nuevos:

Industria 4.0 y Redes determinísticas. Estudio de papers que permitan tener un entendimiento más acabado de todo lo estudiado y a su vez que propongan soluciones para problemáticas parecidas a la atacada en esta tesis de estudios. Algunos de los papers leídos se detallan en la sección de referencias y van desde [12] hasta [20].

- **Herramientas de trabajo.** Instalación del simulador de redes Cooja que viene incluido en Contiki OS. Agregado de la exensión RealSim para la simulación más realista de calidades de enlace. Instalación del compilador MSP430. Estudio de ejemplos básicos, corriendo el software con y sin consola. Primer acercamiento a sistemas Ubuntu y programación en Python y Javascript. Repaso de lenguaje C.

- **Simulaciones.** Realizar cambios en los códigos fuente de Cooja de manera de implementar los mecanismos propuestos para mejorar el rendimiento de las redes y con el objetivo de poder lograr una buena PDR, un retardo mínimo y que varíe poco.

- **Optimizaciones.** Propuesta de mejoras y su consecuente adopción en función de lo que se pueda corroborar en los resultados que se observen.

- **Redacción de paper.** Expresar la problemática atacada y la solución final alcanzada en un artículo de divulgación científica.

- **Redacción de informe.** Ensayo de tesis.

- **Exposición de tesis.** Presentación en Argentina del trabajo realizado.

15 Apéndice: Artículo de divulgación científica

Como resultado del trabajo realizado en esta tesis, se está trabajando en un paper titulado *"From Best-effort Solutions to Deterministic Wireless Industrial Networks"* a publicar a nivel internacional en el jornal IEEE Transaction on Industrial Informatics.

16 Glosario

- A
  - **ACK.** Acuse de recibo.
  - **AP.** Padre alternativo.
  - **ASN.** Número de ranura absoluto.
- B
  - **Bd.** Difusión Ancha.
- C
  - **Channel Hopping.** Salto de canales.
  - **CoAP.** Protocolo de Aplicación Restringido.
- D
  - **DAG.** Grafo Acíclico Dirigido.
  - **DAO.** Objeto de Anuncio de Destino.
  - **DAO-ACK.** ACK de mensaje DAO.
  - **Delay.** Retardo.
  - **DIO.** Objeto de Información DODAG.
  - **DIS.** Solicitación de Información DODAG.
  - **DODAG.** DAG Orientado hacia un Destino.
  - **DP.** Padre por default (según RPL).
- E
  - **EB.** Mensaje de tipo Beacon mejorado.
  - **ETX.** Número promedio de transmisiones.
- I
  - **ICMP.** Protocolo de Mensajes de Control de Internet.
  - **IEEE.** Instituto de Ingeniería Eléctrica y Electrónica.
  - **IETF.** Grupo de Trabajo de Ingeniería de Internet.

  - **IoT.** Internet de las cosas.
  - **IPv6.** Protocolo de Internet Versión 6 (direcciones de 128 bits).

- J
  - **Jitter.** Variabilidad, fluctuación (del retardo).

- L
  - **LAP.** Campo del LFB que indica el AP de un nodo.
  - **LBR.** Router de borde de baja potencia y con pérdidas.
  - **LDP.** Campo del LFB que indica el DP de un nodo.
  - **LFB.** Mensaje de tipo Beacon de LeapFrog Collaboration.
  - **LGP.** Campo del LFB que indica el Abuelo de un nodo.
  - **LL.** Campo del LFB que indica la capa a la que un nodo pertenece.
  - **LLN.** Red de baja potencia y con pérdidas.

- M
  - **MAC.** Control de acceso al medio.
  - **MP2P.** Multipunto a punto.

- O
  - **Overhearing.** Sobre-escucha.

- P
  - **PANA.** Protocolo de autenticación para el acceso a la red.
  - **PDR.** Tasa de entrega de paquetes.
  - **PER.** Tasa de error de paquetes.
  - **PLC.** Comunicaciones mediante línea de potencia.
  - **PPS.** Padres Posibles.
  - **P2MP.** Punto a Multipunto.
  - **P2P.** Punto a punto.

