Universidad Zaragoza
1542

Trabajo Fin de Máster

Desarrollo de un entorno experimental WiFi para el diseño y evaluación de algoritmos de control de tasa

Development of a WiFi experimental environment for the design and evaluation of rate control algorithms

Autor

David del Cos Azón

Director

José Ruiz Mas

Máster Universitario en Ingeniería de Telecomunicación

Escuela de Ingeniería y Arquitectura

2023

Resumen

Nuestra sociedad avanza hacia un mundo inalámbrico en el que cada vez más dispositivos portátiles incorporan al menos una tecnología inalámbrica de acceso radio. Estas tecnologías buscan proporcionar mayores áreas de cobertura y una conectividad más rápida en su afán de implantar soluciones innovadoras y mejorar la accesibilidad y calidad de los servicios. En este contexto, Wi-Fi es la tecnología preferida para escenarios de interior, donde es más fácilmente accesible y presenta ventajas para los despliegues de redes locales privadas en términos de modelo de negocio.

Este Trabajo Fin de Máster se centra en la incorporación de la función de control de tasa de transmisión en entornos de redes definidas por software inalámbricas SDWN (*Software Defined Wireless Networks*) ya desarrollados. La incorporación de esta funcionalidad es imprescindible para desarrollar nuevos algoritmos de control y coordinación que puedan utilizar las características del tráfico de red y toda la información obtenida mediante la monitorización del entorno para mejorar su eficiencia. A través de este mayor nivel de inteligencia, se pretende cubrir las demandas de ancho de banda y proporcionar una mayor calidad de servicio (*Quality of Service*). Esta línea de trabajo es la que viene desarrollando el grupo CeNIT (*Communications Networks and Information Technologies*).

En la realización del trabajo se han implementado y probado los algoritmos de control de tasa ONOE, ARF (*Auto Rate Fallback*) y AARF (*Adaptative Auto Rate Fallback*). Se han realizado pruebas de los algoritmos en un entorno experimental con diferentes condiciones (en entorno estático, en movimiento, en presencia de interferencias, con varios usuarios y con variaciones del entorno). De esta manera se ha comprobado el efecto de las diferentes condiciones del entorno en el funcionamiento de los algoritmos de control de tasa y se han analizado cuales son los parámetros a tener en cuenta para mejorar su rendimiento.

Contenido

1. Introducción ... 10
1.1. Contexto del trabajo ... 10
1.2. Objetivos ... 11
1.3. Estructura de la memoria ... 11
2. Estado del arte ... 13
2.1. IEEE 802.11 ... 13
2.2. Estándares Wi-FI: evolución y características clave ... 15
2.2.1. Evolución estándares ... 15
2.2.2. Características clave de la tecnología Wi-Fi ... 17
2.3. Algoritmos de control de tasa ... 22
2.3.1. Algoritmos basados en tramas perdidas y/o throughput ... 23
2.3.2. Algoritmos basados en SNR y RSSI ... 27
2.3.3. Basados en la pérdida de tramas con diferenciación de colisiones ... 28
2.3.4. Algoritmos basados en redes neuronales: *Deep Reinforcement Learning* ... 30
2.3.5. Conclusiones ... 30
3. Diseño del entorno experimental ... 32
3.1. Aspectos técnicos previos a considerar ... 32
3.1.1. Inyección de tramas ... 32
3.1.2. Drivers, tarjetas Wi-Fi y Linux ... 33
3.1.3. *Radiotap* y *Scapy* ... 35
3.1.4. Obtención de métricas y/o estadísticas ... 35
3.2. Definición del entorno experimental ... 37
3.3. Diseño y estructura del programa ... 39
3.4. Transmisión de tramas: inyección de paquetes ... 40
3.5. Monitorización de paquetes ... 41
3.6. Implementación de los algoritmos ... 43
4. Pruebas experimentales ... 44
4.1. Pruebas estáticas ... 45
4.2. Pruebas en movimiento ... 52
4.3. Pruebas con interferencias por terminal oculto ... 61
4.4. Pruebas con varias estaciones ... 68
4.5. Pruebas con variaciones del entorno (paso de personas) ... 69
5. Conclusiones ... 71
5.1. Trabajo Futuro ... 73
5.2. Problemas encontrados ... 73
ANEXO I: Tabla de índices MCS para 802.11n/ac ... 76

Índice de figuras

Figura 1 Transmisión de una trama de datos con DCF 13
Figura 2 Problema del terminal oculto 14
Figura 3 Entrega de un paquete de datos con DCF utilizando tramas RTS/CTS para solucionar el terminal oculto 14
Figura 4 Problema del terminal expuesto 14
Figura 5 Evolución de la tecnología y estándares Wi-Fi 16
Figura 6 Canales de la banda ISM de 2.4GHz para 802.11 b/g. (El canal 14 solo aparece en Japón). . 17
Figura 7 Banda de 5 GHz en las diferentes regiones (UE, EEUU, India y China) y canalizaciones para Europa [1] 17
Figura 8 Channel bonding en 802.11n [1] 18
Figura 9 Formatos de trama Non-HT y mixto (Non-HT y HT) para 802.11n 19
Figura 10 Tecnología SU-MIMO (Single User) y MU-MIMO (Multi User) en 802.11ac [1] 20
Figura 11 Trama MAC 802.11n 21
Figura 12 Algoritmo ONOE 25
Figura 13 Modificación del envío de RTS/CTS añadiendo RSH (Reservation SubHeader) 28
Figura 14 Algoritmo CARA 29
Figura 15 Aprendizaje por refuerzo (Deep Reinforcement Learning) 30
Figura 16 Estructura de los drivers Full-MAC y Soft-MAC [7] 34
Figura 17 Estadísticas mediante la estructura rtnl_link_stats64 36
Figura 18 Estadísticas específicas del protocolo con ethtool 36
Figura 19 Estadísticas definidas por el driver con ethtool 36
Figura 20 Escenario básico: PC Linux, punto de acceso (AP), espía y estación Wi-Fi (STA) 38
Figura 21 Estructura del programa 39
Figura 22 Escenario para las medidas en estático: visión del laboratorio 2.05 para medir el punto 1 (izquierda) y mapa del escenario completo con los puntos de 1 a 8 (derecha) 45
Figura 23 Varias realizaciones de la eficiencia en los 8 puntos del escenario para ONOE original (a), con ventana de 200 ms (b), con umbral de créditos de 3 (c) y con umbral de reintentos del 30% (d). Transmisión de 100 paq/s y longitud L= 1000 Bytes desde el AP a la STA. 46
Figura 24 Histograma de MCS para los diferentes puntos de medida. ONOE original(a), con ventana de 200 ms(b), con umbral de créditos 3(c) y con umbral de reintentos del 30%(d). Transmisión de 100 paq/s y longitud de 1000 Bytes desde el AP a la STA 47
Figura 25 Varias realizaciones de la eficiencia para ARF y AARF. Transmisión de 100 paq/s con una longitud L=1000Bytes desde el AP a la STA. 48
Figura 26 Histograma de MCS para los diferentes puntos de medida. ARF(a), AARF(b). Transmisión de 100 paq/s y longitud de 1000 Bytes desde el AP a la STA 49
Figura 27 Tasa promedio de transmisión para ONOE y sus variantes (a) y ARF/AARF (b). Transmisión de 100 paq/s y longitud L=1000 Bytes desde el AP a la STA. 50
Figura 28 Escenario para las medidas en movimiento: vista del carro con la STA móvil (izquierda) y plano de la prueba (derecha). 52
Figura 29 Algoritmo ONOE original (W=1s, C=10, R=50%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 53
Figura 30 Algoritmo ONOE ventana 200ms (W=200ms, C=10, R=50%) : Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 53
Figura 31 Algoritmo ONOE umbral crédito 3 (W=1s, C=3, R=50%) : Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3

tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 54
Figura 32 Algoritmo ONOE reintentos 30% (W=1s, C=10, R=30%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 54
Figura 33 Algoritmo ONOE ventana y créditos (W=200ms, C=3, R=50%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 55
Figura 34 Algoritmo ONOE ventana y reintentos (W=200ms, C=10, R=30%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 56
Figura 35 Algoritmo ONOE créditos y reintentos (W=1s, C=3, R=30%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 56
Figura 36 Algoritmo ONOE var. 3 parámetros (W=200ms, C=3, R=30%) : Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 57
Figura 37 Algoritmo ARF: Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 57
Figura 38 Algoritmo AARF: Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes. 58
Figura 39 Algoritmo ARF: Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 20 paq/s y longitud de L = 1000 Bytes. 58
Figura 40 Algoritmo AARF: Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 20 paq/s y longitud de L = 1000 Bytes. 59
Figura 41 Escenario para las pruebas de interferencias (terminal oculto)..... 61
Figura 42 Resistencia de ONOE original (100p/s, 1000B) a interferencias de 100 p/s y longitud de 1000 bytes. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 62
Figura 43 Resistencia de ONOE con ventana de 200 ms (100paq/s, 1000B) a interferencias de 100 paq/s y 1000 Bytes. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b)..... 62
Figura 44 Resistencia de ONOE con umbral de reintentos del 30% (100p/s, 1000B) a interferencias de 100p/s y 1000Bs. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 63
Figura 45 Resistencia de ONOE con umbral de reintentos del 30% y ventana de 200 ms (100p/s, 1000B) a interferencias de 100 p/s y 1000B . Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 63
Figura 46 Resistencia de ARF (100p/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 64

Figura 47 Resistencia de AARF (100paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 65
Figura 48 Resistencia de ONOE original (20paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 65
Figura 49 Resistencia de ONOE con ventana de 200ms (20paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 66
Figura 50 Resistencia de ARF (20paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 66
Figura 51 Resistencia de AARF (20paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 67
Figura 52 Prueba con 2 estaciones: STA 2 (azul) en movimiento con 3 tramos (acercándose, alejándose y acercándose al AP) y STA 1 (verde) en reposo. Transmisión de 52 paq/s para el STA 1 y de 46 paq/s para el STA 2. La longitud del paquete es L=1000B. El algoritmo utilizado es ONOE con ventana de 200ms. 68
Figura 53 Escenario para la medida de las variaciones del entorno 69
Figura 54 Paso de personas cerca de la STA: Eficiencia calculada en pasos (200ms, 1 y 10s) para ONOE original (a) y ONOE con ventana de 200 ms (b). Transmisión de 100 paq/s con longitud L=1000 Bytes desde el AP a la STA..... 70

Acrónimos

AARF – *Adaptative Auto Rate Fallback*

ACK – *Acknowledgement* (Reconocimiento)

AP – *Access Point* (punto de acceso)

ARF – *Auto Rate Fallback*

BARA – *Beacon Assisted Rate Adaptation* (Adaptación de tasa asistida por *beacons*)

CARA – *Colisión Aware Rate Adaptation*

DCF – *Distributed Coordination Function* (Función de coordinación distribuida)

DRL – *Deep Reinforcement Learning* (Aprendizaje por refuerzo profundo)

DSSS – *Direct Sequence Spread Spectrum* (Espectro ensanchado por secuencia directa)

FER – *Frame Error Rate* (Tasa de error de tramas)

GI – *Guard Interval* (Intervalo de guarda)

IEEE – *Institute of Electrical and Electronics Engineers* (Instituto de ingenieros eléctricos y electrónicos)

ISM – *Industrial Scientific Medical* (Industrial Científico Medico)

MAC – *Media Access Control* (Control de acceso al medio)

MCS – *Modulation and Coding Schemes* (Esquemas de modulación y codificación)

MIMO – *Multiple Input Multiple Output* (Múltiples Entradas Múltiples Salidas)

MLME – *MAC Layer Management Entity*

OFDM – *Orthogonal Frecuency-Division Multiplexing* (Multiplexación por división de frecuencias ortogonales)

OFDMA – *Orthogonal Frecuency-Division Multiple Access* (Acceso múltiple por división de frecuencias ortogonales)

PLME – *Physical SubLayer Management Entity*

QAM – *Quadrature Amplitude Modulation* (Modulación de amplitud en cuadratura)

QoS – *Quality of Service* (Calidad de servicio)

RBAR – *Received Based Auto Rate* (Tasa automática basada en el receptor)

RFMON – *Radio Frecuency MONitor* (Monitor de radiofrecuencia)

RRAA – *Robust Rate Adaptation Algorithm*

RSSI – *Received Signal Strength Indicator* (Indicador de fuerza de la señal recibida)

RSSLA – *Received Signal Strength Link Adaptation*

SDM – *Spatial División Multiplexing* (Multiplexación por división espacial)

SDN – *Software Defined Networks* (Redes definidas por software)

SME – *Station Management Entity* (Entidad de gestión de la estación)

SNR – *Signal to Noise Ratio* (Relación señal a ruido)

STA – *Station* (Estación)

STBC – *Space-Time Block Coding* (Codificación en bloque espacio-tiempo)

WLAN – *Wireless Local Area Network* (Red de área local inalámbrica)

1. Introducción

1.1. Contexto del trabajo

Nuestra sociedad avanza hacia un mundo inalámbrico en el que cada vez más dispositivos portátiles incorporan al menos una tecnología inalámbrica de acceso radio. Estas tecnologías buscan proporcionar mayores áreas de cobertura y una conectividad más rápida en su afán de implantar soluciones innovadoras y mejorar la accesibilidad y calidad de los servicios. En este contexto, Wi-Fi es la tecnología preferida para escenarios de interior, donde es más fácilmente accesible y presenta ventajas para los despliegues de redes locales privadas en términos de modelo de negocio. Esta elección se apoya en su bajo coste, menor complejidad de dispositivos y equipos, facilidad de despliegue y mayor control de los datos.

Los despliegues WLAN (*Wireless Local Area Network*) incluyen un conjunto de APs (*Access Points*) gestionados por la misma entidad y suelen desplegarse en escenarios como aeropuertos, centros comerciales o incluso ciudades enteras. Con frecuencia estas soluciones con APs iguales, conocidas como "*Enterprise* Wi-Fi", son propietarias, cerradas y costosas, lo que en la mayoría de los casos las hace inasequibles para muchas organizaciones. Además, muchas de ellas suelen carecer de la flexibilidad, escalabilidad y dinamismo necesarios para optimizar la utilización de las redes Wi-Fi. Por ello, el objetivo es alcanzar un nuevo nivel de coordinación en propuestas de despliegue WLAN que incluyan un conjunto de AP heterogéneos en el mismo dominio con funciones avanzadas como la clasificación del tráfico, el equilibrio de la carga, la planificación de frecuencias, los traspasos inteligentes o el control de potencia, haciendo uso de *hardware* comercial de bajo coste (*Comercial off-the-shelf network cards*) y *software* abierto.

La aparición de las redes definidas por software (*Software Defined Networks*, SDN) facilita un nuevo nivel de coordinación en las redes Wi-Fi (*Software-Defined Wireless Networks, SDWN*), aportando a la red un nivel de programación que permite supervisar el entorno inalámbrico y utilizar algoritmos más inteligentes para las funcionalidades a proporcionar. Esta línea de trabajo es la que viene desarrollando el grupo CeNIT (*Communications Networks and Information Technologies*) con soluciones integradas en pruebas de concepto que demuestran la viabilidad de las propuestas realizadas. La programabilidad de las soluciones propuestas supone la utilización de *scripts* que generalmente hacen uso del modo de operación denominado monitor de los dispositivos Wi-Fi comerciales de bajo coste para así tener un control total de las tramas MAC transmitidas (*raw frames*). La contrapartida es que funcionalidades básicas tales como el escaneo de redes (*active and passive scanning*), autenticación, asociación o control de tasa de transmisión también deben ser programadas por los investigadores sobre estos dispositivos, ya que quedan anuladas las funcionalidades desarrolladas por los fabricantes de las tarjetas Wi-Fi.

En este contexto, este Trabajo Fin de Máster se centra en la incorporación de la función de control de tasa de transmisión en pruebas de concepto de entornos SDWN ya desarrollados. La incorporación de esta funcionalidad es imprescindible para desarrollar, aprovechando la programación de la red, nuevos algoritmos de control y coordinación que puedan utilizar las características del tráfico de red y toda la información obtenida mediante la monitorización del entorno para mejorar su eficiencia. A través de este mayor nivel de inteligencia, se pretende cubrir las demandas de ancho de banda y proporcionar una mayor calidad de servicio (*Quality of Service*).

1.2. Objetivos

Como se ha comentado en el apartado anterior, el propósito de este trabajo es la integración de la función de control de tasa de transmisión en pruebas de concepto de entornos SDWN ya desarrollados. Por ello se plantea como objetivo principal el desarrollo de un entorno experimental Wi-Fi que permita el diseño y evaluación de algoritmos de control de tasa de transmisión en el punto de acceso, aplicables a las transmisiones en sentido descendente (AP hacia STA). Para ello se definen los siguientes objetivos específicos:

- Proponer, implementar y evaluar algoritmos de adaptación de tasa en el entorno a partir de la monitorización de parámetros físicos (por ejemplo, niveles de señal recibido) o estadísticas de nivel de enlace (por ejemplo, pérdida de paquetes).

- Crear un entorno experimental que permita integrar comunicaciones inalámbricas Wi-Fi monitorizables a distintas tasas de transmisión en sentido descendente (del AP a las STAs). Los equipos que formen parte del entorno actuarán como AP y estaciones Wi-Fi (STAs). Será preciso controlar los procesos de inyección de tramas conforme a las peculiaridades de cada versión del estándar Wi-Fi e implementar los algoritmos de adaptación de tasa en el AP. Se ha de garantizar el correcto ajuste temporal de todos los procesos del sistema (transmisión, monitorización de parámetros, adaptación de tasa).

- Analizar los distintos tipos de algoritmos de adaptación de tasa en cuanto a su rendimiento en un escenario real con distintas condiciones del entorno (estático, movimiento, con interferencias, variaciones del entorno…).

- Mejorar la eficiencia de los dispositivos que hacen uso de la tecnología Wi-Fi contribuyendo de esta manera a los Objetivos de Desarrollo Sostenible (Meta 9.4).

1.3. Estructura de la memoria

En este primer capítulo se ha presentado el contexto del trabajo realizado y los principales objetivos del mismo. En el segundo capítulo se introduce el estado actual del estándar 802.11 y sus versiones, incidiendo en las características clave en la evolución de Wi-Fi. Su conocimiento facilitará la comprensión de la inyección de tramas en sus distintas opciones y variantes tanto *software* como *hardware*. Tras ello, se presentan los algoritmos de adaptación de tasa y se analizan cuáles son las mejores opciones para implementar en nuestro escenario teniendo en cuenta la complejidad y la disponibilidad de los parámetros de entrada disponibles.

En el tercer capítulo, se presenta el entorno experimental diseñado con las opciones *hardware* y *software* elegidas. Inicialmente, se introducen los aspectos más relevantes relativos al procedimiento para llevar a cabo una inyección controlada de tramas que permita una evaluación precisa de lo que ocurre en la interfaz radio. Asimismo, se presentan las posibles alternativas a considerar para la obtención de las métricas que deben dar soporte a la implementación y evaluación de los algoritmos de control de tasa. Tras ello, se presenta la estructura del programa que controla el escenario con sus tres partes diferenciadas: el transmisor, el algoritmo de control de tasa y el espía o monitor de red. A lo

largo del capítulo se incide en los problemas que han aparecido durante el desarrollo y las soluciones aportadas.

En el cuarto capítulo se realizan las pruebas con los algoritmos de adaptación de tasa seleccionados en las distintas situaciones en las que puede encontrarse una STA como son las situaciones de reposo, movimiento, en presencia de interferencias causadas por un terminal oculto o la influencia de las variaciones del entorno de la STA.

Por último, en el quinto capítulo se presentan las conclusiones del trabajo realizado, detallando las ventajas e inconvenientes de la implementación del escenario experimental propuesto. También se presentan las conclusiones de los algoritmos analizados mostrando su comportamiento en las diferentes condiciones del entorno, y recopilando las características que debe tener un buen algoritmo de adaptación de tasa para afrontar las distintas situaciones de entorno Wi-Fi. Finalmente, se muestran las líneas futuras de trabajo.

En el anexo I se muestra la tabla de índices MCS completa para el estándar 802.11n y 802.11ac.

2. Estado del arte

En este capítulo se introduce el estado actual del estándar 802.11 y sus versiones a lo largo de los años, incidiendo en las características clave en la evolución de Wi-Fi. Su conocimiento facilitará la comprensión de la inyección de tramas en sus distintas opciones y variantes tanto *software* como *hardware*. Por último, se presentan los diferentes algoritmos de adaptación de tasa presentes en la literatura y se analizan cuáles son las mejores opciones para implementar en nuestro escenario teniendo en cuenta la complejidad y la disponibilidad de los parámetros de entrada para la ejecución de dichos algoritmos.

2.1. IEEE 802.11

El estándar 802.11 es una familia de normas inalámbricas creada en 1997 por el IEEE (Instituto de Ingenieros Eléctricos y Electrónicos). La tecnología fue impulsada por un conjunto de empresas reunidas en torno a la alianza Wi-Fi, que buscaban una tecnología que permitiese comunicar diferentes dispositivos de manera inalámbrica independientemente del fabricante. Rápidamente se convirtió en el estándar más utilizado del mercado para la creación de redes locales inalámbricas (WLAN). Estas normas 802.11 modifican los niveles más bajos de la arquitectura OSI, la capa física y de enlace MAC, para poder transmitir tramas por el medio inalámbrico.

A diferencia de otras tecnologías inalámbricas como pueden ser las redes móviles, 802.11 utiliza las bandas de frecuencia no reguladas de 2,4GHz (banda libre ISM, industrial, científica y médica) y/o 5GHz y, por tanto, gratuitas y de acceso libre sin licencia. El problema de estas bandas es la saturación e interferencia existente ya que cualquier dispositivo puede emitir en ellas. Por ello se introduce el límite de potencia de transmisión a 1 mW. Además, para garantizar el uso oportunista del espectro evitando o limitando las colisiones entre transmisiones de dispositivos Wi-Fi o con otras tecnologías, 802.11 incorpora el protocolo de acceso al medio CSMA/CA (*Carrier Sense Multiple Access with Colission Avoidance*), que permite a diferentes usuarios o estaciones interactuar sin un punto central que coordine las transmisiones. Así, cuando una estación quiere transmitir una trama, primero escucha el medio para saber si se encuentra libre y después tras esperar un tiempo determinado (DIFS, DCF *InterFrame Spacing*) transmite una trama de datos. Por su parte, el receptor tras recibir la trama de datos espera un tiempo SIFS (*Short Inter-Frame Space*), más pequeño que DIFS para tener prioridad sobre las tramas de datos, y responde con una trama de reconocimiento ACK (*Acknowledgement*) para confirmar la recepción. Si el emisor no recibe esta trama durante un tiempo indicado, asume que los datos se han perdido y, tras un tiempo de espera aleatorio (*backoff*), retransmite de nuevo la trama de datos. Este proceso se puede observar en la Fig. 1.

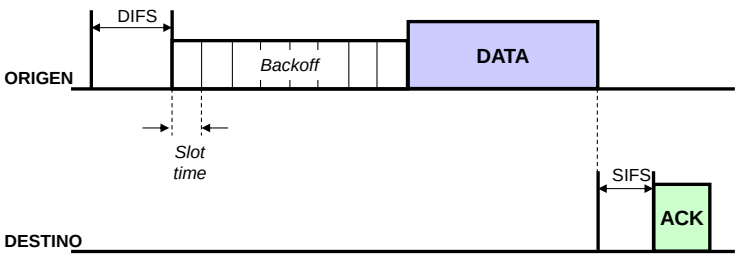


Figura 1 Transmisión de una trama de datos con DCF

Sin embargo, CSMA/CA presenta algunas deficiencias como la del terminal oculto. Este problema aparece cuando una estación intenta transmitir al encontrar el canal libre y otra estación que está fuera

de su rango de cobertura (pero que alcanza también al mismo destinatario) intenta transmitir a la vez. El destinatario puede recibir a la vez la señal de las dos estaciones produciéndose colisiones. La Fig.2 escenifica esta situación donde A quiere transmitir a B al encontrar el canal libre, pero existe el problema de que transmitan a la vez A y C (terminal oculto) colisionando en la zona de recepción de B al no tener constancia A de que existe C y viceversa.

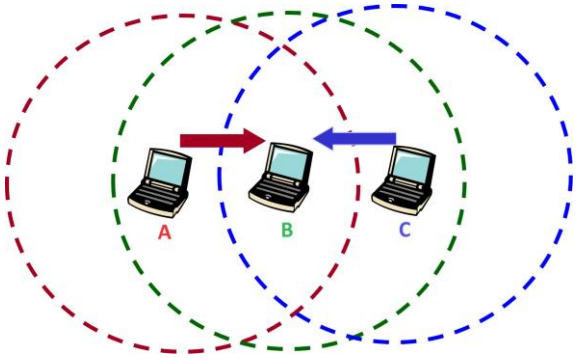


Figura 2 Problema del terminal oculto

Esta situación se puede mitigar en gran medida utilizando las tramas de control RTS/CTS. Primero se envía la trama RTS (*Request to Send*) para notificar al destinatario y al resto de estaciones que queremos transmitir una trama. Si el destinatario está listo, envía de vuelta una trama CTS (*Clear To Send*), indicando que está listo para recibir datos y notificando de esta manera a otros usuarios para que no transmitan durante un tiempo. En la Fig.3 se puede ver el uso de las tramas RTS/CTS en el caso del terminal oculto.

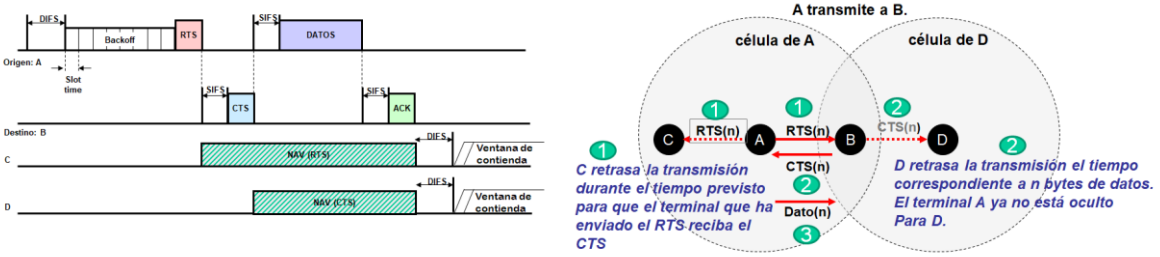


Figura 3 Entrega de un paquete de datos con DCF utilizando tramas RTS/CTS para solucionar el terminal oculto

Además del problema mencionado, también se puede dar el problema del terminal expuesto como puede verse en la Fig.4. Esta situación se da cuando una estación que quiere transmitir a un nodo escucha una transmisión de otra estación a un destino distinto al suyo, con lo que asume que el medio está ocupado y no transmite. Este problema también puede ser minimizado con la utilización de las tramas RTS/CTS.

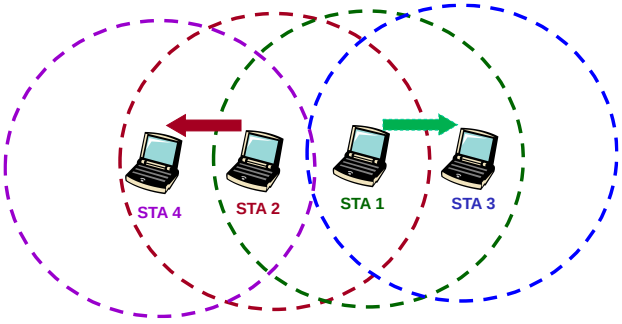


Figura 4 Problema del terminal expuesto

2.2. Estándares Wi-FI: evolución y características clave

En este apartado se presenta la evolución de los estándares Wi-Fi con todas las mejoras y cambios tecnológicos que se han añadido para aumentar el rendimiento y solucionar los problemas que aparecen al transmitir en un medio inalámbrico compartido. Además, se abordan las características clave en esta evolución cuya comprensión facilitan muchos de los aspectos a tener en cuenta en la inyección de tramas.

2.2.1. Evolución estándares

Las diferentes versiones del estándar se ordenan según letras (a, b, g, n, ac y ax) donde las letras intermedias no mencionadas corresponden a pequeñas mejoras que se han añadido, pero no definen una versión completa del estándar. Posteriormente se ha añadido un orden numerado (Wi-Fi 1, 2, 3, 4, 5 y 6) similar a las redes móviles como se puede ver a continuación:

- Wi-Fi (802.11): Es la versión original que se creó en 1997 con unas tasas de 1 y 2 Mbps transmitiendo mediante señales infrarrojas. Incorpora el protocolo de acceso al medio CSMA/CA. Se encuentra obsoleto a día de hoy.

- Wi-Fi 1 (802.11b): Se introdujo en el mercado en 1999 alcanzando una gran popularidad. Utiliza la banda de 2,4 GHz y tiene una tasa de transmisión desde 1 a 11 Mbps. Utiliza 14 canales frecuenciales con solapamiento.

- Wi-Fi 2 (802.11a): Se estandarizó en 1999 y llego al mercado en 2002. Con una velocidad de transmisión desde 6 hasta 54 Mbps, soporta la transmisión en la banda de 5 GHz, evitando de esta manera la saturación y las posibles interferencias de la banda de 2.4 GHz. Utiliza 52 subportadoras con la tecnología OFDM. El alcance es inferior a 802.11b al utilizar una frecuencia mayor.

- Wi-Fi 3 (802.11g): Se introdujo en 2004, aumentando la capacidad de transmisión de hasta 54 Mbps. Destaca la compatibilidad con versiones anteriores utilizando la banda de 2,4 GHz. De la misma manera que su predecesor, está basado en OFDM.

- Wi-Fi 4 (802.11n): Fue ratificado en 2009. Está basado en OFDM e incorpora la tecnología MIMO (múltiples antenas) y agregación de canales de 20 MHz (hasta 40 MHz). Soporta 2,4 y 5 GHz y aumenta la tasa de transmisión hasta 600 Mbps con modulaciones hasta 64-QAM. Además, incorpora un campo para aplicar calidad de servicio (QoS) y agregación de tramas.

- Wi-Fi 5 (802.11ac): Desarrollado entre 2011 y 2013 y ratificado en 2014. Únicamente permite utilizar la banda de 5 GHz, aumentando el número de los canales. Incorpora la tecnología MU-MIMO (*Multi User* MIMO) en el *downlink*, lo que permite que un punto de acceso

transmita a varios usuarios a la vez. La tasa aumenta hasta 6,9 Gbps con modulaciones hasta 256-QAM. Tiene canales de hasta 80 MHz y soporta agregación de canales hasta 160 MHz.

- Wi-Fi 6 (802.11ax): Fue ratificado en 2021. Añade la banda de 6 GHz. Introduce MU-MIMO en el *uplink*, con un punto de acceso recibiendo desde varios clientes simultáneamente. Basado en la tecnología OFDMA. Incorpora modulaciones de más alto nivel hasta 1024 QAM lo que hace aumentar la tasa de transmisión hasta 9,6 Gbps.

- Wi-Fi 7 (802.11be): Es la última evolución de la tecnología Wi-Fi prevista para mayo de 2024.

La Fig. 5 presenta de manera resumida la evolución la tecnología y estándares Wi-Fi.

