ISSN 1688-2806

Universidad de la República
Facultad de Ingeniería

Aprendizaje profundo para la asignación de recursos en redes 5G

Tesis presentada a la Facultad de Ingeniería de la Universidad de la República por

Lucas Inglés

en cumplimiento parcial de los requerimientos para la obtención del título de Magister en Ingeniería Eléctrica.

Directores de Tesis

Claudina Rattaro ..... Universidad de la República
Pablo Belzarena ..... Universidad de la República

Tribunal

Alberto Castro ..... Universidad de la República
Germán Capdehourat ..... Universidad de la República
Matías Richart ..... Universidad de la República

Director Académico

Claudina Rattaro ..... Universidad de la República

Montevideo
martes 28 marzo, 2023

Aprendizaje profundo para la asignación de recursos en redes 5G, Lucas Inglés.

ISSN 1688-2806

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.1).
Contiene un total de 123 páginas.
Compilada el martes 28 marzo, 2023.
http://iie.fing.edu.uy/

Agradecimientos

Creo que una persona no puede alcanzar el éxito sin el apoyo y el trabajo en equipo de las personas que la rodean. Lo mismo sucede con este tipo de proyectos, los cuales son el resultado de la intervención de muchas personas.

En primer lugar, quiero expresar mi profundo agradecimiento a mis tutores, Claudina y Pablo. Tuve la suerte que en 2020 Claudina me acercara esta propuesta de proyecto sobre redes móviles. A lo largo de este tiempo, ella no solo demostró ser una excelente investigadora e ingeniera, sino también una gran mentora y es la principal impulsora de mi carrera en la investigación.

Asimismo, quiero destacar el papel esencial que Pablo desempeñó en el desarrollo y el desenlace de mi proyecto. Su experiencia en redes inalámbricas y su compromiso con el proyecto fueron fundamentales para tomar decisiones clave y lograr los objetivos propuestos.

Por otro lado, quisiera agradecer a mis padres Verónica y Omar y a toda mi familia por su apoyo incondicional en todas las nuevas experiencias que decidí emprender.

También quiero agradecer al Instituto de Ingeniería Eléctrica en su totalidad, y en especial al departamento de telecomunicaciones. Agradezco a Federico y Gabriel por brindarme las condiciones necesarias para desarrollar mi formación personal y profesional. A mis compañeros del Instituto, quiero agradecer especialmente a Camilo por todos los consejos que me brindó tanto para la documentación como para el código. También agradezco a Romina por su amabilidad al ofrecerse a ayudarme con las correcciones necesarias. Finalmente, quiero mencionar a Martín por su apoyo y compañía durante todo el proceso de la tesis.

Por último, un agradecimiento muy especial a mi pareja Emilia, quien me acompañó de cerca en todo este proceso y dedicó su tiempo y esfuerzo para ayudarme con la organización y la escritura de este documento.

Este trabajo se enmarcó en los proyectos de investigación “Inteligencia Artificial aplicada a redes 5G”(Agencia Nacional de investigación e Innovación, Uruguay, Project FMV_1_2019_1_155700), “Herramientas de simulación de redes móviles de futura generación”(FVF-2021-128– DICYT, Fondo Carlos Vaz Ferreira, Convocatoria 2021, Dirección Nacional de Innovación, Ciencia y Tecnología, Ministerio de Educación y Cultura, Uruguay) y “Convergencia entre redes 5G/6G y redes ópticas: un enfoque holístico”(Proyectos de I+D 2022, Comisión Sectorial de Investigación Científica, UdelaR).

Mi más sincero agradecimiento a todos los involucrados en este proyecto, quienes

hicieron posible la realización de esta investigación.

Ningún viento es favorable
para quien no sabe adonde va.

Séneca

Esta página ha sido intencionalmente dejada en blanco.

Resumen

La creciente demanda de conectividad se manifiesta en la cantidad de tráfico y la tendencia a ampliar el alcance de las redes móviles para satisfacer nuevas necesidades. Las redes móviles de quinta generación (5G) se han convertido en el nuevo paradigma de las redes de comunicación con nuevas características y servicios. Estas redes deben permitir la coexistencia de clientes con diferentes requerimientos de servicio, mientras garantizan la autonomía e independencia entre ellos, lo cual constituye un desafío para la asignación de recursos. *Network Slicing* se presenta como la herramienta clave para abordar este problema. Implica el particionamiento de los recursos de radio con el objetivo de cumplir con el nivel de servicio acordado con cada grupo de usuarios.

El objetivo general del presente trabajo es analizar e implementar algoritmos de asignación de recursos basados en aprendizaje profundo para redes 5G, bajo el paradigma de *network slicing*. Para ello se utiliza el simulador Py5cheSim, desarrollado en el Instituto de Ingeniería Eléctrica de UdelaR.

Como resultado principal se destaca el análisis comparativo de dos algoritmos de asignación de recursos entre *slices* que se basan en el aprendizaje profundo y se evalúan bajo un mismo escenario operativo. Así, se analiza su potencial para ser implementados en una red real y se llega a la conclusión de que uno de ellos es la opción más adecuada. Dicha elección se fundamenta en una serie de criterios técnicos y de rendimiento, que demuestran las ventajas que ofrece en términos de eficiencia y precisión en la asignación de recursos.

Por otra parte, se realizan aportes al simulador, tales como la incorporación de nuevos perfiles de tráfico, el ajuste de mecanismos de funcionamiento y la creación de una librería para facilitar el desarrollo de nuevos algoritmos. Muchos de estos aportes enriquecen los resultados principales, pues habilitan la experimentación en diversos entornos de tráfico.

Esta página ha sido intencionalmente dejada en blanco.

Tabla de contenidos

Acrónimos

- **2G**: Segunda generación de redes móviles
- **3G**: Tercera generación de redes móviles
- **3GPP**: 3rd Generation Partnership Project
- **4G**: Cuarta generación de redes móviles
- **5G**: Quinta generación de redes móviles
- **5GCN**: 5G Core Network.
- **AI**: Artificial Intelligence
- **AMC**: Adaptive Modulation and Coding
- **ANTEL**: Administración Nacional de Telecomunicaciones
- **BWP**: Bandwidth Part
- **CORESET**: Control Resource Set
- **CQI**: Información de Calidad del Canal
- **CRB**: Common Resource Block
- **CSI-RS**: Indicador del Estado del Canal
- **DCI**: Downlink Control Information
- **DRL**: Deep Reinforcement Learning
- **eMBB**: Enhanced Mobile Broadband
- **GCD**: Greatest Common Divisor
- **HARQ**: Hybrid Automatic Repeat Request
- **IIE**: Instituto de Ingeniería Eléctrica
- **IoT**: Internet of Things
- **kHz**: Kilohertz

- **LCM**: Least Common Multiple
- **LTE-M**: Long-Term Evolution para Máquinas
- **LTE**: Long-Term Evolution
- **MCS**: Esquema de Modulación y Tasa de Código Óptimas
- **MHz**: Megahertz
- **NR**: New radio
- **OFDM**: Orthogonal Frequency-Division Multiplexing
- **PDCCH**: Physical Downlink Control Channel
- **PDSCH**: Physical Downlink Shared Channel
- **PUSCH**: Physical Uplink Shared Channel
- **RB**: Resource Block
- **RE**: Resource Element
- **SCS**: Subcarrier spacing
- **SINR**: Relación Señal a Ruido más Interferencia
- **UCI**: Uplink Control Information
- **UE**: User Equipment

PARTE I:

INTRODUCCIÓN

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 1

Motivación

En los últimos treinta años, las tecnologías de la información y la comunicación (TICs) han experimentado una evolución sin precedentes, con una dinámica impulsada por la sociedad y sus necesidades de consumo en constante cambio. Así, han respondido a las exigencias contemporáneas, avanzando hacia redes móviles más complejas y enfocadas en la transmisión de datos. Como resultado, hoy disponemos de la red celular de cuarta generación, que cumple con las necesidades actuales de manera efectiva.

A pesar de los avances en tecnología, el mercado se está desplazando hacia un escenario con mayores demandas hacia la red. Se espera que el uso de la red celular por parte de nuevas aplicaciones -automóviles inteligentes, aplicaciones médicas, gemelos digitales, entre otros- aumente las exigencias en términos de tasa de datos, retardo y calidad de servicio. Estas futuras exigencias superarán las capacidades de la infraestructura de la red de cuarta generación actual.

La tecnología de las redes móviles está avanzando hacia la era de la quinta generación (5G). Esta tecnología es estandarizada por la 3GPP (*3rd Generation Partnership Project*) a nivel internacional en colaboración con la industria inalámbrica. Su objetivo es establecer especificaciones comunes para el desarrollo, implementación y despliegue de sistemas de comunicación 5G. El organismo 3GPP publica las especificaciones del estándar, incluyendo procedimientos y capacidades. La primera publicación ocurrió en 2018 con la especificación "Release 15" y, desde entonces, continúa su desarrollo. Como se puede observar en la Figura 1.1, se espera un crecimiento sustancial en la cantidad de dispositivos utilizando esta tecnología en los próximos años.

La tecnología 5G ofrece importantes avances en la velocidad de transmisión, latencia, cobertura y movilidad. Esto es posible gracias a la nueva infraestructura de radio (NR) y a un nuevo núcleo de red (5GCN). La red 5G alcanza su alto desempeño mediante la implementación de una amplia gama de características innovadoras; incorporando los avances científicos y la tecnología de punta alcanzada hasta el momento. Sin embargo, su reciente surgimiento trae consigo una miríada de desafíos que aún están sin resolver. Entre otras cosas, no basta con una avanzada infraestructura, sino que es necesario contar con algoritmos inteligentes que desempeñen una asignación eficaz de los recursos. Estos algoritmos no son estandarizados

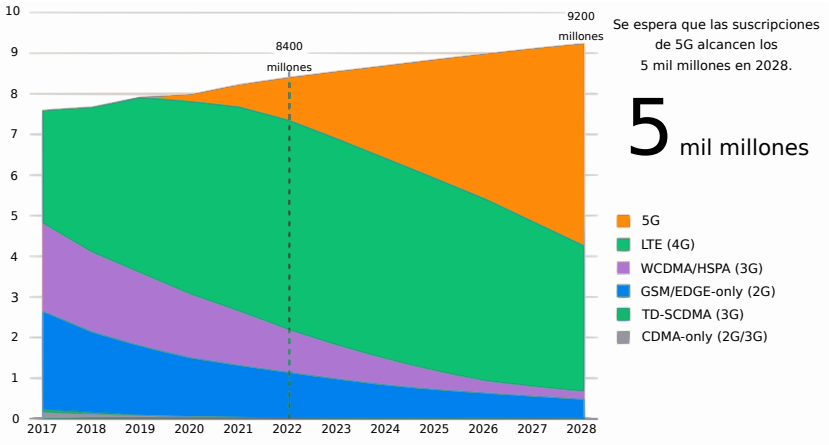


Figura 1.1: Sucripciones de tecnología celular por mil millones. Adaptada de [12].

sino producto de múltiples investigaciones y su adopción e implementación depende de cada proveedor de equipamientos de red.

Por otro lado, el surgimiento de 5G coincide con el momento culmen en la evolución del paradigma del aprendizaje por computadora. Su evolución se ve impulsada por avances significativos en la capacidad de procesamiento de datos y en la investigación de nuevas técnicas de aprendizaje. Entre estas técnicas, destaca el aprendizaje por refuerzo, el cual ha experimentado un considerable crecimiento. Cada vez son más comunes las aplicaciones basadas en inteligencia artificial (AI por su sigla en inglés) y todavía no hay horizonte para su crecimiento. En este contexto, y en base a lo mencionado anteriormente, lograr la integración entre 5G y estas nuevas tecnologías emergentes brindan una gran oportunidad para el desarrollo de soluciones eficaces y sostenibles. En particular, la integración en la inteligencia de la red y en la gestión de sus recursos es un camino prometedor que amerita esfuerzos de investigación y desarrollo por parte de la academia.

Por su parte, el despliegue de una red celular implica afrontar grandes costos, así como una extensa etapa de planificación y relevamiento. Es por ello, un campo difícil de acceder desde los sectores de investigación que buscan involucrarse en la tecnología. En la mayoría de los casos la obtención de datos no es posible y aún menos la implementación y prueba de cualquier nueva característica. Es por ello indispensable contar con algún medio alternativo en el cual simular escenarios de prueba y comprobar y comparar el rendimiento de distintas implementaciones. Con lo reciente que es la tecnología de quinta generación, las plataformas de simulación son escasas y limitadas, pero constituyen un insumo fundamental.

En este marco, el objetivo del presente trabajo es implementar y evaluar algoritmos del estado del arte en la gestión de recursos para redes 5G, específicamente aquellos que utilizan aprendizaje por refuerzo profundo (DRL por su sigla en inglés). Para ello, se utilizará un simulador de código abierto y ampliaremos sus funcionalidades para obtener resultados más realistas. De este modo, se proporcionará una base sólida para futuras investigaciones en el campo.

1.1. Antecedentes

Una amplia gama de investigaciones precede nuestro trabajo. En primer lugar, se describe en [54] un algoritmo LSTM diseñado para predecir la capacidad de un proveedor para satisfacer la demanda, considerando las condiciones del canal y los recursos asignados. Por otro lado, el desafío de la gestión de recursos ha sido abordado utilizando una gran variedad de métodos. Los trabajos [56], [11] y [33] buscan una solución a partir de heurísticas, mientras que [57] utiliza algoritmos genéticos. Asimismo, son numerosos los trabajos que utilizan aprendizaje de máquina para resolver los problemas de asignación. En [13], [36], [48] y [34] se realizan revisiones sobre técnicas de aprendizaje por refuerzo enfocados en redes de datos, se analizan distintos enfoques y se estudia su implementación. Además, comparan estas técnicas con el estado del arte y describen sus ventajas y desventajas.

Yendo a investigaciones más aplicadas, en [37] se presenta una implementación de aprendizaje por refuerzo profundo para la orquestación de una red 5G, utilizando un concepto ya desarrollado en la literatura pero recientemente aplicado a este contexto: el aprendizaje distribuido. Por otro lado, los trabajos [35], [10] y [49] presentan algoritmos concretos sobre la gestión de recursos, siendo los dos primeros los implementados en el presente trabajo. En ellos se plantea la asignación de recursos teniendo como objetivo cumplir los requerimientos de servicio de diferentes grupos de usuarios. En particular, en [10] se utiliza el mencionado concepto de aprendizaje distribuido. Profundizaremos sobre este tema más adelante en el trabajo.

Asimismo, la tecnología de quinta generación es objeto de estudio de gran relevancia en el Instituto de Ingeniería Eléctrica (IIE) y una variedad de investigaciones se han desarrollado en esta línea. En [20] se realiza un estudio sobre técnicas de asignación de recursos y se presenta una forma alternativa para gestionarlos. Al mismo tiempo, un simulador de 5G nace del trabajo en [45]. Esta plataforma sería utilizada como base de desarrollo para múltiples investigaciones, tales como [15], [25] y [22]. En esta última se realiza un esfuerzo por incorporar nuevas características al simulador mencionado, el cual constituye la principal herramienta utilizada para la presente investigación.

Es importante mencionar que, así como se realizan esfuerzos por crear entornos de simulación de redes móviles, hay una nueva vertiente que busca implementar este tipo de redes utilizando radio definida por *software*. Los dos proyectos de mayor relevancia en este ámbito son OpenAirInterface [42] y srsRAN [5]. Este tipo de plataformas brindan la posibilidad de estudiar escenarios más realistas que las simulaciones hechas en *software*. Además, el IIE se encuentra actualmente desarrollando una maqueta de red 5G a través de ellas.

1.2. 5G en el Uruguay

En nuestro país, se está realizando una transición a esta nueva tecnología de redes móviles y hay interés tanto de la industria como por parte del gobierno para su despliegue.

La Administración Nacional de Telecomunicaciones (ANTEL) lanzó con éxito su primera red comercial de 5G en abril de 2019, convirtiéndose en la primera operadora en hacerlo en América Latina. Este importante hito en la tecnología de comunicación fue posible gracias al despliegue de la red en zonas de Maldonado y Nueva Palmira, utilizando la banda de 28 GHz.