- R
  - **RFC.** Pedido de comentarios: publicación de la IETF.
  - **Router.** Dispositivo de enrutamiento.
  - **RPL.** Protocolo de ruteo IPv6 para LLNs.
- S
  - **Scheduling, Schedule.** Cronograma, agenda.
  - **Siblings.** Hermanos.
  - **Slotframe.** Marco de Ranuras (de tiempo).
- T
  - **TCP.** Protocolo de Control de Transmisión.
  - **Timeslot.** Ranura de tiempo.
  - **TSCH.** Tiempo dividido en ranuras con salto de canales.
- U
  - **UDP.** Protocolo de Datagrama de Usuario.
- W
  - **WPAN.** Red de área personal inalámbrica.
  - **WSN.** Red inalámbrica de sensores.
- Símbolo
  - **6LoWPAN.** IPv6 sobre WPANs de bajo consumo.
  - **6TiSCH.** IPv6 sobre el modo TSCH de IEEE 802.15.4e.

17 Bibliografía

[1] P. Thubert. "An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4 draft-ietf-6tisch-architecture-10". *IETF, Work in Progress*, 10 June 2016.

[2] JP. Vasseur. "Terms Used in Routing for Low-Power and Lossy Networks". *IETF, RFC 7102*, Enero 2014.

[3] Olfa Gaddour and Anis Koubâa. "RPL in a nutshell: A survey". *J. Comput. Net., Volume 56, Issue 14*, pages 3163–3178, 28 September 2012.

[4] Aishwarya Parasuram. "An Analysis of the RPL Routing Standard for Low Power and Lossy Networks". *Electrical Engineering and Computer Sciences University of California at Berkeley*, 14 May 2016.

[5] JP. Vasseur and al. "RPL: The IP routing protocol designed for low power and lossy networks". *Internet Protocol for Smart Objects (IPSO) Alliance*, Abril 2011.

[6] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander. " RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks". *IETF, RFC 6550*, Marzo 2012.

[7] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko. "The Trickle Algorithm". *IETF, RFC 6206*, Marzo 2011.

[8] JP. Vasseur, M. Kim, K. Pister, N. Dejean, and D. Barthel. "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks". *IETF, RFC 6551*, Marzo 2012.

[9] P. Thubert. "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)". *IETF, RFC 6552*, Marzo 2012.

[10] O. Gnawali and P. Levis. "The Minimum Rank with Hysteresis Objective Function". *IETF, RFC 6719*, Septiembre 2012.

[11] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. "A High-Throughput Path Metric for Multi-Hop Wireless Routing".

Proceedings of the 9th annual international conference on Mobile computing and networking, pages 134–146, Septiembre 2003.

[12] Quan Wang and Jin Jiang. "Comparative Examination on Architecture and Protocol of Industrial Wireless Sensor Networks Standards". *IEEE Communications Surveys and Tutorials, Vol. 18, No. 3*, pages 2197–2219, 2016.

[13] T. Watteyne, M. Palattella, and L. Grieco. "Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT): Problem Statement". *IETF, RFC 7554*, 2015.

[14] Qin Wang, Thomas Watteyne, Pascal Thubert, and Maria Rita Palattella. "Terminology in IPv6 over the TSCH mode of IEEE 802.15.4e draft-ietf-6tisch-terminology-08". *IETF, Work in Progress*, 16 December 2016.

[15] Mukhopadhyay and Subhas Chandra. *Internet of Things*. 2014.

[16] J. A. Gutierrez, E. H. Callaway, and R. Barett. "IEEE 802.15.4 low-rate wireless personal area networks: Enabling wireless sensor networks". *IEEE Standards Office*, 2003.

[17] A. Watteyne, T.and Mehta and K. Pister. "Reliability Through Frequency Diversity: Why Channel Hopping Makes Sense". *Proceedings of the 6th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN)*, pages 116–123, October 2009.

[18] D. Dujovne and M. Richardson. "IEEE802.15.4 Informational Element encapsulation of 6tisch Join Information draft-richardson-6tisch-join-enhanced-beacon-02". *IETF, Work in Progress*, 18 July 2017.

[19] X. Vilajosana, K. Pister, and T. Watteyne. "Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration". *IETF, RFC 8180*, May 2017.

[20] D. Dujovne and MR. Palattella. "6TiSCH 6top Scheduling Function Zero (SF0) draft-ietf-6tisch-6top-sf0-05". *IETF, Work in Progress*, 2 July 2017.

[21] S. Duquennoy, X. Vilajosana, and T. Watteyne. "6TiSCH Autonomous Scheduling Function (ASF) draft-duquennoy-6tisch-asf-00". *IETF, Work in Progress*, 3 July 2017.