Wi-Fi generations					
	Wi-Fi 4	Wi-Fi 5	Wi-Fi 6	Wi-Fi 6E	Wi-Fi 7 (expected)
Launch date	2007	2013	2019	2021	2024
IEEE standard	802.11n	802.11ac	802.11ax		802.11be
Max data rate	1.2 Gbps	3.5 Gbps	9.6 Gbps		46 Gbps
Bands	2.4 GHz and 5 GHz	5 GHz	2.4 GHz and 5 GHz	6 GHz	1–7.25 GHz (including 2.4 GHz, 5 GHz, 6 GHz bands)
Security	WPA 2	WPA 2	WPA 3		WPA3
Channel size	20, 40 MHz	20, 40, 80, 80+80, 160 MHz	20, 40, 80, 80+80, 160 MHz	20, 40, 80, 80+80, 160 MHz	Up to 320 MHz
Modulation	64-QAM OFDM	256-QAM OFDM	1024-QAM OFDMA		4096-QAM OFDMA (with extensions)
MIMO	4x4 MIMO	4x4 MIMO, DL MU-MIMO	8x8 UL/DL MU-MIMO		16x16 MU-MIMO

Source: IEEE, Intel Corporation, Wi-Fi Alliance

Figura 5 Evolución de la tecnología y estándares Wi-Fi

La evolución presentada ha supuesto cambios tanto a nivel de enlace (MAC) como a nivel físico. Se podría decir que los cambios MAC, exceptuando la incorporación de adaptaciones para soportar calidad de servicio (QoS) incorporadas en la mejora 802.11e, no se han producido hasta la versión 802.11n con la introducción de la agregación de tramas. Los cambios en el nivel físico han sido una constante en todas las versiones cambiando esquemas de transmisión radio (DSSS, OFDM, OFDMA), ampliando las bandas de frecuencia y canales utilizados, pasando por la mejora en las modulaciones utilizadas (64-QAM a 1024-QAM y 4096 QAM) hasta las sucesivas mejoras gracias a la incorporación de tecnologías MIMO (*Multiple Input Multiple Output*).

2.2.2. Características clave de la tecnología Wi-Fi

En el este apartado profundizaremos en alguna de las características clave de nivel físico de la tecnología W-Fi mencionadas en el apartado anterior, teniendo en cuenta su posterior uso en la inyección de tramas.

- **Canales**

Los canales y frecuencias utilizados se han ido modificando constantemente a lo largo del estándar para poder soportar las nuevas funcionalidades y solucionar problemas que han aparecido como son la saturación de las bandas y las interferencias. Uno de los principios de 802.11 es transmitir en bandas gratuitas de acceso libre sin licencia. Por ello, en primeras versiones utiliza únicamente la banda libre de 2.4 GHz como se puede ver en la Fig.6, que muestra los 14 canales solapados de 22 MHz (con separación de 5 MHz) utilizados en 802.11 g/n. Los canales a utilizar dependen de la región donde se encuentre el dispositivo. Así, el mundo fue dividido en 3 regiones por la ITU-R (Grupo de radiocomunicaciones de la ITU). Por ejemplo, en Europa solo es posible utilizar los canales del 1 al 13 donde sólo es posible utilizar 3 canales (1, 6 y 11) sin solapamiento. En Norteamérica se definieron únicamente 11 canales.

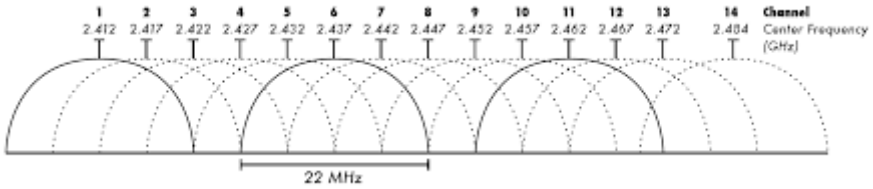


Figura 6 Canales de la banda ISM de 2.4GHz para 802.11 b/g. (El canal 14 solo aparece en Japón).

Posteriormente se añade la banda de 5 GHz, con la regulación sobre su uso que puede observarse en la Fig.7.

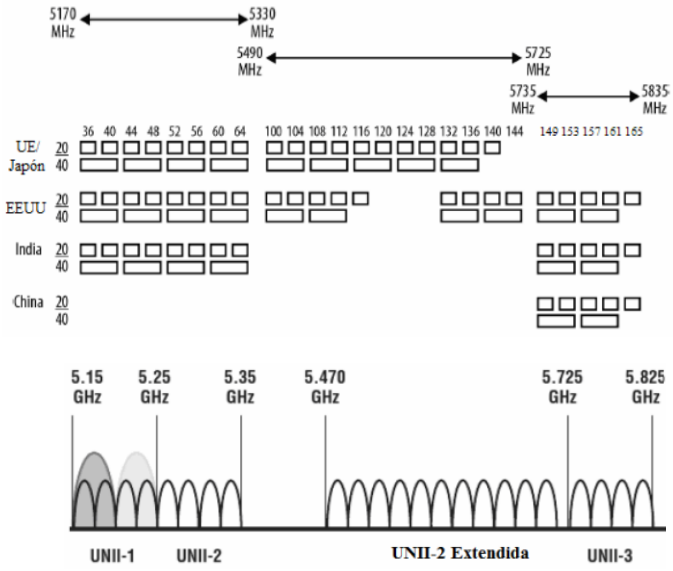


Figura 7 Banda de 5 GHz en las diferentes regiones (UE, EEUU, India y China) y canalizaciones para Europa [1]

Se puede ver que es posible utilizar canales de 20 MHz o agregar dos canales para utilizar 40 MHz. La agregación de canales, conocida como *channel bonding* e introducida por primera vez en el estándar 802.11n (ver Fig. 8) permite usar dos canales adyacentes de 20MHz para que se comporten como uno de 40MHz. De esta manera es posible incrementar la tasa de transmisión en la comunicación con una STA. En la banda de 5GHz definen 23 canales no solapados al utilizar OFDM. Destaca un alcance menor respecto a la banda de 2.4 GHz, pero presenta la ventaja de que los canales no se interfieren entre sí como en la banda de 2.4GHz. Se divide en 4 bandas: UNII-1, UNII-2, UNII-2 extendida y UNII-3.

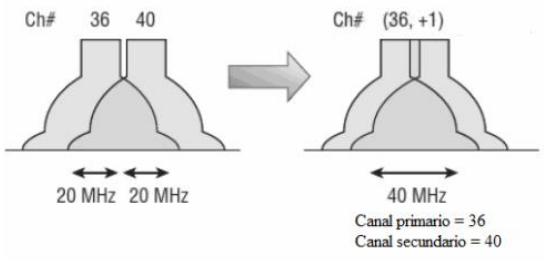


Figura 8 Channel bonding en 802.11n [1]

- **Modulación OFDM y formato de trama física**

Partiendo del estándar 802.11n (aunque se usan modulaciones DSSS en la banda de 2.4GHz), si nos centramos en éste, evoluciones posteriores y la banda de 5GHz, podemos decir que la capa física en Wi-Fi se basa de forma generalizada en el uso de la modulación OFDM, incluyendo distintos formatos de transmisión posibles (denominados MCS: *Modulation and Coding Schemes*) resultado de diferentes combinaciones de modulaciones BPSK, QPSK, 16-QAM y 64-QAM y tasa de codificación de 1/2, 2/3, 3/4 y 5/6. Por otra parte, con respecto a versiones anteriores del estándar (802.11a/g) también OFDM, las mejoras incorporadas en 802.11n que permiten elevar la tasa de transmisión hasta 600Mbps se basan en las técnicas básicas a nivel físico ya mencionadas (uso de esquemas MIMO, *channel bonding*, ampliación de los esquemas de MCS) a las que se suman mejoras a nivel MAC (por ejemplo, la agregación de tramas) pero también en otros cambios sobre la capa física OFDM como la posibilidad de optar por varias configuraciones para los intervalos de guarda en OFDM y elevar el número de subportadoras útiles para datos de 48 a 52. En la Tabla 1 se muestran los principales parámetros físicos de 802.11n (que heredan las versiones posteriores) con respecto a 802.11a/g.

Tabla 1 Parámetros de IEEE 802.11n OFDM comparados con IEEE 802.11a/g

Standards	IEEE 802.11a/g	IEEE 802.11n	
		Mandatory	Optional
Maximum transmission rate (Mbps)	54	130	600
Bandwidth(MHz)	20	20	40
FFT size	64	64	128
Number of subcarrier (data + pilot)	52 (48+4)	56 (52+4)	114 (108+6)
Multi-antenna scheme	signal antenna	2 Tx MIMO	3,4 Tx MIMO Tx Beam forming STBC
Channel coding	Convolutional code (1/2, 2/3, 3/4)	Convolutional code (1/2, 2/3, 3/4, 5/6)	LDPC (1/2, 2/3, 3/4, 5/6)
Modulation	BPSK, QPSK, 16-QAM, 64-QAM		
Spatial stream	1	1 ~ 2	1 ~ 4
Guard interval(ns)	800	800	400
Subcarrier interval	312.5 KHz	312.5 KHz	312.5 KHz
FFT period	3.2 μs	3.2 μs	3.2 μs
Symbol period	4 μs	4 μs	4 μs

Todos estos factores redundan en un incremento de tasa de transmisión, pero requieren la consideración de nuevos formatos de trama física que mantengan la interoperabilidad con dispositivos de versiones anteriores. La Fig. 9 muestra el formato de trama física específica requerido para la capa física OFDM y los formatos para facilitar la transmisión con MIMO (formato *High Throughput*).

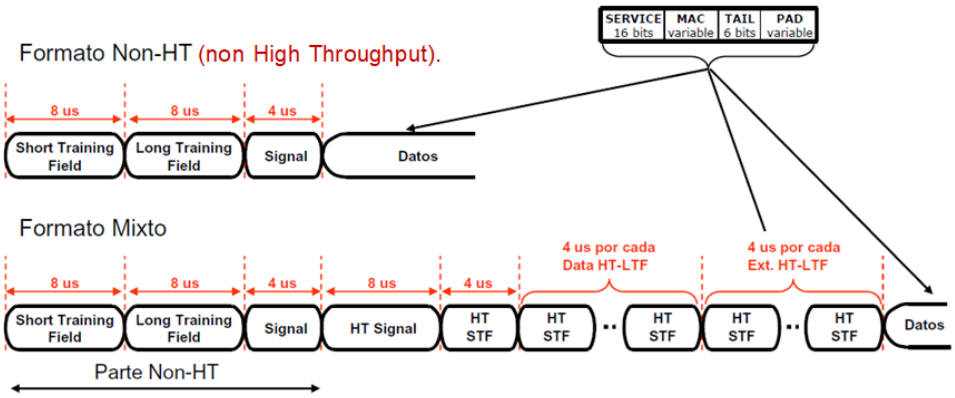


Figura 9 Formatos de trama Non-HT y mixto (Non-HT y HT) para 802.11n

Con respecto al intervalo de guarda (GI, *Guard Interval*), hay que tener en cuenta que los entornos Wi-Fi se ven muy afectados por la propagación multicamino al producirse una gran cantidad de reflexiones de la señal en las paredes y obstáculos en los interiores de edificios. En el caso de utilizar la modulación OFDM, para evitar que los símbolos recibidos por distintos caminos con un retardo diferente se interfieran es preciso definir el intervalo de guarda entre símbolos OFDM. Consiste en una extensión del símbolo actual que absorbe la diferencia de retardo entre varios caminos. En general, el tiempo de guarda debe ser 4 veces la dispersión del retardo multicamino (*delay spread*), esto es, la diferencia entre el tiempo de llegada del primer componente de la señal y la última componente multicamino. El valor de *delay spread* depende del entorno de propagación, inicialmente estimado en 200 ms para entornos interiores, por lo que el valor estándar en 802.11 es de 800 ns. Más adelante se demostró que el retardo *delay spread* se encuentra alrededor de 50-75 ns, por lo que se introdujo en 802.11n la posibilidad de elegir opcionalmente un intervalo de guarda corto (SGI) de 400 ns [2].

Cuanto más pequeño es el intervalo de guarda mayor es la información útil que se transmite lo que permite aumentar la capacidad. Con una duración del símbolo de 3,2 µs y usando el intervalo estándar (0,8 µs) el *overhead* es del 25%, mientras que al utilizar el intervalo corto (0,4 µs) el *overhead* se reduce al 10% [2]. Sin embargo, es preciso elegir adecuadamente el formato GI porque en caso de elegir un GI demasiado pequeño para el entorno, se produce interferencia intersimbólica y en consecuencia aumenta la pérdida de paquetes. Esto obviamente provoca pérdida de capacidad

- **MIMO (Multiple Input Multiple Output)**

Como se ha comentado, una de las técnicas para incrementar la tasa es la aplicación de técnicas MIMO. Hasta la versión 802.11g del estándar sólo se utilizaba un único flujo (*stream*) y la transmisión y recepción se realizaba mediante una única antena en transmisión y recepción. A esta tecnología se le conoce como SISO (*Single Input Single Output*). Con la llegada del estándar 802.11n se introduce la capacidad de utilizar una o varias antenas en transmisión y recepción. Esta tecnología conocida como MIMO permite aprovechar las características de propagación multicamino del entorno Wi-Fi para aumentar la capacidad y la fiabilidad del sistema.

Hay dos técnicas básicas de transmisión con múltiples antenas en MIMO: Multiplexación espacial (*Spatial Multiplexing*, SM) y Diversidad espacial (*Spatial Diversity*, SD). La multiplexación espacial es una técnica que permite aumentar significativamente las tasas de datos y la capacidad de las redes

inalámbricas. Esta técnica involucra la transmisión simultánea de múltiples flujos de datos (2 en el caso de un MIMO 2x2) sobre el mismo canal de frecuencia (a cada uno de ellos se le denomina *spatial stream*), multiplicando efectivamente la capacidad del canal sin aumentar el ancho de banda. La clave radica en el uso de múltiples antenas tanto en el transmisor como en el receptor (tantas en cada equipo como canales se multiplexan espacialmente), con canales incorrelados entre pares de antenas transmisoras y receptoras, lo que permite la separación de los flujos de datos individuales en recepción. Además, en función del número de dispositivos simultáneos con que se establece la comunicación, la tecnología MIMO puede ser dividida en SU-MIMO (*Single User-MIMO*) y MU-MIMO (*Multi User-MIMO*). Por el contrario, la diversidad espacial es una técnica que permite mejorar la calidad de la señal y la fiabilidad de la comunicación. Esta técnica transmite varias copias del mismo flujo de datos a través de diferentes caminos espaciales que, por diversidad, puede proporcionar una ganancia que ayuda a combatir los efectos de la atenuación y la interferencia en los canales inalámbricos.

Inicialmente aparece la opción SU-MIMO, donde solo un único usuario puede recibir datos, mientras que otros usuarios deben esperar a encontrar el canal libre para transmitir. En posteriores versiones a 802.11n se añade la posibilidad de que un AP transmita a distintos usuarios a la vez utilizando varias antenas con MU-MIMO. De esta manera los usuarios no tienen que esperar a que un usuario anterior termine su recepción. Es útil en situaciones específicas como puede ser una red con muchos usuarios estáticos para reducir el retardo de las transmisiones y de esta manera mejorar la calidad del servicio (QoS) en aplicaciones que tengan requisitos temporales como pueden ser el audio y el vídeo. En general supone un aumento de la capacidad de transmisión. Permite hasta 4 usuarios simultáneos como se muestra en la Fig.10 para 802.11ac (si hay suficientes antenas tanto en transmisión como en la recepción).

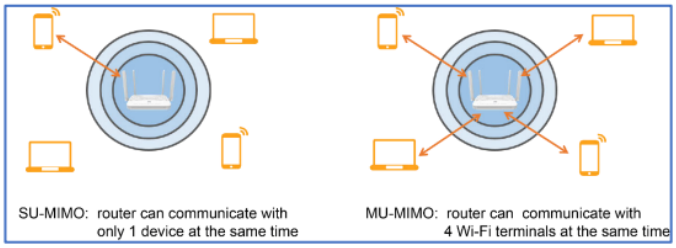


Figura 10 Tecnología SU-MIMO (Single User) y MU-MIMO (Multi User) en 802.11ac [1]

- **MCS (*Modulation Coding Schemes*)**

Todas las versiones del estándar Wi.Fi incluyen distintos formatos de tasa de transmisión posibles, cada uno de ellos con requisitos de relación señal ruido diferentes para garantizar una tasa de error determinada en la transmisión. De hecho, la adaptación de tasa (objetivo de este proyecto) lo que busca es seleccionar en cada momento la tasa de transmisión más adecuada en función de las condiciones de canal ya que en caso de establecer una tasa fija se corre el riesgo de sufrir muchas pérdidas de paquetes o infrautilizar la capacidad del canal. Desde la versión 802.11n las velocidades de datos se definen por un esquema de codificación y modulación MCS. En 802.11a/g, las velocidades se definían en base a la modulación y estaban comprendidas entre los 6 y 54 Mbps. Sin embargo, en 802.11n las velocidades de datos dependen de varios factores: la modulación, el número de *streams* espaciales, la anchura del canal, el intervalo de guarda y la tasa de codificación [2]. La introducción de estos nuevos parámetros que modifican la tasa de transmisión lleva al grupo de trabajo de IEEE 802.11 a definir unos índices que agrupan de manera sencilla las diferentes tasas de transmisión según las combinaciones de parámetros. De esta manera aparecen las tablas de índices MCS donde se contemplan todas las opciones de transmisión. En la Tabla 2 se puede ver un fragmento de la tabla de

MCS completa para 802.11n y 802.11ac. En el caso del estándar 802.11n se utilizará la columna HT MCS (*High Throughput*). Se puede ver que los índices MCS van desde 0 hasta 7 en este recorte, en la tabla completa llega a 23. En 802.11ac se añaden los índices VHT (*Very High-Throughput*) y en 802.11ax los índices de alta eficiencia (*High-Efficiency*).

Por ejemplo, si se escoge el índice 6 para transmitir (círculo rojo), esto implica que la modulación será 64-QAM con una tasa de codificación de ¾ y un solo *stream* espacial. Además, se ha seleccionado un ancho de banda de 20 MHz y un intervalo de guarda corto de 400 ns. Todos estos parámetros determinan que la velocidad de transmisión será de 65 Mbps (círculo verde). En el *Anexo I* se puede ver la tabla completa.

Habitualmente, y para un buen funcionamiento de la inyección de tramas, se han de fijar el ancho de banda y el intervalo de guarda al configurar la tarjeta Wi-Fi, y solo se modifica el MCS durante la transmisión. De esta manera, se toma una columna fija en la tabla y variando el índice MCS se aumenta o se reduce la tasa de transmisión. Cada índice MCS necesita un valor de RSSI y una relación señal a ruido mínima para recibir correctamente los símbolos de la modulación.

Tabla 2 Recorte de la tabla de índices MCS (802.11 n/ac)

802.11n and 802.11ac — MCS, SNR and RSSI

HT MCS	VHT MCS	Modulation	Coding	20MHz Data Rate 800ns	20MHz Data Rate 400ns	20MHz Min. SNR	20MHz RSSI	40MHz Data Rate 800ns	40MHz Data Rate 400ns	40MHz Min. SNR	40MHz RSSI
1 Spatial Stream											
0	0	BPSK	1/2	6.5	7.2	2	-82	13.5	15	5	-79
1	1	QPSK	1/2	13	14.4	5	-79	27	30	8	-76
2	2	QPSK	3/4	19.5	21.7	9	-77	40.5	45	12	-74
3	3	16-QAM	1/2	26	28.9	11	-74	54	60	14	-71
4	4	16-QAM	3/4	39	43.3	15	-70	81	90	18	-67
5	5	64-QAM	2/3	52	57.8	18	-66	108	120	21	-63
6	6	64-QAM	3/4	58.5	65	20	-65	121.5	135	23	-62
7	7	64-QAM	5/6	65	72.2	25	-64	135	150	28	-61
	8	256-QAM	3/4	78	86.7	29	-59	162	180	32	-56
	9	256-QAM	5/6			31	-57	180	200	34	-54

- **Configuración de la trama de nivel MAC**

El formato de trama de nivel MAC, mostrado en la Fig. 11, da soporte al protocolo CSMA/CA descrito anteriormente. En los procesos de inyección de tramas se deberán tener en cuenta la configuración adecuada del campo de control y la estructura dependiendo del tipo de trama a intercambiar. Por ejemplo, el número de campos de dirección presentes varía según los distintos tipos de trama (datos, control, gestión). Asimismo, si se incluyen o no mecanismos de control de la QoS, si se aplica o no agregación de cabeceras, etc.

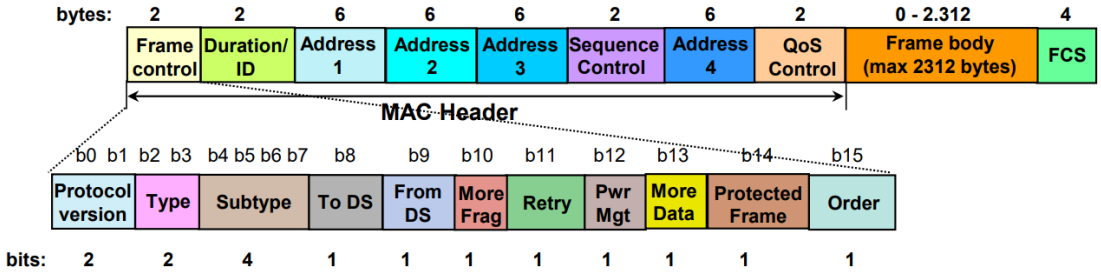


Figura 11 Trama MAC 802.11n

2.3. Algoritmos de control de tasa

La implementación de algoritmos de control de tasa es común en las redes Wi-Fi para ajustar adecuadamente el esquema de transmisión (MCS) a las diferentes condiciones de canal entre AP y STAs y evitar así llenar el canal con transmisiones con una alta probabilidad de error.

A continuación, se realiza una primera clasificación de los algoritmos basándose en enfoques generales.

-*Basado en parámetros según la capa (física/enlace):*

- Parámetros de nivel físico. SINR. Tiempo de transmisión de la trama. Tasa. *Throughput*. BER/FER
- Parámetros de nivel de enlace. Número de ACKs recibidos. Número de reintentos de transmisión.
- Nivel físico y enlace (*Cross-layer*).

-*Basada en métricas:*

- Medidas de señal: SNR o RSSI.
- Tasa de pérdidas de paquetes (FER) y/o número de retransmisiones.

-*Según el origen de la información:*

- Lazo abierto: Basados en propia información.
- Lazo cerrado: Requieren realimentación el extremo opuesto.

-*Cálculo de la tasa:*

- Determinista: sin realizar un promediado temporal (utilizando contadores de paquetes y umbrales)
- Estadístico: basado en un periodo temporal de monitorización.

-*Capacidad o no de distinguir entre pérdidas de canal o pérdidas por colisión.*

-*Enfoque del algoritmo*

- Incremento o decremento secuencial de la tasa.
- Cálculo de tasa óptimo.
- Heurístico: basado en prueba y error de tasas de transmisión.

Teniendo en cuenta todos los enfoques anteriores y combinándolos se pueden encontrar diferentes clasificaciones de algoritmos en la literatura. Inicialmente se propusieron algoritmos basados en tramas perdidas y estadísticas de los paquetes por su simplicidad. A continuación, se presentan algoritmos más complejos basados en medidas de señal (SNR o RSSI) donde aparecen varios enfoques según el origen de la información (lazo abierto o cerrado). Después se muestran los algoritmos con diferenciación de colisiones que permiten aumentar la resistencia a interferencias. Por último, se presenta una variante más reciente basada en redes neuronales para ilustrar la variedad de metodologías a aplicar en el control de tasa.

2.3.1. Algoritmos basados en tramas perdidas y/o throughput

Se propusieron inicialmente por su simplicidad frente a otras opciones. Estos algoritmos se basan en monitorizar el número de transmisiones correctas e incorrectas y ajustar la modulación en consonancia con esas medidas.

- ***Auto Rate Fallback* (ARF)**

Es un algoritmo determinista que usa las pérdidas de trama como indicador de la calidad del canal. Procesa las estadísticas paquete a paquete. Muy utilizado en las primeras generaciones.

Utiliza dos umbrales de número de paquetes enviados y con éxito para aumentar la tasa que son fijos. Se cuentan los paquetes enviados y los paquetes con éxito hasta que se supera uno de los dos umbrales para que el algoritmo aumente la tasa. Además, siempre que se alcanza uno de los dos umbrales se reinician ambos contadores.

- Umbral y contador de ACKs o éxitos: Se cuentan los paquetes de datos que se han enviado correctamente a partir de los ACKs recibidos. El umbral se define como 10 paquetes, es decir son necesarios 10 paquetes con éxito para aumentar la tasa.
- Umbral y contador de paquetes enviados o *Timeout*: Cuenta todos los paquetes de datos enviados por el transmisor independientemente de si tienen o no éxito. Su función es evitar mantenerse durante demasiado tiempo en la misma tasa si no se consigue alcanzar el umbral de paquetes con éxito, por lo que el umbral de paquetes enviados debe ser mayor. De esta manera se define en 15 paquetes a partir del umbral de paquetes con éxito (10), multiplicando por un factor constante llamado *timeout* k (1.5). Es decir, es necesario enviar 15 paquetes de datos antes de aumentar la tasa con este contador, estos se cuentan aunque no tengan éxito.

El algoritmo trabaja de la siguiente manera. Hay dos estados: reposo y *probing packet* (prueba de envío de 1 paquete a una tasa mayor):

- Se empiezan a transmitir tramas a la tasa más baja.
- Si se envían correctamente N tramas (umbral de ACKs) consecutivas correctas o se transmiten suficientes paquetes para alcanzar el umbral *timeout* (N*1,5), entonces se incrementa la tasa de transmisión. Se inicializan el contador de tramas con éxito (*succ*) y el de paquetes (*timeout*).
- Si la primera trama enviada a la nueva tasa (llamada *probing packet*) no se envía correctamente entonces se reduce la tasa.
- Si dos o más tramas se pierden en reposo se inicializan los contadores y se reduce la tasa.

El valor N se define como umbral de ACKs y se fija a 10.

La principal limitación del algoritmo es que tiene un comportamiento oscilante. En situaciones estáticas el algoritmo se encuentra constantemente subiendo y bajando la tasa cada N tramas correctas (o 1-2 tramas fallidas) lo que produce que pierda eficiencia. Por ello, se han propuesto otras versiones (como AARF) con umbrales adaptativos. Lo que provocará un aumento más lento de la tasa tras perdidas de paquetes y una reducción de la oscilación entre tasas. Estos umbrales adaptativos se plantean para mejorar únicamente el rendimiento en situaciones estáticas, hay que destacar que el algoritmo AARF podrá seguir trabajando en situaciones de movimiento, pero por su diseño ARF ya se adapta bien a estas variaciones rápidas así que no es necesaria una mejora en este aspecto. Por otra parte, ARF no distingue entre pérdidas debidas a condiciones de canal o pérdidas por colisiones. Si se producen pérdidas debidas a colisiones es importante no reducir la tasa ya que de esta manera es más probable que ocurran al transmitir durante un tiempo mayor un paquete.

- ***Adaptative Auto Rate Fallback* (AARF)**

Su funcionamiento es similar al anterior, pero en este caso se adapta el umbral de N paquetes para aumentar la tasa y a su vez el umbral de paquetes enviados (*timeout*). De esta manera tiene un comportamiento menos oscilante en situaciones estáticas al aumentar más lentamente la tasa tras perdidas de tramas. La modificación consiste en añadir los siguientes factores fijos para adaptar los umbrales, el resto del algoritmo se mantiene.

- *Succ_k*: Definido como 2, se utiliza para doblar el umbral de tramas con éxito (*succ*). Se aplica en el caso de pérdida del *probing packet*.
- *Succ_min*: Toma el valor de 10 paquetes, se aplica al iniciar o reiniciar el algoritmo para volver a inicializar el umbral de éxitos.
- *Succ_max*: Es el valor máximo que puede tomar el umbral de éxitos o ACKs, también limita el *timeout* a 75 al definirse a partir de él.
- *Timeout_min*: Toma el valor de 15 paquetes. Se calcula como el umbral de éxitos mínimo multiplicado por un factor 1,5 (*timeout_k*).

El funcionamiento del algoritmo es el siguiente. Mantiene los dos estados de ARF: reposo y *probing packet* (prueba de envío de 1 paquete a una tasa mayor):

- Se empiezan a transmitir tramas a la tasa más baja.
- Si se envían correctamente N tramas o el *timeout* expira (1,5*N tramas transmitidas con o sin éxito) entonces se incrementa la tasa de transmisión. A continuación, se intenta transmitir el *probing packet* con la nueva tasa. Se inicializan el contador de tramas con éxito (*succ*) y el de paquetes (*timeout*).
- Si la primera trama enviada a la nueva tasa (*probing packet*) no se envía correctamente entonces se reduce la tasa y se dobla el límite de tramas a 2N. Con un límite máximo definido por el factor *succ_max*. Se inicializan el contador de tramas con éxito (*succ*) y el de paquetes (*timeout*).
- Si no se pierde el paquete de prueba, se mantiene N como el umbral y la nueva tasa. No se inicializan los contadores.
- Si hay dos fallos consecutivos en reposo se inicializan los umbrales y los contadores y se reduce la tasa.

- ***ONOE***

Fue desarrollado por *Mad Wifi* y se encuentra implementado comercialmente en chips *Atheros*. Se trata de un algoritmo en lazo abierto basado en contabilizar paquetes correctos e incorrectos. La principal diferencia con los anteriores es que promedia las estadísticas de los paquetes recibidos en una ventana temporal (1 segundo), lo que hace que sea menos sensible a las variaciones rápidas (ráfagas de errores o cambios en rápidos del canal). Todo esto es a costa de una actualización lenta de la tasa de transmisión, que puede dar lugar a un mayor número de pérdidas o ineficiencia en el uso del canal al utilizar tasas más bajas de las que podría según el nivel de SNR. Este algoritmo no realiza diferenciación de colisiones.

Las estadísticas de paquetes que considera son las siguientes. Se calculan cada ventana de tiempo de W=1s utilizando todos los paquetes recibidos en este periodo. No tienen memoria, se reinician cada ventana.

- *Num*: Número de tramas de datos transmitidas, se cuentan las tramas independientemente de si tienen éxito o no.

- *Retries*: Número de reintentos en promedio de las tramas transmitidas. La tecnología Wi-Fi tras una trama fallida vuelve a intentar transmitir la trama (reintento o *retry*) durante un número determinado de veces (en nuestro caso 6). A partir de esta variable el algoritmo obtiene una medida de las pérdidas en la transmisión, que provocan que una trama tenga que retransmitirse.
- *Succ*: Número de tramas enviadas con éxito. Se comprueba que una trama de datos ha sido confirmada por un ACK.

Además, para añadir un seguimiento temporal del rendimiento al algoritmo en ventanas anteriores y poder ajustar la tasa en consecuencia define el siguiente contador.

- Contador de créditos: Es un contador inicializado en 0 que se incrementa o decrementa en una unidad (crédito) según el rendimiento en cada ventana temporal. Si se alcanzan 10 créditos aumenta la tasa.

El algoritmo se muestra en la Fig.12. Su objetivo es encontrar la mejor tasa para obtener una pérdida de tramas menor del 50%.

Tras una ventana W de monitorización:

- Si más del 50% de los paquetes se retransmiten y se envían un número suficiente de paquetes (10), entonces se reduce la tasa y se inicializan los créditos a 0.
- Si menos del 10% de los paquetes necesitan retransmitirse, se aumentan el crédito en 1. En este caso, si se alcanzan 10 créditos se aumenta la tasa y se inicializan los créditos a 0.
- Si más del 10% de los paquetes necesitan retransmitirse se reduce el número de créditos en 1.
- Si no se envía ninguna trama con éxito, se espera un segundo para volver a adaptar la tasa.

El funcionamiento del algoritmo ONOE se puede ver en el diagrama de flujo de la Fig.12.