El cinco de julio de 2022 la Unidad Reguladora de Servicios de Comunicaciones [6] establece en los decretos 113 [16], 114 [17] y 115 [18] la autorización a las tres mayores operadoras del país a desarrollar pruebas técnicas. Si bien el decreto permite pruebas hasta setiembre del mismo año, abre las puertas a la apropiación de esta tecnología. A fines del año 2022 se publicó el decreto número 83 [38] en el cual se dicta la licitación de tres bloques de espectro a realizarse en marzo de 2023. Estos bloques pertenecen a la banda de 3.5 GHz y se utilizarán para servicios de redes 5G. Una vez licitada la banda de frecuencias ya será posible contar con esta red a nivel comercial.

Las operadoras han evidenciado una predisposición hacia el despliegue de estas redes. Más aún, se maneja la posibilidad de contar con una infraestructura única. Durante una entrevista, un representante de la dirección de una de las empresas se pronunció con las siguientes palabras:

> Dadas las características de esta nueva tecnología en términos de la cantidad de radiobases que requiere, consideramos conveniente evaluar la oportunidad de desplegar en Uruguay una 'red 5G única' para el país, que utilicemos todos los operadores del mercado (a diferencia de los despliegues de 2G, 3G y 4G, en los que cada empresa desarrolló su propia red). En el marco de políticas de sostenibilidad, hay experiencias a nivel mundial de iniciativas de compartición de infraestructura que han permitido, de esta forma, despliegues más eficientes, que posibilitan, además, llegar a zonas geográficas desatendidas, reduciendo el impacto de la infraestructura en el ambiente y alcanzando un adecuado equilibrio entre la cooperación y la competencia [43].

Esta opinión es compartida por los proveedores de conectividad y denota la necesidad de algoritmos eficientes que permitan la gestión efectiva de los recursos. En este enfoque novedoso de infraestructura conjunta, es determinante el concepto de *network slicing*. Todo ello es un indicador de la relevancia de investigaciones como la del presente documento. Uruguay no queda al margen del avance tecnológico, y es fundamental para continuar su desarrollo la investigación y la formación de recursos humanos en la materia.

1.3. Contribuciones

El presente trabajo de maestría contribuye significativamente desde múltiples aristas. En primer lugar, se realiza diversos aportes a la herramienta de simulación existente, como se describirá más adelante. En particular, se destaca el diseño e implementación de una biblioteca asociada, que se orienta al desarrollo de algoritmos de asignación de recursos. Como producto de estos resultados se publicó el trabajo [46]. Asimismo, se contribuye al simulador en tanto se solucionan bugs y se mejoran varias funcionalidades.

Por otra parte, se realiza el análisis, implementación y comparación de dos algoritmos de asignación de recursos basados en aprendizaje por refuerzo. El aporte radica en el análisis de cada algoritmo mediante la evaluación de su desempeño y escalabilidad, entre otros. Los resultados obtenidos en esta investigación se han difundido y presentado en importantes eventos académicos internacionales. En primer lugar, se han presentado en el Encuentro Latinoamericano sobre Inteligencia Artificial, KHIPU, celebrado en Montevideo en 2023 [32]. Además, se han compartido en el evento Network Traffic Measurement and Analysis, organizado por la TMA PhD School en los Países Bajos en 2022 [51], así como en la Jornada de Jóvenes Investigadores celebrada en Bolivia en el mismo año [14]. Asimismo, se ha publicado un trabajo sobre el simulador y los aportes realizados en el CLEI Electronic Journal (ver referencia [46]).

Finalmente, resulta pertinente subrayar que el presente trabajo se realiza en el marco de dos proyectos de investigación, lo cual trae consigo un valioso intercambio con diversos actores. También se destacan como productos colaterales dos proyectos de fin de carrera que ha dirigido el autor; uno sobre mejoras a la herramienta de simulación y otro sobre la implementación de una maqueta de red 5G basada en SDR.

1.4. Organización del documento

En vista del objetivo de investigación descrito en la Sección 1, la documentación ha sido dividida en tres secciones principales: “Introducción”, “Entorno de Simulación” y “Aportes”. Esta estructuración permitirá abordar los objetivos planteados de manera clara y efectiva, y garantizará la coherencia y cohesión del estudio.

En los capítulos dos y tres de la introducción, se presentarán los conceptos y fundamentos teóricos en los que se basa nuestro trabajo. En el capítulo dos, se introducirá la tecnología 5G desde una perspectiva técnica, desde los casos de uso hasta los aspectos importantes que intervienen en la comunicación en esta tecnología, y se definen el problema de la asignación de recursos y algunas de sus implementaciones. En el capítulo tres, se introducirá al lector en el paradigma del aprendizaje por refuerzo, construyendo desde sus conceptos básicos hasta llegar a la familia de algoritmos utilizados en nuestra investigación.

El segundo bloque titulado “Entorno de Simulación” tiene como objetivo introducir la herramienta utilizada en el estudio. Este incluye un solo capítulo que comienza con una discusión sobre la pertinencia de utilizar un simulador de redes móviles, presenta los simuladores más relevantes a nivel internacional y finaliza con la introducción y justificación de la elección del simulador empleado en el estudio.

El tercer bloque de la investigación, “Aportes”, está compuesto por tres capítulos: “Aportes al simulador”, “Algoritmos de asignación de recursos” y “Conclusiones”. En el primer capítulo se describen las modificaciones realizadas al simulador y las nuevas características implementadas. En el segundo, se aborda el problema de la asignación de recursos desde una perspectiva práctica y se presentan los resultados de dos implementaciones de algoritmos basados en el estado del arte. Finalmente, el último capítulo presenta las conclusiones generales del estudio, proporcionando

una visión hacia futuras investigaciones.

Capítulo 2

Tecnología 5G

Este capítulo está dedicado a introducir al lector a la tecnología de quinta generación de redes celulares; aspectos relevantes y prácticos que contribuyen al entendimiento del presente trabajo. Sus características serán introducidas a nivel conceptual, sin perjuicio de lo cual en cada sección se incluirán referencias bibliográficas en donde se puede profundizar sobre cada tema.

2.1. Casos de uso

Las redes celulares que preceden a 5G eran exiguas para contemplar la convivencia de tráfico heterogéneo. Debido a ello, era necesario contar con el despliegue de redes específicas que cubran las necesidades de sus usuarios. Algunos ejemplos de ello son Narrowband IoT o LTE-M, soluciones que cuentan con grandes restricciones de transmisión de datos, capacidad y movilidad.

Una de las grandes novedades de 5G es la posibilidad de integrar dispositivos con requerimientos de servicio diferentes bajo la misma red. Para este fin, aprovechando las grandes prestaciones que ofrece 5G; ITU-R M.2083-0 en [50] define tres escenarios de uso que atienden a las demandas previstas hacia el futuro de la siguiente manera:

- Banda ancha móvil mejorada (eMBB): Utilizada para brindar acceso a contenido y servicios multimedia a través de dispositivos móviles. Es especialmente útil en escenarios donde se necesita una conexión de alta velocidad y una experiencia de usuario fluida, por ejemplo, en áreas con alta densidad de usuarios. Además, la Banda Ancha Móvil también se puede utilizar para brindar cobertura en áreas amplias, donde se requiere movilidad y una tasa de datos de usuario mejorada en comparación con las tasas de datos existentes. En general, la Banda Ancha Móvil es una herramienta importante para mejorar la conectividad y la accesibilidad a Internet en diferentes contextos.

- Comunicaciones ultra confiables y de baja latencia (URLLC): Tal como describe su nombre, este escenario requiere un alto nivel de confiabilidad y baja latencia en la comunicación. Algunos ejemplos de aplicaciones que pueden beneficiarse de esta tecnología incluyen el control de procesos industriales a

través de redes inalámbricas, cirugías médicas a distancia, la automatización de sistemas de distribución en redes inteligentes y la seguridad en el transporte. Estas aplicaciones necesitan un rendimiento y una disponibilidad de alta calidad para funcionar de manera óptima.

- Comunicaciones masivas de tipo máquina (mMTC): Implica un escenario con gran cantidad de dispositivos conectados. En general estos dispositivos transmiten una baja cantidad de datos que no son sensibles al retraso. Además, este escenario contempla dispositivos económicos que cuentan con una vida útil de la batería muy larga.

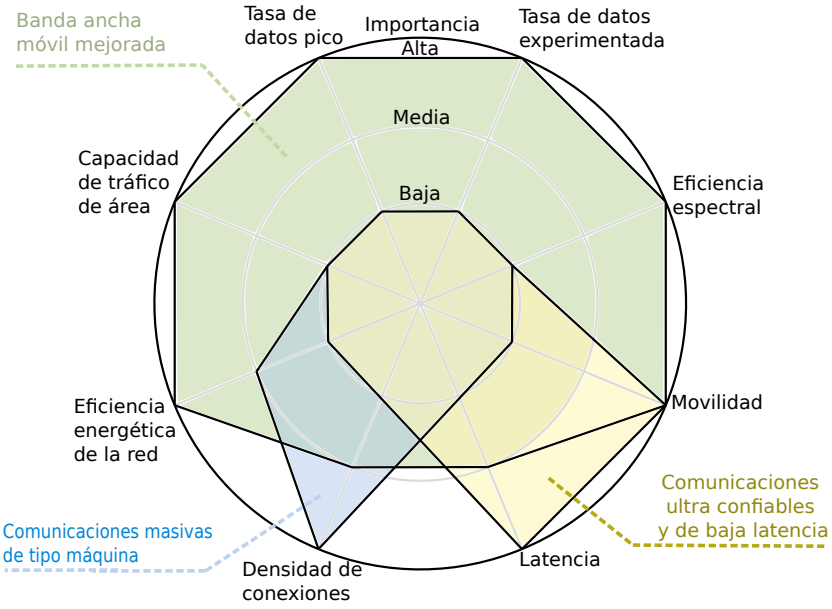


Figura 2.1: Características del tráfico en 5G, imagen adaptada de [40].

En conclusión, la tecnología 5G es capaz de integrar dispositivos con requerimientos de servicio diversos en la misma red, lo que permite atender a las demandas previstas hacia el futuro de manera más eficiente. Los escenarios definidos abarcan diferentes aplicaciones y necesidades, como el acceso a contenidos multimedia, el control de procesos industriales, la automatización de sistemas de distribución y la conectividad de dispositivos masivos. La Figura 2.1 muestra las características del tráfico en 5G y cómo estos escenarios de uso se ajustan a ellas.

2.2. Tecnología de transmisión

Para comprender la tecnología de quinta generación de redes móviles es necesario introducir los conceptos que vertebran su funcionamiento, esto es, su estructura de transmisión. En esta sección presentaremos su tecnología; para mayor rigurosidad de detalles se puede consultar en [19] y en la especificación de 3GPP 38.211 [8].

Para la transmisión, NR utiliza el sistema de modulación OFDM, el cual ofrece grandes ventajas en la mitigación de los efectos desfavorables del canal inalámbrico en entornos de comunicaciones móviles. Además, permite dividir el ancho de banda disponible en un gran número de canales de subportadoras más estrechos, lo que aumenta la capacidad de transmisión de datos y mejora la calidad de la señal.

La tecnología 5G debe contemplar distintos escenarios de despliegue y adaptarse a las necesidades de tráfico de sus clientes. Por eso, el sistema NR admite numerologías OFDM variables. La numerología determina la separación entre subportadoras (SCS, por su sigla en ingés), lo que impacta en el ancho de banda total y la duración de símbolo OFDM. Siendo que la cantidad total de subportadoras es constante, se tiene para los anchos de banda de 50/100/200/400 MHz un SCS de 15/30/60/120 kHz, respectivamente. En la Figura 2.2 se puede observar el efecto de aumentar la numerología considerando 12 subportadoras: a medida que aumenta el SCS, también lo hace el ancho de banda en la misma medida.

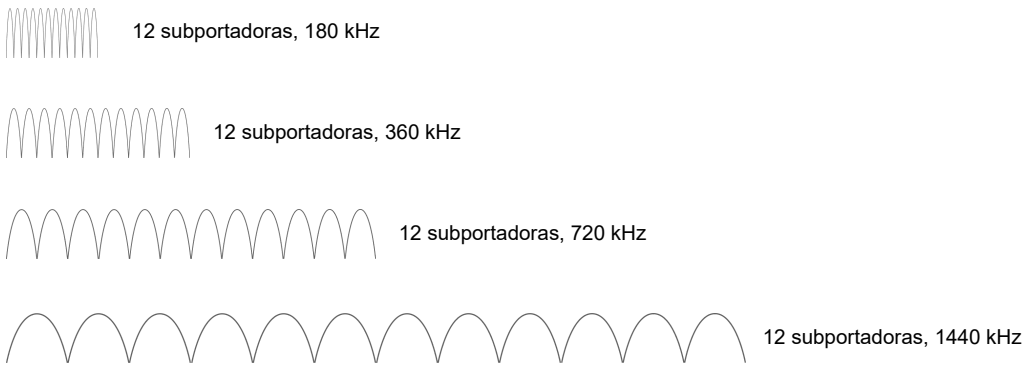


Figura 2.2: Doce subportadoras para distintas numerologías.

La transmisión en esta tecnología se realiza mediante cuadros que tienen como finalidad organizar y estructurar la información. A través de la definición de cuadros es posible transmitir y recibir datos de manera eficiente y sincronizada. Cada cuadro tiene una duración de 10 ms que a su vez se divide en subcuadros de 1 ms. Por otro lado, cada subcuadro contiene ranuras (o *slots*) temporales. Cada *slot*, entre otras cosas, determina el tiempo durante el cual se realiza la asignación de recursos, profundizaremos en este tema más adelante en el texto. En 5G, 14 símbolos OFDM forman un *slot*. Sin embargo, la duración de los símbolos OFDM depende de la separación entre subportadoras. Ergo, la duración de un *slot* queda determinada por la numerología y por consiguiente la cantidad de *slots* por subcuadro es variable. La Figura 2.3 ilustra la estructura de cuadro, subcuadro y ranuras.

Las unidades fundamentales de la tecnología de transmisión son la del *resource element* (RE) y *resource block* (i.e. RB). Un RE se define como una subportadora con duración de un tiempo de símbolo OFDM; un RE puede ser inequívocamente identificado dada la tupla (k, l), donde k es el índice en el dominio de la frecuencia y l identifica a la posición del símbolo en el dominio temporal. Por otro lado, un RB se define únicamente en el dominio de la frecuencia como 12 subportadoras

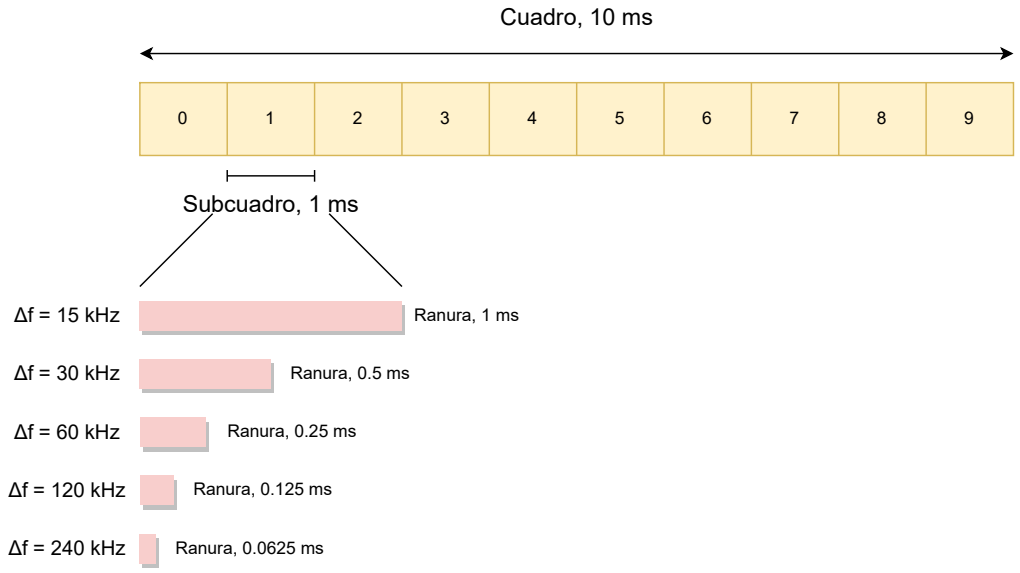


Figura 2.3: Estructura de cuadro.

consecutivas. La Figura 2.4 es una representación gráfica de ambos elementos en una grilla donde el eje horizontal contiene el tiempo y el eje vertical contiene el dominio frecuencial.