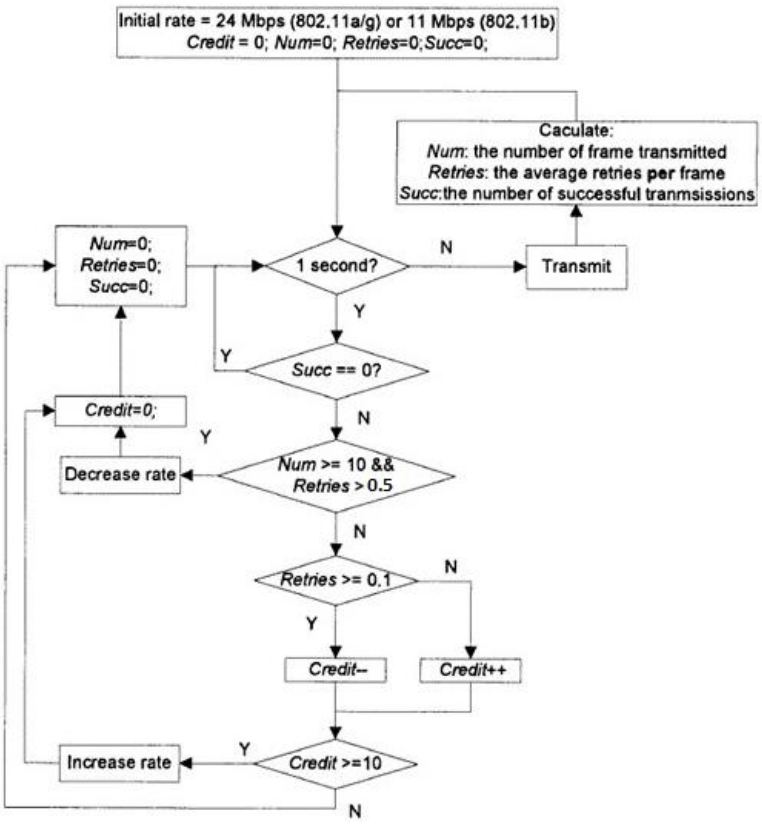


Figura 12 Algoritmo ONOE

- ***Minstrel***

Algoritmo ampliamente utilizado en dispositivos Wi-Fi comerciales que surgió en 2005. Tiene un planteamiento heurístico que prueba diferentes tasas (*retry-chains*) y lleva una cuenta de los éxitos en cada tasa. Evoluciona a la versión *Minstrel-rts* que tiene en cuenta colisiones. Se puede ver como una mejora o evolución de ONOE al utilizar también una ventana temporal para evaluar el algoritmo. Prueba diferentes tasas definidas con una cadena (*retry chain*) siguiendo unos criterios predefinidos, cada ventana temporal de 100 ms evalúa el rendimiento de estas tasas (*statistics of transmissions*) y se elige con cuál transmitir (*normal transmisión*). Además, destina un porcentaje pequeño de los paquetes enviados a probar tasas diferentes a las definidas en la cadena (*sampling transmisión*) para actualizarlas si su rendimiento es bajo.

Estos 4 conceptos de diseño se muestran en detalle a continuación.

- *Retry chain mechanism*: utiliza 4 pares de reintentos (r0, c0), (r1, c1), (r2, c2) y (r3, c3). Primero se envía un paquete a tasa r0 durante c0 reintentos. Si no se envía correctamente, se prueba con (r2, c2). Así sucesivamente hasta el final de la cadena.
- *Statistics of transmission*: almacena las estadísticas de transmisión para cada tasa y se aplica una media móvil (EWMA) para evaluar cada 100ms.
- *Normal Transmission*: durante la transmisión normal, se eligen r0 y r1 como las tasas que han obtenido los dos mayores *throughputs* tras aplicar la media móvil. La tasa r2 corresponde a la tasa con mayor porcentaje de éxitos. La tasa r3 es la que ofrece una menor tasa de transmisión.
  Sampling Transmission: durante la transmisión *sampling* se prueban tasas diferentes a r0-r3 para poder actualizar las tasas antiguas si estas no permiten un *throughput* óptimo.

- ***Sample Rate***

El algoritmo *Sample Rate* [3] intenta optimizar constantemente el tiempo de transmisión de los paquetes. Para ello busca la tasa que conseguiría el menor tiempo de transmisión promedio. Primero mide el tiempo de transmisión total de todos los paquetes enviados a una tasa teniendo en cuenta el tiempo desde el envío de la trama hasta la recepción del ACK. Después, el tiempo de transmisión total se promedia entre las tramas que se han enviado con éxito (confirmadas con ACK). Se van guardando las tasas escogidas y su rendimiento (tiempo de transmisión promedio) como aparece más adelante en la tabla 3. Para eliminar información obsoleta la tabla se reinicia cada ventana de 10 segundos.

El funcionamiento del algoritmo es el siguiente:

- Inicialmente se empieza a transmitir con la mayor tasa posible.
  - Se reduce la tasa si hay 4 perdidas consecutivas (sin contar reintentos). Entonces la tasa y su rendimiento se eliminan de la memoria (marcando un tiempo infinito) para que no se puedan elegir (hasta que se pruebe de manera aleatoria esa tasa). Se reducirá la tasa de esta manera hasta que consiga encontrar una tasa que sea capaz de enviar paquetes con éxito.
- Cada 10 paquetes enviados se enviará un solo paquete con una tasa aleatoria que sea mayor a la actual y con menor tiempo promedio.
  - Si hay menos pérdidas se guarda el tiempo de transmisión.
- En cualquier punto de la transmisión se elige para el siguiente paquete a enviar la tasa de transmisión que ha obtenido el menor tiempo de transmisión promedio.

En la tabla 3 se puede ver el cálculo del tiempo de transmisión promedio. Tiene en cuenta los intentos de transmisión (*Tries*), los paquetes enviados con éxito (*ACK'ed*) y los paquetes fallidos (*Succ. Fails*). De esta manera guarda en memoria las tasas de transmisión probadas anteriormente (*Bit-rate*) y su

tiempo de transmisión promedio (*Avg. TX Time*). Como ejemplo, se puede ver en la primera fila para la tasa 11 Mbps al ocurrir 4 fallos de transmisión de un paquete consecutivos (los reintentos no cuentan como fallo), se ha dejado de utilizar la tasa y se marca el tiempo como infinito. A continuación, se ha reducido la tasa hasta 5,5 Mbps donde se han conseguido transmitir 100 paquetes con éxito sin ningún fallo, así que divide el tiempo de transmisión total entre los 100 paquetes para calcular el tiempo promedio. En el resto de las tasas (1 y 2 Mbps) no se han transmitido paquetes porque no ha sido necesario reducir la tasa.

Tabla 3 Cálculo del tiempo de transmisión [3]

Destination	Bit-rate	Tries	Packets Ack'ed	Succ. Fails	Total TX Time	Avg TX Time	Lossless TX Time
00:05:4e:46:97:28	11	16	0	4	250404	∞	1873
00:05:4e:46:97:28	5.5	100	100	0	297600	2976	2976
00:05:4e:46:97:28	2	0	0	0	0	-	6834
00:05:4e:46:97:28	1	0	0	0	0	-	12995

Minimizando el tiempo de transmisión el algoritmo consigue aumentar el *throughput* además de reducir el número de variaciones en la tasa de transmisión utilizada. Tiene un mejor rendimiento que ARF, AARF y ONOE [4].

2.3.2. Algoritmos basados en SNR y RSSI

El control de tasa basado en medidas de señal como la SNR o RSSI es ampliamente utilizado en los sistemas avanzados como LTE, y ofrece un rendimiento superior. Se basan en que los paquetes enviados con un índice MCS determinado necesitan un SNR y RSSI mínimo para que los símbolos se reciban correctamente. En general son los más complejos ya que es necesario determinar cómo obtener la medida de señal para que tenga validez cuando se toma la decisión de cambiar de tasa de transmisión. En algunos casos es necesaria una realimentación (lazo cerrado) del otro extremo añadiendo complejidad al ser necesario enviar la medida de señal. Surgen varios algoritmos dependiendo donde y como obtienen las medidas de señal.

- ***Received Signal Strength Link Adaptation* (RSSLA)**

En [5] se plantea el siguiente algoritmo para adaptar la tasa de transmisión desde una STA a un AP basándose únicamente en medidas de potencia de la señal recibida (RSS) como indicador del canal. El artículo tiene en cuenta que el AP no puede enviar a la STA ninguna medida de señal según el estándar Wi-Fi que utiliza (802.11b), por lo que se basa en medidas locales del STA. La STA considera la potencia de transmisión de los paquetes del AP conocida y fija según el estándar, y mide la potencia de recepción de los paquetes recibidos (enviados por el AP) para estimar el canal. De esta manera puede conocer si existen variaciones en el canal. Asume para simplificar el algoritmo que el canal es simétrico (aunque puede no serlo) y utiliza la estimación del enlace desde el AP a la STA para estimar el enlace desde la STA al AP.

De esta manera, se analizan todas las tramas que la STA escucha y se guarda su RSS de transmisión (conocido y fijo por el estándar) y de trama (medido por la STA). Este algoritmo utiliza los RSS guardados para escoger la tasa de transmisión y los compara con umbrales para aumentar o reducir la tasa.

Se ajustan las variaciones temporales del RSS de manera adaptativa con el siguiente filtro paso bajo, donde rss es el RSS actual y RSS_{n-1} es el RSS adaptado anterior.

$$RSS_n = (1 - a) * R_{n-1} + a * rss \quad \text{(Ec.1)}$$

- ***Beacon Assisted Rate Adaptation*** **(BARA)**

Es un algoritmo de adaptación de tasa para estaciones [6]. Sigue el mismo planteamiento que el algoritmo anterior pero obteniendo únicamente el RSS de las tramas *beacon* que envía el AP y que recibe la STA. Esto es debido a que analizar todas las tramas que escucha la estación es costoso. Por ello solo utiliza el RSS de los *beacons* enviados por el AP cada 100 ms reduciendo el coste computacional de analizar todas las tramas de datos del AP. BARA es adecuado para adaptar el tráfico *uplink* desde los clientes al AP. Es eficiente energéticamente ya que los clientes pueden estar en modo de ahorro de energía mientras realizan la adaptación de tasa. Tras unos intervalos de *beacon* miden y guardan el RSS de los *beacon*, reciben y envían tráfico al AP y vuelven al modo de ahorro de energía.

- ***Receiver Based Auto Rate*** **(RBAR)**

Usa las tramas RTS/CTS de CSMA/CA. En este algoritmo el receptor envía el SNR que observa al transmisor para utilizarlo como indicador del canal. Se modifica el RTS/CTS para añadir más información (tasa, SNR, longitud del paquete). Una variante de este algoritmo consiste en tomar la decisión de cambiar la tasa en el transmisor en vez de que el receptor proponga una tasa. Un ejemplo de algoritmo teórico es *IdealWifi* que trata de maximizar el *throughput* y tener un BER menor a 10^{-5} [7].

El algoritmo trabaja de la siguiente manera:

- El transmisor escoge la mayor tasa posible y la envía en el RTS junto con la longitud del paquete.
- El receptor obtiene el SNR del RTS recibido, según el SNR propone una tasa de transmisión y la envía al transmisor junto con la longitud del paquete.
- Hay nodos que pueden no recibir la tasa que propone el receptor con el CTS así que el transmisor envía un mensaje extra llamado RSH (*Reservation SubHeader*).

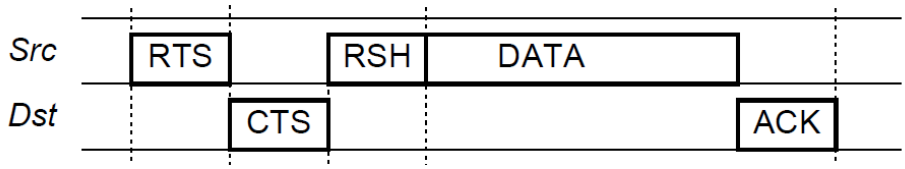


Figura 13 Modificación del envío de RTS/CTS añadiendo RSH (Reservation SubHeader)

2.3.3. Basados en la pérdida de tramas con diferenciación de colisiones

Generalmente las tramas se pueden perder por que la calidad del canal no es suficientemente buena para la tasa elegida o por colisiones. Si las tramas se pierden por colisiones no se debería modificar la tasa de transmisión ya que, como se ha comentado anteriormente, esto lo único que hace es empeorar la situación. Por tanto, interesa desarrollar algoritmo que identifiquen cuando una pérdida se produce debido a una colisión para no adaptar la tasa. Por tanto, el problema abordado por estos algoritmos es saber cómo llevar a cabo la diferenciación.

- ***Collision Aware Rate Adaptation* (CARA)**

Este algoritmo introduce las tramas RTS/CTS tras la pérdida de un paquete para indicar a otros dispositivos que va a transmitir para, de esta manera, evitar posibles colisiones.

El algoritmo trabaja de la siguiente manera:

- Se envía una trama de datos utilizando RTS/CTS.
- Si la trama se envía correctamente, se mantiene la tasa y no se utiliza RTS/CTS en la siguiente transmisión.
- Si la trama se pierde, se reduce la tasa y se vuelve a utilizar RTS/CTS en la siguiente transmisión.

En la Fig.14 se pueden visualizar la máquina de estados (esperando nuevo paquete de datos, transmisión de RTS y transmisión de datos) que se corresponden con el algoritmo definido anteriormente. Al enviar las tramas RTS/CTS tras una perdida se asegura de que la siguiente trama no se pueda perder por interferencias. Además, al desactivar RTS/CTS tras una transmisión correcta consigue eliminar la carga no útil que añaden estas tramas en el canal cuando no es necesario.

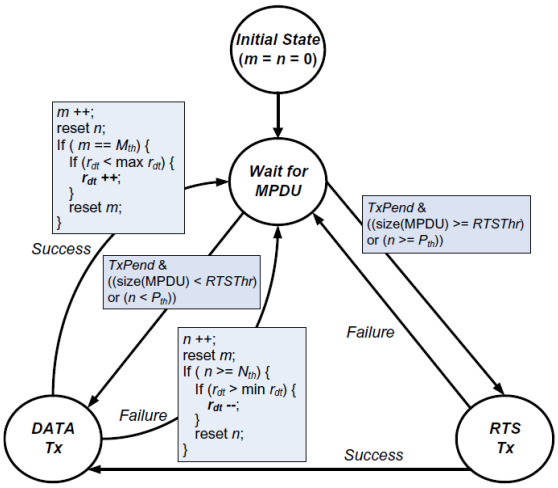


Figura 14 Algoritmo CARA

- ***Robust Rate Adaptation Algorithm* (RRAA)**

Utiliza las tramas RTS/CTS después de la pérdida de una trama. Usa un porcentaje de tramas perdidas en una ventana temporal y RTS adaptativo para reducir el *overhead*. Tiene una variable RTSw, que es el número de tramas a partir de una perdida que se van a transmitir con RTS/CTS. Esta variable se aumenta sumando y se reduce exponencialmente.

El funcionamiento de RRAA es el siguiente:

- Para modificar la tasa se calcula el porcentaje de tramas perdidas en una ventana (20-100ms).
- Si el porcentaje de perdidas < P1, se incrementa la tasa.
- Si el porcentaje de perdidas > P2, se reduce la tasa.
- En otros casos no se modifica la tasa.
- Si una trama se pierde sin RTS, se aumenta el contador RTSw.
- Si la trama se transmite correctamente (con o sin RTS), se reduce exponencialmente el RTSw=RTSw/2.

Mejora a ARF, AARF, *SampleRate* y RRAA si existe movilidad en la transmisión [4].

2.3.4. Algoritmos basados en redes neuronales: *Deep Reinforcement Learning*

En [8] se presenta un algoritmo que combina el *reinforcement learning* (RL) y el *deep learning* [9]. En el aprendizaje por refuerzo (RL) se utiliza un agente computacional que determina que acción se debe tomar en un entorno para maximizar la recompensa. Por ejemplo, tomando un planteamiento similar al algoritmo ONOE, se podrían tomar del entorno los datos de número de paquetes enviados (*num*), número de paquetes con éxito (*succ*) y número de retransmisiones (*retries*). Se podrían comparar con los anteriores valores, ver si han empeorado o mejorado respecto la última ventana y tomar una decisión sobre la red neuronal. Aprende de los errores, en este caso, si empeora el número de paquetes con éxito o *retries* modifica los parámetros de la red para que en otros casos similares a este reduzca la tasa, es decir, va aprendiendo de diferentes situaciones que pueden aparecer. De esta manera se puede adaptar a medida a ese entorno.

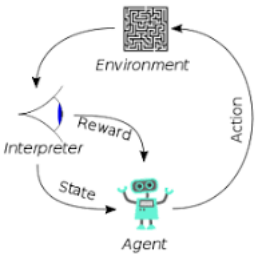


Figura 15 Aprendizaje por refuerzo (Deep Reinforcement Learning)

Los pasos para formular el problema de RL se pueden encontrar en el siguiente enlace [10], donde el autor aborda una aplicación no relacionada con control de tasa, aunque los principios del problema son los mismos. Siguiendo los pasos del artículo se definen los siguientes principios.

- Entorno: Mundo real donde opera el agente. El entorno de envío de paquetes entre AP y STA.
- Estado(S): SNR promedio observada en el intervalo de tiempo de transmisión.
- Recompensa (R): Un valor numérico que indica si ha empeorado o mejorado el rendimiento. Se utiliza como recompensa el producto entre la tasa de tramas con éxito (FER) y la MCS normalizada de las tramas transmitidas en el intervalo de tiempo de transmisión.

$$reward = \frac{MCS_n}{MCS_7}\ x\ FER, n\ \epsilon\ [0,1,\dots,7] \qquad \text{(Ec.2)}$$

- Acción: Utilizar uno de los índices MCS desde 0 a 15 en la próxima transmisión.

2.3.5. Conclusiones

Tras presentar los diferentes algoritmos, se resumen los pros y contras de cada uno de ellos teniendo en cuenta, además, el propósito de este trabajo que pasa por implementación de estos algoritmos en un entorno experimental.

- Los algoritmos basados en RSSI o SNR son utilizados en sistemas más complejos (por ejemplo, LTE) y en general ofrecen mejores resultados. Realizar este tipo de medidas puede ser complejo y debe tenerse en cuenta donde se realiza la medida (transmisor o receptor), que promediado de la medida se realiza y el retardo que puede suponer enviar la medida al transmisor si se realiza en el receptor. Todos estos factores pueden hacer que la medida de SNR o RSSI estimada no tenga validez cuando se tome la decisión del cambio de tasa.
- Los algoritmos basados en la pérdida de tramas tienen diferentes rendimientos según su diseño. Se necesitan estadísticas como son el número de tramas originales o las tramas enviadas con éxito y fallidas, factibles de obtener en un escenario experimental.

- Los algoritmos que diferencian colisiones posibilitan poder hacer frente a las posibles interferencias y, por ello, son utilizados en tarjetas Wi-Fi comerciales. Están basados en el envío de tramas RTS/CTS, lo que los hace más complicados en la obtención y gestión de estadísticas, sobre todo, en su seguimiento temporal.
- Los algoritmos basados en redes neuronales se presentan como una nueva opción que puede adaptarse a los entornos ajustando los parámetros de la red, pero su implementación para trabajar en tiempo real y con un desarrollo ágil puede ser complicada.

Los algoritmos basados en la pérdida de tramas, por la disponibilidad y facilidad para obtener las estadísticas, serían los que más ajustan a las necesidades de este trabajo. La posibilidad de variar parámetros asequibles como son el umbral de reintentos y créditos o la ventana temporal de actualización permitirá a los algoritmos adaptarse a las diferentes situaciones en una transmisión y ajustar la eficiencia en distintas situaciones (estático, movimiento, presencia de interferencias…). Además, plantean la posibilidad de confrontar modos de funcionamiento distintos (trabajar paquete a paquete o con ventanas de paquetes) que permitirían afrontar situaciones muy diversas y enriquecedoras.

3. Diseño del entorno experimental

En este capítulo se presenta el escenario diseñado para el entorno experimental, que incluye los equipos y dispositivos a utilizar. Como paso previo, se introducen los aspectos más relevantes relativos al procedimiento para llevar a cabo una inyección controlada de tramas que permita una evaluación precisa de lo que ocurre en la interfaz radio. Se detallan las distintas opciones y variantes tanto software como hardware. Asimismo, se presentan las posibles alternativas a considerar para la obtención de las métricas que deben dar soporte a la implementación y evaluación de los algoritmos de control de tasa. Seleccionadas los mejores enfoques conforme a las restricciones prácticas en las que se desarrolla el proyecto, se describe el entorno experimental propuesto. Tras ello, se introduce la estructura de programa diseñada y que gobierna el entorno diseñado. Finalmente, se profundiza en los 3 bloques software que configuran nuestro entorno: transmisión de tramas o inyección, monitorización de red o captura de tramas y control de tasa o implementación de los algoritmos.

3.1. Aspectos técnicos previos a considerar

3.1.1. Inyección de tramas

En el contexto investigador sobre redes IEEE 802.11, resulta extremadamente valioso poder inyectar/transmitir cualquier trama controlando de forma manual sus características de cara a observar lo que ocurre en la interfaz radio y los resultados sobre la operación del sistema, sin necesidad de estar autenticado en ninguna estación. Para ello se requiere disponer de un *kernel* de Linux. De hecho, un gran número de experimentos Wi-Fi se realizan utilizando Linux con tarjetas de red comerciales (*commercial off-the-shelf network cards*). El ejemplo más llamativo de ello es la investigación en ciberseguridad (*suite* de herramientas *aircrack-ng*), donde se envían tramas inesperadas o malformadas a un dispositivo bajo prueba [11]. En los últimos años, otros ámbitos como la redes SDWN por su programabilidad hacen uso de esta inyección de tramas para crear entornos experimentales (pruebas de concepto) que permitan aportar funcionalidades inteligentes y avanzadas a las redes Wi-Fi. Lógicamente, estos entornos necesitan aportar inicialmente funciones básicas como el escaneo, la asociación, el control de potencia o el control de tasa.

¿Cómo se puede llevar a cabo la inyección de tramas? Para responder a esta pregunta es necesario conocer los modos de operación que ofrece una tarjeta Wi-Fi. Estos modos son los siguientes:

- ***Managed mode***

  Este es el funcionamiento para un usuario normal (por defecto) que utiliza una estación Wi-Fi. La STA se conecta a un AP que le proporciona conectividad. De esta manera una estación forma parte de una red Wi-Fi en modo infraestructura.

- ***AP/Master mode***

  Este modo permite al dispositivo comportarse como un punto de acceso (AP) y trabajar en modo infraestructura.

- **Ad-hoc (IBSS, *Independent Basic Set*)**

  Las estaciones Wi-Fi se comunican entre ellas directamente sin necesidad de un punto de acceso (AP).

- ***Monitor mode***

  El modo monitor también llamado RFMON (*Radio Frecuency MONitor*) [12] permite inyectar y monitorizar el tráfico recibido en un canal inalámbrico a través de un controlador de interfaz de red inalámbrica (WNIC). Para ello no es necesario asociarse con un AP ni formar parte de una red ad-hoc. Dentro del modo monitor se distinguen dos opciones [11]:

  - ***Pure monitor Mode:*** En este modo la tarjeta de red se usa únicamente en modo monitor.
  - ***Mixed monitor mode:*** En este modo la tarjeta de red es utilizada por una o más interfaces (virtuales) en modo cliente o AP y también por una o más interfaces (virtuales) en modo monitor. El término interfaz(es) no monitor se refiere a las interfaces que funcionan en modo cliente o AP.

En general, los *scripts* Wi-Fi desarrollados en el espacio de usuario de Linux suelen utilizar el modo de operación monitor para tener control sobre las tramas completas (en bruto o *raw frames*) que se transmiten por la interfaz de red inalámbrica. Sin embargo, las tarjetas de red comerciales suelen ser cajas negras sobre las que no se tiene un control total, lo que significa que no hay garantía de que las tramas inyectadas (*raw frames*) se transmitan según lo previsto. Esto significa que no está claro cómo de fiables son estos *scripts* en la práctica. Dependiendo de la configuración experimental, pueden surgir distintos problemas al inyectar tramas Wi-Fi, y estos problemas pueden hacer que los scripts se comporten mal [11]. Los principales problemas suelen ser que las cabeceras son sobrescritas antes de la transmisión, las tramas se reordenan de manera diferente a como se ha indicado desde el espacio de usuario o que algunas tramas puedan perderse. Todos estos problemas deben tenerse en cuenta al diseñar el escenario.

3.1.2. Drivers, tarjetas Wi-Fi y Linux

El *kernel* de Linux tiene soporte integrado para la inyección de tramas sin procesar (*raw frames*). Desde el espacio de usuario (*scripts* de usuario) se utiliza la interfaz *nl80211* para enviar comandos que son gestionados por el módulo *cfg80211* del *kernel*. Profundizando en esto, podemos decir que existen dos tipos de *drivers* Wi-Fi en Linux según donde y como implementan la capa MAC [11].

- ***Full*-MAC:** Implementa la entidad de gestión de la subcapa MAC (*MAC Sublayer Management Entity*, MLME) en *hardware* o *firmware*. MLME se encarga de las funciones iniciales de la red como son la autenticación y asociación de dispositivos, envío de *beacons* o el escaneo de redes. Además, también implementa en controladores las funciones de SME (*Station Management Entity*) que se encarga de interactuar entre la capa MLME y la capa PLME (*Physical SubLayer Management Entity*), que gestiona la coordinación DCF, por ejemplo. Las tramas inyectadas se pasan directamente desde el módulo *cfg80211* al *driver* y éste se encarga de las cabeceras *Radiotap* (comentadas en apartados posteriores).

- ***Soft*-MAC:** Como su nombre indica implementa parte de la capa MAC en *software*. Se apoya en el módulo del *kernel* denominado *mac80211* para implementar partes del MLME en *software*. Las tramas inyectadas se pasan desde el módulo *cfg80211* al módulo *mac80211* y este módulo se encarga de las cabeceras *Radiotap*.

Los *drivers Full*-MAC son más eficientes ya que realizan menos llamadas a módulos superiores e implementan ciertas funcionalidades directamente en *hardware* o *firmware*. El *driver* Soft-MAC se apoya en el módulo *mac80211* para interactuar con el módulo *cfg80211* por lo que es menos eficiente

y más lento. La Fig.16 [7] muestra las relaciones entre los distintos módulos comentados anteriormente (*mac80211*, *cfg0211* y *nl80211*) y como se utilizan según si el driver es *Full*-MAC o *Soft*-MAC. Además, también se pueden observar llamadas o *callbacks* que se realizan a registros. En la parte superior se puede ver la frontera entre espacio de usuario y espacio de *kernel*. La API *nl80211* nos permite interactuar con el *kernel* desde el espacio de usuario gracias a los comandos como *ifconfig*, *iwconfig*, *iw*, entre otros.

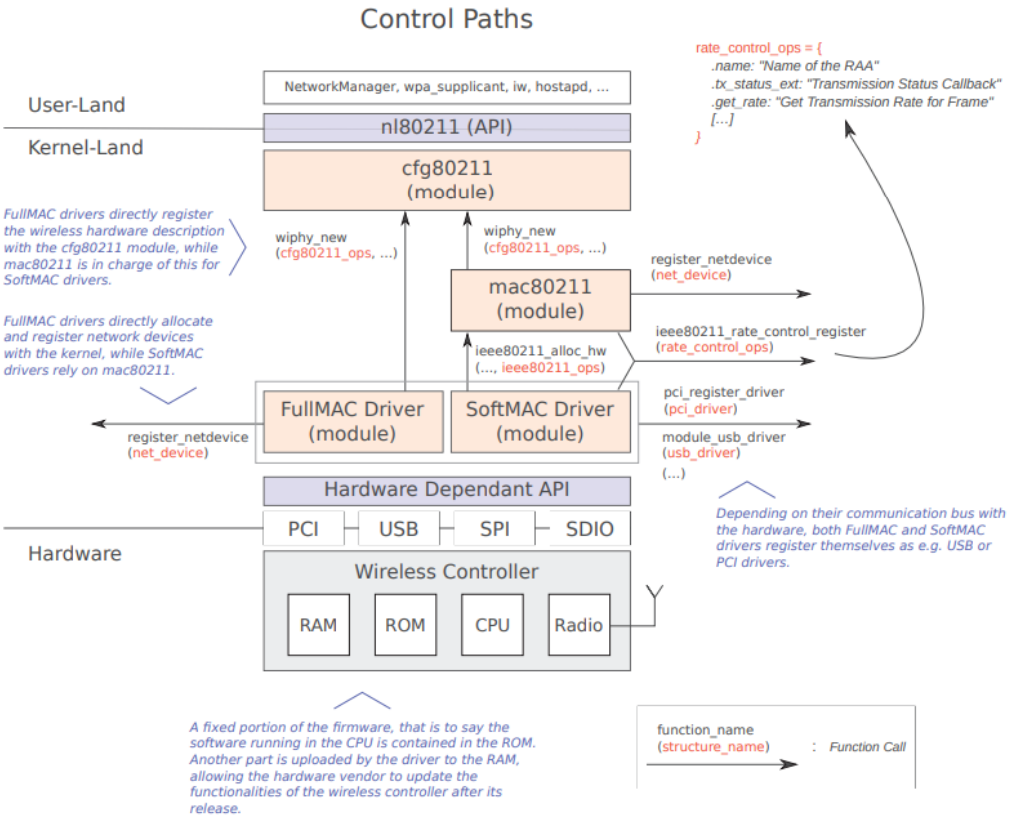


Figura 16 Estructura de los drivers Full-MAC y Soft-MAC [7]

Las tarjetas USB Wi-Fi pueden tener el *driver* implementado en el *kernel* (*in-kernel*) o fuera de él (*out-of-kernel*). Los *drivers in-kernel* están integrados con el *kernel* de Linux y su ejecución es más rápida mientras que los drivers *out-of-kernel* trabajan en el espacio de usuario y necesitan más pasos para comunicarse con el *kernel*. Los *drivers out-of-kernel* son instalados fuera del sistema operativo Linux y llevan una evolución distinta al mismo mientras que los dispositivos con un driver *in-kernel* pueden ser elegidos en la propia instalación del sistema operativo Linux y evolucionan con él. Se puede encontrar una clasificación de *chipsets* Wi-Fi donde se especifica el tipo de *driver* que utilizan en el enlace [14].

No todas las tarjetas inalámbricas comerciales y, en concreto, su *driver,* permiten la inyección de tramas. Hemos de trabajar con *drivers* abiertos que lo posibiliten. Ejemplo de ello son las tarjetas *TP-Link TLWN722N* (*atheros, driver in-kernel)* y *Alfa AWUS036ACM* (*rtl8812au, driver out-kernel*). Esta última fue la primera tarjeta comercial (su *driver*) que permitió la inyección de tramas hasta 802.11ac. Además, es capaz de soportar monitor activo o capacidad de enviar ACKs al recibir datos en modo monitor. Esta es una característica clave para el escenario de pruebas experimental en el que se quiere realizar un control de tasa de transmisión. La versión del *kernel* de Linux utilizada debe ser compatible con la versión del *driver* para esta tarjeta. Esto ocurre, por ejemplo, en la tarjeta TP-Link ya que su *driver* es *in-kernel* y se va actualizando junto al *kernel*.

3.1.3. *Radiotap* y *Scapy*

Normalmente, el sistema operativo rellena todos los campos de una trama Wi-Fi. Sin embargo, y como hemos comentado, algunos sistemas operativos también admiten la inyección de tramas 802.11 sin procesar en las que el usuario tiene control total sobre el contenido de la trama de la trama. Al inyectar este tipo de tramas 802.11 sin procesar, también deben proporcionarse varios parámetros como la tasa de bits de transmisión, canal de transmisión, ancho de banda del canal, si se debe retransmitir la trama caso de no recibir ACK, si utiliza métodos de evitar colisiones tales como el *CTS-to-self*, etc. Los programas del espacio de usuario especifican estos parámetros anteponiendo a la trama Wi-Fi una cabecera denominada *Radiotap* [13]. Esta cabecera contiene información de control para especificar exactamente cómo se ha de transmitir la trama [14]. Permiten fijar la tasa de transmisión (MCS), el ancho de banda y el intervalo de guarda, entre otros parámetros. Asimismo, el *kernel* también antepone a las tramas recibidas esta cabecera *Radiotap* para informar con una serie de metadatos relacionados con la trama recibida, información tal como la tasa en bits por segundo, la intensidad de la señal, etc. Estas cabeceras no se transmiten por la red si no que sirven para ordenar a la tarjeta de red como transmitir y, por otro lado, ver como se han recibido las tramas por parte de dicha tarjeta. Por ejemplo, en el caso de inyectar tráfico en 802.11n, la cabecera de *Radiotap* utiliza el campo 19 (MCS) para indicar valores como el índice MCS, *el* intervalo de guarda (*short or long*), el ancho de banda (20, 40, 20L, 20U), el tipo de FEC (BCC, LDPC) o el número de STBCs. Todo esto permite que *Radiotap* sea usado por herramientas especializadas en inyección de tramas (*aircrack-ng*) y de monitorización de tráfico (*tcpdump*) así como programas estándar como *hostap*. Para el objetivo de este trabajo, permite controlar de manera sencilla la tasa de transmisión utilizada y poder implementar algoritmos de control de tasa.

La inyección de tramas en este trabajo requiere poder crear las tramas en crudo o *raw*, es decir, desde cero especificando los diferentes campos de la trama (direcciones origen y destino, datos, tipo y subtipo de tramas…) ilustrados en la Fig.11. Esto requiere la utilización de *scripts* o programas en el espacio de usuario que permitan hacerlo. Para ello, hay que elegir en primer lugar el lenguaje de programación y las librerías. *Python* es elección que presenta las mejores condiciones buscadas como son su facilidad de aprendizaje, rapidez de desarrollo y efectividad para crear un entorno experimental. Además, aporta una gran variedad de librerías de apoyo disponibles junto con una comunidad de desarrolladores muy activa. Destacar entre este apoyo todo lo relacionado con las redes como puede ser, por ejemplo, la librería *Scapy*. Esta librería permite crear tramas Wi-Fi con la función *Dot11*. Creada la cabecera 802.11, se añaden los niveles superiores y, finalmente, se concatenan los datos propiamente dichos, unos datos de relleno dependiendo de la longitud del paquete deseada en la prueba. Este entramado debe ir precedido de la cabecera *Radiotap* con las condiciones deseadas.

Una vez definida la trama Wi-Fi con las diferentes cabeceras el siguiente paso es inyectarla a través de la tarjeta de red. *Scapy* permite inyectar un paquete por una interfaz Wi-Fi mediante funciones como *sendp*. En siguientes apartados se incluirán detalles respecto al desarrollo software efectuado.

3.1.4. Obtención de métricas y/o estadísticas

Los algoritmos de control de tasa necesitan tener un seguimiento sobre lo que está ocurriendo con los paquetes enviados para poder adaptar la tasa en consecuencia. Como se ha visto, requieren apoyarse en parámetros de nivel físico, por ejemplo, potencia recibida en la antena, o de nivel de enlace tales como número de paquetes número de paquetes transmitidos originalmente, número de paquetes transmitidos con éxito, el número de reintentos que se han realizado, etc. La decisión de los algoritmos

a implementar en este proyecto viene condicionada por la viabilidad del acceso a los distintos tipos de métricas y/o estadísticas disponibles.

Se plantean varias opciones para la obtención de estas métricas y/o estadísticas.

- **Cabeceras *Radiotap*:** Estas cabeceras nos permiten obtener diferentes métricas al recibir una trama. De esta manera se puede saber el número de reintentos que una trama ha utilizado para transmitirse con el campo 17 (*Data retries*) y si la transmisión de un paquete ha fallado por excesivos reintentos con el campo 15 (*TX-Flags*). El número de paquetes original debería obtenerse contando los paquetes de datos que no son reintentos (recuento de reintentos a 0). El nivel de señal en la antena o RSSI en dBm se encuentra en el campo 5 llamado *Antenna signal*. Lógicamente, se necesita capturar y filtrar estos paquetes para poder leer su cabecera *Radiotap*.

- **Debug driver[1]:** Una fuente de estadísticas son los propios ficheros de *debug* del driver donde se escriben a partir de la información de la tarjeta de red el número de paquetes transmitidos, el número de reintentos y el número de paquetes fallidos, entre otros parámetros. Estos ficheros pueden ser leídos vía línea de comandos específicos desde sistema operativo, habiendo 3 fuentes para obtener información de estadísticas [15]. La primera fuente son las *standard interface statistics*, accediendo a la estructura *struct rtnl_link_stats64* mediante el siguiente comando:

```
$ ip -s -s link show dev ens4u1u1
6: ens4u1u1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500
  link/ether 48:2a:e3:4c:b1:d1 brd ff:ff:ff:ff:ff:ff
  RX: bytes  packets  errors  dropped overrun mcast
  74327665117 69016965 0       0       0       0
  RX errors: length   crc     frame   fifo    missed
             0        0       0       0       0
  TX: bytes  packets  errors  dropped carrier collsns
  21405556176 44608960 0       0       0       0
  TX errors: aborted  fifo   window heartbeat transns
             0        0       0       0       128
  altname enp58s0u1u1
```

Figura 17 Estadísticas mediante la estructura rtnl_link_stats64

La segunda fuente son las *protocol-specific statistics* accediendo vía los propios interfaces. Ejemplo de ello es el comando *ethtool*, como se puede ver a continuación.

```
$ ethtool -S eth0 --groups eth-phy eth-mac eth-ctrl rmon
Stats for eth0:
eth-phy-SymbolErrorDuringCarrier: 0
eth-mac-FramesTransmittedOK: 1
eth-mac-FrameTooLongErrors: 1
eth-ctrl-MACControlFramesTransmitted: 1
eth-ctrl-MACControlFramesReceived: 0
```

Figura 18 Estadísticas específicas del protocolo con ethtool

Por último, la tercera fuente son las *driver-defined statistics* que se pueden ver utilizando también *ethtool*.

```
$ ethtool -S ens4u1u1
NIC statistics:
   tx_single_collisions: 0
   tx_multi_collisions: 0
```

Figura 19 Estadísticas definidas por el driver con ethtool

[1] Se debe tener en cuenta el tiempo de acceso a ficheros si planteamos un acceso desde programa de usuario.

- **Monitorización de red:** Esta opción propone monitorizar la red con herramientas como *tcpdump* o *scripts* basados en librerías como *Scapy* y, a partir de los paquetes capturados, obtener métricas y calcular estadísticas. Se puede aplicar un primer filtrado para saber si un paquete recibido corresponde a datos o ACKs, conocer si un paquete es un reintento con el campo *retry* de la trama MAC. En cualquier caso, es preciso tener en cuenta las relaciones y secuencialidad de los paquetes conforme a la implementación de los procesos a nivel MAC.

La opción de cabeceras *Radiotap* permite acceder a un gran número de parámetros, pero no todos los drivers implementan el acceso a ciertos campos de la cabecera Radiotap por ser de carácter opcional (campo 17, *data retries*). En el caso de ficheros de *debug*, sólo se puede acceder a ciertos parámetros y no parece viable discernir entre distintas comunicaciones con distintos dispositivos Wi-Fi desde un único equipo (un AP con distintas STAs). La obtención de estadísticas vía monitorización de red (por ejemplo, vía *Scapy*) es más compleja que las anteriores, pero nos da más libertad para escoger los campos que necesitamos de las tramas. Su implementación sería similar a la de *Radiotap* en cuanto a que necesitamos también monitorizar la red y procesar las distintas tramas, eso sí, de un modo más complejo que *Radiotap*.

Tras lo expuesto, se decide implementar la obtención de métricas y estadísticas con *Scapy* (monitorización de red) ya la utilización de ficheros de debug del driver solo nos dejaría acceder a unos pocos e insuficientes parámetros y la opción de cabeceras *Radiotap* es dependiente de la implementación del driver de la tarjeta de red.

3.2. Definición del entorno experimental

Tras el análisis realizado anteriormente se procede a diseñar el escenario del entorno experimental. Los elementos hardware y software utilizados en el escenario son los siguientes:

- PC Linux con versión de OS de Linux Debian 8.11 con versión de Kernel 4.19.28.
- Tarjetas USB Wi-FI (Alfa AWUS036ACH).
- Driver rtl8812au v5.2.20
- Python (versión 3.4).
- Librería Scapy (versión 2.5.0).

Para los nodos Wi-Fi (AP, STAs), se utilizarán las tarjetas de red *Alfa* mencionadas con el *driver rtl8812au* (*out-of-kernel*) por su capacidad de inyección, siendo la versión del *driver* compatible con el *kernel* en ese momento disponible[2]. La inyección y monitorización de tramas se realizará con el lenguaje de programación *Python* y la librería *Scapy* utilizando la cabecera *Radiotap*, tal como se ha comentado en apartados anteriores. Estas herramientas permitirán obtener vía monitorización de red las métricas y/o estadísticas necesarias para los algoritmos de control de tasa utilizados. La monitorización ha sido la opción elegida para obtener estadísticas ya que los drivers mencionados no son capaces de proporcionar campos opcionales del *Radiotap* ni la información de *debug* necesitada.

El escenario de monitorización de la red se construye tomando como base un único PC bajo *Linux*, donde se conectarán todos los dispositivos involucrados en la transmisión y recepción de tramas. El escenario básico está compuesto por una tarjeta de red que actúa como AP y transmite paquetes a otra tarjeta que actúa como STA y recibe los paquetes, estando los dos conectados a un punto central que los controla (equipo PC).

[2] Durante la realización de este trabajo se buscaron y probaron versiones más actualizadas de ambos (*kernel* y *driver*), estableciendo la relación entre ellos existente [9].

Con respecto a la monitorización de los parámetros necesarios para la implementación de los algoritmos de control de tasa, teniendo en cuenta que se opta por algoritmos en lazo abierto como ONOE, ARF y AARF, se planteó inicialmente extraer las métricas a partir de la monitorización de la red en el AP. Estos algoritmos necesitan métricas como pueden ser el número de paquetes enviados, las retransmisiones o los paquetes con éxito y fallidos. Para decidir si la transmisión es correcta o no, basta con monitorizar la recepción de ACKs. A priori, el AP es el elemento adecuado para acceder a toda esta información, sin embargo, la realidad es que hay una serie de limitaciones prácticas que no lo hacen posible. El AP no captura todos los paquetes en la antena. Captura los paquetes recibidos de la red pero no los transmitidos en la interfaz radio. En cuanto a estos últimos, la aplicación *tcpdump* muestra los paquetes transmitidos por el programa de espacio de usuario antes de llegar al *driver*. Por ejemplo, si inyectamos un paquete y es transmitido hacia la red, en caso de pérdida, es el *driver* el que decide la retransmisión y ésta no puede ser capturada y, por tanto, monitorizada. Por tanto, no es posible contabilizar directamente el número de paquetes originales y retransmisiones efectuadas.

La monitorización en el receptor o STA no tiene sentido salvo que se hubiera optado por la implementación de mecanismos en lazo cerrado. Vistas las limitaciones prácticas y tratándose de una plataforma de evaluación experimental, se decidió (sin pérdida de validez para la evaluación de prestaciones) incorporar un tercer dispositivo 'espía' cuyo único objetivo es monitorizar la red para obtener estadísticas. El dispositivo espía será una tarjeta de red que se coloca cerca del AP procurando, idealmente, unas condiciones de recepción de transmisiones recibidas desde los STAs idénticas a las del AP. El dispositivo espía debe escuchar los mismos paquetes que el AP para que las estadísticas obtenidas sean correctas. Tal y como se comentará más adelante, AP y dispositivo espía es imposible que estén en idénticas condiciones por lo que hay que tomar una serie de medidas de precaución y ajuste para que el sistema opere de forma correcta.

Tomando las consideraciones anteriores, se presenta el escenario diseñado para el envío de los paquetes en la Fig.20, donde el AP envía paquetes a la STA y el espía monitoriza la transmisión para obtener las estadísticas. Tal y como se ha explicado, se dispone de un PC Linux donde se conectan y se controlan todos los dispositivos Wi-Fi de manera centralizada. Se decidió centralizar este punto (equipo PC) donde se ejecutan los *scripts* ya que es necesario tener una sincronización entre los diferentes bloques que se requieren en el diseño funcional del escenario (transmisión, monitorización, decisión del algoritmo de adaptación de tasa), además de facilitar la realización de las pruebas y la obtención de *logs*. El punto de acceso (AP) se encuentra fijo junto al PC y el espía, y la STA puede variar su posición dependiendo de la prueba, ya que se dispone de un cable USB de varios metros. En las próximas figuras que muestran escenarios se omiten el ordenador y el espía, aunque siguen presentes.

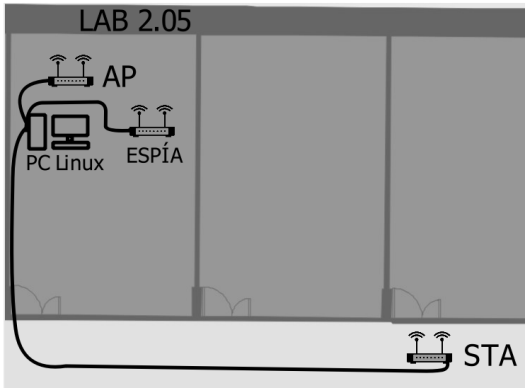


Figura 20 Escenario básico: PC Linux, punto de acceso (AP), espía y estación Wi-Fi (STA)

3.3. Diseño y estructura del programa

La idea es crear módulos separados con funciones diferenciadas para agilizar el desarrollo y aislar fuentes de error al añadir funcionalidades en un bloque. Además, se busca probar diferentes algoritmos sin modificar excesivamente la estructura. Por ello, se plantea un diseño en bloques del programa, dividido en tres partes claramente diferenciadas, sincronizadas y conectadas entre sí: el transmisor o inyector de tramas, el monitor de red o espía y control de tasa o implementación de los algoritmos. La Fig.21 muestra este diseño en bloques formado por 3 ficheros diferentes en lenguaje de programación *Python* que se ejecutan en paralelo como 3 hilos en el sistema. El primero se encarga de la inyección de paquetes (transmisor.py), el segundo ejecuta el algoritmo seleccionado (algoritmo.py) y el tercero se encarga de obtener las estadísticas de los paquetes (espía.py).

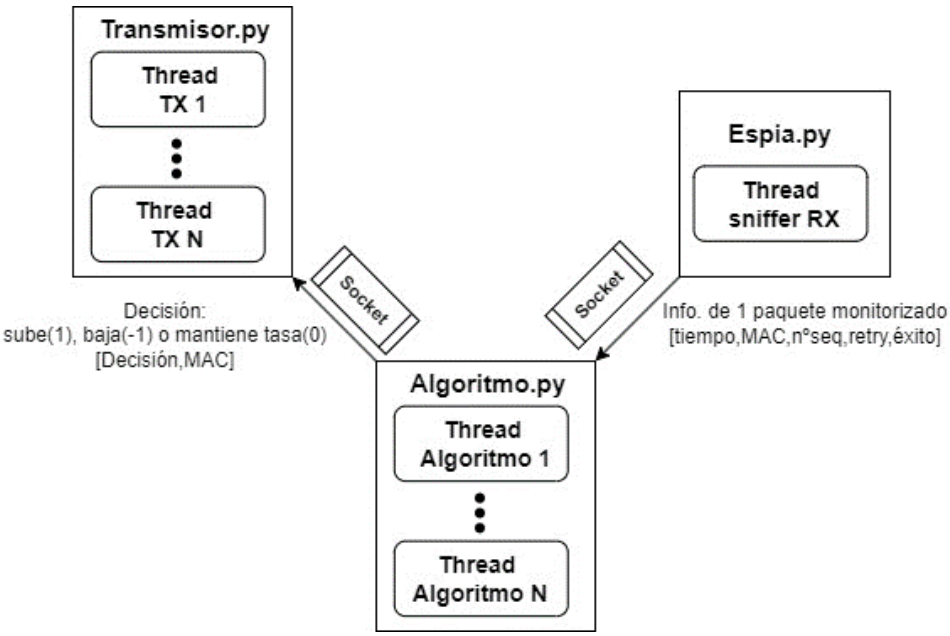


Figura 21 Estructura del programa

Hay que tener en cuenta que la solución debe operar en tiempo real por lo que hay que tener en cuenta los siguientes aspectos:

- **Tiempo de ejecución**: La ejecución del programa no debe ralentizar al algoritmo de control de tasa. Esto puede ser complicado en los algoritmos que trabajan paquete a paquete. Por ejemplo, si la tasa de paquetes es 100 paq/s el algoritmo debe responder en menos de 10 ms.
- **Obtención de estadísticas**: Las estadísticas de los paquetes se deben obtener lo suficientemente rápido como para no ralentizar el algoritmo.
- **Varios usuarios**: Si se pretende controlar la tasa de varias estaciones o usuarios la ejecución del algoritmo por parte de un usuario no debe ralentizar a otros usuarios. Para ello se lanzan varios *threads* o hilos que ejecutan el algoritmo en paralelo dependiendo del número de usuarios.
- **Memoria:** Hay que gestionar la memoria de las variables que se vayan almacenando durante el tiempo para no agotar el espacio de memoria.
- **Sincronización:** Los tres bloques trabajan en el mismo punto y ejecutan varios hilos, es importante la sincronización, siendo necesarios semáforos cuando aparecen variables compartidas.

Teniendo en cuenta todos estos aspectos además de las consideraciones de diseño analizadas en los apartados anteriores, se propuso la estructura de programa que se muestra en la Fig.21.

3.4. Transmisión de tramas: inyección de paquetes

Tras analizar las diferentes opciones en apartados anteriores, la inyección de paquetes se realiza mediante la librería *Scapy* (función *sendp*). Inicialmente, se configuran vía línea de comandos las tarjetas Wi-Fi a utilizar (ancho de banda, modo monitor…) y después hay que crear los paquetes e inyectarlos. Estas acciones exigen todo un trabajo previo arduo de pruebas que validen la correcta inyección de tramas con las distintas opciones existentes de parámetros de *Radiotap* y funciones de *Scapy*. En principio el *driver* de la tarjeta Wi-Fi puede indicar que soporta la transmisión con un parámetro al pertenecer a una versión del estándar 802.11, pero se debe conocer la capacidad real soportada. Dependiendo del fabricante algunas funcionalidades no son soportadas o simplemente no existe el *hardware* o *software* para implementarlas. Un ejemplo son los *spatial streams*, donde una tarjeta que soporta 802.11n en teoría puede tener hasta 4. Sin embargo, el número real estará limitado por el número real de antenas que tenga (por ejemplo, en nuestro caso 2). Además, la capacidad para implementar multiplexación espacial con ese número de antenas dependerá también de la disponibilidad de esa misma configuración en el receptor y, no solo eso, además, será necesario que la disposición de equipo transmisor y receptor sea tal que los caminos entre recorridos por la señal entre las antenas estén incorrelados. Por tanto, no basta con configurar la tarjeta vía comando, sino que es preciso contrastar si la configuración indicada es factible en el entorno experimental desplegado.

Para el desarrollo del proyecto se escogió el estándar IEEE802.11n para la transmisión. El objetivo ha sido evitar la complejidad de las versiones superiores como por ejemplo 802.11ac que incorpora el mecanismo Block-ACK (no se recibe un ACK por trama sino por bloque de tramas) y que obligaría a contemplar algoritmos de adaptación de tasa más complejos. Es preciso recalcar que el entorno trabaja con una mera inyección, sin asociación ni autenticación ni ningún tipo de funciones necesarias para trabajar en modo infraestructura. Los principales parámetros para 802.11n son los siguientes[3].

- **Guard_Interval** (long GI ‘0’, short GI ‘1’)
- **MCS_bandwidth** (20 MHz ‘0’, 40 MHz ‘1’, 20 MHz L ‘2’, 20 MHz U ‘3’)
- **HT_format** (mixed ‘0’, greenfield ‘1’)
- **MCS_index** (valor numérico del MCS: 0,1,…,23)

Primero se debe indicar que está presente el campo MCS (*present*=”MCS”), dentro del MCS hay que indicar los campos o bits conocidos (*knownMCS*) que son los parámetros anteriores, y por último indicar uno a uno el valor de estos campos (*guard_interval*=0, *MCs_index*=0,…).

Respecto a la cabecera de la trama MAC (Dot11), los parámetros principales son.

- **type** (Datos ‘2’, ACK ‘1’)
- **subtype** (Datos ‘0’)
- **Direcciones MAC de los equipos inalámbricos del escenario** (addr1:receptor, addr2:transmisor)

Después se añaden las cabeceras de nivel superior y un relleno para poder trabajar con diferentes longitudes en el campo de datos. El paquete final (Pkt) tiene el siguiente aspecto con formato del lenguaje *Python*:

Pkt=RadioTap(present=”MCS”, knownMCS=”MCS_index+MCS_bandwidth+guard_interval+HT_format”,HT_format=0,MCS_index=MCS_vector[MCS], MCS_bandwidth=1,guard_interval=0) / **Dot11**(type=2, subtype=0, addr1=AP_MAC_R[usuario], addr2=AP_MAC_T, addr3=AP_MAC_T) / **ip / udp / relleno1000**)

[3] Existen otros parámetros opcionales del estándar (STBC, LDPC…) que no se utilizan.

Una vez está definido el paquete (Pkt), hay que enviarlo por la interfaz del transmisor (AP) mediante la función *sendp*. Se debe especificar el paquete, la interfaz, marcar como tiempo real, señalar la variable *loop* (una lista para poder modificarla) y el intervalo de paquetes.

Sendp (Pkt, iface=IFACE_T, realtime=True, loop1=loop1[usuario], inter=interv, verbose=False)

Para probar el control de tasa de transmisión se deben enviar tramas continuamente con una tasa de envío de paquetes constante. Por ello, en la función, se especifica el intervalo o tiempo entre paquetes y la transmisión continua se consigue marcando la variable *loop*. De esta manera se comporta como un bucle que transmite paquetes a una tasa constante. Cuando sea necesario cambiar la tasa del paquete se modifica la lista loop1[usuario], al apuntar a una dirección de memoria hace que la variable que hemos pasado a *sendp* cambie y salga del bucle de envío inmediatamente para cargar el paquete con la nueva tasa.

Se ha de tener en cuenta que con este método aparecen varias limitaciones en la inyección de paquetes. En primer lugar, el tiempo de ejecución de la función *sendp* no es despreciable, y provoca que la inyección de paquetes esté limitada a 100 paq/s. En segundo lugar, el control de tasa se basa en cambiar la tasa de transmisión o el índice MCS con el que se transmite el paquete en el momento adecuado. Para ello fue necesario modificar la librería de transmisión para cambiar la tasa inmediatamente en el siguiente paquete ya que de otra manera la función *sendp* no permitía cambiar el paquete que se enviaba. Era necesario esperar a que terminara de transmitir un número de paquetes fijo que venía dado por la variable *count*, provocando un retardo en el cambio de tasa. La modificación del tipo de la variable *loop* a una lista (objeto mutable) posibilita el poder salir del bucle y poder cambiar la tasa de transmisión.

3.5. Monitorización de paquetes

La monitorización de paquetes se realiza desde un dispositivo espía como se ha explicado anteriormente. Los paquetes se capturan utilizando la librería *Scapy* mediante la función *sniff*. Hay que especificar la interfaz Wi-Fi donde se capturan (en este caso la interfaz correspondiente al espía) y la función que ejecuta al recibir un paquete. Esta función denominada *handler* se encarga de filtrar el paquete según el tipo, obtener métricas y calcular las estadísticas.

Sniff (iface=IFACE_R, prn=handler)

Se diseña la función *handler* como una máquina de estados, donde el estado corresponde al tipo de paquete recibido (datos, ACK, reintentos u otros paquetes no dirigidos a la STA). Inicialmente, la función filtra el paquete por cabecera para saber en qué estado estamos. Cuando se recibe un paquete de datos original (no es reintento) se guarda su información (tiempo, MAC y nº secuencia). Por el momento no se puede decir si el paquete ha tenido éxito o no, debemos esperar al ACK usando la variable *esperoACK*. El siguiente paquete puede ser un ACK (entonces el paquete se marca como éxito y se envía al algoritmo), otro paquete cualquiera que nada tiene que ver con la secuencia de paquetes enviados por el AP (no se tendrá en cuenta y esperamos al siguiente), un reintento de ese paquete (se marca como fallo el paquete original o reintento anterior y el actual será un nuevo reintento a la espera de ser confirmado) o el siguiente paquete con un nuevo número de secuencia (si no se ve el ACK, se confirma correcto el paquete anterior si no ha habido el máximo de reintentos antes, en caso contrario se opta por considerarlo incorrecto en una aproximación pesimista del resultado de la transmisión).

Para entender las decisiones adoptadas hay que tener en cuenta varias situaciones que se dan en el escenario de pruebas y que vienen derivadas de que la monitorización se hace en el espía y no en el AP:

- Para tener claro que capturamos en el espía todos los paquetes que salen de la interfaz radio en el AP debemos asegurarnos que el AP y el espía, por una parte, están colocados cerca pero a una distancia suficiente como para saber que la potencia recibida no es tan alta como para saturar el receptor y, por otra parte, que en una configuración MIMO los caminos están incorrelados.
- Con respeto a la captura de los ACKs enviados por la o las STAs, se pueden dar dos situaciones:
    1. El AP recibe el ACK pero no lo hace el espía. En ese caso, el AP envía un paquete nuevo. Por ello, la recepción de un paquete con un número de secuencia nuevo, independientemente de que no se haya recibido un ACK, debe ser interpretado como una transmisión correcta. Esta decisión es segura siempre que no se haya completado el número de retransmisiones. Si se ha alcanzado el número máximo no se sabe si la transmisión ha sido correcta o no y se toma como incorrecta.
    2. El AP no recibe el ACK pero si lo recibe el espía. En ese caso, el AP envía una retransmisión y el paquete no debe considerarse correcto. En consecuencia, en el espía, la recepción de un ACK por sí solo no basta para considerar la transmisión correcta. Es preciso verificar que no se producen retransmisiones.

Lo comentado implica una gran variedad de casos que pueden ocurrir y que se tienen en cuenta observando en una máquina de estados las posibles combinaciones. Además, los paquetes pueden perderse por limitaciones propias de *Scapy* por lo que el código de monitorización también ha de tener en cuenta estas pérdidas.

En la recepción de los paquetes se detectó un error no esperado en situaciones en las que se envían un gran número de reintentos (en puntos alejados con mala cobertura). La tasa de paquetes por segundo se multiplica (por 7 si ponemos el límite de reintentos a 6) y esto hace que cuando la tasa de paquetes originales es ya alta el volumen de paquetes a capturar esté por encima de la capacidad de escucha de la función de recepción, provocando que algunos paquetes no se reciban. Se han encontrado referencias donde también aparece este problema al transmitir a altas tasas de paquetes con *Scapy*. Esto se solucionó desde el código de recepción comprobando de forma explícita número de secuencia y reintentos para controlar los paquetes no monitorizados.

En cuanto a la opción de manejar transmisiones dirigidas a varios STAs, en los programas desarrollados para la transmisión e implementación de los algoritmos de adaptación se han definido las variables, como vectores de tamaño variable (N), ajustando al número de STAs a manejar. Asimismo, en ambos programas se ejecutan en paralelo las transmisiones y los algoritmos como *threads* o hilos.

Sin embargo, en el caso del monitor espía, aunque se planteó la posibilidad de lanzar en paralelo varios hilos monitores (uno por estación), fue descartada ya que se pueden calcular las estadísticas de varios usuarios con una sola máquina de estados. Esto es debido a que el punto de acceso transmite el paquete original de un usuario y su *driver* las posibles retransmisiones a continuación (no intercala usuarios entre retransmisiones), para después continuar con la transmisión de otro paquete nuevo al mismo u otro usuario. Al no mezclarse las transmisiones dirigidas a varios STAs, la máquina de estados inicial continúa siendo válida con algunos cambios. Al añadir más usuarios hay que tener en cuenta a quien pertenece el último paquete que falta por confirmar y para ello se añade la variable *ultimo_usuario* que lo identifica.

3.6. Implementación de los algoritmos

Para la implementación del bloque de algoritmos se propone un diseño general del programa que lleva a cabo las siguientes acciones: recibir métricas y/o estadísticas desde el monitor de paquetes (espía.py), ejecutar el algoritmo deseado (ONOE, ARF…) y enviar la decisión al transmisor (subir, bajar o mantener la tasa).

El programa realizado (Algoritmo.py) recibe vía *socket* vectores con las métricas sobre cada paquete de datos recibido por el espía, que se van acumulando en una matriz donde cada fila es un paquete a la espera de ser procesados por el algoritmo. El vector recibido contiene el tiempo de llegada del paquete, la trama MAC del receptor (posibilita distinguir a los usuarios o STAs), el número de secuencia del paquete, si es un reintento y por último el espía determina si ese paquete ha tenido éxito teniendo en cuenta diferentes condiciones que se explican en el apartado anterior (ACKs, NSeq siguientes…). Los vectores de cada paquete que se van almacenando en una matriz de paquetes se pueden ver en la Tabla 4:

Tabla 4 Matriz de paquetes del algoritmo

	Tiempo de llegada	MAC STA	Número de secuencia	Reintento	Éxito
Paquete 1	1.01	00:c0:ca:a4:7b:ae	400	0	1
Paquete 2	1.02	00:c0:ca:a4:7b:ae	401	0	1
…	…	…	…	…	…
Paquete N	10.01	00:c0:ca:a8:87:a5	3900	1	0

El procesado de las métricas de cada paquete depende del algoritmo a utilizar.

Se van a implementar los algoritmos ONOE, ARF y AARF. Las características de estos algoritmos son las siguientes. El algoritmo ONOE realiza un promediado estadístico de los paquetes recibidos en un periodo temporal (ventana de 1 segundo) y a partir de ellos obtiene ciertos parámetros (paq. enviados, transmitidos con éxito y retransmitidos) para tomar la decisión (subir, bajar o mantener la tasa). A diferencia de ONOE, los algoritmos ARF y AARF son deterministas y se basan en contar los paquetes recibidos uno a uno y compararlos con umbrales para tomar las decisiones.

Por tanto, ONOE necesita procesar todos los paquetes recibidos filtrando la matriz de paquetes en una ventana de 1 segundo para tomar decisiones. Los algoritmos ARF y AARF trabajan paquete a paquete por lo que van procesando la información recibida conforme llega vía *socket.*

El algoritmo es implementado como un *thread* o hilo por usuario (STA) que se ejecuta continuamente en un bucle y procesa la información. Una vez ha tomado una decisión sobre reducir, mantener o aumentar la tasa de transmisión, la envía vía *socket* al programa de transmisión (Transmisor.py). Se envía una línea por decisión que contiene la orden a ejecutar (aumentar=1, reducir=-1 y mantener=0) y también la MAC de la STA (permite tener más usuarios o STAs en el sistema).

Tabla 5 Vector de decisión enviado al transmisor

Decisión	MAC
1	00:c0:ca:a4:7b:ae

4. Pruebas experimentales

Una vez definido el entorno experimental se realizan las pruebas con los diferentes algoritmos para evaluar su comportamiento ante las distintas situaciones que pueden aparecer a un usuario que utiliza una STA y en función de distintos parámetros de configuración del algoritmo. En primer lugar, se prueba el rendimiento en condiciones estáticas, después con un movimiento constante y por último se analiza la influencia del entorno con una prueba de interferencias causadas por un terminal oculto y otra prueba simulando el paso de personas cerca de la STA.

Como norma general se genera un tráfico con una tasa constante de 100 paq/s y una longitud de 1000 Bytes. Ocasionalmente, se miden las prestaciones con tasas de 20 y 60 paq/s para evaluar el impacto de la tasa de generación de paquetes. No obstante, conviene destacar que la tasa de transmisión en paquetes por segundo no es muy alta debido a problemas prácticos de inyección y, por otra parte, el escenario de pruebas limitaba el número de STAs. En este contexto, hay que tener en cuenta que incluso en los escenarios de mayor tasa (100 paq/s), la carga de la red no será muy alta y, como es de esperar, se consiguen transmitir prácticamente todos los paquetes originados. Para verificarlo, se evalúa el grado de satisfacción alcanzado, medido como el número de paquetes transmitidos correctamente con respecto al número de paquetes originales que se intentan transmitir. Como se verá más adelante, se consigue transmitir el 100% de los paquetes en todos los casos.

En estas condiciones, no tiene sentido utilizar como parámetro de evaluación y comparación el *throughput* total alcanzado (tasa efectiva de transferencia en bps) y por ello se hace necesario considerar otras métricas alternativas.