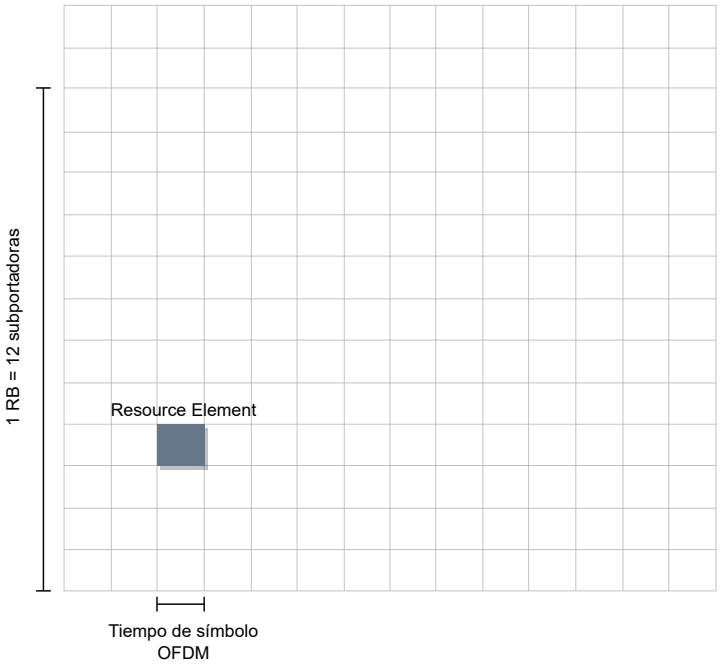


Figura 2.4: *Resource Element* y *Resource Block*.

El *resource block* es la mínima unidad de recursos que puede ser asignado a

un usuario. Mientras que, a través de *resource elements*, se transmiten señales de control y sincronismo. Ambos conceptos son unidades eidéticas para la asignación de recursos y, por consiguiente, del funcionamiento de la red.

2.3. Acceso al medio

La comunicación se logra a través de mecanismos bien definidos por el estándar. En particular, para *downlink* cada dispositivo monitorea el canal de control (abreviado como PDCCH por su sigla en inglés) al menos una vez por *slot*. Este canal contiene toda la información de control necesaria para que el UE pueda transmitir o recibir datos con la radiobase. Para las transmisiones de bajada (o *downlink*) de radiobase a UE, el mensaje de información de control para bajada (*DCI*) contiene la información de cuáles *resource blocks* acarrean los datos destinados al dispositivo, así como qué esquema de modulación utilizar y con qué tasa de código. Luego, el UE es capaz de recibir los datos de la comunicación a través del canal de datos físico en enlace descendente (PDSCH). La Figura 2.5 ilustra este mecanismo.

Figura 2.5: Acceso al medio en downlink.

Para la comunicación de subida (*uplink*), el mecanismo es diferente. En primer lugar, el UE le informa a la radiobase su voluntad de transmitir a través del mensaje de control de subida (UCI). La radiobase recibe esa información, utiliza el DCI y, en base a sus criterios de asignación, le anuncia los recursos en el que el UE puede transmitir y los parámetros de transmisión a utilizar. Luego, el UE puede transmitir sus datos a través del canal físico en enlace ascendente (PUSCH).

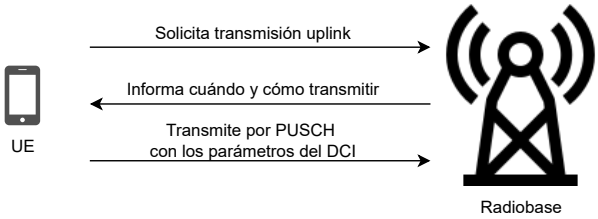


Figura 2.6: Acceso al medio en uplink.

Los mensajes de control DCI y UCI cuentan con distintos formatos, cada uno con sus casos de uso. Ambos resumen toda la información necesaria para una transmisión exitosa. Esto es: información sobre la calidad del canal; modulación

y tasa de código convolucional; sistema de solicitud de reconocimientos híbrida (HARQ); partición de ancho de banda (BWP) y diferentes reportes sobre el estado de la comunicación. En el presente capítulo introduciremos los conceptos de BWP y HARQ.

2.4. Recursos para el control de datos

Los conjuntos de recursos de control (i.e. CORESET), definidos en [7], son un conjunto de recursos en tiempo y frecuencia en donde el canal de control descendente (PDCCH) puede ser transmitido. Este conjunto de recursos es configurado por la red de manera semi-estática y puede encontrarse en cualquier posición arbitraria en el dominio temporal y frecuencial, con la restricción de tener como máximo la duración de 3 símbolos OFDM en el tiempo.

A diferencia de tecnologías de redes móviles anteriores, CORESET no se distribuye en todo el dominio frecuencial sino en porciones del espectro, dado que los UE no soportan la totalidad de la banda frecuencial. Con este fin, se utiliza su propia unidad de recursos llamados elementos de control de canal (CCE); constituidos por 12 RE en frecuencia en un tiempo de símbolo OFDM. El canal PDCCH puede ocupar 1, 2, 4, 8 o 16 CCEs.

Para que el UE pueda encontrar el canal de control, realiza una exploración del medio sin previo conocimiento. El dispositivo intenta decodificar el PDDCH utilizando varias combinaciones de parámetros, entre ellos, el índice de CCE y el RNTI [1].

2.5. Partición del ancho de banda

En su tecnología predecesora LTE, el ancho de banda de la portadora es constante de 20 MHz, lo cual implica que todos los dispositivos de la red son diseñados para operar en ella. Sin embargo, como se mencionó anteriormente, NR soporta numerología variable; de ahí que se pueden alcanzar extensos anchos de banda que, en consecuencia, no todos los dispositivos pueden soportar.

Para ello, 5G define el concepto de *Bandwidth part* (BWP), que implica la posibilidad de que los dispositivos se limiten a un particionamiento del ancho de banda y, de esta manera, poder operar en la red. El BWP es definido por una numerología y una cantidad consecutiva de RBs, iniciando desde un *common resource block* (CRB) específico (ver Figura 2.7). Cada dispositivo puede estar configurado con hasta cuatro posibles BWP diferentes para bajada y subida, siendo solo uno de ellos utilizado para un instante de tiempo dado.

[1] RNTI es utilizada para decodificar el DCI e identificar a cada UE

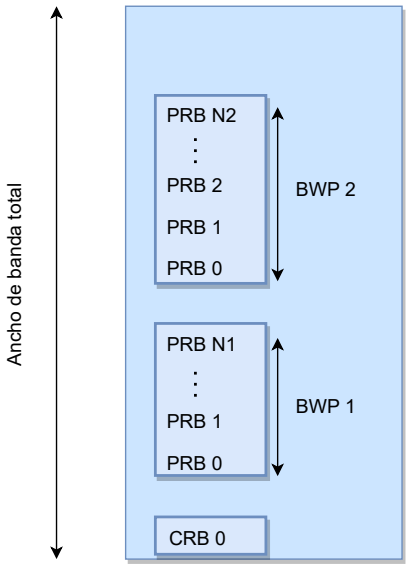


Figura 2.7: Representación de varios BWP en el ancho de banda total.

2.6. Adaptación al enlace

Las redes 5G tienen un mecanismo bien definido a través del cual obtienen la información de la calidad de señal que recibe cada dispositivo. A partir de ella, las redes utilizan técnicas de Modulación y Codificación Adaptativa (AMC por su sigla en inglés) con el fin de adaptar la señal para que el dispositivo final pueda recibir sin errores la información transmitida.

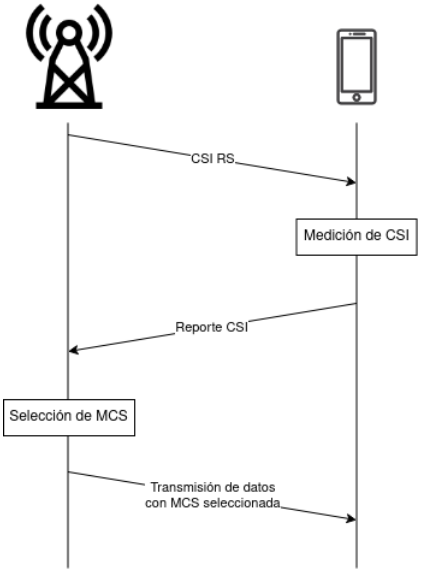


Figura 2.8: Mensajes involucrados en la adaptación al enlace.

Por un lado, 5G utiliza una señal llamada “indicador del estado del canal”

(CSI-RS), a través de la cual es posible estimar el canal en cada subportadora (ver Fig. 2.8). Específicamente, las medidas utilizadas son la señal de referencia de potencia no nula (NZP) y la señal de referencia de potencia nula (ZP). La primera es utilizada para medir la potencia de la señal, mientras que la segunda, para medir la interferencia. Utilizando ambas en conjunto es posible obtener una medida de la relación señal a ruido (SINR) con la cual el dispositivo final recibe la señal. No obstante, la medida reportada por el dispositivo es el mapeo de SINR a un valor estandarizado que lleva el nombre de "información de calidad del canal" (CQI). Describe la calidad de la señal recibida y es reportada a través de los mensajes de información del estado del canal (CSI).

Por otro lado, las redes celulares 5G utilizan el sistema de solicitudes de repetición híbrida (i.e. HARQ). HARQ es un esquema de corrección de errores y reconocimientos utilizado por estas redes. Cuando no llega el reconocimiento de un mensaje enviado, eso significará su pérdida, lo cual sugiere ajustar la modulación y tasa de código utilizada para garantizar la recepción de futuros mensajes. Conjuntamente con las anteriores señales de referencia, conforma un importante insumo para conocer el estado del enlace.

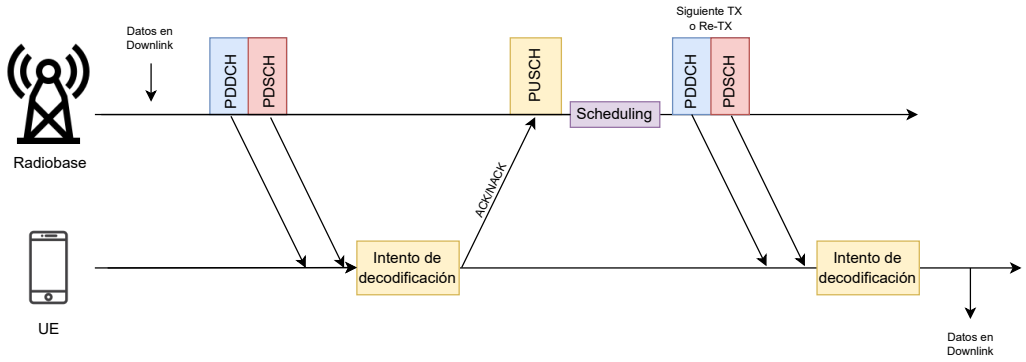


Figura 2.9: Mecanismo de retransmisiones en bajada.

Las medidas obtenidas son utilizadas por la radiobase a través de los mecanismos bucle interno de adaptación al enlace (ILLA) y bucle externo de adaptación al Enlace (OLLA). La primera toma en consideración las medidas reportadas de la calidad de canal de la señal en recepción, mientras que la segunda toma los reconocimientos del esquema HARQ. En base a ambas, la radiobase determina el esquema de modulación y la tasa de código óptimas (MCS por su sigla en inglés) en la transmisión de datos.

En la Figura 2.9 se ilustra el mecanismo de decodificación de recepción de un mensaje que contiene errores. En primer lugar, el UE detecta que hay errores en el mensaje. Luego, envía un mensaje de error a la radiobase. La radiobase agenda nuevamente la transmisión y en el *slot* correspondiente envía el mensaje. En caso de que el UE no haya decodificado errores, se finaliza la transacción enviando un acuse de recibo.

2.7. Network Slicing

Como se introdujo anteriormente, una de las grandes novedades de 5G es la incorporación de *network slicing*. Una rebanada de red (o *network slice*) es una red lógica utilizada para atender las necesidades de un grupo de usuarios específico. Es posible de esta manera contar con niveles de servicio diferentes bajo la misma infraestructura de red. Por ejemplo, una *slice* puede estar destinada a asegurar los recursos necesarios para el tráfico de dispositivos celulares -los cuales no tienen grandes restricciones de latencia y tasa de datos- mientras que otra *slice* puede garantizar los recursos y servicios utilizados para aplicaciones de ultra baja latencia -como pueden ser autos inteligentes o aplicaciones médicas-, como se ilustra en la Figura 2.10.

El paradigma de *network slicing* es un elemento clave para el éxito de la implementación de 5G y su capacidad para cumplir con los requisitos y expectativas de una amplia variedad de usuarios y aplicaciones. Es la principal habilitadora para la coexistencia de los diferentes perfiles de tráfico definidos en la Sección 2.1. Una de las grandes novedades a resaltar es que *network slicing* no solo incluye el núcleo de la red sino también la red de acceso inalámbrica.

Más aún, el paradigma de *network slicing* abre las puertas a un nuevo modelo de negocio, en donde una única infraestructura de red brinda servicios a operadores -o inquilinos-. Los inquilinos, a su vez, brindan su propio servicio a sus clientes teniendo la posibilidad de agregarles servicios de valor agregado. Por lo que es posible que este nuevo paradigma cambie la forma en que funciona el mercado de redes móviles.

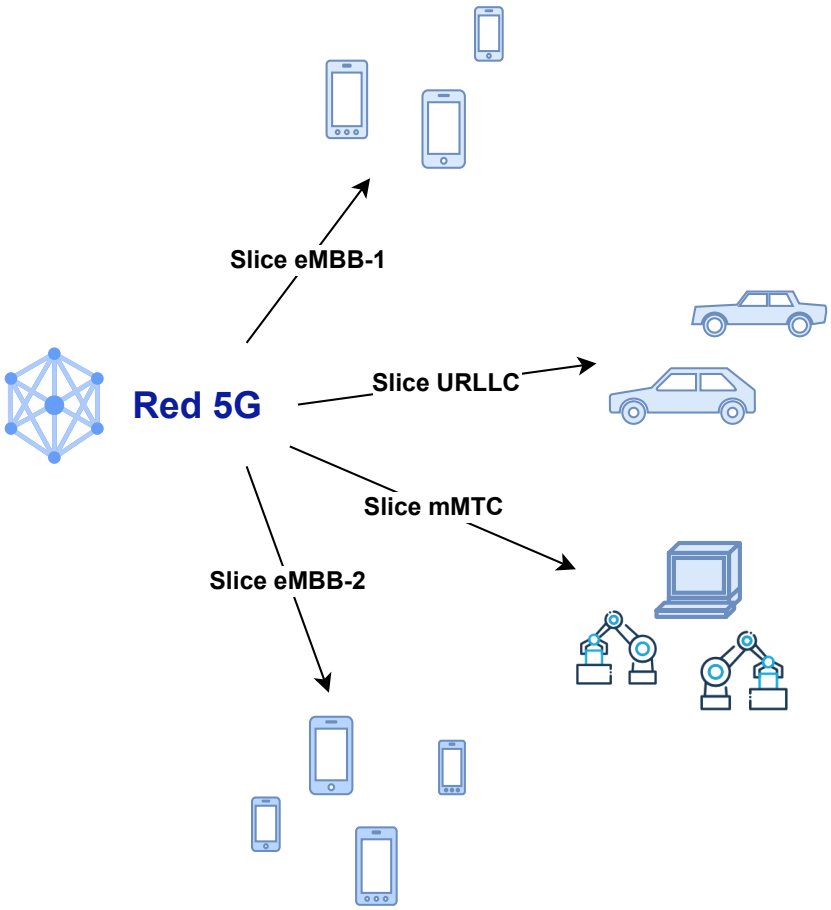


Figura 2.10: Representación de *Network Slicing* en una red 5G.

2.8. Asignación de recursos

La asignación de recursos es el objeto de estudio del presente trabajo. Por ello es de especial importancia y, como tal, esta sección está dedicada al estudio de los mecanismos involucrados.

En la sección 2.3 se introdujo cómo los usuarios de la red interactúan con la radiobase tanto para transmisión de datos ascendente como descendente. Luego, es esta entidad la que determina cuándo y a qué dispositivo son asignados los recursos de tiempo y frecuencia. Asimismo, es quien determina los parámetros de transmisión, incluida la tasa de datos que debe usarse. La asignación dinámica de recursos implica que el *scheduler*, para cada *slot*, determina qué dispositivos pueden transmitir y cuáles recibir. Ya que la asignación de recursos se realiza con alta frecuencia, esta asignación puede contemplar rápidas variaciones en el flujo de la demanda y de la calidad del canal de radio. Sin embargo, es necesario contar con técnicas eficientes y equitativas de asignación de recursos que garanticen la calidad de servicio de los usuarios finales.