Para evaluar y comparar los algoritmos se van a considerar las siguientes métricas:

Eficiencia: se calcula como el cociente entre el número de paquetes enviados con éxito en el intervalo de medida y el número de paquetes/ráfagas enviadas, incluyendo repeticiones, en el intervalo de tiempo de medida. El parámetro nos da una idea de cómo de eficiente ha sido el algoritmo en cuanto a ocupación del canal para alcanzar los objetivos de transmisión. A mayor número de repeticiones requeridas menor eficiencia.

En las pruebas estáticas el intervalo de monitorización de la eficiencia será la duración de la prueba eliminando un transitorio inicial. En cambio, en las pruebas de movimiento (incluidas las pruebas en presencia de interferencias provenientes de nodos ocultos), puesto que la eficiencia va cambiando en el tiempo, los valores de la eficiencia se obtendrán promediados en ventanas con 3 intervalos de duración diferentes (200ms, 1 s y 10 s). Esto permitirá una mejor visualización y comparación de las prestaciones ofrecidas por los distintos algoritmos en términos de capacidad de adaptación efectiva a corto plazo a los cambios del entorno de propagación.

Distribución estadística de las MCSs utilizadas en cada uno de los puntos de las pruebas estáticas. Nos permite visualizar el grado de estabilidad de los algoritmos. Para facilitar la visualización de las implicaciones que tienen estas distribuciones estadísticas en la tasa de transmisión, en este mismo tipo de escenario estático, se obtendrá la tasa física promedio utilizada en las transmisiones correctas. Para ello basta con promediar la tasa física utilizada en cada uno de los paquetes transmitidos correctamente durante la duración de la prueba. Por contra, en las pruebas de movimiento se comparará la variación del índice MCS de los paquetes transmitidos respecto al tiempo, que corresponde a la tasa de transmisión instantánea alcanzada durante el trayecto.

Los experimentos a realizar se detallan al comienzo de las pruebas. Como ya se comentó en el capítulo 2, los algoritmos basados en la pérdida de tramas, por la disponibilidad y facilidad para obtener las estadísticas, son los que más ajustan a las necesidades de este trabajo. Se analizan en orden los algoritmos ONOE y sus variantes, ARF y AARF.

4.1. Pruebas estáticas

En primer lugar, se realizan las pruebas en estático. Normalmente un usuario que utiliza una STA pasa la mayor parte del tiempo en reposo. Por ejemplo, un usuario que pasa horas con un ordenador portátil o con un teléfono móvil en un escritorio de trabajo. En general, no van a existir grandes variaciones del nivel de señal o del canal como en el caso del movimiento por lo que un buen diseño de un algoritmo no debería estar constantemente reduciendo y aumentando la tasa ya que perderá eficiencia.

La descripción de las pruebas en estático se puede ver en la Fig.22. El AP se encuentra en una posición fija en el laboratorio 2.05 junto al PC Linux y el dispositivo Wi-Fi espía (obtiene las estadísticas de los paquetes). Se definen 8 puntos en el mapa donde se colocará la STA y se realizarán medidas por separado. Primero se analiza ONOE y sus variaciones y después ARF y AARF.

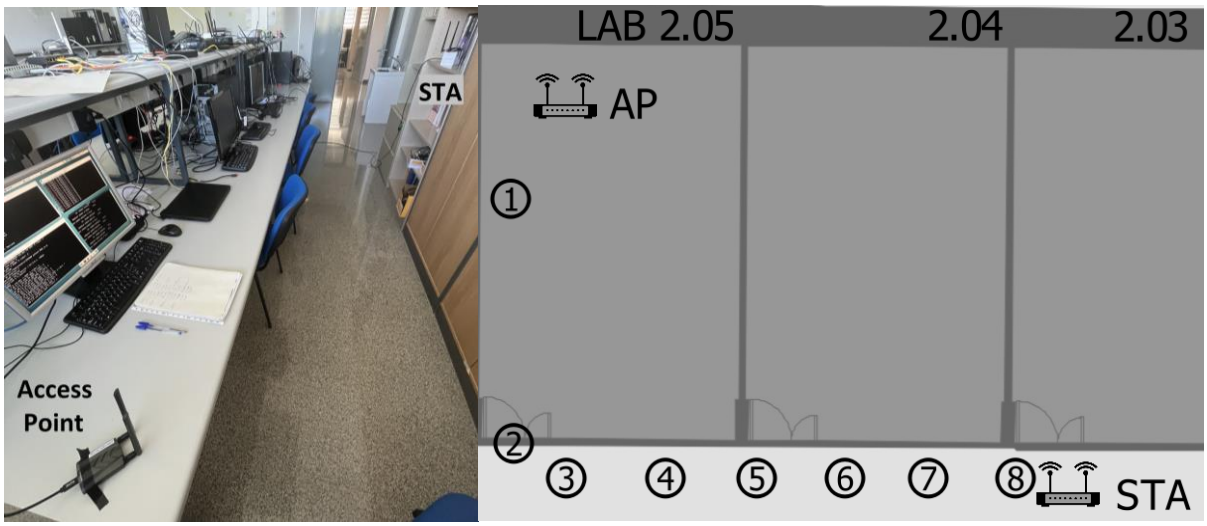
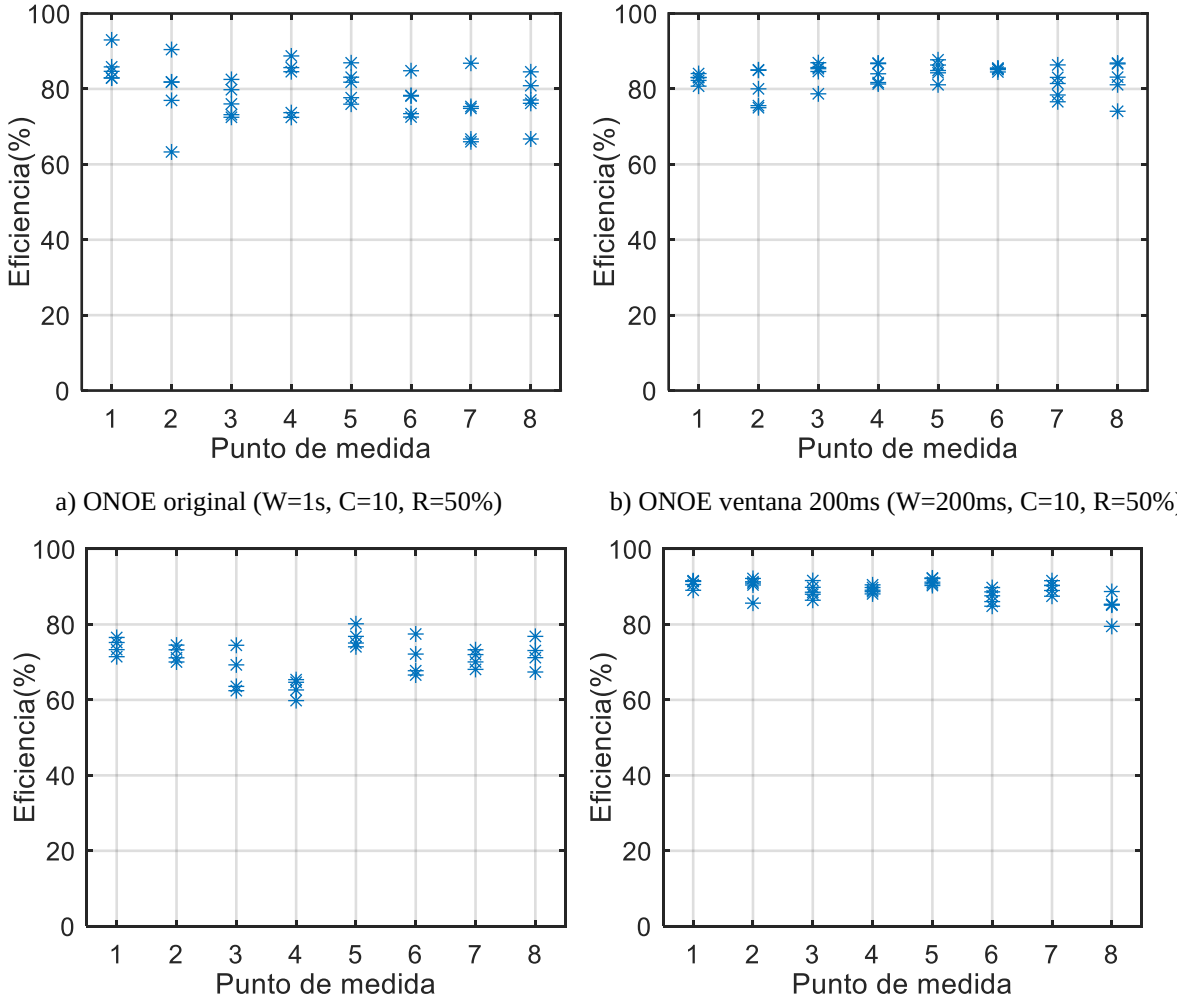


Figura 22 Escenario para las medidas en estático: visión del laboratorio 2.05 para medir el punto 1 (izquierda) y mapa del escenario completo con los puntos de 1 a 8 (derecha)

En primer lugar, se va a analizar el efecto de los parámetros de ONOE (umbral de créditos (C), umbral de reintentos (R) y ventana de actualización (W)) en la eficiencia de la transmisión. Se van a comparar el algoritmo ONOE original (W=1s, C=10 y R=50%) con una variante en la que el tamaño de la ventana de actualización se reduce a 200 ms (W=200ms, C=10 y R=50%), la variante reduciendo el umbral de créditos a 3 (W=1s, C=3, R=50%) y por último reduciendo el umbral de reintentos al 30% (W=1s, C=10, R=30%). El umbral de créditos C indica cuantas ventanas de tiempo con paquetes enviados con un éxito superior al 90% (porcentaje de reintentos inferior al 10%) deben ocurrir para aumentar de tasa. El umbral de reintentos R indica el promedio de reintentos por trama (debidos a errores en tramas anteriores) en una ventana de actualización que debe alcanzarse para reducir la tasa. La duración de los experimentos en estático es de 3 minutos. El AP transmite a la STA una tasa constante de 100 paq/s y una longitud de 1000 Bytes. A partir de las medidas se van generando unos *logs* o registros donde se almacenan todos los datos importantes de la transmisión para generar los resultados gráficos.

Experimentalmente se ha observado que en distintas realizaciones para un mismo algoritmo y parámetros de configuración aparecen variaciones significativas de la eficiencia que, dependiendo de los parámetros escogidos, pueden ser más o menos relevantes. Ante la inviabilidad práctica de repetir los experimentos el número de veces necesarios para obtener un buen promediado y una buena caracterización estadística, se opta por un análisis cualitativo basado en varias repeticiones. En la Fig.23 se muestran varias realizaciones de las medidas por separado en los 8 puntos del escenario anterior para los casos de ONOE original, reduciendo la ventana a 200 ms mientras se mantiene C=10

y R=50%, reduciendo el umbral de créditos a C=3 manteniendo W=1s y R=50%, y reduciendo el umbral de reintentos al 30%, mientras se mantiene W=1s y C=10. Se puede ver que para el ONOE original (Fig.23.a) existe una gran dispersión de los resultados entre las distintas realizaciones. Esto es debido a que tiene un umbral de pérdidas muy alto (umbral de reintentos del 50%). Por otra parte, tiene una ventana de promediado alta (ventana de 1 s) que permite compensar mejor picos de pérdidas de paquetes con periodos de transmisión correctos. Es decir, promedia más las variaciones en las condiciones de canal. La combinación de ambas cosas puede hacer que el algoritmo se mantenga en una tasa en condiciones promedio de canal bastante diferentes. El rango de valores es amplio (10 a 50% sin subir ni bajar de tasa). Ocasionalmente, puede soportar tasas que provocan perdidas elevadas (hasta el 50%) durante bastante tiempo, lo que reduce la eficiencia de la transmisión. En otras ocasiones, la tasa de pérdidas puede ser menor (por ejemplo, del 20%) arrojando valores de eficiencia significativamente mayores. Si se modifica el tamaño de la ventana de 1s a 200 ms (Fig.23.b), la capacidad de promediado de las condiciones de canal es menor lo que permite un seguimiento más fino del canal en cada ventana W. Por otra parte, una actualización más rápida (junto a un promediado menor de errores y éxitos) permite bajar la tasa más rápido en situaciones con pérdidas elevadas (>50%). Esto hace que se reduzca la varianza de la eficiencia del algoritmo en un mismo punto. Como puede observarse en la Fig 23.b, el valor no depende tanto de la realización, sino que se observa una variación en los rangos de la eficiencia entre unos valores más acotados.



a) ONOE original (W=1s, C=10, R=50%)

b) ONOE ventana 200ms (W=200ms, C=10, R=50%)

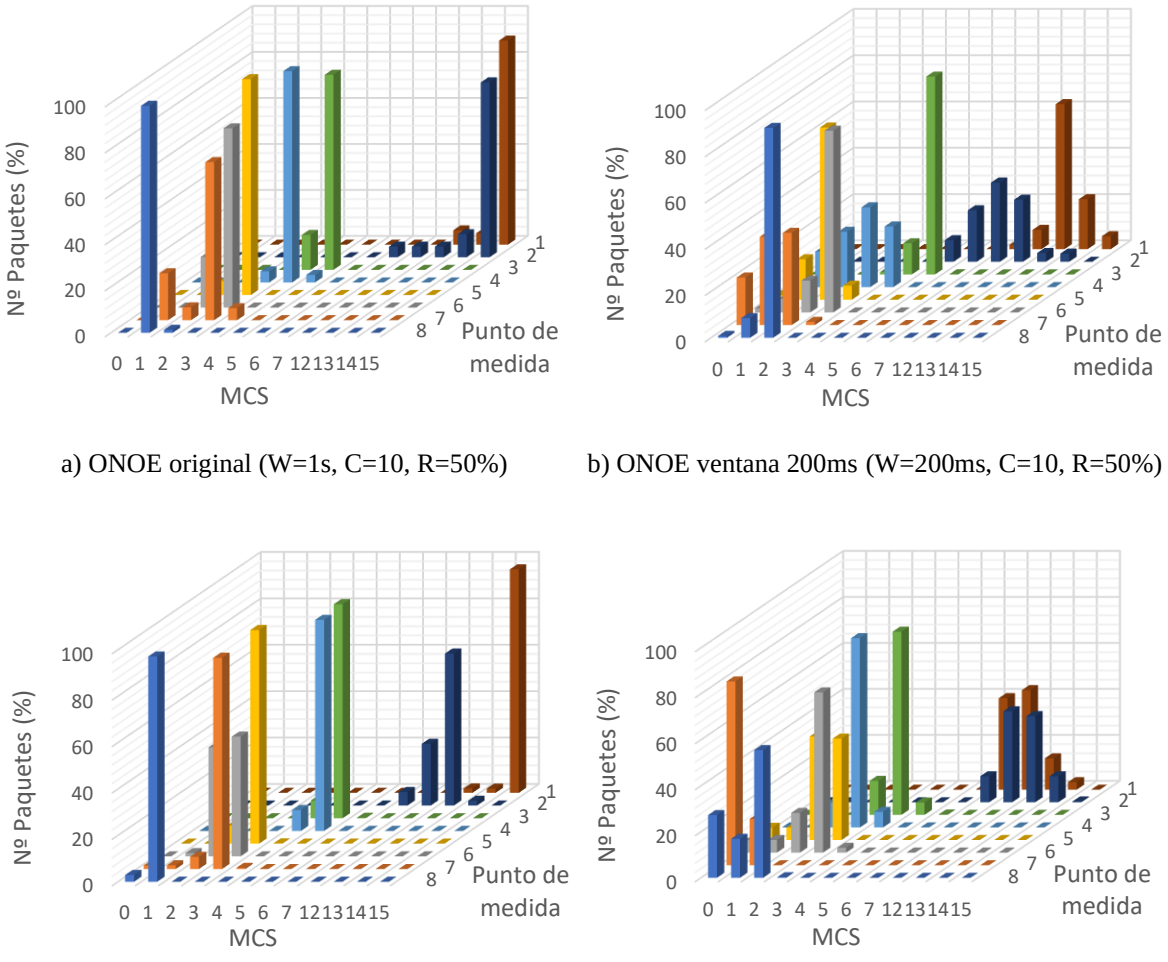
c) ONOE umbral crédito 3 (W=1s, C=3, R=50%)

d) ONOE reintentos 30% (W=1s, C=10, R=30%)

Figura 23 Varias realizaciones de la eficiencia en los 8 puntos del escenario para ONOE original (a), con ventana de 200 ms (b), con umbral de créditos de 3 (c) y con umbral de reintentos del 30% (d). Transmisión de 100 paq/s y longitud L= 1000 Bytes desde el AP a la STA.

Al reducir el número de créditos a 3 (Fig.23.c) la eficiencia se reduce respecto a las otras configuraciones, esto es debido a que se intenta transmitir con más frecuencia en tasas más altas como se podrá ver más adelante, lo que provoca un aumento de las pérdidas. Para el caso de umbral de reintentos del 30% (Fig.23.d) se obtiene una eficiencia mayor (90%) y una variación más acotada que en los casos anteriores. Frente a la versión original, la ventana de promediado es igual pero los parámetros hacen que el algoritmo sea más conservador. Al buscar reducir las pérdidas con el umbral del 30% (frente al 50% en el original), el algoritmo reacciona más rápido a la presencia de pérdidas elevadas reduciendo la tasa de transmisión. Esto aumenta la eficiencia a costa de transmitir con índices MCS bajos para su nivel de señal como veremos en las siguientes gráficas. Por otra parte, el rango de pérdidas permitidas es menor, lo que reduce la dispersión entre las distintas realizaciones.

En la Fig.24 se presenta el histograma de MCS para los 8 puntos de medida utilizando diferentes parámetros del algoritmo ONOE. Los histogramas representan el porcentaje de paquetes que se han enviado con una MCS determinada (0, 1, 2, 3, 4, 5, 6, 7, 12, 13, 14 o 15) en cada punto de medida del 1 al 8. Para la representación se utiliza la primera realización de las medidas individuales de 3 minutos de cada uno los puntos de medida de la Fig.23.



a) ONOE original (W=1s, C=10, R=50%)

b) ONOE ventana 200ms (W=200ms, C=10, R=50%)

c) ONOE umbral crédito 3 (W=1s, C=3, R=50%)

d) ONOE reintentos 30% (W=1s, C=10, R=30%)

Figura 24 Histograma de MCS para los diferentes puntos de medida. ONOE original(a), con ventana de 200 ms(b), con umbral de créditos 3(c) y con umbral de reintentos del 30%(d). Transmisión de 100 paq/s y longitud de 1000 Bytes desde el AP a la STA

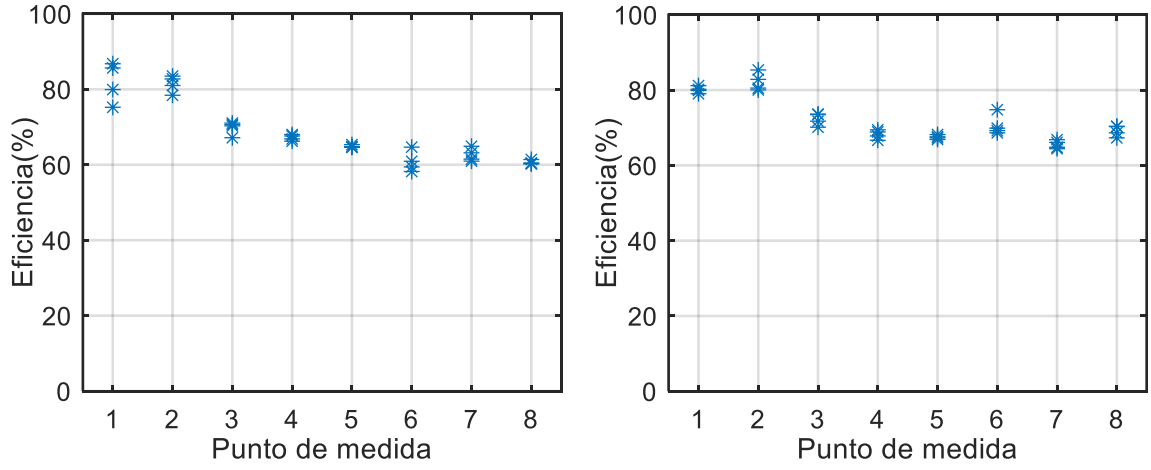
Se puede ver que reduciendo el tamaño de la ventana desde 1 s en ONOE original (Fig.24.a) hasta 200 ms (Fig.24.b) la dispersión de las MCS utilizadas en un mismo punto aumenta (el promediado de

errores y éxitos es menor por lo que cambia más). Se dan dos situaciones según la zona de cobertura. Si la cobertura es buena (puntos 1, 2, 3) se hacen adaptaciones a la baja más rápidas que con la ventana de 1 s. Esto implica que se da en menos ocasiones el fenómeno ocasional de que el algoritmo se queda con una MCS que implica una tasa promedio de error de hasta el 50%. Por el contrario, tiende a adaptarse más variando la tasa. Cuando existe mala cobertura el algoritmo puede ser más veces optimista y escoger MCS mayores. No podemos concluir que la variante de ONOE con ventana de 200 ms sea más eficiente en promedio, pero se ha visto que es menos disperso, por este motivo, al menos en condiciones estáticas de la STA, sería interesante aplicar una configuración con una ventana reducida. En la Fig.24.c se muestra el efecto de reducir el número del umbral de créditos a 3 manteniendo la ventana a 1 s. La ventana de 1 s fuerza a esperar en una MCS al menos un segundo aunque la tasa de error sea alta (si es <50%). Sin embargo, al reducir el número de créditos a 3 se favorece llegar a MCS más altas.

En la Fig.24.d de muestra el efecto de reducir el umbral de errores al R=30%. Como es de esperar, esta opción implica una mayor adaptación de las MCSs lo que se traduce en una mayor dispersión de las MCSs utilizadas en cada localización del despliegue.

Tras analizar ONOE, se presentan los resultados de ARF y AARF. La principal diferencia respecto a ONOE es que trabajan monitorizando el resultado de las transmisiones paquete a paquete en lugar de promediados en ventanas temporales de tamaño fijo. ARF y AARF incorporan también umbrales para aumentar la tasa según el número de ACKs recibidos (éxitos) y según el número de paquetes transmitidos con la variable *timeout*. Como punto de partida ambos requieren dos transmisiones erróneas para bajar la tasa y N=10 correctas para incrementarla. No obstante, AARF aplica un algoritmo de adaptación dinámica del número N.

Se realizan el mismo tipo de pruebas que con el algoritmo ONOE. En primer lugar, se analiza en la Fig.25 la variación de la eficiencia en distintas realizaciones. Se transmite con una tasa de 100 paq/s y una longitud de 1000 Bytes desde el AP a la STA, que se coloca en cada uno de los 8 puntos de medida. Se realizan varias realizaciones en el mismo punto de una duración de 3 minutos.



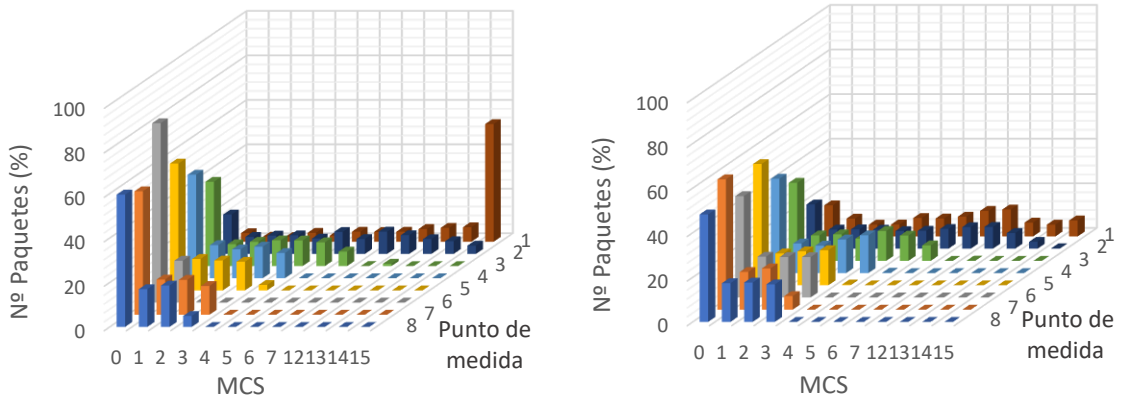
a) ARF b) AARF

Figura 25 Varias realizaciones de la eficiencia para ARF y AARF. Transmisión de 100 paq/s con una longitud L=1000Bytes desde el AP a la STA.

Comparando los dos casos en la Fig.25 con los obtenidos en ONOE se puede observar una reducción apreciable de la eficiencia de ambos algoritmos. Esta reducción es más acusada para el ARF, particularmente cuando las condiciones de canal son peores (puntos 5 a 8). Por otra parte, los resultados en diferentes realizaciones no presentan apenas variaciones a diferencia de lo que ocurre en el algoritmo ONOE. Esto puede deberse a que al trabajar contabilizando número de paquetes exitosos

o erróneos, adapta más rápidamente la tasa, promediándose más los resultados obtenidos con las distintas MCSs. Hay que tener en cuenta que solo se requieren dos transmisiones erróneas para bajar la tasa, y con 10 transmisiones correctas se incrementa. En las condiciones de prueba consideradas (generación de 100 paq/s) el incremento de tasa podría producirse en tan solo 100ms. Sin embargo, al basar en las decisiones en resultados puntuales, no se considera de forma tan adecuada las condiciones de canal promedio que está observando la STA. Con frecuencia se eleva la tasa en base a unas condiciones estimadas más optimistas de las reales o que cambiarán en las transmisiones siguientes. Esto provoca una reducción de la eficiencia, siendo más apreciable en ARF que en AARF. AARF mejora levemente la eficiencia como era de esperar al hacer uso de los umbrales variables y más amplios en cuanto número de éxitos requeridos para aumentar la tasa que en ARF según se observan errores en la transmisión. Se puede ver en el punto 8 que mejora desde un 60% en ARF (Fig.25.a) hasta un 70% en AARF (Fig.25.b).

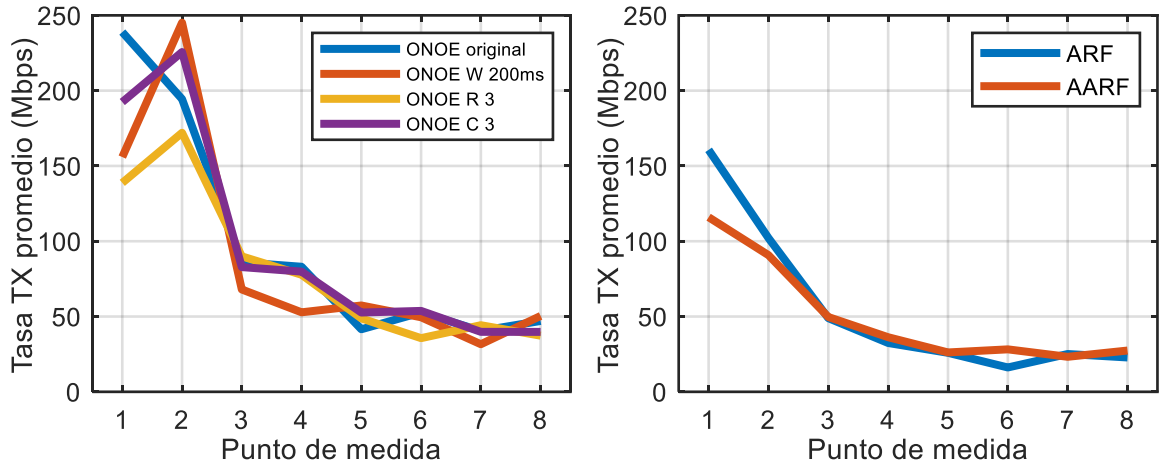
En la Fig.26 se presenta el histograma de MCS para los 8 puntos de medida utilizando los algoritmos ARF y AARF. Los histogramas representan el porcentaje de paquetes que se han enviado con una MCS determinada (0, 1, 2, 3, 4, 5, 6, 7, 12, 13, 14 o 15) en cada punto de medida del 1 al 8. Para la representación se utiliza la primera realización de las medidas individuales de 3 minutos de cada uno los puntos de medida de la Fig.25. En el caso de AARF (Fig.26.b) se puede ver una mayor dispersión de índices MCS. Esto es debido al umbral adaptativo para aumentar la tasa que incorpora AARF. Este provoca que tras una transmisión fallida, el umbral requerido de éxitos se doble, requiriendo más tiempo para volver a tasas superiores y generando una mayor dispersión.



a) ARF b) AARF

Figura 26 Histograma de MCS para los diferentes puntos de medida. ARF(a), AARF(b). Transmisión de 100 paq/s y longitud de 1000 Bytes desde el AP a la STA

Por último, para comparar el rendimiento de ONOE, ARF y AARF se presenta la tasa de transmisión promedio de los paquetes enviados correctamente. Se calcula como el promediado de las tasas físicas de transmisión, correspondientes al índice MCS con el que se han transmitido, de los paquetes transmitidos con éxito durante los 3 minutos de la primera realización de las medidas de la Fig.23 y la Fig.25 y en cada punto de medida. En general la tasa promedio de transmisión se reduce al pasar del algoritmo ONOE (Fig.27.a) a los algoritmos ARF/AARF (Fig.27.b). Destaca en ARF (Fig.27.b) la tasa en el punto de medida 1, se puede ver que es superior a AARF, esto es debido a que transmite un número de veces mayor la tasa más alta (MCS 15) como aparece en la Fig.26.a.



a) ONOE original y variantes (W, R y C)

b) ARF y AARF

Figura 27 Tasa promedio de transmisión para ONOE y sus variantes (a) y ARF/AARF (b). Transmisión de 100 paq/s y longitud L=1000 Bytes desde el AP a la STA.

Tras analizar las pruebas en estático se presentan las conclusiones obtenidas en las pruebas en estático.

Sobre el rendimiento del algoritmo ONOE y sus variantes (ventana, umbral de créditos y reintentos):

- Sobre la ventana de actualización temporal del algoritmo, se ha visto que reduciendo el tamaño de la ventana temporal (W) se promedian menos los errores y los éxitos y en consecuencia disminuye la dispersión en los resultados de eficiencia. Se obtiene de esta manera una eficiencia más estable y acotada sin depender tanto de la realización. Por otro lado, utilizar una ventana grande puede ser contraproducente al mantenerse el algoritmo transmitiendo con tasas de perdidas altas (hasta 50% en el caso ONOE original) durante el tiempo de la ventana, mientras que con una ventana reducida se puede ajustar antes.

- Sobre el umbral de créditos o éxitos para aumentar la tasa (C), se ha visto que reducir el umbral de créditos fuerza al algoritmo a transmitir con tasas más altas rápidamente, lo que combinado con una ventana de actualización lenta puede provocar una baja eficiencia.

- Sobre el umbral de reintentos o pérdidas (R), al reducir el umbral de reintentos estamos forzando al algoritmo a ser conservador y reducir la tasa de transmisión antes para evitar las pérdidas por lo que en general la eficiencia aumenta. Esto es a costa de utilizar tasas más bajas de transmisión.

Sobre el rendimiento de los algoritmos ARF y AARF:

- La eficiencia en la transmisión se reduce respecto a ONOE en las mismas condiciones de generación de paquetes (100 paq/s). Generalmente se encuentra por debajo del 80% pudiendo llegar al 60 % en algunos puntos de medida. Esta reducción es debida en buena parte a que el tiempo utilizado para tomar las decisiones de adaptación de tasa se reduce mucho al trabajar con un número de éxitos y errores en lugar de ventanas de tamaño fijo en estas condiciones de evaluación. Esto es particularmente cierto en el caso de la comparación con la versión original del ONOE donde la ventana es W=1s. ONOE en cada ventana W promedia el resultado de la transmisión de 100 paquetes y además requiere considerar el resultado de al menos C=10 ventanas para incrementar la tasa, frente a ARF o AARF donde se requiere el equivalente a una única ventana de 100ms para elevarla.

- Como se aprecia en ARF y AARF, en general no es bueno trabajar con ventanas tan pequeñas ya que con frecuencia provoca la elección de MCS por encima de las soportables.