Más aún, las técnicas de asignación de recursos no están estandarizadas, dejando

a los operadores decidir por aquellas que les sean más convenientes. Sin embargo, sí define mecanismos a través de los cuales cada sistema de *scheduling* entrega los recursos a los usuarios finales.

En este contexto, la obtención de algoritmos que cumplan con las necesidades de toda la red es un desafío a abordar. Es por ello un tema central para toda red móvil y aún más en las redes de quinta generación en donde conviven muchas restricciones de servicio bajo la misma infraestructura.

Por otro lado, la incorporación del paradigma de *network slicing* trae consigo una nueva forma de asignación de recursos. Es por ello que se definen dos etapas para el alojamiento de recursos, a saber:

- Asignación *Inter-Slice*: Es la asignación de los recursos de la celda entre las *slices* disponibles.
- Asignación *Intra-Slice*: Es la asignación de los recursos de la *slice* entre los *UE* pertenecientes a esta.

En la Figura 2.11 se pueden observar ambas etapas en conjunto. En la primera etapa, la radiobase asigna sus recursos a las *slices*, mientras que en la segunda, cada *slice* asigna recursos a sus dispositivos. Este esquema de asignación permite flexibilidad y es una de las grandes habilitadoras a una red heterogénea. El *scheduler Inter-Slice* resulta de gran importancia para garantizar los niveles de servicio acordados con los usuarios finales. En tanto que el *scheduler Intra-Slice* tiene significado local a la *slice* y su tarea es asignar eficientemente sus recursos. En esta misma línea, ambos *schedulers* viven en distintas escalas temporales de asignación. El primero tiene una granularidad no definida en el estándar, pero se asume en el orden de los 500 ms; mientras que la segunda depende de factores como la numerología, pero con tiempos cercanos a 1 ms.

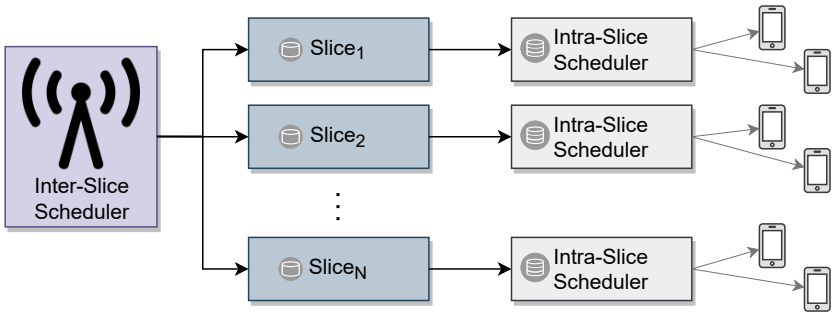


Figura 2.11: *Inter-Slice scheduler.*

Es importante señalar la relevancia de la asignación *inter-slice*, pues se realiza de manera "macroscópica" y un error se traduce en un gran deterioro del desempeño de la red. En este sentido, un error a nivel *inter* es más perjudicial que un error en la asignación *intra* por dos razones:

- Las *slices* representan grupos de usuarios, si su asignación no es buena, la de aguas abajo tampoco lo será. Además, puede tener implicaciones en el

servicio como proveedor de infraestructura. A modo de ejemplo, una mala asignación a este nivel en una infraestructura arrendada a diversos operadores se traduce en un problema a gran escala que se hereda a los inquilinos.

- El tiempo entre asignaciones consecutivas es muy superior en la asignación *inter-slice*. Esto se traduce en que un error puede prolongarse en el tiempo tanto como dure la granularidad temporal.

Estas características vuelven prioritarios a los algoritmos *inter-slice* como objeto de estudio para el despliegue de redes 5G.

2.8.1. Algoritmos de *Scheduling*

La asignación de recursos implica la decisión de repartir los finitos recursos disponibles en función de una política de asignación. Para ello, es necesario tener distintas medidas que permitan conocer el estado actual de los usuarios. Estas medidas pueden ser la tasa de transferencia (o *throughput* en inglés), calidad del enlace, historial de transmisiones pasadas, nivel de servicio acordado, entre otras.

En particular, los algoritmos más utilizados son *Round Robin*, *Proportional Fair*, *Best CQI* y *MaxMin*. Basándonos en el trabajo escrito en [21], los introduciremos a continuación.

- *Round Robin*: Este algoritmo asigna los recursos de manera equitativa, sin depender del estado del canal ni de las métricas de los usuarios. Primero genera una lista ordenada de los usuarios, para luego ir asignando recursos de manera secuencial hasta repartirlos todos. Es uno de los algoritmos de asignación más simples y, debido a su sencillez, es ampliamente utilizado en redes móviles. No obstante, la falta de información del canal produce baja eficiencia espectral, lo que se traduce en un desaprovechamiento de la red. Además, este algoritmo provee justicia en términos de cantidad de recursos asignados, pero no es justo considerando las características de transmisión de los usuarios.

- *Best CQI*: El objetivo del algoritmo *Best CQI* es elegir al usuario que cuenta con la mejor calidad de canal para una determinada RB en un intervalo de tiempo. Este enfoque tiene un buen rendimiento en términos de *throughput*, pero presenta cierta falta de equidad desde el punto de vista de los usuarios. Solo aquellos usuarios con altos índices CQI -o con el CQI más alto- podrán enviar y recibir datos mediante este algoritmo. El usuario con la mejor calidad de canal será asignado con recursos, siempre y cuando la calidad del canal de los demás usuarios no supere la de dicho usuario. Así, solo unos pocos usuarios podrán aprovechar al máximo los recursos en el caso del "Best CQI". Los usuarios que se encuentran cerca de la radiobase utilizarán la mayor parte de los recursos, mientras que aquellos situados en la periferia quedarán sin conexión.

- *MaxMin*: Este algoritmo busca maximizar el mínimo *throughput* de los usuarios y, de esta manera, brindar justicia a la red. No obstante, dado que

aquellos usuarios que tienen poco *throughput* son en general los que tienen condiciones inestables del canal, se les asignará más recursos, lo cual produce un deterioro en el *throughput* total del sistema.

- *Proportional Fair*: A cada usuario se le asigna una métrica dada por la razón entre el *throughput* instantáneo y el *throughput* medio.

$$m_n = \frac{I_n}{\overline{T_n}}$$

Donde m_n es la métrica del n-esimo usuario, I_n su *throughput* instantáneo y $\overline{T_n}$ su *throughput* medio. Luego, se asignará al usuario de mayor métrica los recursos disponibles.

Este algoritmo tiene un gran componente de justicia, pues pondera la capacidad de transmisión de los usuarios con la media de transmisiones pasadas. Por lo que, si un usuario no transmitió durante muchos tiempos de asignación, tendrá mayor prioridad para hacerlo. A su vez, la capacidad de transmisión de los usuarios será considerada.

- *Proportional Fair* con exponentes: Es una variante del recién descrito; en este caso se toma el numerador y denominador con diferentes exponentes:

$$m_n = \frac{I_n^{\alpha}}{\overline{T_n}^{\beta}}$$

Siendo α y β arbitrarios. En función de su elección, es posible dar más importancia a la capacidad de los usuarios o a la historia de los mismos. También es posible la obtención de diferentes algoritmos al manipular ambos exponentes. Por ejemplo, en el caso de tener $\alpha = 1$ y $\beta = 0$ se obtiene un algoritmo que asignará recursos al usuario con mayor *throughput*.

Estos algoritmos de asignación aplican como algoritmos *Inter-Slice* e *Intra-Slice*. En el caso de los primeros, las medidas utilizadas tienen en cuenta cualidades de la *slice*, mientras que en la segunda las medidas son directamente de los usuarios. El algoritmo de facto puede ser una combinación de los expuestos o un desarrollo propietario, pues su elección está sujeta a los intereses del operador. Es por tanto de especial interés explorar distintos abordajes existentes e incorporar técnicas novedosas a su desarrollo.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 3

Aprendizaje por refuerzo

En el contexto del presente trabajo, el aprendizaje por refuerzo fue el método empleado para la implementación de un algoritmo capaz de resolver el problema de la asignación de recursos. Es por tanto inexorable introducir al lector a este paradigma en el transcurso de este capítulo. Aunque el aprendizaje por refuerzo es un tema de gran alcance, en este capítulo solo se tratarán de manera breve o superficial sus fundamentos teóricos, basados en [52], [2], [53], [13] y [26].

Para presentar los conceptos, se utilizará como ejemplo didáctico el ilustrado en la Figura 3.1. Allí se puede observar una antena que representa a la radiobase celular y dos dispositivos celulares conectados a ella. El objetivo de la red es lograr un buen desempeño en términos de calidad de servicio en los dispositivos. Para ello deberá decidir cómo asignar sus recursos a través de técnicas eficientes de *scheduling*.

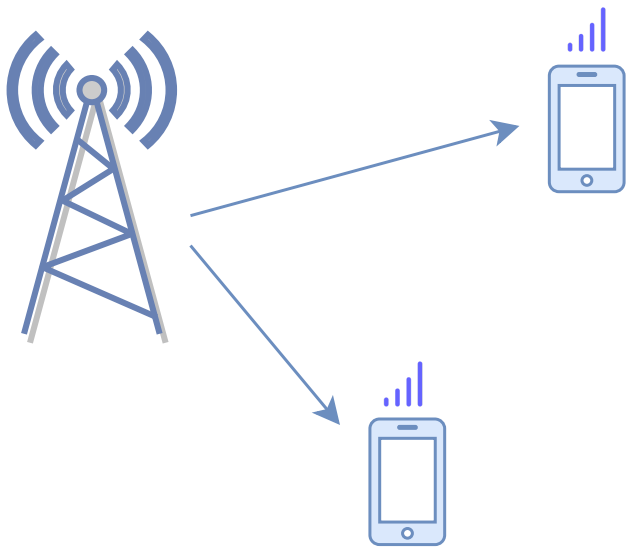


Figura 3.1: Ejemplo didáctico: una radiobase y dos dispositivos celulares conectados a él.

3.1. Fundamentos

3.1.1. Interacción Agente-Entorno

El aprendizaje por refuerzo comprende un área de la inteligencia artificial enfocada a resolver el problema de cómo un agente puede aprender a tomar decisiones eficientes en un entorno dado a través del uso de recompensas y castigos.

Más específicamente, el agente y el entorno interactúan en instancias discretas de tiempo. En cada una de ellas, el agente recibe una representación del estado del entorno, $s_t \in S$, donde S es el conjunto de estados posibles. En base a ese estado s_t, el agente elige una acción $a_t \in A(s_t)$, donde $A(s_t)$ es el espacio de acciones posibles para el estado s_t. La acción realizada por el agente tendrá un efecto sobre el entorno, el cual cambiará, pasando al estado s_{t+1}. Además, el entorno le devolverá al agente una recompensa r_{t+1}, utilizada como retroalimentación de si la acción a_t en s_t fue o no conveniente. Esta interacción se puede observar en la Figura 3.2.

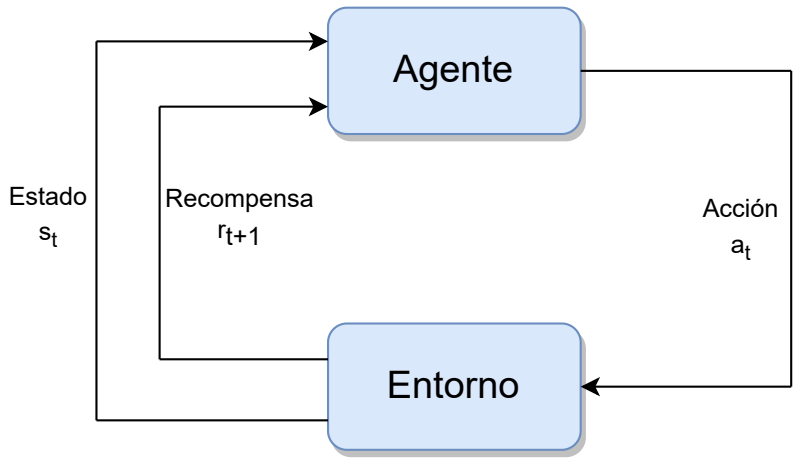


Figura 3.2: Diagrama general de aprendizaje por refuerzo. Adaptado de [52].

En cada instante de tiempo t, la acción a tomar en el estado actual s_t se determina siguiendo la política π_t, donde $\pi_t(s, a)$ representa la probabilidad de realizar la acción a en el estado s en el tiempo t. En otras palabras, la política define el comportamiento del agente para un instante dado. En algunos casos, la política puede ser una función simple o una tabla de búsqueda, mientras que en otros, puede involucrar extensos cálculos.

En el ejemplo presentado en la introducción de este capítulo, la antena y los dispositivos celulares constituyen el entorno; el algoritmo de *scheduling*, el agente; la asignación de recursos, la acción; y la calidad de servicio, la recompensa. El agente -el algoritmo de asignación- interactúa con el entorno -la red- a través de la decisión de a qué dispositivos se asignarán recursos. Luego, el agente percibirá a través de la recompensa -la calidad de servicio reportada- cuán conveniente fue la acción tomada.

3.1.2. Recompensa y Retorno

Cada vez que el agente toma una acción, el entorno le proporciona una recompensa numérica. El agente tiene como meta maximizar la recompensa total recibida a largo plazo. La señal de recompensa determina si los eventos son positivos o negativos para el agente. Dependiendo de la acción actual y del estado del entorno, la recompensa variará. El agente no puede alterar cómo se asigna la recompensa, pero sí puede influir en ella a través de sus acciones, tanto directa como indirectamente a través de cambios en el entorno. La señal de recompensa es la base para modificar la política; es decir, si una acción elegida por la política tiene una recompensa baja, puede ser modificada para elegir otra acción en esa situación en el futuro. Las señales de recompensa pueden ser funciones aleatorias del estado del entorno y de las acciones tomadas.

Por otro lado, como se mencionó con anterioridad, el objetivo es maximizar la recompensa a largo plazo. Para ello se define el concepto de retorno, denotado como G_t para un tiempo t dado. Es la suma de las recompensas desde t en adelante. Es decir:

$$G_t = R_t + R_{t+1} + R_{t+2} + ... + R_{t+T} \tag{3.1}$$

Donde T es el tiempo para el que se alcanza el objetivo del agente. O en otras palabras, es el tiempo para el cual el agente alcanza el estado terminal.

Sin embargo, el retorno definido en 3.1 tiene el problema de que puede divergir para escenarios en los que el estado terminal no es claro. Es por ello que es común utilizar una noción diferente de retorno definido de la siguiente manera:

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \gamma^3 R_{t+3} + ... = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{3.2}$$

Donde el valor γ lleva el nombre de factor de descuento. Este valor es utilizado para que en G_t tenga mayor importancia las recompensas cercanas en el tiempo; haciendo que las recompensas más alejadas tengan menor peso. Esta definición del retorno cobra sentido al ser empleada para la obtención de la función de valor.

En el ejemplo didáctico previamente mencionado, se ilustró que la señal de recompensa se obtiene a partir de la evaluación de la calidad del servicio por parte de los usuarios. Por consiguiente, si se ejecuta una acción en el tiempo t siguiendo una política π, el retorno se definirá como el desempeño reportado por los usuarios correspondiente al momento t, sumado a los desempeños futuros ponderados por sus respectivos factores de descuento γ^{t-1}.

3.1.3. Función de valor y de valor-acción

En relación a una política π dada, se hace referencia al *valor* del estado s bajo π como $v_\pi(s)$. Dicho valor representa la estimación del retorno futuro esperado al seguir la política π a partir del estado s. A diferencia de la señal de recompensa, que refleja una ganancia inmediata, la función de valor nos proporciona información

acerca de los beneficios a largo plazo. En MDPs[1] es posible expresar el valor en s mediante la siguiente fórmula:

$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s\right] \tag{3.3}$$

Donde a v_π se la define como función de valor. La función de valor para el estado s es el valor esperado del retorno si el agente inicia en el estado s y sigo en adelante el comportamiento π.

Análogamente, denotamos *acción-valor* al retorno esperado bajo la política π de estar en el estado s y tomar la acción a. Sea $q_\pi(s, a)$ esta medida, tenemos que:

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s, A_t = a\right] \tag{3.4}$$

El paradigma del aprendizaje por refuerzo busca la política que maximice la recompensa a largo plazo. De esta manera decimos que la política π es mejor que π' si $v_\pi(s) > v_{\pi'}(s) \forall s \in S$. La política que cumple que es mayor o igual que todas las demás, lleva el nombre de *política optima* y se denota π_*. Entonces para la función de valor y función de acción-valor, se tiene:

$$v_{\pi_*} = \max_\pi v_\pi(s),$$

$$q_{\pi_*} = \max_\pi q_\pi(s, a).$$

Al mismo tiempo, si se observa la definición de la función de valor y la de acción-valor, ambas se vinculan según la siguiente ecuación:

$$v_{\pi_*} = \max_{a \in A(s)} q_{\pi_*}(s, a) \tag{3.5}$$

Sí logramos caracterizar a la función de *acción-valor* de alguna forma, para todo el conjunto de estados, podemos obtener una política que maximice el retorno. Volviendo al ejemplo ilustrado, si se supone un estado s en el que ambos dispositivos tienen paquetes para transmitir, y se supone dos acciones posibles:

- a1: Asignar la mitad de los recursos a cada usuario.
- a2: No asignar recursos a ninguno de ellos

Entonces, el valor de $q_{\pi_*}(s, a1)$ será por consiguiente mayor al de $q_{\pi_*}(s, a2)$. Por lo tanto, es natural que la política seleccione $a1$ por sobre $a2$ en el estado planteado.

[1]Markov Decision Process

Optimalidad de Bellman

A partir de las nociones introducidas anteriormente es posible ahora introducir la ecuación de optimalidad de Bellman:

$$q_*(s,a) = \mathbb{E}_{\pi_*}\left[R_{t+1} + \gamma \max_{a'} q_*(s',a')\right] \tag{3.6}$$

Esta ecuación nos dice que, para cada par de estado-acción en un momento dado, el valor esperado a partir del estado inicial s y tomando la acción a y siguiendo la política óptima π_* será igual a la recompensa inmediata esperada más el valor esperado descontado a largo plazo del próximo par estado-acción (s', a').

La ecuación 3.6 es la raíz del aprendizaje por refuerzo. Pues, al hallar q_* la obtención de la política óptima es directa. Ya que para cada estado s, simplemente se debe hallar la acción que maximiza $q_*(s,a)$.

3.2. Q-Learning

Q-Leaning es un algoritmo de aprendizaje por refuerzo a través del cual es posible obtener una solución para la función de *acción-valor* y, con ello, la obtención de una política óptima. El algoritmo fue introducido por Watkins et al. en 1989, quién además prueba en [55] que el algoritmo converge bajo ciertas hipótesis.

El algoritmo de *Q-Learning*, en su forma más simple, es definido de la siguiente manera:

Entrada: un conjunto de estados S, un conjunto de acciones A, una función de recompensa R, un factor de descuento γ, y un parámetro de aprendizaje α

Inicializar una tabla $Q(s,a)$ con valores arbitrarios para todos los pares $(s,a) \in S \times A$;

repetir

  Seleccionar un estado inicial s;

  repetir

    Elegir una acción a basada en una política de selección de acciones, como ϵ-greedy;

    Ejecutar la acción a y observar la recompensa r y el nuevo estado s';

    Actualizar la tabla $Q(s,a)$ utilizando la ecuación de Q-learning:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

    $s \leftarrow s'$;

  hasta que *estado final*;

hasta que *convergencia*;

Algoritmo 1: Q-learning

Agregando el índice temporal a los estados y acciones, queda claro que un paso en *Q-Learning* queda definido por:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right] \tag{3.7}$$

El proceso de aprendizaje sigue una dinámica similar al método de diferencias temporales (TD) [53]. En cada iteración, se actualiza la estimación de la función de *valor-acción* a través de la fórmula de actualización de diferencia temporal (TD), la cual se basa en la diferencia entre la recompensa real obtenida y la estimación previa.

En particular, Q-Learning es de mucho interes para el presente trabajo debido a sus siguientes particularidades:

- Actualiza los valores Q (función de acción valor) entre dos episodios consecutivos; no debe esperar por una colección de episodios como en los métodos *Monte-Carlo* (*Time differential method*).
- La política de actualización de la función Q es irrelevante a la política de muestreo (*Off-policy*).
- El agente estima los valores Q para cada par estado-acción, del cual deriva la política óptima; no predice el próximo estado previo a tomar la acción (*Model-free).*

Este método supone contar con una tabla $Q \in S \times A$ que contenga toda la información de la función de *valor-acción.* En el ejemplo didáctico, si consideramos que el espacio de acciones está dado por a_1 y a_2 como en la Sección 3.1.3. Y consideramos los estados:

- s_1: ambos celulares tienen paquetes para transmitir.
- s_2: ningún celular tiene paquetes para transmitir.

	a_1	a_2
s_1	q_1	q_2
s_2	q_3	q_4

Tabla 3.1: Tabla Q en un problema de dos acciones y dos estados posibles.

Para el escenario didáctico, se dispone de una matriz de tamaño 2×2. Cada uno de sus valores q_i se actualizará utilizando el Algoritmo 1 hasta obtener una solución óptima. Una vez que se tenga la tabla Q con sus valores finales, se buscará para cada estado s el índice a que tenga el mayor valor de q, construyéndose así la política que maximizará la recompensa.

Sin embargo, la técnica descrita presenta algunas limitaciones. Pues es necesario calcular $q(s, a)$ para cada combinación de estado-acción. En otras palabras, en su forma original, Q-Learning implica tener una tabla con tantas columnas como acciones posibles y filas como estados posibles. Esto podría ser útil en un número

limitado de aplicaciones, pero en la realidad, muchos escenarios requieren espacios de estados de una gran magnitud, como por ejemplo Go con 10^{170}, Backgammon con 10^{20}, o aplicaciones que involucran espacios continuos, como robots en entornos industriales, etc. Esto conlleva un alto costo en términos de uso de memoria para almacenar la tabla Q, además de no ser eficiente en términos de entrenamiento ya que para tener el modelo entrenado, es necesario visitar cada estado -para actualizarlo- repetidas veces.

3.3. Deep Q-Learning

Considerando que los estados escalan con la complejidad del sistema y con la granularidad de los estados, resolver una gran cantidad de escenarios se vuelve costoso e irrealizable. Sin embargo, si fuese posible aproximar el valor de $q(s, a)$ como una función paramétrica se resolvería este inconveniente y sería posible entrenar modelos en sistemas significativamente más complejos.

En [39] se presenta un hito en el campo del aprendizaje por refuerzo mediante la implementación del algoritmo DQL (*Deep Q-Learning*), revolucionando el paradigma existente y sentando las bases para nuevos avances en el área. En él, presentan el primer modelo de aprendizaje profundo que logra aprender una política directamente de un espacio de entrada de alta dimensionalidad; utilizando aprendizaje por refuerzo. Utilizan el algoritmo aplicado a juegos de Atari, en donde entrenan a un agente que logra un desempeño significativo.

El objetivo central de esta técnica es aproximar a la función Q a través de una red neuronal utilizando un enfoque de aprendizaje profundo, de manera que la red neuronal pueda aprender a predecir el valor Q para cualquier estado y acción en el entorno del agente. Una vez que entrenada la red, el agente puede utilizarla para tomar decisiones en tiempo real, seleccionando la acción que maximiza el valor Q estimado para el estado actual.

Para entrenar a la red neuronal se define a partir a partir de la Ecuación 3.6 la función de perdida de la siguiente manera:

$$L_i(\theta_i) = \mathbb{E}_{s,a\sim p(.)}\left[(y_i - Q(s, a; \theta_i))^2\right] \tag{3.8}$$

El termino $Q(s, a; \theta_i)$ es la función Q para el estado s y la acción a mientras que y_i es la etiqueta. El valor de la etiqueta y_i se define de la siguiente manera:

$$y_i = \mathbb{E}_{s'\sim\epsilon}\left[r + \gamma max_{a'} Q(s', a'; \theta_{i-1})|s, a\right] \tag{3.9}$$

Es importante destacar que la función de pérdida juega un papel fundamental en el proceso de aprendizaje de la red neuronal, ya que es la medida que se utiliza para evaluar la calidad de las predicciones de la red y, por tanto, para guiar el ajuste de sus parámetros.

Por su parte, la etiqueta en 3.9 comprende el valor de la recompensa actual sumado al valor de la recompensa que se obtendrá luego. Algo llamativo de ello es que utilizamos nuestra misma red neuronal (con los parametros en tiempo $i - 1$)

para estimar a nuestra etiqueta. No obstante, en ello se basa la teoría de *Deep Q-Learning*.

Por otro lado, es interesante observar que por diseño, la red neuronal se entrena para obtener un conjunto de valores Q a partir de un estado s, tal como se ilustra en la Figura 3.3. Y luego, se toma la acción óptima como la correspondiente al máximo valor Q.

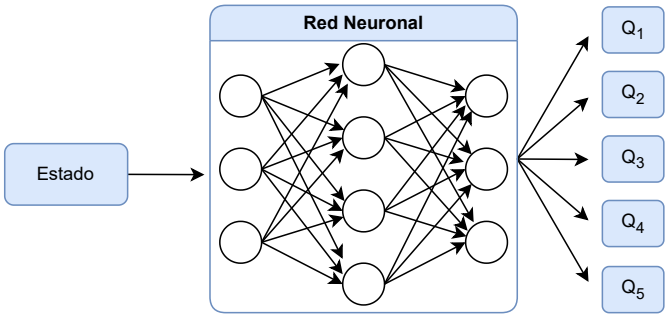


Figura 3.3: Diagrama general de DQL; para una estado obtengo varios q.

Otro abordaje podría ser diseñar una red neuronal que aproxime directamente $Q(s, a)$, es decir, que para cada par estado-acción, devuelva su valor q correspondiente. Sin embargo esto implica que para conocer la acción óptima, se deba evaluar la red para cada acción en dicho estado, lo cual lo hace computacionalmente más costoso.

Como se menciona en [26], dos incorporaciones clave hacen al éxito de *Deep Q-Learning* como algoritmo de aprendizaje. Estos son: la "memoria de experiencia" y el concepto de red objetivo Q^-.

En el modelo de DQN, a medida que el agente interactúa con el entorno y va obteniendo nuevas experiencias, estas son almacenadas en una memoria específica denominada "memoria de experiencia". Posteriormente, se extrae un subconjunto de estas experiencias de manera aleatoria para entrenar la red neuronal. Es importante destacar que durante este proceso algunas muestras pueden ser utilizadas múltiples veces, mientras que otras pueden no ser utilizadas en absoluto.

Este enfoque permite descorrelacionar las experiencias, es decir, reducir la dependencia entre las muestras de la memoria y garantizar que el aprendizaje se realice de manera más eficiente. Asimismo, al utilizar una selección aleatoria de las experiencias almacenadas en la memoria, se evitan secuencias específicas de experiencias y por consiguiente un aprendizaje sesgado o poco representativo del entorno.

Por otro lado, la red objetivo es un componente fundamental en el modelo DQN. La red objetivo es una copia de la red neuronal utilizada para aproximar la función Q, pero se utiliza para estimar los valores Q objetivo durante el entrenamiento. Esta red se mantiene fija durante un cierto número de iteraciones de entrenamiento de manera de evitar inestabilidades y garantizando una actualización de los pesos más suave. El algoritmo de aprendizaje, incluyendo la memoria de experiencia, se puede observar a través del Pseudocódigo 2.

Entrada: Un entorno de aprendizaje E, una tasa de descuento γ, un tamaño de lote N, una tasa de aprendizaje α, un número de episodios M, un tamaño de buffer de repetición B, un intervalo de actualización de red de destino C, un conjunto de acciones A, una función de aproximación de la función de valor Q, una política de selección de acción ϵ-greedy
Salida: La función de valor aprendida Q

Inicializar Q con pesos aleatorios;
Inicializar buffer de repetición D de tamaño B;
Inicializar la red de destino Q^- con los mismos pesos que Q;
para $m = 1$ **a** M **hacer**
  Reinicializar el entorno y obtener el estado inicial s;
  para *cada paso del episodio* **hacer**
    Seleccionar una acción a con probabilidad ϵ y $1 - \epsilon$, respectivamente;
    Ejecutar la acción a en el entorno y obtener la siguiente observación s' y la recompensa r;
    Actualizar el estado actual: $s \leftarrow s'$;
    Almacenar la transición (s, a, r, s') en D;
    si *La cantidad de muestras en D supera un tamaño arbitrario.* **entonces**
      Muestrear un mini-lote de N transiciones (s_i, a_i, r_i, s'_i) aleatoriamente de D;
      Calcular el objetivo y_i para cada transición:

$$y_i = r_i + \gamma \operatorname{máx} a' Q^-(s'_i, a')$$

      Actualizar la función de valor Q mediante la minimización de la pérdida cuadrática media:

$$L_i(\theta) = (y_i - Q(s_i, a_i; \theta))^2$$

      Actualizar los pesos de Q mediante el descenso de gradiente: $\theta \leftarrow \theta - \alpha \nabla \theta L_i(\theta)$;
      Si el número de pasos del episodio es un múltiplo de C, copiar los pesos de Q en Q^-: $Q^- \leftarrow Q$;
    fin
  fin
fin

Algoritmo 2: Algoritmo de DQN.

DQL es una valiosa herramienta para el desarrollo de algoritmos en entornos complejos. Ofrece mayor estabilidad que el algoritmo original de Q-Learning, resultado de utilizar la memoria de experiencia conjuntamente con la red neuronal. Además, permite manejar problemas con grandes espacios de estados y acciones, abriendo la posibilidad de utilizar está técnica en una amplia gama de problemas. Y por último, a través de este método, es posible predecir el valor Q de estados a los que el algoritmo no haya recorrido anteriormente.

Todas estas ventajas tienen una importancia significativa en entornos reales. Es posible escalar el ejemplo didáctico a un escenario con muchas más acciones y estados. Se puede, por ejemplo, considerar una red que incluye acceso y núcleo, con cientos o miles de dispositivos celulares conectados y miles de posibles acciones -sobre cómo distribuir sus recursos-. La Figura 3.4 busca ilustrar el mecanismo de aprendizaje en *Deep Q-Learning*. El entorno, recibe una acción y retorna estado y recompensa. Las cuales son almacenadas en la memoria de experiencia. Al mismo tiempo, el estado alimenta la red neuronal de evaluación, de la cual se obtienen los valores q y con ello la acción a realizar en la siguiente iteración. Luego de almacenar una determinada cantidad de experiencias, se actualizan las redes neuronales siguiendo el Pseudocódigo 2.

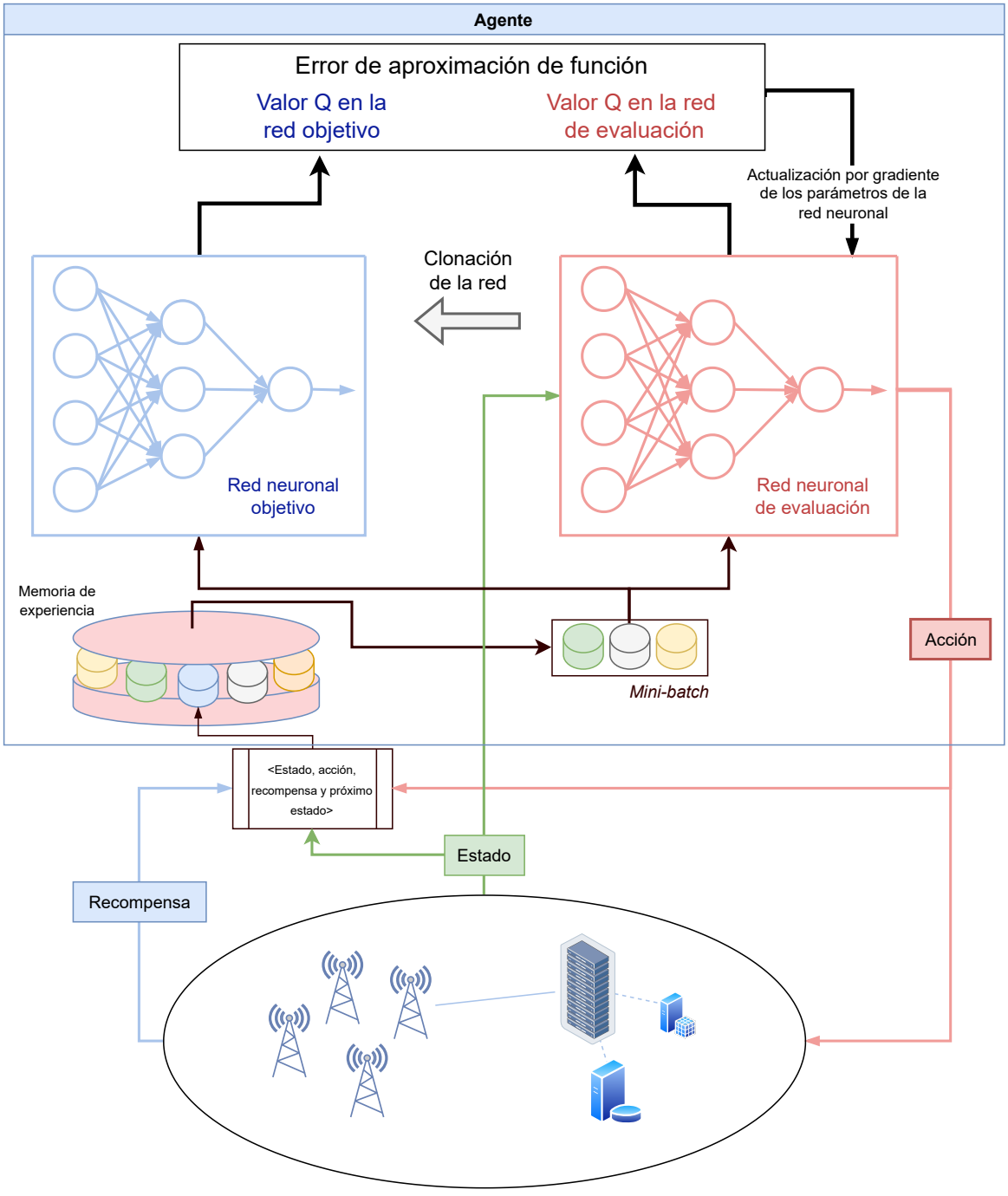


Figura 3.4: Proceso de aprendizaje de DQL sobre un entorno complejo. Adaptado de [35].