- Sobre los umbrales de éxitos para subir la tasa (definidos como umbral de ACKs y *timeout* en ARF), la versión adaptativa de estos umbrales que propone AARF supone una leve mejora de la eficiencia ya que reduce el número de oscilaciones entre MCS.

El análisis realizado en las pruebas estáticas no es suficiente, se deben probar otras condiciones que pueden aparecer a un usuario que utiliza una STA. En el siguiente apartado se introducen las pruebas en movimiento para comprobar la capacidad de los algoritmos a adaptarse al cambio de las condiciones de señal y del entorno.

4.2. *Pruebas en movimiento*

Tras realizar las pruebas en estático, se va a determinar el rendimiento de los algoritmos en movimiento. De esta manera, se prueba la capacidad del algoritmo a adaptarse al cambio de condiciones promedio de canal y del entorno. El experimento se muestra en la Fig. 28. Se utiliza un carro con ruedas en el que se monta el STA como se puede ver. El AP se encuentra fijo en la misma posición que en las pruebas estáticas. La medida comienza desde el punto más alejado (punto 8) y se avanza a una velocidad constante con la ayuda de un metrónomo hasta la posición 2. Una vez en la posición 2, se vuelve hasta la posición 8 de la misma manera. Finalmente, se realiza el trayecto desde el punto 8 al 2. Se definen así 3 tramos de aproximadamente un minuto cada uno como se puede ver en las figuras. En primer lugar, se analizará ONOE y sus variantes y después ARF y su versión adaptativa AARF.

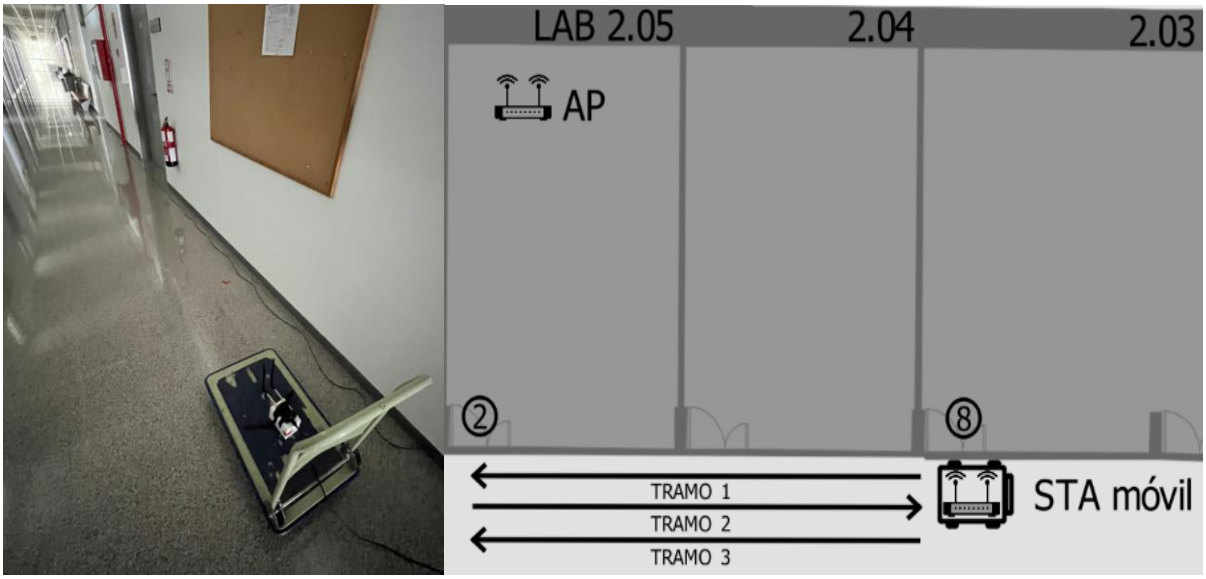


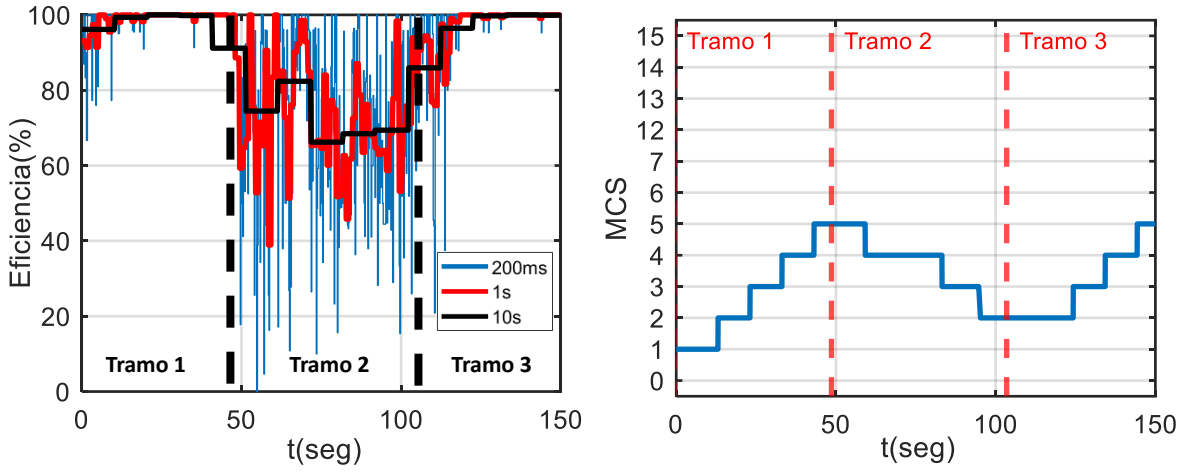
Figura 28 Escenario para las medidas en movimiento: vista del carro con la STA móvil (izquierda) y plano de la prueba (derecha).

El primer algoritmo sobre el que se van a realizar los test es ONOE. Este algoritmo, tal y como se ha visto anteriormente, tiene diferentes parámetros que variar (ventana (W) y umbral de créditos (C) y reintentos (R)). Se plantean unas pruebas del tipo DoE (*Design of Experiments*) donde variamos parámetros por separado y observamos el resultado con el fin de encontrar los parámetros que maximicen la eficiencia del algoritmo en todos los tramos. Las métricas que se van a utilizar en la evaluación son las mismas que las definidas para el escenario estático.

En primer lugar, se analiza como varía la eficiencia en la transmisión y el índice MCS a lo largo del trayecto. Para la evaluación de la variación de la eficiencia a lo largo del recorrido, se muestra eficiencia promedio calculada en ventanas de 200 ms (azul), 1 s (rojo) y 10 s (negro) como se puede ver en las siguientes figuras. El objetivo es poder visualizar mejor el comportamiento de los algoritmos. Se presentan los tres tramos del trayecto como se explica al inicio del capítulo separados por líneas discontinuas. Primero se muestran las pruebas realizadas con el algoritmo ONOE original y a continuación variando los diferentes parámetros por separado.

En el caso de ONOE original (Fig.29.a) la eficiencia se mantiene alta al acercarse al AP (tramos 1 y 3). Hay que tener en cuenta que las condiciones de canal mejoran en estos desplazamientos. Por otra parte, partiendo del punto 8 con una MCS baja (Fig.29.b), debido al valor de W=1s y C=10, el algoritmo incrementa la MCS de una forma muy conservadora. En la Fig. 29.b puede verse como en el punto 2 solo se alcanza la MCS=5. Se constata que se están usando índices MCS probablemente más

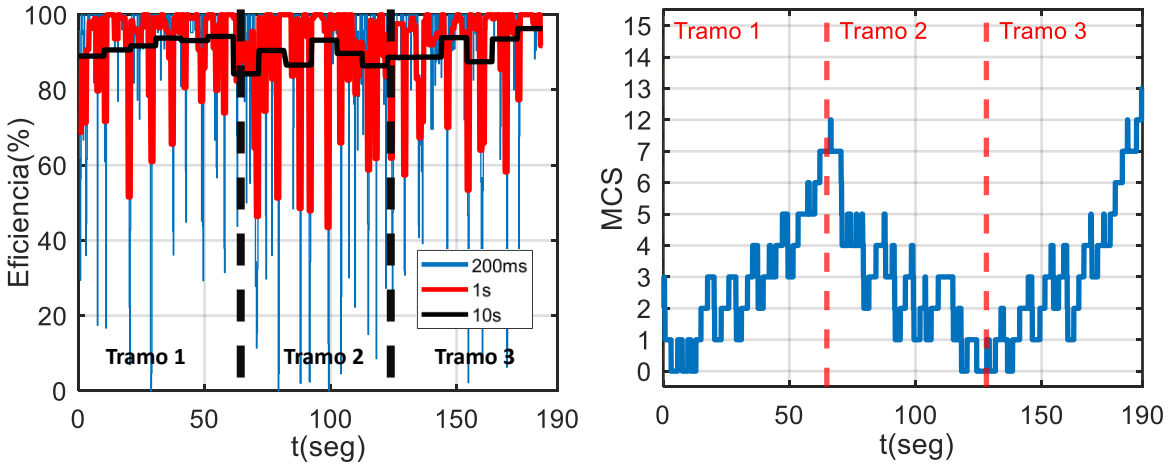
bajos de los soportables según el nivel de señal. Esto hace que la tasa de paquetes erróneos sea muy baja y en consecuencia la eficiencia muy alta. Por el contrario, al alejarse la STA del AP en el tramo 2 a la ventana de actualización lenta no le permite adaptarse a tiempo a las degradaciones de SINR y, en consecuencia, la eficiencia cae de forma muy apreciable.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 29 Algoritmo ONOE original (W=1s, C=10, R=50%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.

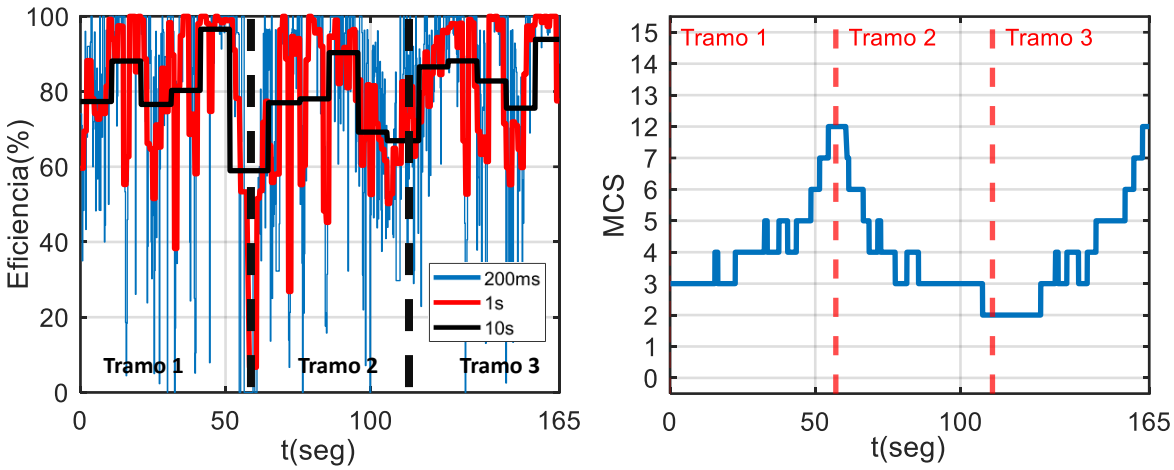
Sin embargo, si se reduce la ventana W a 200 ms (Fig.30) vemos una eficiencia más estable en todos los tramos ya que la actualización ahora es más rápida y además el promediado de errores y éxitos es menor con lo cual hay más oscilaciones en la elección de MCSs en cada tramo (Fig.30.b). En los tramos 1 y 3, la MCS se actualiza más rápidamente en sentido de subida y en consecuencia aumenta la probabilidad de transmitir paquetes usando MCS (se alcanzan MCS altas) que requieren un nivel SINR por encima del real. Como consecuencia, aumenta la probabilidad de pérdida de paquetes y decrece la eficiencia con respecto al ONOE original. Sin embargo, en el tramo 2, esta actualización rápida en sentido decreciente (a MCS más conservadoras) permite reducir significativamente la tasa de errores.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 30 Algoritmo ONOE ventana 200ms (W=200ms, C=10, R=50%) : Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.

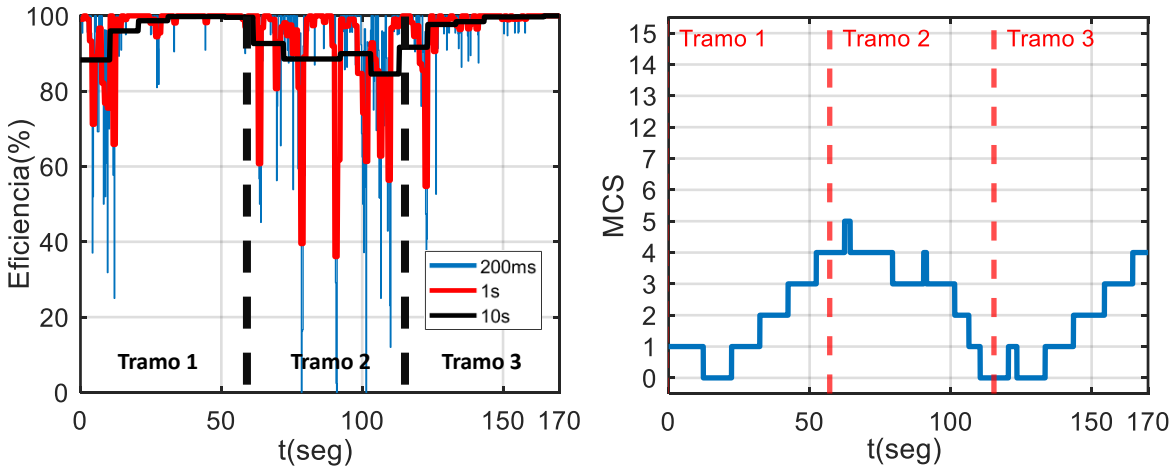
Si se varía únicamente el umbral de créditos colocándolo a 3 (Fig.31) se observa que el rendimiento en general es peor que en los casos anteriores. En los tramos 1 y 3, con esta configuración se requiere menos tiempo para incrementar la tasa o índices de MCSs (igual que ocurre cuando se baja el tamaño de W a 200ms), lo que acaba en una penalización de la eficiencia y a su vez, se ve afectado negativamente en el tramo 2 en el proceso de reducción de la MCS igual que ocurre en ONOE original (a) ya que comparten la ventana de 1 s.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 31 Algoritmo ONOE umbral crédito 3 (W=1s, C=3, R=50%) : Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.

Por último, al variar el umbral de reintentos al 30 % (Fig. 32) la eficiencia mejora en todos los tramos ya que es una parametrización más conservadora. Mantiene el número de reintentos bajo forzando a transmitir con MCS más bajas tal y como se ve en la Fig. 32.b. En los tramos 1 y 3 la MCS se sitúa por debajo del ONOE original. Por otra parte, en el tramo 2 la reducción del umbral de pérdidas consigue bajar la MCS rápidamente al degradarse la calidad de la señal, lo que hace mejorar su eficiencia en comparación con ONOE original(a) desde el 70% hasta el 87% en el tramo 2 (Tabla 6).

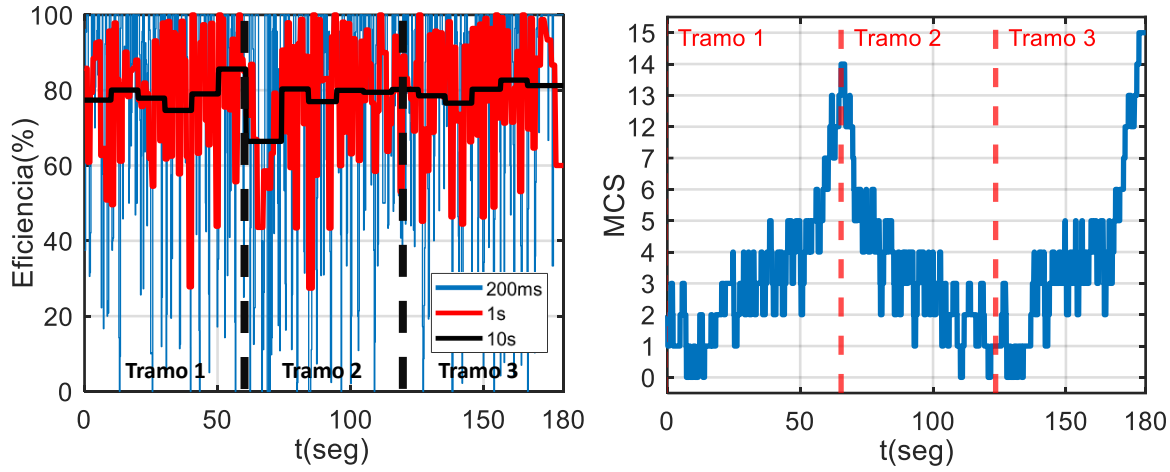


a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 32 Algoritmo ONOE reintentos *30% (W=1s, C=10, R=30%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.*

A continuación, se analiza el efecto de variar 2 y 3 parámetros de ONOE a la vez. Para la comparación, de manera similar a la prueba anterior, se van a visualizar conjuntamente las medidas de eficiencia, calculada en ventanas de 200ms, 1 s y 10 s, y la variación temporal del índice MCS utilizado en la transmisión respecto al tiempo.

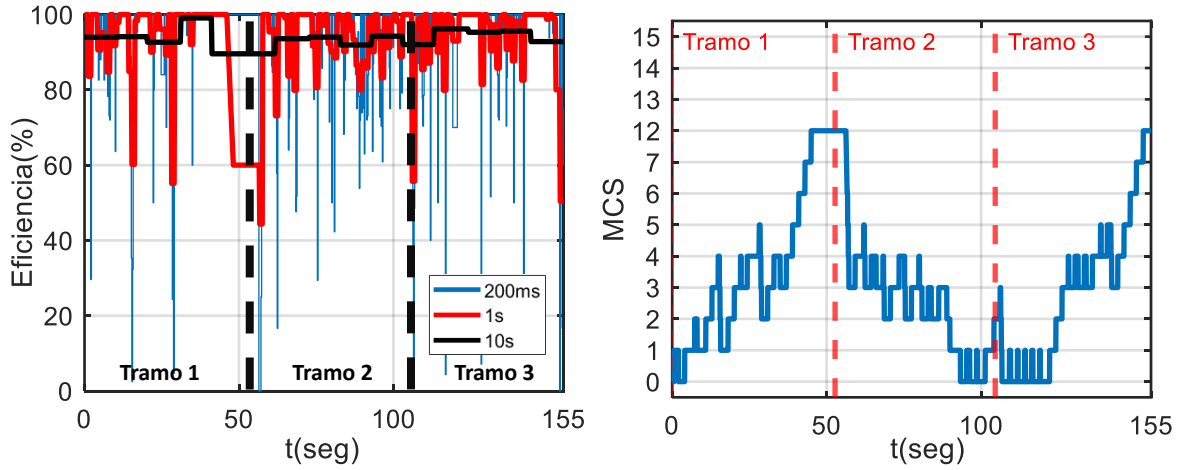
En primer lugar, evaluamos el caso de disminuir el umbral de créditos a 3 y la ventana a 200 ms (Fig.33). Comparándolo con el caso anterior en el que se disminuye solo la ventana a 200 ms (Fig.30), se puede ver que se pierde eficiencia en los tramos 1 y 3 al reducir el umbral de créditos C. Como se ve en la Fig.33.b el algoritmo incrementa demasiado rápido la MCS alcanzando valores más altos que en el caso de la Fig.30.b , lo que provoca una eficiencia baja. En el tramo 2, podría esperarse que el comportamiento fuera similar ya que la tendencia es a decrecer la MCS usando solo la ventana de W=200ms. No obstante, se observa como al modificar el umbral de créditos a 3 decae la eficiencia. Esto es debido a que, aunque en promedio la calidad del canal decae al alejarnos hacia el punto 8, el canal es variable y, puntualmente, es posible que haya intervalos en los que se considere que es posible incrementar la tasa. En el caso de W=200ms y C=3, el intervalo mínimo para tomar una decisión en este sentido es 600ms frente a los 2 s que tendríamos en el caso de considerar W=200ms y C=10. Por tanto, como se aprecia en la Fig 33.b las oscilaciones entre MCS son mucho mayores en el tramo 2 en la configuración W=200ms y C=3 que en el caso de W=200ms y C=10. En consecuencia, más veces se sobreestima la MCS posible y decrece la eficiencia.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 33 Algoritmo ONOE ventana y créditos (W=200ms, C=3, R=50%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.

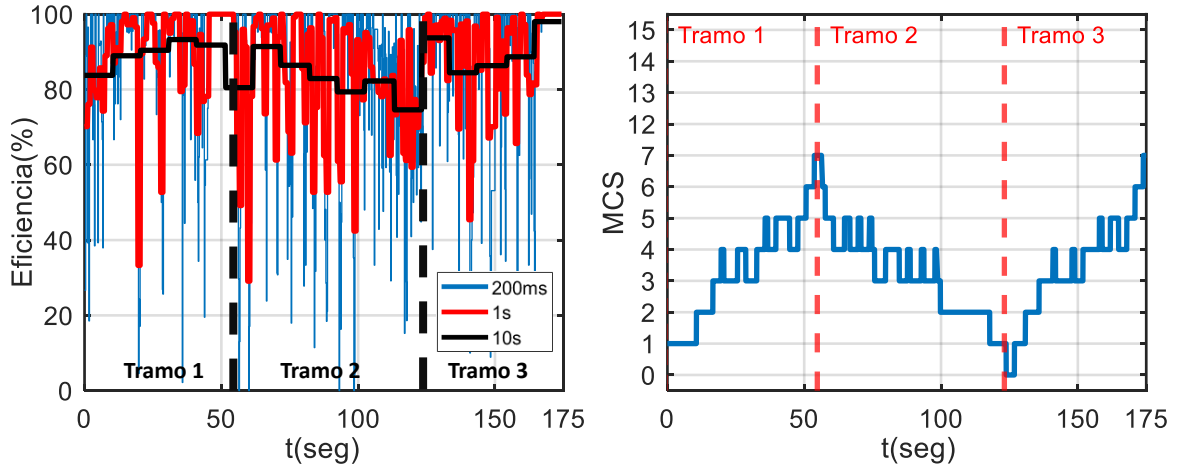
En segundo lugar, al modificar la ventana a 200 ms y los reintentos al 30% (Fig.34) se puede ver una eficiencia alta y estable (Fig.34.a). Son dos parámetros que combinados ofrecen un muy buen rendimiento. Permite una actualización rápida de la tasa gracias a la ventana reducida, pero corrige más rápidamente la elección de MCSs más altas de las deseables al reducir la tasa en presencia de un 30% de pérdidas. Esto último limita la degradación que se produce en los tramos 1 y 3 y, por otra parte, acelera la elección de MCSs más bajas en el tramo 2, tal y como se ve en la Fig.34.b.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 34 Algoritmo ONOE ventana y reintentos (W=200ms, C=10, R=30%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.

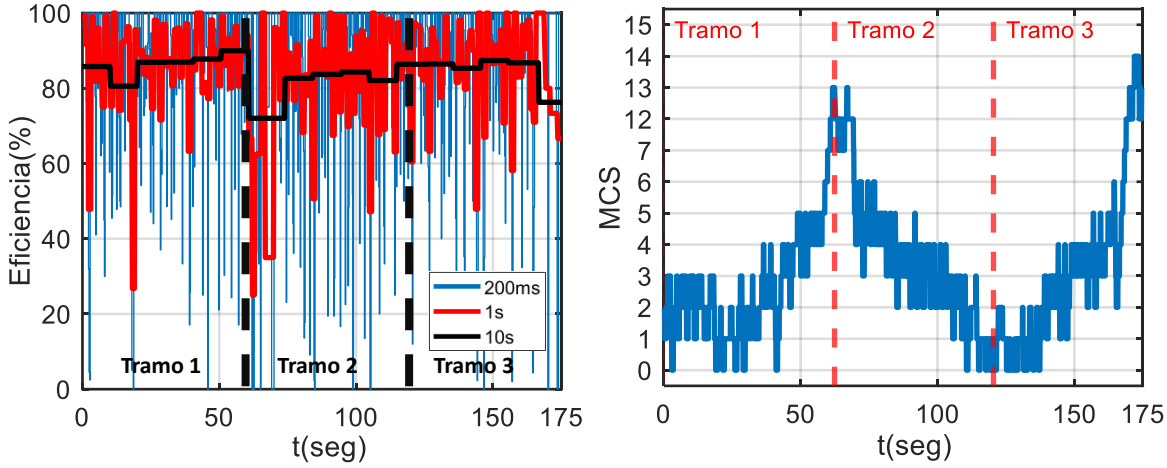
Sobre el caso en el que reducimos los créditos a 3 y el umbral de reintentos al 30% (Fig.35), observamos que los resultados empeoran con respecto a la opción de considerar únicamente una reducción del umbral de reintentos, si bien mejoran con respecto a reducir únicamente los créditos. En líneas generales, reducir los créditos sin reducir el valor de W supone subir rápidamente la tasa pero tener una capacidad de reacción a la baja lenta, solo compensada por la reducción del umbral de reintentos.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 35 Algoritmo ONOE créditos y reintentos (W=1s, C=3, R=30%): Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.

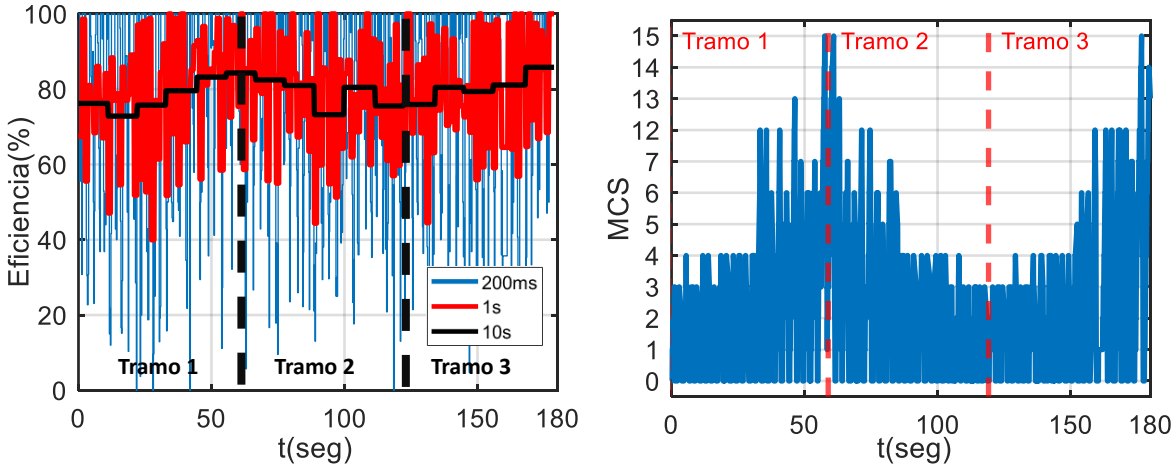
Por último, en la figura 36 se reducen todos los parámetros (ventana y umbrales de reintentos y créditos reducidos). En general la eficiencia se encuentra sobre el 80% mostrando un comportamiento similar al de la Fig.33 con una leve mejora gracias al uso de R=30%.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

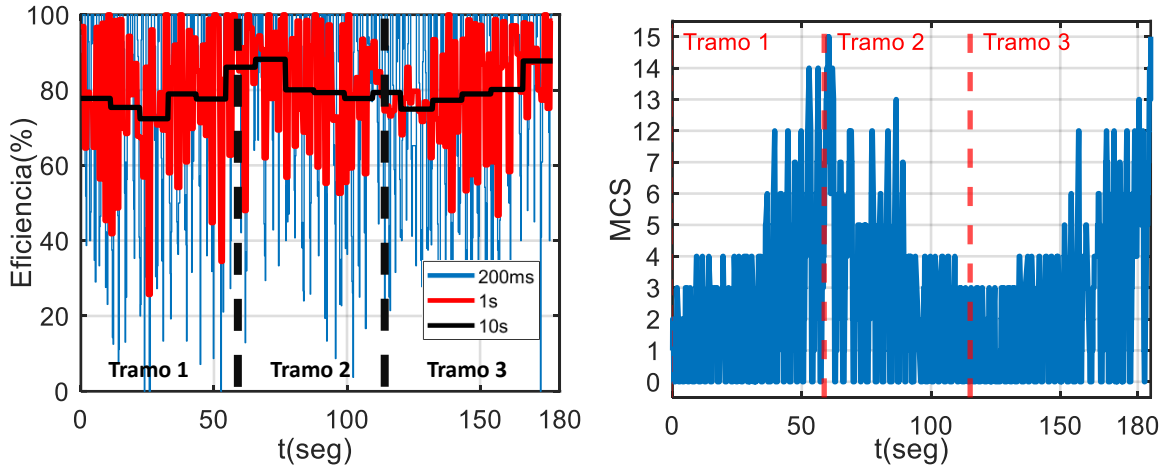
Figura 36 Algoritmo ONOE var. 3 parámetros (W=200ms, C=3, R=30%) : Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.

Tras analizar los resultados del algoritmo ONOE, en la siguiente figura se presentan las pruebas en movimiento de los algoritmos ARF (Fig.37) y su variante AARF (Fig.38). Se muestran las gráficas de eficiencia calculada en pasos (200ms, 1 s y 10 s) y la evolución del índice MCS a lo largo del tiempo de manera conjunta. La eficiencia es similar en ambos algoritmos (Fig.37.a y Fig.38.a). Destaca la gran variabilidad de las tasas transmitidas en los dos casos (Fig.37.b y Fig.38.b), alternando tasas altas con bajas continuamente. Esta configuración es muy sensible a los cambios del entorno.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

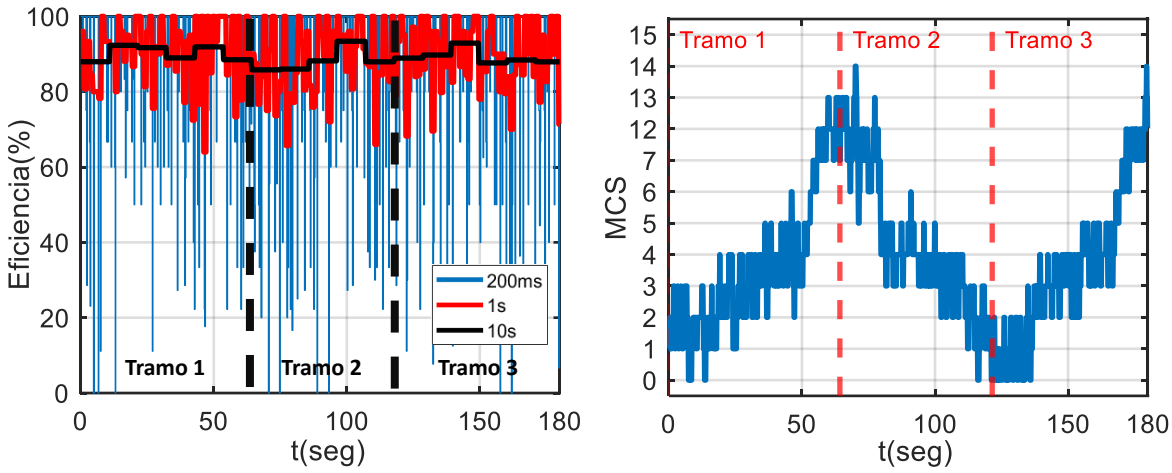
Figura 37 Algoritmo ARF: Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 38 Algoritmo AARF: Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 100 paq/s y longitud de L = 1000 Bytes.