Esta página ha sido intencionalmente dejada en blanco.

PARTE II:

ENTORNO DE SIMULACIÓN

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 4

Entorno de simulación

El presente capítulo es fundamental para comprender el desarrollo de la investigación. En él se describe el entorno de simulación utilizado, se analizan sus características y se explica su funcionamiento. Como parte del trabajo de maestría, se ampliaron las funcionalidades del simulador y se corrigieron y adaptaron algunos aspectos prácticos para seguir los estándares establecidos. También se desarrolló una librería que facilita la implementación de nuevos *schedulers*. Además, se construyeron y evaluaron diversos *schedulers* como prueba de concepto.

4.1. Simulador como herramienta de investigación

Contar con una herramienta de simulación permite crear un modelo de un sistema real o hipotético y luego evaluar cómo este sistema se comportaría en diferentes situaciones. Esto es especialmente útil en el área de las redes móviles, pues es costoso realizar pruebas en el mundo real. Un entorno de simulación permite a los usuarios identificar problemas potenciales o incluso encontrar soluciones optimizadas. Además, la realización de cualquier experimento resulta más rápida y eficiente en un entorno simulado que en uno real. Por otro lado, una herramienta de simulación es valiosa para cualquier persona que trabaje con sistemas complejos. Permite a los usuarios evaluar y optimizar el rendimiento de un sistema de manera rápida y económica así como visualizar y comprender cómo funciona un sistema de manera más clara.

No obstante, todo resultado o análisis producto de un trabajo basado en simulaciones debe tener en cuenta las limitaciones inherentes de un entorno que no es real. De hecho, una cuestión que puede plantearse en relación con las simulaciones es ¿cuánto se aproximan las soluciones halladas a una red real?

No es fácil caracterizar una red comercial o real, aún más cuando se trata de sistemas muy complejos como lo son las redes móviles. En escenarios reales, la obtención de medidas puede lograrse mediante estrategias activas o pasivas. Las primeras involucran la experimentación a través de la creación de tráfico. Las segundas involucran la obtención de medidas a partir del tráfico ya existente. Sin embargo, la intervención trae aparejada una serie de consideraciones, tanto

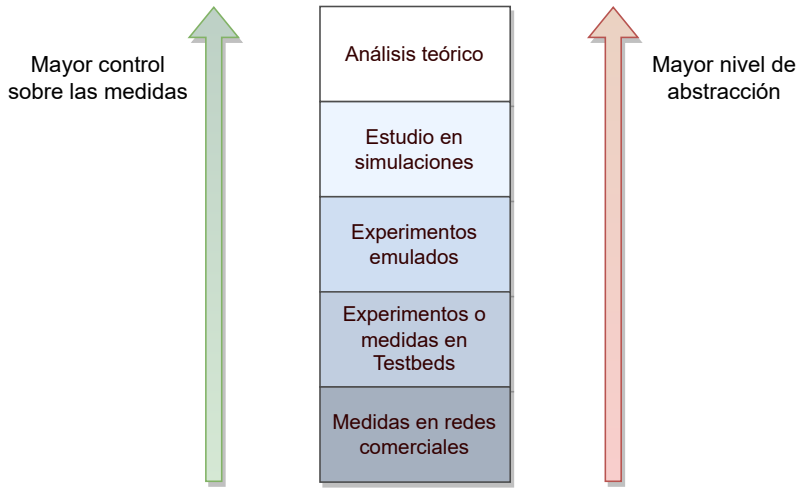


Figura 4.1: Compromiso entre control y nivel de abstracción. Adaptado de [12].

de seguridad como de privacidad de la información e incluso un deterioro en la *performance* de la red en el caso de las mediciones activas -lo cual afectará también a las medidas-. Lo anterior sugiere que, a medida que se utilizan métodos de evaluación a más bajo nivel, existe un compromiso con el control que se tiene sobre el sistema y, por consiguiente, de las medidas. En la Figura 4.1 se puede observar los métodos posibles para la evaluación de una tecnología y este compromiso mencionado.

Si bien se alcanza un alto nivel de abstracción en un entorno de simulación, es de mucho valor contar con uno. Y es que acceder a datos de una red comercial no está al alcance del investigador en la mayoría de los casos. Además, es importante trabajar en mejorar y extender las soluciones ya existentes para lograr una herramienta que pueda ser adoptada por la industria y la academia.

4.2. Simuladores más relevantes a nivel internacional

Actualmente las opciones de *software* de simulación de redes 5G son muy limitadas. En [27] se analizan los entornos de simulación más relevantes y se señalan las características que hacen del simulador una herramienta útil de investigación -y como tal, es la fuente de esta sección-.

Un software de simulación de redes 5G es una herramienta muy compleja, en tanto implica la incorporación de una miríada de especificaciones del estándar. Entre otros, debe soportar:

- Aspectos relacionados con la transmisión en la capa física tales como protocolos de acceso, transmisión de mmWave, massive MIMO.
- Características del control de acceso como *Scheduling*, Adaptación al Enlace, etc.

- Aspectos generales como configuración dinámica de la red, modelado del canal, y soporte para los casos de uso descritos en 2.1

Integrar todas las características de 5G en un mismo simulador muchas veces no es posible; los esfuerzos en el desarrollo de simuladores integran parcialmente las funcionalidades que se describen en el estándar. Algunos de los más relevantes son presentados en la Tabla 4.1 conjuntamente con sus principales características.

Tabla 4.1: Característica de los simuladores. Esta tabla es una adaptación de [27].

Simulador	Características principales
NYUSIM [31]	-Modelado estadístico del canal. -Tiene una interfaz fácil de usar. -Soporta frecuencias desde 2 hasta 73 GHz.
Vienna Simulator [47]	-Incluye simulaciones a gran escala con múltiples nodos y orientaciones multi-capa. -Está disponible de forma gratuita.
WiSE [30]	-Simulaciones a nivel de sistema de redes 5G para orientaciones multi-capa. -Código fuente disponible para simulaciones y validación.
GTEC 5G Simulator [23]	-Simulaciones de enlace de sistemas 5G -Implementaciones de transceptor.
5G Toolbox by Matlab [1]	-Simulaciones de nivel de enlace. -Modelado de canal. -Generación de señales.
5G-LENA [44]	-Modelado de tráfico de red. -Enfoque en la latencia. -Modelado de consumo de energía.

4.3. Py5cheSim

Py5cheSim (ver Figura 4.2) es un entorno de simulación de red 5G desarrollado en el Instituto de Ingeniería Eléctrica. Surge originalmente como resultado del trabajo de maestría en [45]. Sin embargo, el potencial de dicha contribución fue tal que pronto se convirtió en un proyecto más ambicioso, con la participación de diferentes investigadores y desarrolladores.

Figura 4.2: Logo del simulador Py5cheSim.

Después de analizar las opciones disponibles descritas en la sección 4.2 y compararlas con Py5cheSim, se consideró a este último como la opción más adecuada debido a sus características. En particular, el gran diferenciador de Py5cheSim es su capacidad para proporcionar soporte a *network slicing*.

A su vez, Py5cheSim es de código abierto, lo cual permite a los usuarios descargar, utilizar y modificar su código fuente. Además, admite mayor integridad sobre su funcionamiento, fomenta la cooperación en su desarrollo y promueve la transparencia al poder conocer internamente su código.

Por otra parte, Py5cheSim es fidedigno en cuanto al procedimiento para la asignación de recursos. Es decir, incorpora los procedimientos y especificaciones descritas en el estándar. Este aspecto es fundamental teniendo en cuenta la discusión en 4.1, pues garantiza que los resultados obtenidos sean fiables y precisos. De esta manera, permite validar y contrastar los resultados obtenidos en la investigación con los de una red real.

Asimismo, Py5cheSim permite simulaciones a alto nivel. Es esta línea, logra un nivel de abstracción que no requiere un profundo entendimiento de cada aspecto en la comunicación sino aquellas más relevantes. De este modo, logra un enfoque que facilita su adopción por parte de la comunidad de investigadores.

Por último, el simulador fue desarrollado en Python, lo cual lo hace versátil por diseño. Esto permite aprovechar las características del lenguaje orientado a objetos y las librerías ya existentes sobre el aprendizaje automático.

Sumado a todo lo anterior, el simulador está diseñado como *framework* para el desarrollo de *schedulers*. Lo que es más, el proyecto trae consigo ejemplos de posibles *schedulers* para utilizar como referencia.

Algunas otras características puntuales del simulador a considerar son:

- La comunicación duplex a través de división de tiempo (TDD) y división de frecuencia (FDD).
- La definición de diferentes perfiles de tráfico dentro de los escenarios descritos en la Sección 2.1.

Al comparar Py5cheSim con los simuladores en la Tabla 4.1, se considera que es el que mejor se ajusta a uno de los objetivos del presente proyecto: la investigación e implementación de técnicas de *scheduling Inter-Slice*. Por otra parte, es pertinente destacar que, al tratarse de una solución desarrollada en el Instituto de Ingeniería Eléctrica, promueve la investigación y desarrollo nacional.

4.3.1. Funcionamiento general

Para entender las contribuciones de esta investigación al simulador, es necesario tener un conocimiento previo del funcionamiento de la versión original. En esta sección se introduce al lector -desde una perspectiva general- los diferentes módulos con los que cuenta el simulador y cómo interactúan entre sí.

En su versión base, el simulador consta de ocho módulos. Estos son:

- IntraSliceSch.py
- InterSliceSch.py
- Scheds_Intra.py
- Scheds_Inter.py
- Slice.py
- Simulation.py
- Cell.py
- Results.py
- UE.py

Toda simulación que se desee realizar en el simulador mencionado debe tener su configuración establecida en el archivo "*Simulation.py*". En este módulo se definen las características de la celda, de las *slices* y de los perfiles de tráfico. Así como la elección del tipo de *scheduler inter* e *intra-slice*. "*Simulation.py*" no es parte del simulador como tal sino un *script* de *python* que actúa como punto de partida del simulador, una vez configurado basta con ejecutarlo desde una terminal para comenzar la simulación.

Por otro lado, la configuración de los perfiles de tráfico -y, por tanto, de los grupos de terminales (UEs)- se realiza a través de la clase UE definida en "UE.py". En este módulo se definen las clases y métodos necesarios para la creación y manejo del tráfico de los usuarios. Además, define la clase UE, la cual cuenta con múltiples métodos que simulan el comportamiento de un dispositivo celular: encolamiento de paquetes y conexión a la radiobase, entre otros. Asimismo, el módulo contiene la definición de la clase PacketFlow, que gestiona la generación de tráfico. Esto lo logra a través de los atributos definidos en él: la tasa de arribo de paquetes y su tamaño.

El módulo "*Cell.py*" se encarga de inicializar todo lo relacionado con la celda. Allí se encuentra el método *inter-slice* para asignar recursos y también se encuentra el manejo de estadísticas.

Por otro lado, "*InterSliceSch.py*" tiene los métodos necesarios para la asignación de recursos entre *slices*. Desde aquí se llama al método "*resAlloc*", el cual contiene el algoritmo de asignación de recursos. Para lograr mayor control y flexibilidad en el desarrollo de estos algoritmos, el método "*resAlloc*" se sobrescribe desde el módulo "*Scheds_Inter.py*" donde conviven las diferentes implementaciones de algoritmos. A su vez, este último método es invocado una vez por cada tiempo de asignación *inter-slice*, definido en "*Simulation.py*".

Una vez que las *slices* tienen los recursos asignados, "*IntraSliceSch.py*" tiene todos los métodos necesarios para distribuirlos entre los usuarios. Los métodos incluyen el cálculo de los bloques de transmisión, encolamiento para transmisión, selección de modulación y tasa de código, entre otros. Este módulo cuenta con su propio método "*resAlloc*", esta vez para la asignación a las terminales. Análogamente al caso anterior, en "*Scheds_Intra.py*" conviven todos los métodos de asignación de recursos *intra-slice*.

Por último, *"Results.py"* contiene métodos auxiliares para el procesamiento de estadísticas, la creación de gráficos y una clase para gestionar los grupos de UE. Además, tiene un método mediante el cual define el valor de SINR de cada usuario.

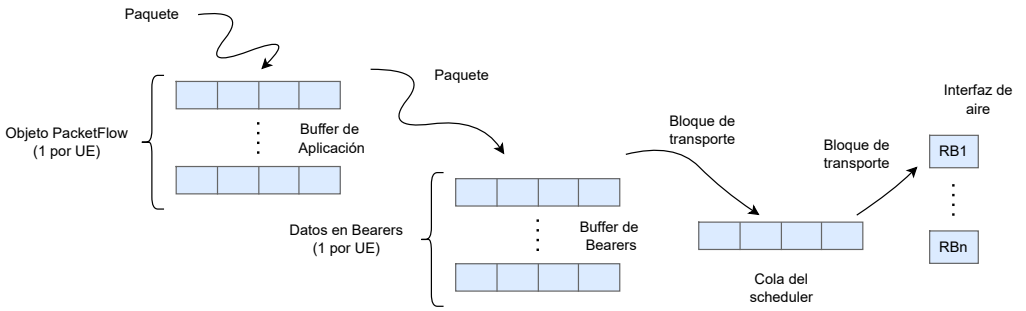


Figura 4.3: Flujo de paquetes, desde su creación hasta que es enviado a la interfaz de aire. Adaptado de [45].

Como se mencionó con anterioridad, los paquetes a transmitir pasan por diversos procesos de encolamiento. Según los atributos del objeto *PacketFlow*, se generan y encolan los paquetes a nivel de capa de aplicación. Luego, estos pasan al buffer de transmisión o de "*bearers*". Este buffer está definido para simular el encolamiento en el enlace de transmisión. Posteriormente con los paquetes encolados en este último, se fragmentan y concatenan aquellos para ser transmitidos a través del tamaño de un bloque de transporte (*TBS* por su sigla en inglés de *Transport Block Size*). La Figura 4.3 ilustra este proceso.

Más detalles de los módulos se proporcionarán en lo que resta del documento. Cabe destacar que los métodos de asignación de recursos están bien definidos y separados del módulo principal. Por lo tanto, para todo nuevo algoritmo que se desee desarrollar, basta con obtener las medidas o estadísticas necesarias para el algoritmo de facto e implementarlo allí.

4.3.2. Reflexión sobre el simulador

El simulador cuenta con mucho potencial como entorno de desarrollo de *schedulers*. En base a la experimentación con el simulador y su evaluación, destacamos las siguientes características:

- Es posible realizar modificaciones puntuales con facilidad debido a su gran modularidad.
- Cuenta con múltiples métodos de recopilación de estadísticas, que son guardadas en forma de archivo de texto al finalizar la simulación, habilitando su posterior análisis. Además, a partir de ellas es posible generar múltiples gráficas en carpetas bien definidas.
- El estilo de programación es claro, aunque no cuenta con comentarios ni documentación que facilite su entendimiento.

- Se valora que -en la mayoría de los casos- se tomaron rigurosamente las especificaciones y procedimientos del estándar.

No obstante, las medidas de rendimiento de las terminales y de las *slices*, las características del enlace y el estado de la red -entre otros- se deben obtener de los módulos principales *"IntraSliceSch.py"* e *"InterSliceSch.py"*. Es allí donde ocurre todo el proceso de encolamiento y gestión de recursos. Por lo tanto, en la versión base es responsabilidad del desarrollador comprender y estudiar el funcionamiento interno de dichos módulos para implementar nuevos algoritmos de asignación. En principio supone un obstáculo, que requiere de un profundo conocimiento de estos módulos, incluyendo nociones del estándar para incorporar mejoras y nuevas implementaciones. La implementación de nuevos *schedulers* podría ser más directa si se obtuviesen más fácilmente las mediciones. A pesar del gran esfuerzo realizado para que el simulador cumpla con el estándar, hay algunos aspectos en los que no se ha podido seguir con todo rigor. Algunos de ellos, de poca relevancia, como lo son los procedimientos de conexión de las terminales, descritas en la Sección 2.3 y Sección 2.4. La incorporación de estas características no es requerida en este simulador de alto nivel debido a su impronta simplista y a que no afectan a la gestión de recursos. Sin embargo, es importante tener en cuenta que en otras situaciones puede ser necesaria su incorporación.

Por otro lado, hay algunos otros aspectos a mejorar en cuanto a procedimientos que sí deben ser ajustados en el simulador, así como características adicionales que podrían enriquecer al simulador. Algunas de las observaciones son las siguientes:

- El simulador original no tiene incorporada la característica de *bandwidth part*, introducida en la Sección 2.5. Esto significa que no hay límites en la cantidad de recursos que pueden ser asignados a las terminales.

- El método utilizado para calcular el tamaño del bloque de transmisión (TBS) es incorrecto y se basa en aproximaciones. Es fundamental realizar esta medida de manera precisa para la validación de resultados.

- Actualmente, se utiliza una distribución de Pareto para modelar tanto la tasa de arribo de paquetes como el tamaño de los mismos. Sin embargo, es posible que diferentes aplicaciones tengan comportamientos y distribuciones diferentes. Es relevante incorporar distintas formas de generar tráfico para enriquecer el simulador.

- Cuando se realizan simulaciones en redes con muchos usuarios conectados, el simulador puede ralentizarse. Esto puede constituir un obstáculo en algunas situaciones, pero es posible sortearlo si se realizan sobre equipos con muchas prestaciones o al tener consideraciones de escala sobre las simulaciones.

- En cuanto a la adaptación al enlace, el simulador cuenta con varias simplificaciones de este mecanismo. No se incorpora un sistema de reconocimientos como *HARQ* (ver Sección 2.6). La asignación de esquema de modulación y tasa se obtuvo de métodos empíricos (ver capítulo 4.1.2 de [45]).

- Sumado a lo anterior, se incorpora únicamente un mapeo de esquema de modulación y tasa. Siendo que el estándar permite dos (ver GPP TS 38.214 [9]) para distintos escenarios.

Visto las consideraciones del simulador, se consideró pertinente realizar modificaciones al simulador como herramienta metodológica, precedente al estudio de las técnicas de asignación *inter-slice*.

En los siguientes capítulos se presentarán los aportes al simulador como herramienta de trabajo, los algoritmos implementados en el mismo y sus resultados. Y el documento finalizará con las conclusiones del presente trabajo.

PARTE III:

APORTES

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 5

Aportes al simulador

En el Capítulo 4 se introdujo Py5cheSim como el simulador seleccionado para el proyecto, se justificó su elección y se realizó un análisis detallado de sus principales características. Este análisis permitió identificar algunos aspectos que requerían mejoras para enriquecer el simulador como herramienta de trabajo.

En este capítulo se presentan los aportes realizados al simulador utilizado durante la investigación, a saber:

1. Se ha realizado una serie de mejoras en el simulador, incluyendo la adición de nuevas características para la generación de paquetes y la obtención de estadísticas. Además, se han llevado a cabo ajustes en los módulos existentes para asegurar su cumplimiento con las especificaciones y procedimientos del estándar. Estas actualizaciones y modificaciones permiten un mejor funcionamiento del simulador y una mayor precisión en los resultados obtenidos.

2. Se ha creado Py5cheLiSA, una biblioteca diseñada para simplificar y facilitar el desarrollo de algoritmos de *scheduling*.

5.1. Adición de nuevas características

Para garantizar la efectividad de una investigación en la que se utiliza un simulador, es fundamental que este cuente con las características necesarias que permitan obtener resultados precisos y confiables. Con este objetivo en mente, se ha tomado la decisión de agregar nuevas características al simulador, basadas en diversas necesidades identificadas:

- En primer lugar, se busca que el simulador se ajuste completamente a las especificaciones del estándar, garantizando así su total compatibilidad y validez en el contexto de la investigación.

- Por otro lado, se ha considerado fundamental la capacidad del simulador para emular distintos perfiles de tráfico, lo que permitirá obtener resultados más representativos y realistas.

- En línea con lo anterior, se ha reconocido la importancia de poder activar una *slice* en un momento específico, lo que brinda una mayor flexibilidad en la simulación de diferentes escenarios.
- Finalmente, se ha incluido la necesidad de medir el *delay* de los paquetes transmitidos con el fin de evaluar la eficiencia del simulador.

Con estas nuevas características, se espera lograr una simulación más precisa y efectiva, con resultados confiables y de alta calidad.

Cálculo de TBS

En la Sección 4.3.1, se mencionó el bloque de transporte como la unidad lógica a través de la cual se transmiten los datos por la interfaz de aire. El tamaño máximo del bloque de transporte determina la cantidad de datos que se enviarán en un intervalo de tiempo de transmisión (TTI). Este depende de la calidad del enlace y la cantidad de recursos asignados. Para calcular el tamaño del bloque de transporte se utiliza la siguiente fórmula:

$$N_{info} = N_{RE} \cdot R \cdot Q_m \cdot l \tag{5.1}$$

Donde N_{RE} es la cantidad de *resource elements* asignados, R es la tasa de transmisión de datos, Q_m es el número de bits transmitidos por símbolo y l es la cantidad de capas en el caso de utilizar SU-MIMO.

Para conocer el tamaño máximo en la transmisión de datos se toma el tamaño calculado del bloque de transporte. El tamaño máximo en la versión original del simulador se aproximaba por N_{info}. Sin embargo, este criterio, aunque aproximado, puede generar pequeñas desviaciones en el rendimiento de una red simulada.

En el presente trabajo se ajustó el tamaño máximo del bloque de transporte siguiendo la especificación del estándar. En este sentido, si el valor de N_{info} es inferior a 3824, se compara con la Tabla 7.1, en donde se toma el valor inmediatamente inferior como su valor cuantizado. En caso de que N_{info} sea mayor a 3824 se evalúa mediante expresiones matemáticas según su tasa de código. Todas estas expresiones están disponibles en el procedimiento detallado en [9].

Particionamiento de ancho de banda

En la Sección 2.5, se presentó el concepto de particionamiento de ancho de banda (también conocido como *bandwidth part*). Ahora bien, es importante contar con su incorporación para lograr simulaciones que tengan el desempeño de redes reales. El simulador en su versión original no cuenta con dicha característica, por lo que, como parte de la investigación, se adicionó en el simulador según lo especificado en el procedimiento descrito en [9].

Es importante destacar que, dado que el *bandwidth part* afecta directamente al *scheduling intra-slice*, debe incorporarse en el archivo *"Scheds_Intra.py"*. Como nuevo aporte al simulador se han implementado funciones que integran el concepto

de *bandwidth part* para la asignación de recursos y se han modificado los *schedulers* ya existentes para que lo incorporen.

El *bandwitdth part* está relacionado con el *scheduling* en tanto es quien limita la cantidad de recursos asignables a un dispositivo. Así, se implementó una aproximación al particionamiento de ancho de banda (BWP) en el simulador.

Perfiles de tráfico

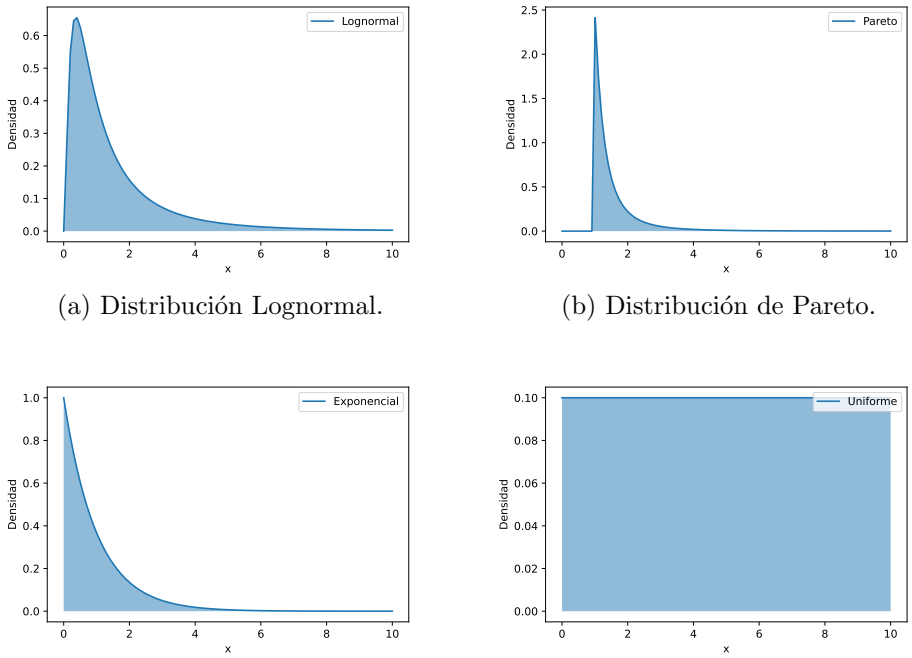
Una de las características relevantes a tener en cuenta en un simulador es la capacidad de representación del tráfico. Tener distintos perfiles de tráfico -es decir, distintas distribuciones de arribo de paquetes y de tamaño- puede enriquecer significativamente un simulador. Esto permite simular situaciones realistas y evaluar cómo se comporta la red en diversos escenarios. Además, utilizar diferentes perfiles de tráfico también permite evaluar la eficiencia de diferentes políticas de gestión de recursos. En resumen, contar con distintos perfiles de tráfico permite obtener una representación más precisa y realista del comportamiento de la red, lo cual es esencial para el diseño y optimización de una red 5G.

Como se mencionó en la Sección 4.3.2, en la versión original del simulador es posible caracterizar el perfil de tráfico de los usuarios anunciando el tamaño medio de los paquetes -en bytes- y el tiempo medio entre arribos de paquetes. Esta información es tomada por el módulo UE.py en donde se incluyen funciones que devuelven el tiempo y tamaño correspondiente. Lo anterior se logra a través de distribuciones de Pareto que toman como media los datos ingresados. En particular, esto se realiza mediante la función *paretovariate* de la librería *numpy* [28]. En la versión original del simulador las funciones que devuelven el valor correspondiente de la distribución son *getPsize* y *getParrRate*. Estas funciones son pertinentes pues es allí en donde se programan nuevas distribuciones de tráfico.

Con el fin de dotar al simulador con nuevas posibilidades para el perfil de tráfico es que se integraron las siguientes distribuciones de tamaño como de arribos de paquetes:

1. Distribución lognormal
2. Distribución uniforme
3. Distribución exponencial
4. Valores constantes

A partir de la combinación de ellas es posible obtener perfiles de tráfico muy diversos y atractivos que simulen el comportamiento de ciertas aplicaciones. El comportamiento cualitativo de estas distribuciones se puede observar en la Figura 5.1. Por ejemplo, al suponer una distribución uniforme de arribos de paquetes y un tamaño constante para los mismos, es posible modelar aplicaciones industriales o de automatización. La distribución lognormal y exponencial, al igual que la de Pareto, pueden ser combinadas para representar tráfico multimedia. Diversos estudios se han realizado sobre este campo [24] [41].



(a) Distribución Lognormal.

(b) Distribución de Pareto.

(c) Distribución Exponencial.

(d) Distribución Uniforme.

Figura 5.1: Distribuciones integradas al simulador Py5cheSim.

Activación/Desactivación de grupos de tráfico

Una característica necesaria para modelar adecuadamente el comportamiento de una red real es la capacidad de incorporar el tráfico de los grupos de usuarios a partir de un momento seleccionado de manera arbitraria.

Como nueva adición al simulador, se incorporó la característica de iniciar y finalizar la transmisión de paquetes de los grupos de usuarios. Este aspecto agrega dinamismo al sistema y enriquece los algoritmos de asignación. Al mismo tiempo, habilita a estudios más profundos de los mismos y a la obtención de resultados más realistas.

La activación y desactivación de los grupos de usuario se puede encontrar en el archivo de configuración del simulador "Simulation.py". Allí se anuncia el inicio y final como un porcentaje del tiempo total de simulación. En la Figura 5.2 se observa un ejemplo de uso de una *slice* configurada para intercambiar datos desde el inicio hasta la mitad del tiempo de simulación.

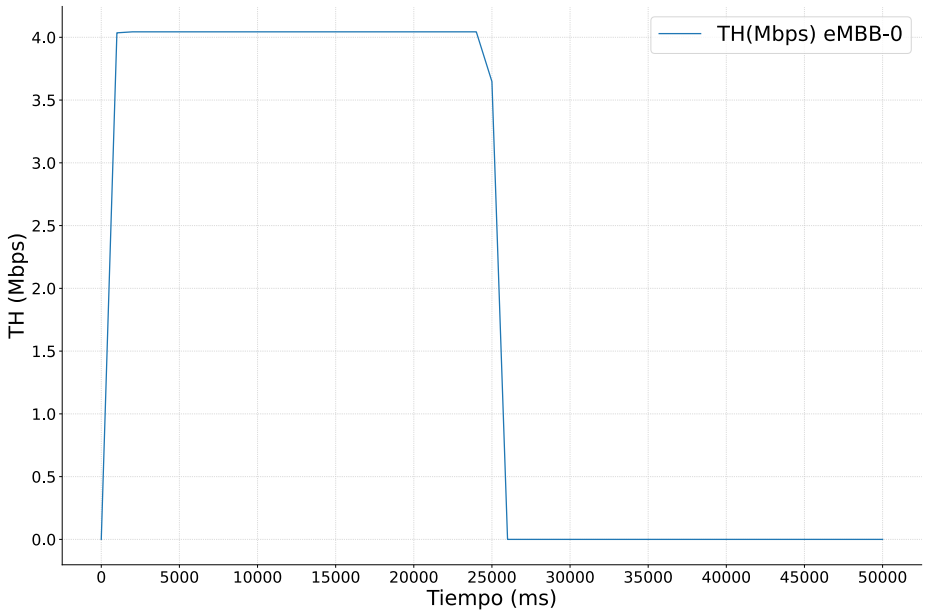


Figura 5.2: *Throughput* de la slice configurada para iniciar con la simulación y finalizar en la mitad del tiempo.

Es interesante observar el comportamiento del algoritmo Round Robin introducido en la Sección 2.8.1. Allí los recursos son asignados de igual manera entre las *slices*, y la variante implementada en el simulador es que si una *slice* no cuenta con tráfico, no se le asignará recursos. En este contexto se implementó un escenario con dos *slices*: eMBB-0 y eMBB-1 (ver Figura 5.3). La primera permanece activada la primera mitad del tiempo de simulación. La segunda está activa desde el primer cuarto del tiempo de simulación hasta el final. Mientras solo una está activada, todos los recursos irán hacia ella. En cuanto ambas comparten el mismo instante de activación -desde el cuarto hasta la mitad del tiempo total-, los recursos se reparten equitativamente entre ellos.