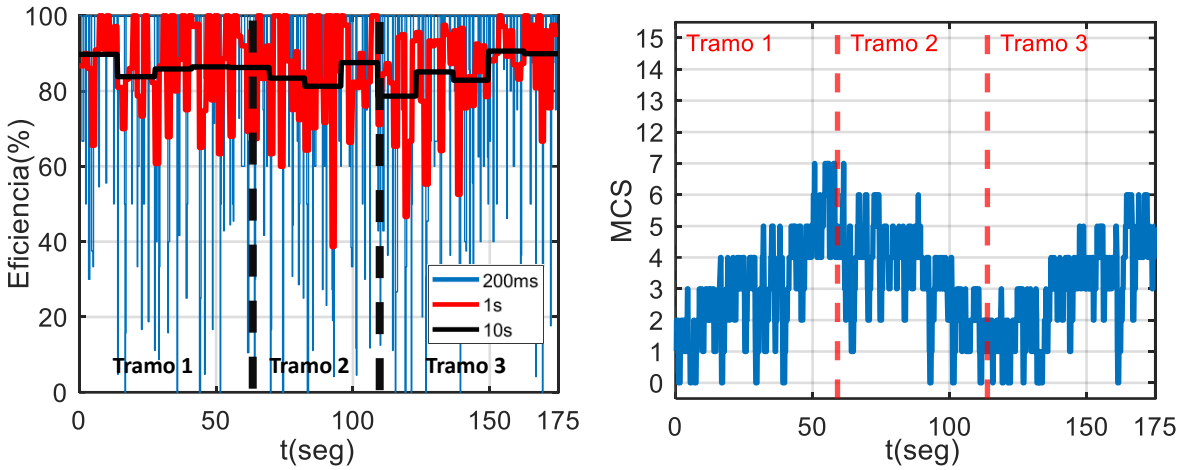
Por último, para ilustrar el efecto de la reducción del número de paquetes transmitidos en ARF (Fig.39) y AARF (Fig.40), se realiza una prueba similar a las anteriores, pero transmitiendo 20 paq/s y manteniendo la longitud a 1000 Bytes. Como se puede ver para ARF en la Fig.39.b, al transmitir un número menor de paquetes el algoritmo no es tan sensible a las variaciones como en el caso anterior. Se obtiene un *feedback* de la red cada 50 ms (20 paq/s) lo que provoca que los índices MCS no se reduzcan constantemente hasta cero (al existir menos probabilidad de acumular tramas fallidas consecutivas) como se puede apreciar en la figura 39.b. En consecuencia, la eficiencia aumenta como se muestra en la Fig.39.a.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 39 Algoritmo ARF: Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 20 paq/s y longitud de L = 1000 Bytes.

De la misma manera para AARF se puede ver que los índices MCS (Fig.40.b) no se reducen hasta cero, destaca en este caso que se alcanzan índices MCS al final del tramo que son menores que en ARF (Fig.39.b). Esto es debido a los umbrales de aumento de tasa variables. La eficiencia para AARF (Fig.40.a) se comporta de manera similar a ARF (Fig.39.a).



a)Eficiencia calculada por pasos (200ms, 1 y 10s)

b)Variación del índice MCS

Figura 40 Algoritmo AARF: Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b). 3 tramos del trayecto separados por líneas discontinuas. Transmisión desde el AP a la STA de 20 paq/s y longitud de L = 1000 Bytes.

Para concluir los análisis en movimiento, en la tabla 6 se recopilan los valores de la eficiencia promedio por tramos de movimiento y el grado de satisfacción en la transmisión de paquetes obtenida para cada elección de parámetros de ONOE realizada anteriormente y los algoritmos ARF y AARF.

En primer lugar, tal y como era esperable y dado que la carga ofrecida al sistema es muy baja, el grado de satisfacción en la transmisión de los datos es muy alta en todos los casos. En líneas generales podemos decir que prácticamente todos los paquetes que se intentan transmitir pueden ser transmitidos. Observar que se consiguen transmitir en torno al 97-98% de paquetes originados (100 paq/s) en todos los casos. Debido a las limitaciones prácticas para cargar el sistema no tiene sentido hacer un análisis en términos de *throughput* efectivo. Por tanto, tal y como se ha explicado al inicio de este capítulo, la comparación de los algoritmos se hace en función de la métrica que hemos definido como eficiencia.

A partir de los resultados se observa que en líneas generales interesa definir un umbral de reintentos del 30% (en lugar de 50%) para mejorar las prestaciones en todos los tramos. Con respecto a los créditos necesarios, en general, la consideración de C=3 empeora los resultados con respecto a considerar C=10. Es demasiado optimista aumentar la tasa tras tres ventanas de transmisión con reintentos por debajo del 10%. Finalmente, la consideración de ventanas de monitorización más pequeñas W=200ms ofrecen mejores resultados que W=1s cuando el terminal debe adaptarse a degradaciones progresivas en la calidad del canal (tramo 2) mientras que se comporta ligeramente peor en los tramos en los que la calidad mejora (tramos 1 y 3). Como puede observarse en la tabla, los mejores resultados promedio consiguen con W=200ms, C=10 y R=30%· y W=1, C=10 y R=30%.

Sobre ARF y AARF, en general se produce una reducción de eficiencia debida principalmente a que no se trabaja con promediados temporales sino contabilizando paquetes correctos e incorrectos, lo que lleva a subir y bajar la tasa con mayor frecuencia. Se hace más evidente al transmitir tasas más altas de paquetes por segundo (100 paq/s) ya que se es más sensible a las variaciones del canal. Cuando hay dos transmisiones erróneas se baja la tasa, lo cual puede ocurrir en tan solo 20ms. En ONOE se requiere tener tasa de error por encima de 30% o 50% en ventanas de promediado de 1 s. Para

incrementarla se requiere un mínimo de 10 transmisiones correctas, lo que ocurre en un mínimo de 100ms aproximadamente. En el caso de ONOE el incremento de tasa se produce después de monitorizar y promediar las transmisiones durante WxC unidades de tiempo (0,2x10 o 1x10 s). Por tanto, en ARF y AARF la aparición de pérdidas consecutivas no se promedia lo que hace bajar al algoritmo de tasa cada con mayor probabilidad. Además, hay que destacar que en general AARF obtiene una mayor eficiencia frente a ARF al utilizar umbrales variables en el tramo 2 en los que se produce una degradación del canal. En este tramo, la eficiencia de AARF aumenta del 71% hasta el 75% . Esta mejora es posible ya que el algoritmo AARF penaliza los incrementos de MCS previos que han supuesto pérdidas de paquetes. En la misma tabla se muestran los resultados de ARF con una tasa de 20 paquetes para ilustrar el efecto de la reducción de la tasa de paquetes. Se ha visto que al trabajar con tasas más pequeñas las condiciones de canal entre una transmisión y la siguiente (50ms para 20paq/s) pueden cambiar más. Al reducir la tasa desde 100 a 20 paq/s en ARF se puede observar un aumento de la eficiencia desde un 72% hasta un 87% en media en los tres tramos. De la misma manera para AARF se observa un aumento desde un 73% hasta un 84% de media. Destaca para AARF que el índice MCS más alto transmitido es 7 mientras que para ARF se ha alcanzado el índice 14, esto es debido a los umbrales variables de AARF que incrementan el número de paquetes necesarios para aumentar la tasa, lo que junto con la reducción del número de paquetes enviados (de 100 a 20 paq/s) provoca que se alcancen tasas más bajas que para AARF con 100 paq/s, donde se alcanzó el índice MCS 14.

Tabla 6 Comparativa de eficiencia por tramos del movimiento y satisfacción global (Algoritmo ONOE, ARF y AARF) a partir de las gráficas de eficiencia y MCS representadas en las figuras anteriores.

Parámetro modificado	Paq/s	Longitud Paquete [Bytes]	Ventana[s]	Créditos	Reintentos[%]	MCS Alta	MCS Baja	Ef. TR1[%]	Ef. TR2[%]	Ef. TR3[%]	Satisfacción
Original	100	1000	1	10	50	5	1	95.4	69.3	97.2	99.2
Ventana	100	1000	0.2	10	50	12	0	89.7	83.2	89.6	98.8
Créditos	100	1000	1	3	50	12	3	79.67	68.83	79.96	98.08
Reintentos	100	1000	1	10	30	5	0	96.5	86.7	97.45	99.6
W y C	100	1000	0.2	3	50	15	0	74.62	72.2	73.8	97.04
W y R	100	1000	0.2	10	30	12	0	93.3	90.45	93.8	99.43
C y R	100	1000	1	3	30	7	0	87.47	77.62	84.88	98.7
W,R y C	100	1000	0.2	3	30	14	0	83.3	78.3	83	98.3
ARF	100	1000	-	-	-	15	0	72.37	71.13	73.24	98.72
AARF	100	1000	-	-	-	14	0	69.66	75.11	74.73	98.45
ARF	20	1000	-	-	-	14	0	88.14	85.21	87.93	98.23
AARF	20	1000	-	-	-	7	0	85.83	83.51	84.35	98.38

4.3. *Pruebas con interferencias por terminal oculto*

El objetivo de este apartado es analizar el comportamiento ante interferencias co-canal de los algoritmos ONOE, ARF y AARF. Hay que tener en cuenta que los algoritmos de adaptación de tasa deben adaptarse a los cambios en condiciones de canal pero cuando las pérdidas de paquetes son debidas a colisiones reducir la MCS no resuelve el problema sino más bien lo agrava. Por este motivo, interesan algoritmos que distinguen pérdidas debido a colisiones o pérdidas debidas a canal. En este caso, ninguno de los algoritmos considerados distingue de forma explícita el origen de las pérdidas, pero si consideramos conveniente analizar su comportamiento en estas condiciones.

Se coloca el AP en la misma posición que en los casos anteriores y la STA sobre la que se van a evaluar las prestaciones en un punto intermedio de la cobertura del AP. En concreto en la posición 4. Se añade un terminal oculto que transmite en la misma frecuencia a otro dispositivo diferente. Para garantizar la condición de terminal oculto, la STA interferente debe encontrarse lo suficientemente alejada para que su cobertura no alcance al AP (círculo rojo) pero si a la STA, y a su vez la cobertura del AP (círculo verde) no debe alcanzar al terminal oculto como se puede ver en la Fig. 41. De esta manera puede ocurrir que al AP y el terminal oculto transmitan simultáneamente y aparezcan colisiones en la zona marcada en la Fig.41.

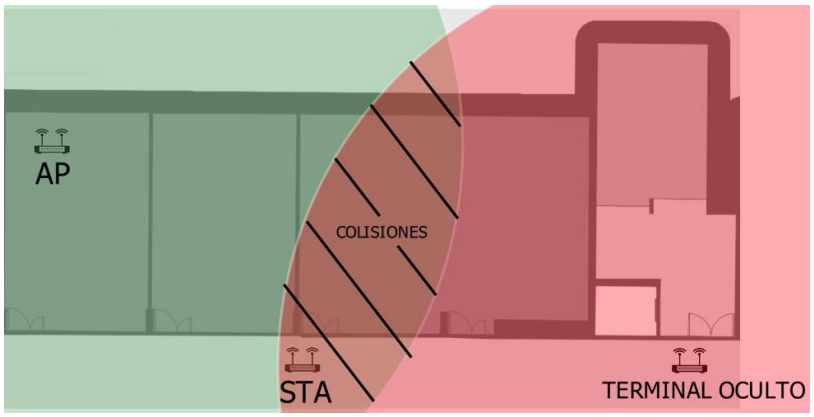
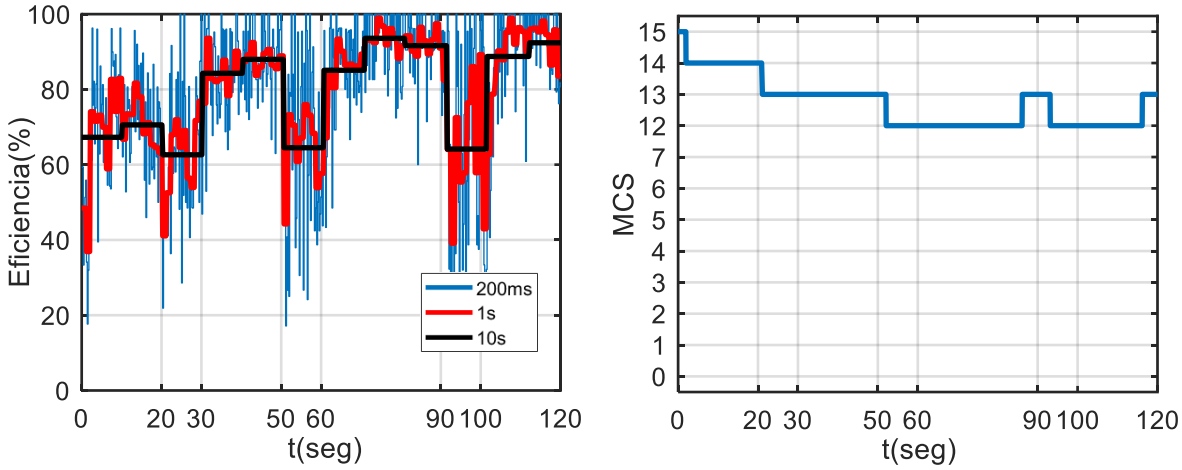


Figura 41 Escenario para las pruebas de interferencias (terminal oculto)

En la prueba de transmisión con interferencias el AP transmite constantemente paquetes (100paq/s) de longitud igual a 1000 bytes a la STA. El terminal oculto alterna periodos de silencio de 20 s con periodos de transmisión de 1000 paquetes a una tasa de 100paq/s, es decir, 10 s de interferencia. Hay 3 periodos de interferencia que comienzan transcurridos 20, 50 y 90 s. Se presentan las medidas de eficiencia promediadas en ventanas de 200s, 1s y 10s igual que en los apartados anteriores y las medidas de variación de MCS respecto al tiempo.

El resultado de la prueba para ONOE original se puede ver en la Fig.42. Se aprecia claramente como la interferencia disminuye la eficiencia de la transmisión periódicamente cuando el terminal oculto transmite (Fig.42.a). Sin embargo, en la Fig.42.b se observa que la MCS solo disminuye un índice en presencia de interferencias ya que en promedio en un segundo las pérdidas son menores del 50%, pasando de la MCS 14 a las 13 o de la MCS 13 a la 12 en las distintas ventanas de transmisión.

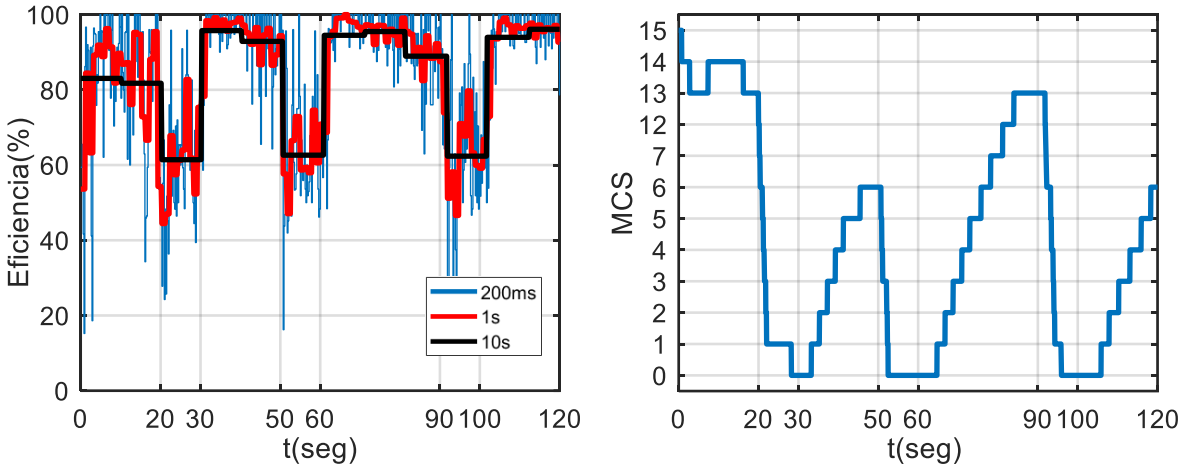
La eficiencia del algoritmo (Fig.42.a) se ve afectada por interferencias (se reduce al 60%), pero no se produce una bajada significativa de la MCS. Por tanto, a partir del comportamiento de la MCS se evidencia que la configuración con una ventana de W=1 y R=50% permite que las pérdidas debido a las colisiones se promedien lo suficiente con las transmisiones correctas sin provocar un descenso innecesario de la MCS.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 42 Resistencia de ONOE original (100p/s, 1000B) a interferencias de 100 p/s y longitud de 1000 bytes. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

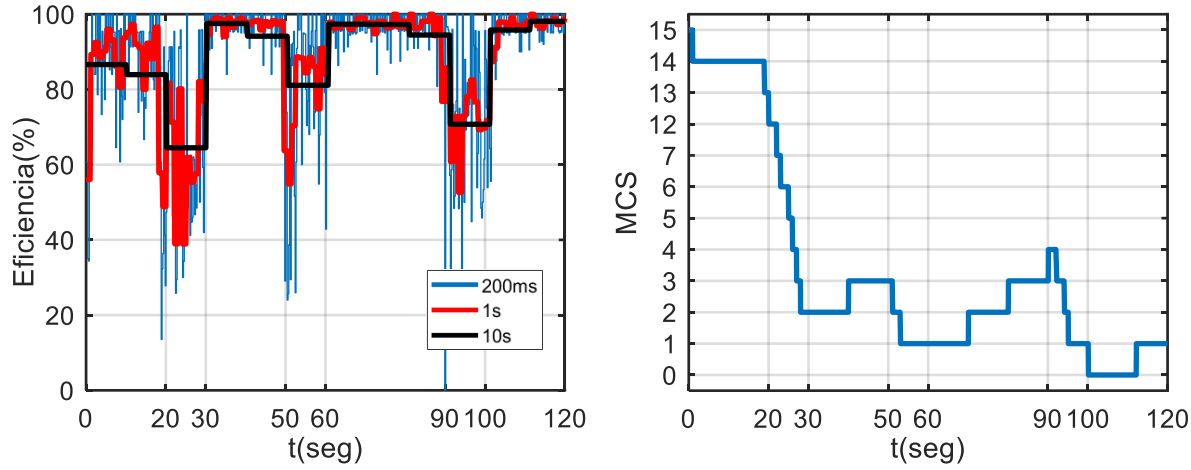
En la Fig.43 se muestran los resultados reduciendo ahora la ventana de ONOE hasta 200 ms. En este caso (Fig.43.b) la reducción de tasa (MCS) es mayor en el periodo de interferencias por que el promediado de errores y éxitos es menor que en el caso anterior (1 s). En este caso se promedia un mínimo de 20 transmisiones en comparación con las 100 de la configuración anterior. Se comporta peor que ONOE original ya que tiene una eficiencia similar del 60% en periodo de interferencias, pero utiliza MCS más bajas. Como se ha comentado, en presencia de interferencias no se debe reducir la tasa ya que de esta manera la transmisión de un paquete tarda más tiempo y es más probable que se vea afectada por estas interferencias, como en este caso en el que se aprecia que una MCS más baja no garantiza mayor eficiencia.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 43 Resistencia de ONOE con ventana de 200 ms (100paq/s, 1000B) a interferencias de 100 paq/s y 1000 Bytes. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

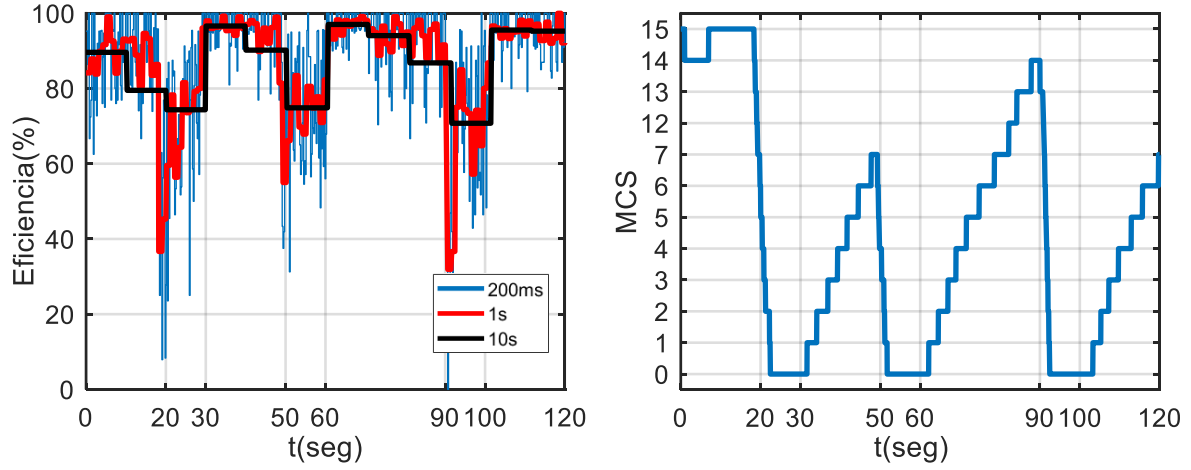
En la Fig.44 se muestran los resultados con un umbral de reintentos del 30% y la ventana original (W=1). En este caso, en la eficiencia (Fig.44.a) se observa el mismo tipo de comportamiento (se identifican claramente los periodos de interferencia) que en los casos anteriores, aunque con un ligero aumento de la eficiencia. Esto era esperable ya que el parámetro R=30% fuerza a efectuar adaptaciones que garanticen una eficiencia por encima del 70%. Vemos que los MCS de la transmisión (Fig.44.b) se mantienen bajos a partir de la primera ventana siendo la configuración incapaz de adaptarse a la situación real de la red. Se produce una bajada rápida en la primera ventana de interferencias debido a que el umbral de pérdidas es del 30% y la subida de MCS es lenta debido a la ventana de 1 s. Por tanto, queda claro que esta combinación de parámetros no se comporta bien ante interferencias.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 44 Resistencia de ONOE con umbral de reintentos del 30% (100p/s, 1000B) a interferencias de 100p/s y 1000Bs. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

En la Fig.45 se muestran los resultados con ventana de 200 ms y umbral de reintentos del 30%. En este caso sí que se puede ver que la eficiencia ha mejorado en los periodos de interferencia con respecto al ONOE original. Se encuentra por encima del 70% (Fig.45.a) pero esta mejora de la eficiencia es debida fundamentalmente a que el umbral de pérdidas para adaptación de tasa se fija en 30%.

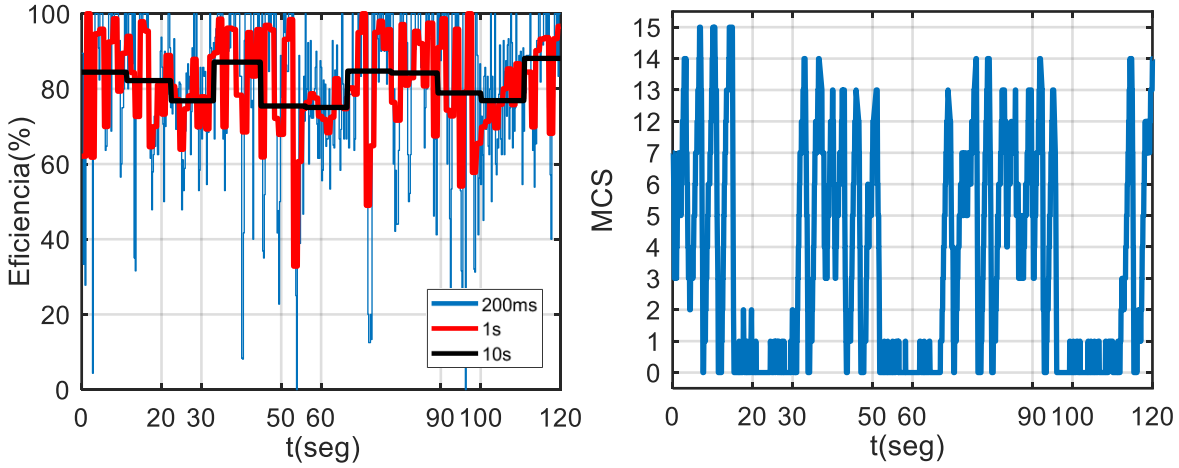


a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 45 Resistencia de ONOE con umbral de reintentos del 30% y ventana de 200 ms (100p/s, 1000B) a interferencias de 100 p/s y 1000B . Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

Visualizando los resultados de la evolución de la MCS (Fig.45.b), se observa una bajada de tasa muy rápida al combinar la ventana de 200 ms y el umbral de reintentos supera el 30%. Sin embargo, a diferencia de la configuración anterior, la definición de una ventana de W=200ms permite también recuperar la tasa original rápido. En líneas generales, podemos decir que esta configuración presenta unas prestaciones parecidas a las de la configuración W=200ms y R=50% pero ambas son peores que la configuración W=1s y R=50%. Por otra parte, reducir el umbral de reintentos al 30% con una ventana de W=1s es claramente inadecuado. A partir de este análisis, parece claro que con el patrón de generación de tráfico definido en la prueba, la mejor configuración en presencia de interferencias es la de ONOE original. Esto difiere de los resultados obtenidos anteriormente en los que en ausencia de este tipo de interferencias la mejor opción es la configuración con ventanas de menor tamaño W=200ms y R=30%.

En la Fig.46 se muestra caso de ARF con el mismo patrón de generación de tráfico. Se puede ver que la eficiencia (Fig. 46.a) en media se ve menos afectada que en los casos anteriores. Se encuentra sobre 80% y puede reducirse al 75% en presencia de interferencias. La poca variación en la eficiencia se debe a que este tipo de algoritmo reacción a la pérdida de paquetes analizando las transmisiones paquete a paquete. En presencia de interferencias reduce la tasa rápidamente al índice MCS 0 tal y como se ve en la Fig. 46.b, y una vez terminan las interferencias consigue volver a transmitir con tasas altas (MCS 14) de manera rápida al utilizar un umbral de éxitos pequeño para subir la tasa. Como puede observarse, igual que se ha observado en algunas de las configuraciones de ONOE, no es robusto ante la presencia de interferencias ya que reduce drásticamente la tasa.

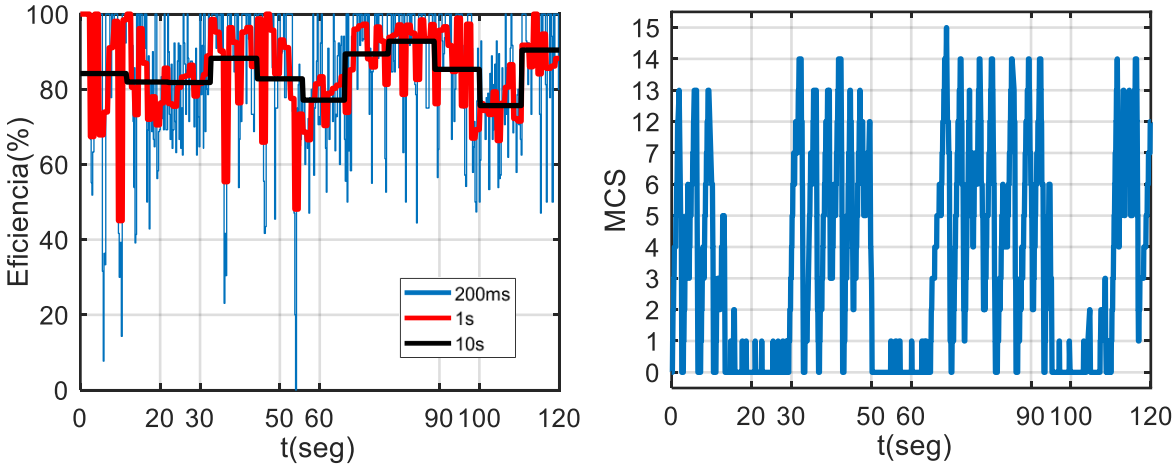


a)Eficiencia calculada por pasos (200ms, 1 y 10s)

b)Variación del índice MCS

Figura 46 Resistencia de ARF (100p/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

En la Fig.47 se analiza el caso adaptativo de ARF (AARF) para comprobar las diferencias que aparecen respecto al caso ARF. La variación de MCS (Fig.47.b) no presenta cambios apreciables respecto a ARF. Se ve afectada de la misma manera. También la eficiencia en la transmisión (Fig.47.a) se ve afectada de manera similar aunque destaca un ligero aumento de la eficiencia debido a los umbrales variables.

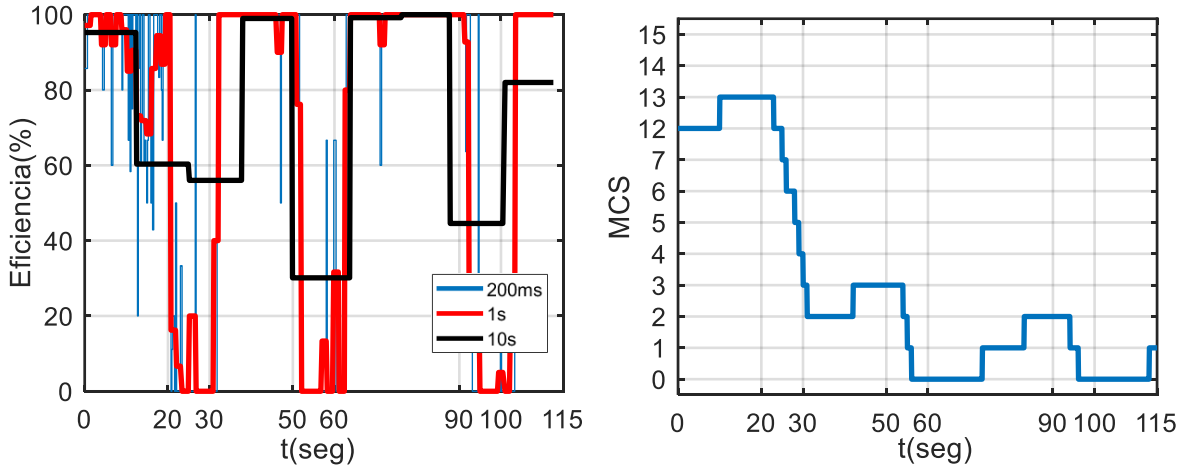


a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 47 Resistencia de AARF (100paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

Tras analizar el caso de interferencia de paquetes 1000 bytes a una tasa de 100 paq/s hacia un transmisor de 100 paq/s, ahora se analiza el efecto de dicha interferencia sobre la adaptación de tasa en un transmisor a 20 paq/s. El objetivo es comprobar cómo se comporta el algoritmo de control de tasa en función de las posibles tasas de paquetes de transmisor, que son diferentes dependiendo de la aplicación o servicio que un usuario utiliza.

En la Fig.48 se muestra la resistencia de ONOE original al reducir su tasa de transmisión a 20 paq/s.



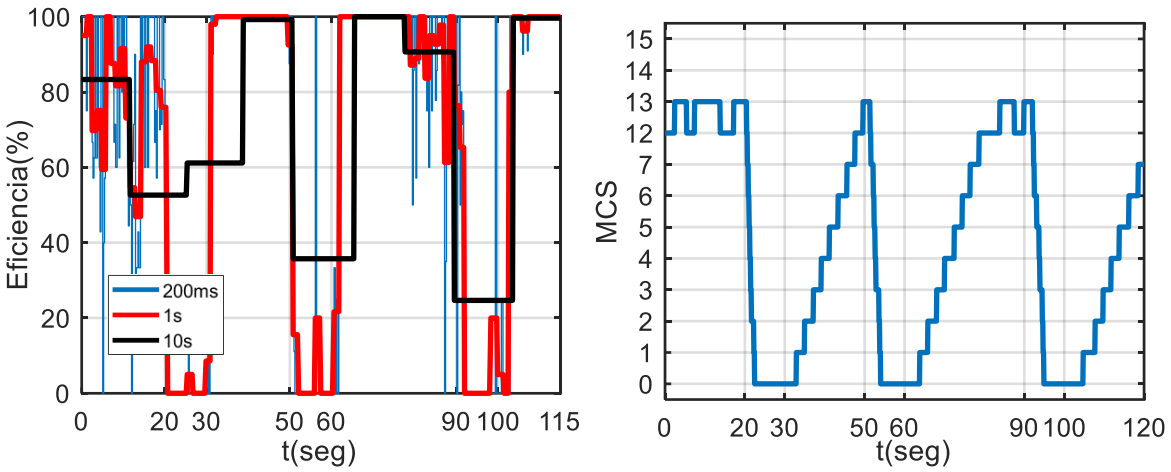
a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 48 Resistencia de ONOE original (20paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

Si se compara la Fig.48 con el caso de transmisión a 100 paq/s (Fig.42) se puede ver que está más afectado por las interferencias. La tasa de transmisión se reduce de manera más rápida (Fig.48.b). Esto

es debido a que al enviar 100 paquetes en una ventana de 1 s se estaba realizando un mayor promediado de los errores y éxitos. En este caso al enviarse únicamente 20 paquetes es más probable que ocasionalmente se alcance un 50% de pérdidas (10 paquetes interferidos) y, en consecuencia, baje la tasa.

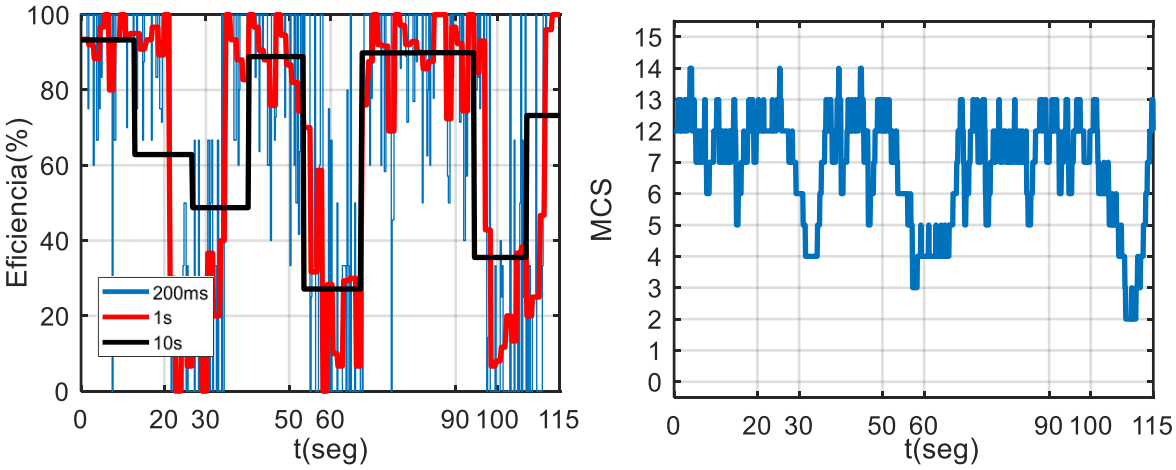
En la Fig.49 se muestra el resultado para ONOE reduciendo la ventana a 200 ms. Las tasas transmitidas (Fig.49.b) se ven afectadas de manera similar al caso con una transmisión de 100 paq/s (Fig.43.b). Destaca en este caso que la eficiencia (Fig.49.b) se reduce hasta el 20% en presencia de interferencias.



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 49 Resistencia de ONOE con ventana de 200ms (20paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

Para el caso de ARF el resultado se muestra en la Fig.50.



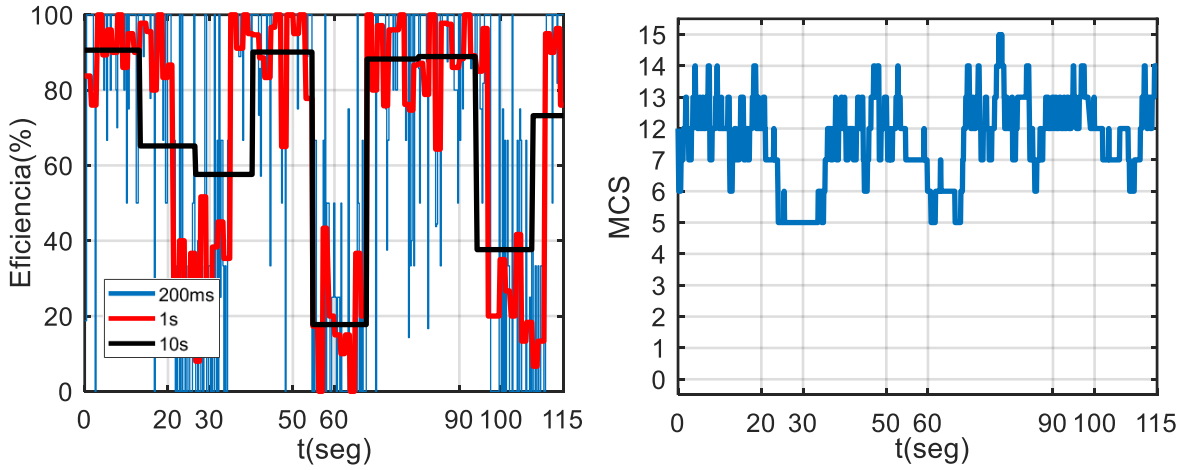
a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 50 Resistencia de ARF (20paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

Si se compara la variación del índice MCS (Fig.50.b) con el caso anterior de transmisor de 100 paq/s (Fig.46.b), se puede ver que la reacción ante las interferencias es menor que en el caso anterior. La tasa de transmisión no disminuye tan rápido (por el mero hecho de que la tasa de transmisión es menor y el intervalo entre colisiones más grande) y, por tanto, en el tiempo que dura la interferencia no alcanza el mínimo como en el caso anterior sino que puede disminuir hasta el índice MCS 2 en el peor caso. Sin

embargo, esto supone que la eficiencia baja, particularmente en los intervalos de interferencia. La elección de MCSs más altas reduce la probabilidad de que un paquete que colisiona con otro pueda ser detectado (por efecto captura) si tiene un nivel de SINR suficiente. En el caso de las modulaciones más bajas (MCS 0, 1, 2) es posible la detección. Como en la Fig.46.a el número de paquetes transmitidos en la ventana con interferencia es 5 veces mayor, aunque las primeras transmisiones sean erróneas, la eficiencia mejora ya que hay un número considerable de paquetes que pueden ser detectados directamente o con un número bajo de retransmisiones. Se puede ver que al reducir la tasa del transmisor a 20 paq/s (Fig.50.a), la eficiencia disminuye desde un 80% hasta un 30%. La eficiencia se ve gravemente afectada.

El resultado para AARF se muestra en la Fig.51, se puede ver que se comporta de manera similar a ARF (Fig.50). Destaca la variación del índice MCS (Fig.51.b), donde la tasa mínima en presencia de interferencias es el MCS 5 (frente al caso de 100paq/s donde alcanzaba MCS 0). La eficiencia (Fig.51.a) se ve afectada de la misma manera que en ARF (Fig.50.a).



a)Eficiencia calculada por pasos (200ms, 1 y 10s) b)Variación del índice MCS

Figura 51 Resistencia de AARF (20paq/s, 1000B) a interferencias de 100 p/s y 1000B. Eficiencia calculada en pasos: 200ms(azul), 1 s (rojo) y 10 s (negro) (a) y variación del índice MCS a lo largo del trayecto (b).

Las principales conclusiones de las pruebas de interferencia se muestran a continuación:

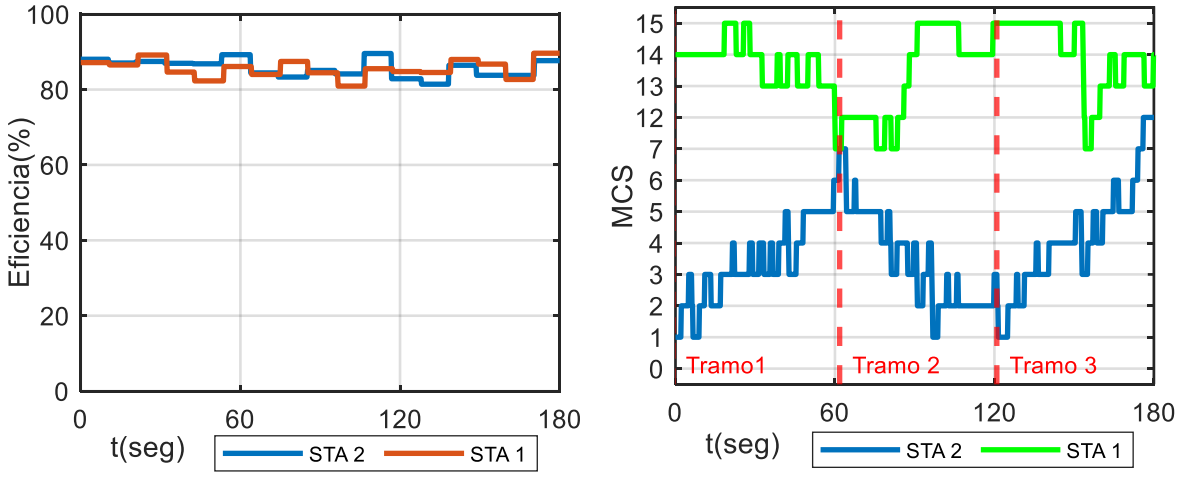
- A priori, interesa no reducir la tasa de transmisión en presencia de interferencias, ya que se ocupa el canal durante un mayor tiempo de transmisión (al transmitir con MCS más bajas) y es más probable ser interferido
- Sin embargo, reducir la tasa de paquetes del transmisor influye en la resistencia a interferencias de los algoritmos.
    - Reducir la tasa de paquetes del transmisor manteniendo la tasa de paquetes del interferente provoca que la eficiencia se reduzca. La variación de las tasas transmitidas se ven más afectada en el algoritmo ONOE, mientras que ARF y AARF logran transmitir con tasas más altas en presencia de interferencias.
- La ventana de actualización del algoritmo influye en la resistencia a interferencias. Ventanas grandes (hasta 1 s) como ONOE original permiten no reducir la tasa de manera rápida en presencia de interferencias, mejorando la tasa promedio de transmisión. Reducir la ventana hasta 200 ms como en la variante de ONOE puede provocar una reducción de tasa rápida que no es beneficiosa, aunque puede recuperar la tasa inicial al acabar la interferencia. El caso extremo es ARF y AARF que reducen la ventana al mínimo trabajando paquete a paquete, esto

produce que baje la tasa rápidamente en presencia de interferencias, con un aumento rápido de tasa al terminar la interferencia.

- Los umbrales de pérdidas tienen influencia en la resistencia a interferencias, se ha visto que umbrales reducidos como ONOE y su variante de umbral de reintentos del 30% causan que el algoritmo baje rápido de tasa en presencia de interferencias y además que se recupere lentamente por su ventana de actualización de 1 s. Esta combinación no es adecuada. Por otro lado, al añadir una ventana de 200 ms al caso de umbral de reintentos del 30% se puede ver que la subida de tasa es ahora rápida lo que produce un aumento de la eficiencia, por lo que puede ser una buena combinación de parámetros.

4.4. Pruebas con varias estaciones

En este apartado se muestra el funcionamiento del entorno con varias estaciones, donde el AP debe controlar la tasa de ambas al mismo tiempo. En la prueba se utilizan 2 estaciones. Dado lo acotado del escenario de evaluación en cuanto a número de STAs, el objetivo no es efectuar análisis de *throughput* en distintas condiciones de carga sino simplemente validar la capacidad de los desarrollos efectuados para manejar convenientemente y de forma individualizada la gestión de tasa de un conjunto de estaciones en función exclusivamente de sus condiciones de canal. La segunda estación (STA 2) está moviéndose de manera similar al apartado 4.2, la primera estación (STA 1) está en reposo en el punto 4. Partiendo de una situación de reposo de la STA 2 en el punto 8 y la STA 1 en el punto 4, la STA 2 comienza el movimiento del tramo 1 acercándose al AP desde el punto más alejado hasta el punto 2. Después en el tramo 2 se aleja del AP desde el punto 2 al 8. Por último, vuelve a acercarse al AP desde el punto 8 al 2. La situación de los puntos de medida se puede ver en los apartados anteriores. Se utiliza el algoritmo ONOE con ventana de 200 ms para las medidas. La tasa de paquetes enviados medida por el STA 1 es de 52 paq/s y el STA 2 envía 46 paq/s. La tasa individual es inferior a la usada en apartados anteriores porque el software utilizado no permite transmitir a ambas STAs con una tasa superior. Al enviar paquetes a dos estaciones a la vez, y debido a limitaciones en la función *sendp* de *phyton*, es más complejo fijar una tasa determinada como ocurría en las pruebas anteriores ya que se tienen que alternar para transmitir y la tasa real tiene variaciones. El tamaño de los paquetes enviados es L=1000 Bytes.



a) Eficiencia calculada en pasos de 10s b) Variación del MCS para la STA1(fija) y STA2(móvil)

Figura 52 Prueba con 2 estaciones: STA 2 (azul) en movimiento con 3 tramos (acercándose, alejándose y acercándose al AP) y STA 1 (verde) en reposo. Transmisión de 52 paq/s para el STA 1 y de 46 paq/s para el STA 2. La longitud del paquete es L=1000B. El algoritmo utilizado es ONOE con ventana de 200ms.

Con respecto a la estación en movimiento, se observa la lógica adaptación de tasa correspondiente a los tres tramos de movimiento, con resultados similares a los obtenidos para una sola STA en las Fig. 30.a y Fig. 30.b. En lo que respecta a la estación fija (STA1), la bajada de MCS es debida al paso entre la STA1 y el AP del carro (y persona que lo mueve) en el que se encuentra la STA 2 en determinados momentos. Esto introduce una atenuación no despreciable durante unos segundos. Este paso que altera el entorno como se muestra en el apartado 4.5. Se puede ver alrededor del primer minuto de medida (fin tramo 1) y del tercer minuto (fin tramo 3). En el primer minuto es más pronunciada porque la trayectoria del carro es de ida y vuelta. Como consecuencia de la reducción de la SINR se produce una adaptación de tasa.

4.5. Pruebas con variaciones del entorno (paso de personas)

Por último, se realiza una prueba para ilustrar el comportamiento del algoritmo cuando el entorno varía por el paso de personas por delante o por los alrededores de la STA. El experimento se muestra en la Fig. 53, se alternan periodos de silencio de 20 s con periodos de movimiento de personas delante y cerca de la STA durante 10 s. Se realiza una transmisión desde el AP a la STA de 100 paq/s con una longitud de L=1000 Bytes.

Figura 53 Escenario para la medida de las variaciones del entorno

En la Fig.54 se muestran los resultados para ONOE original (Fig.54.a) y con ventana de 200 ms (Fig.54.b). Se pueden ver los valles de pérdida de eficiencia en ONOE original cuando se realiza el movimiento. El efecto es similar al apartado anterior cuando ocurren interferencias. Sin embargo, ya que el origen de la degradación de la señal es diferente la reacción del algoritmo ideal no debe ser la misma. En este caso, el objetivo debe ser ajustar adecuadamente la MCS para mantener el nivel de eficiencia. Por el contrario, en el caso de colisiones, las pérdidas son inevitables y no debe reducirse la MCS

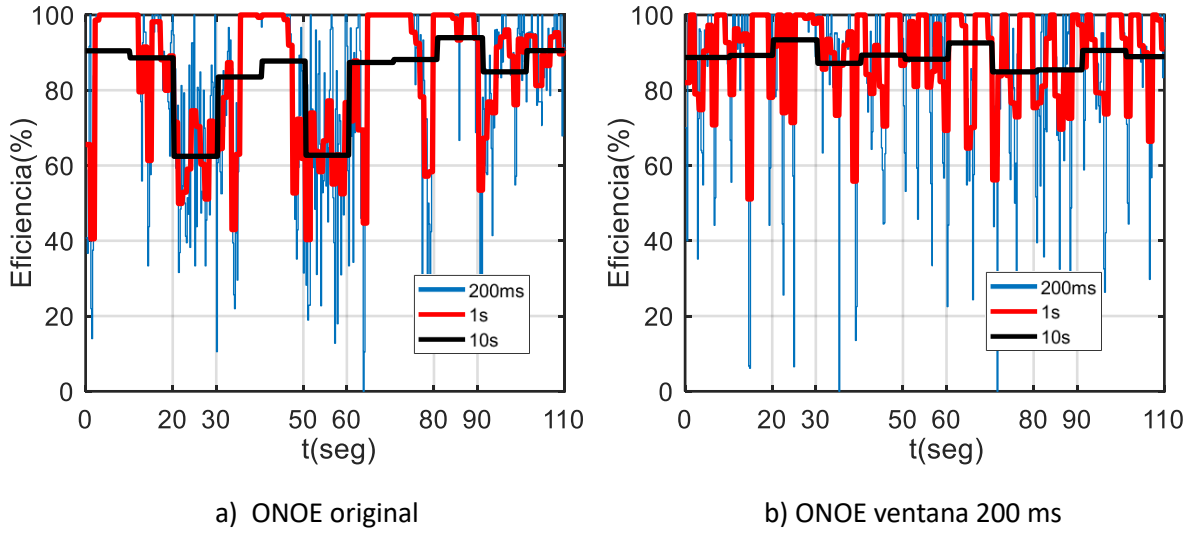


Figura 54 Paso de personas cerca de la STA: Eficiencia calculada en pasos (200ms, 1 y 10s) para ONOE original (a) y ONOE con ventana de 200 ms (b). Transmisión de 100 paq/s con longitud L=1000 Bytes desde el AP a la STA.

Como se ve en las gráficas, utilizando ventanas más pequeñas de 200 ms (Fig.54b) es posible adaptar rápidamente la tasa para compensar las pérdidas de canal provocadas por el movimiento de personas delante de la STA. No ocurre lo mismo en el caso del ONOE original donde responderá tras 1 s.

5. Conclusiones

Por último, en este capítulo se presentan las conclusiones del trabajo realizado, se exponen las lecciones aprendidas sobre el desarrollo de un entorno experimental Wi-Fi y se muestran las características que debe tener un algoritmo de control de tasa para ser eficiente en las diferentes situaciones que pueden aparecer a partir de las pruebas realizadas. Después, se propone un trabajo futuro a partir del entorno creado y se presentan los problemas que han aparecido durante la realización del trabajo.

Las principales conclusiones del trabajo son:

- Se puede crear un entorno experimental Wi-Fi para evaluar algoritmos de control de tasa e investigar nuevas funcionalidades utilizando herramientas hardware comerciales (tarjetas de red *off-the-shell*) y de software libre (*Linux* y *Python*). De esta manera, se acerca el desarrollo de funcionalidades Wi-Fi (realizado principalmente por empresas con software y hardware propietario) al público en general y a la investigación en universidades. Las tarjetas de red presentes (*out-of-kernel*) se pueden ver como una caja negra actualmente sin ofrecer capacidad para modificar sus funcionalidades, por lo que es necesario desarrollar escenarios como en este trabajo e ir construyendo funcionalidades tan básicas como pueden ser el control de tasa, el escaneo de red o la asociación y autenticación de los usuarios entre otras. Esto permitirá aportar funcionalidades inteligentes (y más complicadas) como la QoS o el *network slicing* a nuestros entornos SDWN.

- Se propone un método de inyección y monitorización de tráfico a partir de la librería *Scapy* y las cabeceras *Radiotap*. Son herramientas interesantes para la inyección de tráfico, aunque presentan limitaciones en la monitorización de red. No es posible probar condiciones de carga en la red por la limitación en la tasa de paquetes inyectados por segundo.

- Se han realizado pruebas para analizar el comportamiento de los algoritmos ante las distintas situaciones que pueden aparecer. Se han realizado pruebas con el algoritmo ONOE y variando sus parámetros de configuración y con el algoritmo ARF y su versión adaptativa AARF. Las conclusiones son las siguientes.

    - Sobre las pruebas en estático, tras estudiar ONOE y sus variantes y ARF se ha visto el efecto de los diferentes parámetros (ventana de actualización, umbrales de pérdidas para reducir la tasa y umbrales de éxitos para aumentar la tasa).

        - Ventanas de promediado estadístico de éxitos y errores o de actualización: Las ventanas de larga duración como en el caso de ONOE original (W=1 s) provocan una actualización lenta y una incapacidad para adaptarse a las variaciones rápidas. Reduciendo la ventana hasta 200 ms en ONOE el algoritmo puede adaptarse mejor a las variaciones del canal realizando un promediado menor de errores y éxitos, además de reducir la dispersión en la eficiencia del algoritmo. En el caso de algoritmos como ARF o AARF donde se trabaja contabilizando únicamente número de éxitos o fracasos consecutivos, se observa una reducción de la eficiencia. Por tanto, parece más adecuado llevar a cabo un promediando estadístico que aplicar un método determinista (ARF o AARF). Interesa un tamaño de ventana reducido, pero suficientemente grande para llevar a cabo un promediado estadístico.

- Umbral de éxitos o créditos para aumentar la tasa: Reducir el umbral de créditos o éxitos necesarios para aumentar la tasa provoca que se intente constantemente transmitir con tasas más altas que las recomendadas según el nivel de señal, lo que provoca una reducción de eficiencia. Como ejemplo tenemos ONOE donde reducir el umbral a 3 créditos en lugar de 10, se asocia con una pérdida de eficiencia en general. También se pueden comparar ARF y su versión adaptativa AARF que dobla el umbral de éxitos necesarios para subir en determinadas ocasiones. En general al aumentar el umbral la eficiencia aumenta como se ha visto.

- Umbral de pérdidas para reducir la tasa: En general al reducir el umbral de perdidas necesario para bajar de tasa los algoritmos consiguen una mayor eficiencia en su transmisión al evitar transmitir durante más tiempo con pérdidas elevadas, como si ocurre en ONOE original que acepta unas pérdidas de hasta el 50%. El aumento de eficiencia se consigue a costa de transmitir con índices MCS más bajos.

- Sobre las pruebas en movimiento, se dan dos situaciones dependiendo el sentido del movimiento:

  - La STA se aleja del punto de acceso: En este caso, la calidad de la señal se va degradando e interesa tener un algoritmo que penalice las situaciones con pérdidas y baje la tasa rápidamente, es decir, ventanas de actualización pequeñas y umbrales de pérdidas reducidos para bajar la tasa en estas situaciones.

  - La STA se acerca al punto de acceso: En este caso, la calidad de la señal va aumentando por lo que interesa adaptarse rápido para aprovechar la transmisión con índices MCS superiores. Nos interesa reducir la ventana de actualización también en estas situaciones.

- Sobre la resistencia a interferencias de los algoritmos:

  - La ventana de actualización del algoritmo tiene una influencia sobre la resistencia a interferencias del algoritmo.

    - Ventanas grandes (de hasta 1 s) como puede ser el caso de ONOE original evitan reducir la tasa rápidamente en presencia de interferencias, mejorando la tasa promedio de transmisión. Al reducir la ventana hasta 200 ms se provoca una reducción rápida de la tasa en presencia de interferencias, aunque el aumento de tasa tras las interferencias es rápido. Esta característica no es beneficiosa. En general no interesa reducir la tasa en presencia de interferencias. Por tanto, en este escenario la configuración más apropiada es la del ONOE original frente a la nueva parametrización con W=200ms que se podría recomendar en escenarios libres de interferencias. El caso extremo se produce en los algoritmos ARF y AARF, que trabajan paquete a paquete, lo cual es equivalente a tener una ventana de

actualización de 1-2 paquetes. Esto produce que la tasa se desplome en presencia de interferencias. Tras desaparecer la interferencia vuelve a la tasa original de inmediato.

- Los umbrales de pérdidas afectan a la resistencia a interferencias. Se ha comprobado que umbrales más reducidos para el ONOE, por ejemplo, un umbral de reintentos del 30% causan que el algoritmo baje más rápidamente su tasa cuando empieza el intervalo de interferencia. Además, si se combina con una ventana de actualización grande (1 s) provoca que se recupere lentamente.

5.1. Trabajo Futuro

Se pueden realizar varias líneas de trabajo a partir de este:

- Investigar otras opciones para la monitorización de red u obtención de estadísticas y superar las limitaciones sobre la tasa de inyección de paquetes que se han sido puestas al descubierto.

- Desarrollar nuevos algoritmos de control de tasa teniendo en cuenta las lecciones aprendidas durante la realización del trabajo. Con el avance del hardware de las tarjetas de red pueden ser interesantes los algoritmos basados en redes neuronales.

- Introducir cambios en el entorno desarrollado que posibiliten actuar en función del tráfico inyectado y sus connotaciones temporales. Ello supone diferenciar el tráfico (sistema de colas) y ajustar en función del tipo de servicio a ofrecer y la calidad que se pretende dar (*Quality of Service*, QoS)

5.2. Problemas encontrados

La principal parte del trabajo consiste en poner en marcha un escenario que trabaja en tiempo real, por lo que han aparecido gran variedad de problemas durante la realización. Los problemas principales que se han encontrado son:

- La inyección de tramas puede parecer sencilla, pero hay que tener en cuenta los diferentes parámetros de las tarjetas de red involucradas en la transmisión y probarlos. La capacidad real de transmisión de la tarjeta (anchos de banda, frecuencias, tecnología MIMO según el número de antenas o tecnologías opcionales como STBC y LDPC) puede no corresponderse con lo indicado por el fabricante.

- Se han encontrado problemas de monitorización de red con la librería Scapy. Cuando se transmiten tasas altas de paquetes por segundo no todos los paquetes son procesados por la función. Esto limito las tasas de inyección a 100 paquetes por segundo. El problema aparecía principalmente en puntos con mala cobertura en los que se enviaban gran cantidad de reintentos que aumentaban la tasa global de paquetes por segundo.

- Aparecieron problemas al intentar obtener las estadísticas teniendo en cuenta los tiempos de recepción según el tipo de paquete (Datos, reintentos y ACKs), que son diferentes para dar prioridad a los ACKs según la técnica de coordinación DCF. Al principio se consideró esperar un tiempo máximo ligeramente superior a SIFS (tiempo de respuesta de un ACK definido en DCF) para confirmar el éxito o el fallo en la transmisión un paquete de datos, pero se encontraron dificultades en la medida del tiempo. Finalmente se propuso una máquina de estados y se utilizaron otras métricas para saber si un ACK ha llegado a tiempo para confirmar un paquete de datos.

Referencias

[1] Bravo Bobadilla, S. (2021). Diseño de redes IEEE 802.11 en entornos extremos: Alta densidad. (TFG Grado en Ingeniería de Sistemas de Telecomunicación)

[2] Cano Pérez, L. A. (2013). Proyecto de fin de carrera: configuración y evaluación de redes 802.11n. (TFG Ingeniero de Telecomunicación, Universidad de Granada).

[3] Bicket, J. C. (2005). Bit-rate selection in wireless networks (Massachusetts Institute of Technology). http://hdl.handle.net/1721.1/34116

[4] Pathak, P. H. (2019). Diapositivas del curso: Wireless and mobile computing.

[5] del Prado, J., Choi, S. (2003). Link Adaptation Strategy for IEEE 802.11 WLAN via Received Signal Strength Measurement.

[6] Shaoen Wu, Saad Biaz, Bing Qi, Kehao Zhang (2007). BARA: A Sender Based Rate Adaptation in Wireless Networks.

[7] Grünblatt, R. (2021). From WiFi performance evaluation to controlled mobility in drone networks. http://www.theses.fr/2021LYSE1002/document

[8] Queiros, R., Almeida, E. N., Fontes, H., Ruela, J., & Campos, R. (Jan 1, 2022). Wi-fi rate adaptation using a simple deep reinforcement learning approach.

[9] Deep reinforcement learning. https://en.wikipedia.org/wiki/Deep_reinforcement_learning

[10] Reinforcement learning. https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292

[11] Vanhoef, M., Jiao, X., Liu, W., & Moerman, I. (Jan 24, 2023). Testing and improving the correctness of wi-fi frame injection.

[12] Morrow. (2023). Monitor mode. https://github.com/morrownr/Monitor_Mode

[13] Radiotap. https://www.radiotap.org/

[14] USB wi-fi chipsets.https://github.com/morrownr/USB-WiFi/blob/main/home/USB_WiFi_Chipsets.md

[15] Interface statistics.https://docs.kernel.org/networking/statistics.html

ANEXO I: Tabla de índices MCS para 802.11n/ac

802.11n and 802.11ac MCS, SNR and RSSI

wirelessLAN PROFESSIONALS

HT MCS	VHT MCS	Modulation	Coding	20MHz				40MHz				80MHz				160MHz			
				Data Rate		Min. SNR	RSSI	Data Rate		Min. SNR	RSSI	Data Rate		Min. SNR	RSSI	Data Rate		Min. SNR	RSSI
				800ns	400ns			800ns	400ns			800ns	400ns			800ns	400ns		
1 Spatial Stream																			
0	0	BPSK	1/2	6.5	7.2	2	-82	13.5	15	5	-79	29.3	32.5	8	-76	58.5	65	11	-73
1	1	QPSK	1/2	13	14.4	5	-79	27	30	8	-76	58.5	65	11	-73	117	130	14	-70
2	2	QPSK	3/4	19.5	21.7	9	-77	40.5	45	12	-74	87.8	97.5	15	-71	175.5	195	18	-68
3	3	16-QAM	1/2	26	28.9	11	-74	54	60	14	-71	117	130	17	-68	234	260	20	-65
4	4	16-QAM	3/4	39	43.3	15	-70	81	90	18	-67	175.5	195	21	-64	351	390	24	-61
5	5	64-QAM	2/3	52	57.8	18	-66	108	120	21	-63	234	260	24	-60	468	520	27	-57
6	6	64-QAM	3/4	58.5	65	20	-65	121.5	135	23	-62	263.3	292.5	26	-59	526.5	585	29	-56
7	7	64-QAM	5/6	65	72.2	25	-64	135	150	28	-61	292.5	325	31	-58	585	650	34	-55
	8	256-QAM	3/4	78	86.7	29	-59	162	180	32	-56	351	390	35	-53	702	780	38	-50
	9	256-QAM	5/6			31	-57	180	200	34	-54	390	433.3	37	-51	780	866.7	40	-48
2 Spatial Streams																			
8	0	BPSK	1/2	13	14.4	2	-82	27	30	5	-79	58.5	65	8	-76	117	130	11	-73
9	1	QPSK	1/2	26	28.9	5	-79	54	60	8	-76	117	130	11	-73	234	260	14	-70
10	2	QPSK	3/4	39	43.3	9	-77	81	90	12	-74	175.5	195	15	-71	351	390	18	-68
11	3	16-QAM	1/2	52	57.8	11	-74	108	120	14	-71	234	260	17	-68	468	520	20	-65
12	4	16-QAM	3/4	78	86.7	15	-70	162	180	18	-67	351	390	21	-64	702	780	24	-61
13	5	64-QAM	2/3	104	115.6	18	-66	216	240	21	-63	468	520	24	-60	936	1040	27	-57
14	6	64-QAM	3/4	117	130.3	20	-65	243	270	23	-62	526.5	585	26	-59	1053	1170	29	-56
15	7	64-QAM	5/6	130	144.4	25	-64	270	300	28	-61	585	650	31	-58	1170	1300	34	-55
	8	256-QAM	3/4	156	173.3	29	-59	324	360	32	-56	702	780	35	-53	1404	1560	38	-50
	9	256-QAM	5/6			31	-57	360	400	34	-54	780	866.7	37	-51	1560	1733	40	-48
3 Spatial Streams																			
16	0	BPSK	1/2	19.5	21.7	2	-82	40.5	45	5	-79	87.8	97.5	8	-76	175.5	195	11	-73
17	1	QPSK	1/2	39	43.3	5	-79	81	90	8	-76	175.5	195	11	-73	351	390	14	-70
18	2	QPSK	3/4	58.5	65	9	-77	121.5	135	12	-74	263.3	292.5	15	-71	526.5	585	18	-68
19	3	16-QAM	1/2	78	86.7	11	-74	162	180	14	-71	351	390	17	-68	702	780	20	-65
20	4	16-QAM	3/4	117	130	15	-70	243	270	18	-67	526.5	585	21	-64	1053	1170	24	-61
21	5	64-QAM	2/3	156	173.3	18	-66	324	360	21	-63	702	780	24	-60	1404	1560	27	-57
22	6	64-QAM	3/4	175.5	195	20	-65	364.5	405	23	-62			26	-59	1580	1755	29	-56
23	7	64-QAM	5/6	195	216.7	25	-64	405	450	28	-61	877.5	975	31	-58	1755	1950	34	-55
	8	256-QAM	3/4	234	260	29	-59	486	540	32	-56	1053	1170	35	-53	2106	2340	38	-50
	9	256-QAM	5/6	260	288.9	31	-57	540	600	34	-54	1170	1300	37	-51			40	-48