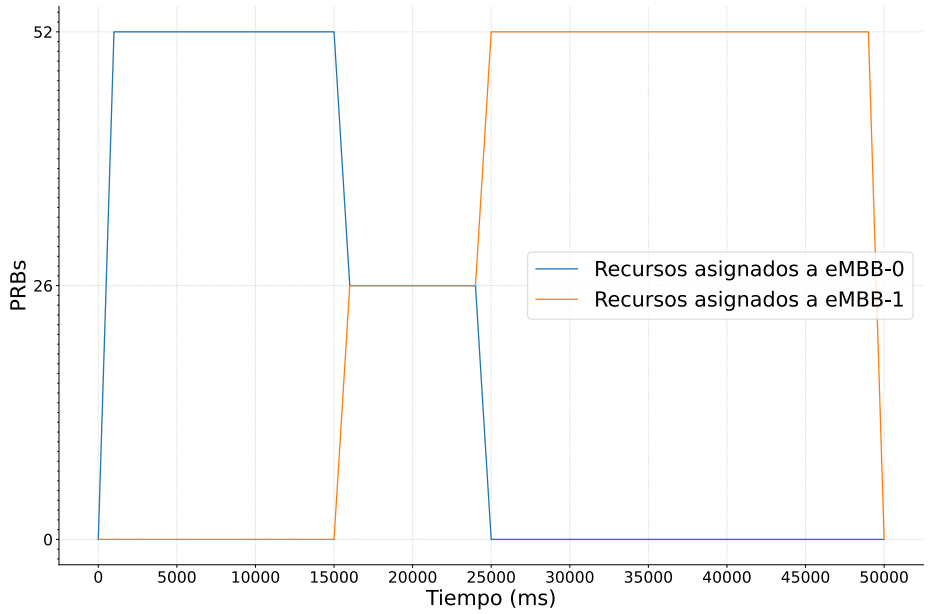


Figura 5.3: Recursos asignados, utilizando *Round Robin* y diferentes zonas de activación entre dos *slices*.

Esta característica será de mucha utilidad en la etapa de implementación de algoritmos de asignación *inter-slice*, pues abre la posibilidad de probar una mayor cantidad de escenarios. La interacción de las *slices* en ausencia o presencia de las demás permite evaluar desde una perspectiva más realista los algoritmos de *scheduling*.

Medición del *delay*

El retraso (o *delay* en inglés) es un concepto clave en las redes de paquetes, ya que se refiere al tiempo que transcurre desde que un paquete de datos es enviado hasta que es recibido. Este retraso puede ser causado por varios factores, siendo el más común la congestión en la red. Tenerlo en cuenta es relevante pues puede afectar el desempeño de la red y la calidad de servicio percibida por los usuarios finales.

En la presente extensión del simulador se ha caracterizado el *delay*, y es posible ahora tomarla como insumo para la asignación de recursos. Se mide el tiempo desde que el paquete está en *buffer* de transmisión hasta que efectivamente es enviado. No obstante, no se incorporó las restricciones en los tiempos (k0, k1 & k2) según lo especificado en [9]. Si bien no es determinante dicha implementación, el concepto de *delay* permite la construcción de este y futuros procedimientos.

5.2. Py5cheLiSA

Otro aporte a resaltar es la creación de la biblioteca Py5cheLiSA (*Py5cheSim Library for Scheduling Algorithms*). Su principal objetivo es permitir al usuario final implementar algoritmos de manera eficiente y sencilla, sin necesidad de estudiar el funcionamiento del simulador completo. Así, se trata de un proyecto de código abierto para Py5cheSim y presenta un valioso conjunto de herramientas para implementar *schedulers intra-slice* e *inter-slice*. En cuanto a los cálculos necesarios para la obtención de métricas, se presenta un enfoque novedoso que aborda cualquier algoritmo de asignación compleja desde un análisis matricial. Además, la biblioteca incluye un módulo que facilita el desarrollo de *schedulers* basados en IA. Para hacerlo fácil de usar, se ha documentado todos los módulos de la biblioteca con *Pydoctor* [3].

A continuación, se explicará el funcionamiento de Py5cheLiSA, que se puede dividir en dos partes interconectadas: una orientada a facilitar el desarrollo de algoritmos *intra-slice* y otra para *inter-slice*.

5.2.1. Py5cheLiSA *intra-slice*

Esta primera parte consiste en tres módulos: "Scheduler Utils", "UE statistics" y "AI Schedulers Utils", diseñados para su uso complementario, según las necesidades del usuario.

Asignación por métricas de usuario

Las prioridades de *Scheduler Utils* consisten en ser una herramienta versátil y eficiente. Como núcleo de la propia biblioteca, contiene las principales herramientas para desarrollar algoritmos de asignación de recursos, implementados mediante la interacción directa con las clases de Py5cheSim (presentadas en el Capítulo 4).

La asignación de recursos *intra-slice* se lleva a cabo observando el estado actual de los dispositivos y utilizando una métrica previamente definida. Para cada usuario, su métrica es el resultado de la operación de diferentes medidas que dan cuenta del estado del usuario, tales como el *throughput pasado* y la SINR, entre otras. Se puede considerar al *scheduler* como un sistema que tiene como entrada el conjunto de medidas de cada usuario y como salida, la asignación de recursos para cada uno de ellos.

No solo la asignación es una tarea difícil, sino que también debe manejarse de manera eficiente, para que no requiera un alto costo computacional.

A partir del trabajo en [20], se ha incluido en este módulo de la biblioteca un enfoque novedoso que permite calcular las métricas de cada uno de los dispositivos UE. Este método considera la manipulación de matrices para dicho objetivo, lo cual resulta de especial utilidad en tanto proporciona una forma poderosa y compacta para que el usuario final resuelva este problema con facilidad. Los autores de [20] presentaron el algoritmo basado en matrices y su implementación diseñada para el simulador 5G Vienna en Matlab.

En la presente implementación, la columna vertebral de este sistema de matrices es la clase framework_UEfactor, la cual contiene la descripción del cálculo de las métricas de los *schedulers*. Esta clase constituye el punto de partida para que el usuario final desarolle su propio algoritmo.

Dentro de la clase principal, se define un atributo "*layer*", componente clave para el diseño de la métrica de los *schedulers*, ya que contiene la descripción de las operaciones realizadas por el algoritmo.

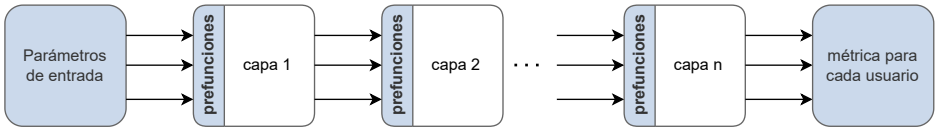


Figura 5.4: Diagrama del funcionamiento matricial de la librería.

Una sola capa puede realizar un conjunto de operaciones de matrices a partir de sus coeficientes. Si se utilizan varias capas consecutivamente, es posible lograr cualquier métrica dada (ver la Figura 5.4 para más detalles). Cada capa tiene tres campos de entrada: el tipo, la matriz de coeficientes y las funciones previas a realizar. Los valores de los coeficientes de la matriz dependerán del tipo de operación que realizará la capa, lo cual depende de su tipo.

Formalmente, se tiene que dada una matriz C de la forma

$$C = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1m} \\ c_{21} & c_{22} & \dots & c_{2m} \\ \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & c_{nm} \end{bmatrix}$$

Es posible definir dos tipos de capas en función del cálculo de la métrica:

- Combinación lineal de entradas (tipo S): La salida Y se definirá como

$$Y = CX$$

- Producto de entradas elevadas a exponentes (tipo P): En este caso, la salida también será un arreglo, donde el componente y_i tendrá la forma

$$y_i = \prod_{j=0}^{n} x_j^{c_{i,j}}.$$

Aunque es posible obtener casi cualquier operación que se proponga utilizando este método, puede resultar necesario realizar funciones que no están contempladas en los tipos de capas, tales como trigonométricas, exponenciales y logarítmicas. Para estos casos particulares, se habilita el campo de *funciones previas*. A través de él, se aplica la función específica a la entrada de la capa x_i, antes de realizar la operación de matrices.

Para ilustrar el funcionamiento de este módulo, a continuación se presenta un sencillo ejemplo de implementación: el algoritmo *Proportional Fair*. Tal como se introdujo en la Sección 2.8.1, el algoritmo de *Proportional Fair* busca asignar recursos, manteniendo una noción de equidad entre los dispositivos que no transmitieron en el pasado reciente. Más especificamente, la asignación depende de la relación entre la capacidad de transmisión actual que puede lograr un UE y la historia de transmisiones pasadas. Por lo tanto, si no se han asignado recursos a un UE en el pasado reciente, hay más posibilidades de hacerlo en el presente. La métrica para cada dispositivo se elige de la siguiente manera:

$$metric_{UE_i} = \frac{actualThroughput^n}{pastThroughtput^m} \tag{5.2}$$

Donde UE_i representa al usuario i, y n y m ponderan el valor final de la métrica. Los valores altos de m priorizan a aquellos dispositivos que no han transmitido en el pasado reciente, mientras que los valores altos de n priorizan al dispositivo con un rendimiento actual alto.

Al utilizar Py5cheLiSA, en particular los módulos *Utils* y *Statistics* correspondientes, se podría seguir los siguientes pasos para construir un algoritmo *Proportional Fair*:

1. Primero, se define un objeto de *scheduling* con una sola capa de tipo P con coeficientes $[n, -m]$. Donde n y m son valores arbitrarios definidos por el usuario. Esta capa imita la ecuación 5.2.

2. Luego, se obtienen las medidas de cada dispositivo. En este caso, para cada usuario se tomará un vector de dos elementos conteniendo el *throughput* alcanzable y el nivel promedio de *throughput* alcanzado en una ventana de tiempo.

3. Posteriormente, del *scheduler* definido y las medidas obtenidas, utilizando la función "assignUEfactor" del módulo "Scheduler Utils" se obtiene la métrica para cada usuario. La Figura 5.5 muestra este proceso.

4. Por último, se busca la máxima métrica entre los usuarios y finalmente se asigna los recursos para cada uno de ellos mediante la función "prbs_allocate" (de *Scheduler Utils*).

Este algoritmo sirve como base de desarrollo de nuevos *schedulers*, modificando únicamente los parámetros de entrada y las capas de procesamiento.

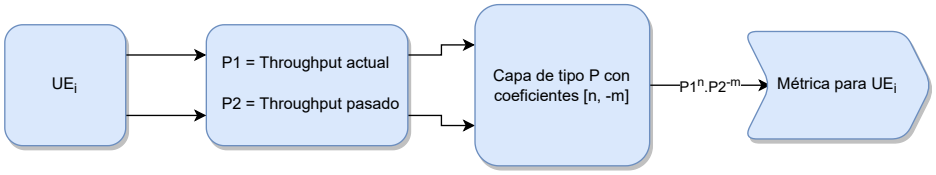


Figura 5.5: Cálculo de métricas.

Obteniendo las estadísticas

Como se mencionó anteriormente, la asignación de recursos se basa en el estado actual del sistema. Teniendo esto en cuenta, se ha desarrollado el módulo de estadísticas (*UE statistics*), que proporciona al usuario final herramientas para obtener mediciones de los dispositivos sin la necesidad de conocer en detalle el núcleo de Py5cheSim. El uso estándar de este módulo se da en conjunto con el módulo *Scheduler Utils*, como se muestra en la Figura 5.6. El módulo *UE statistics* obtiene las mediciones directamente del simulador y las intercambia para ser utilizadas por el módulo principal de la biblioteca.

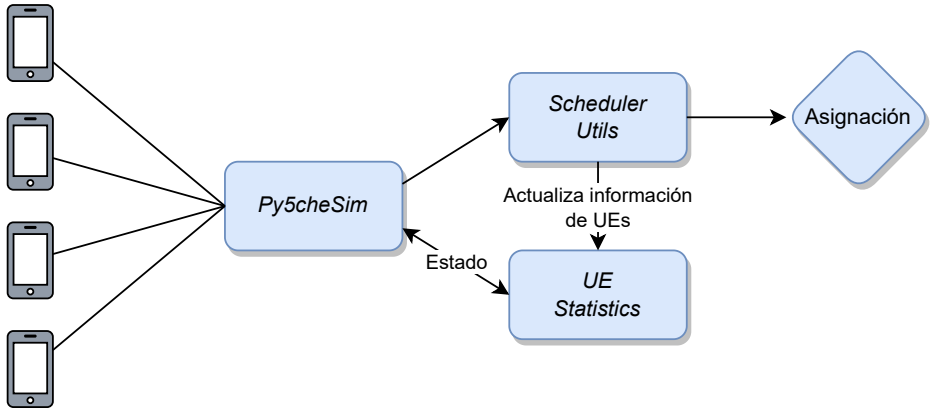


Figura 5.6: Interacción de módulos de Py5cheLiSA.

Algunas estadísticas que es posible obtener con este módulo son:

- Tamaño de la cola de paquetes.
- Channel quality indicator (CQI).
- Índice de modulación y codificación.
- *Throughput* alcanzable.
- *Throughput* medio en una ventana de tiempo arbitraria.
- *Delay* medio.
- Utilización media de los recursos.
- La relación señal a ruido.
- Cantidad de recursos asignados promediado en una ventana de tiempo arbitraria.

Si bien el presente módulo se encuentra en una etapa inicial, se prevé su posterior desarrollo y profundización en futuras versiones.

Inteligencia artificial para el *scheduling*

El *scheduling* involucra resolver el problema de asignar un recurso finito entre un conjunto de usuarios. Este problema se complejiza al considerar diferentes perfiles de tráfico. En particular así sucede en 5G, con sus diferentes requisitos de servicio, como se introdujo en la Sección 2.1. Como consecuencia, se evalúan dos enfoques posibles para el desarrollo de un algoritmo:

- Resolver la asignación como un problema de optimización convencional. En este caso, se obtendría una asignación a través de una métrica dada por una formulación. En estos casos su implementación se daría a través del módulo *Scheduler Utils*.
- Abordar el problema a través de algoritmos de IA.

Dado que la primera opción se puede lograr utilizando los módulos ya presentados, se decidió incorporar un módulo que facilita el desarrollo de algoritmos de IA. Este módulo contempla el desarrollo de *schedulers* basados en aprendizaje supervisado y en aprendizaje por refuerzo. Su principal objetivo es ser una plataforma que guíe el desarrollo de este tipo de *schedulers*. Sin perjuicio de lo cual puede haber algoritmos de *scheduling* que requieran mayores sofisticaciones que las que este módulo puede brindar.

Dentro de este módulo se dispone de varias clases, cada una brindando soporte para un tipo de aprendizaje de máquina. Una de ellas es denominada "DQNAgent", cuyo objetivo radica en brindar las herramientas requeridas para la creación de algoritmos basados en DQL. Esta clase contiene las herramientas necesarias para:

- Crear la red neuronal.
- Entrenar el modelo y actualizar la función Q.
- Realizar predicciones.
- Guardar el modelo.
- Cargar un modelo existente.

Por otro lado, también existe la opción de utilizar el algoritmo básico de Q-Learning sin redes neuronales. Para ello, se facilitan funciones que actualizan la tabla Q a través de la clase "qlearning_scheduler".

Finalmente, también se incorporó herramientas para la confección de algoritmos basados en SVM[1]. Las herramientas mencionadas forman parte de la clase "supervised_scheduler", la cual proporciona las funcionalidades necesarias para la adquisición de datos y su posterior entrenamiento. A modo de ejemplo, en la

[1]Las Máquinas de Vectores de Soporte (SVM, por sus siglas en inglés) son un tipo de algoritmo de aprendizaje automático supervisado utilizado para resolver problemas de clasificación y regresión.

nueva versión del simulador se cuenta con un algoritmo SVM que aprende a asignar recursos como un algoritmo *Proportional Fair*. Durante un período inicial de aprendizaje, el algoritmo es entrenado para que, posteriormente, pueda replicar su comportamiento ante nuevas entradas.

Es fundamental resaltar que el presente módulo no tiene la intención de constituir una solución per se, sino que requiere que el usuario cuente con conocimientos previos acerca del funcionamiento de cada técnica de aprendizaje. Es decir, se espera que el usuario tenga una comprensión previa de las técnicas de aprendizaje pertinentes para poder aprovechar plenamente el contenido de este módulo.

5.2.2. Py5cheLiSA inter-slice

Hasta ahora, se ha descrito la asignación de recursos para dispositivos finales. Por otra parte, Py5cheLiSA también cuenta con módulos destinados al desarrollo de *schedulers inter-slice*. Para ello, se desarrollaron los módulos "Slice Statistics", "AI Scheduler Utils" y se extendió el ya introducido *Scheduler Utils*. El escenario de *scheduling* entre *slices* comparte muchas características con el de *intra-slice*, por lo que no se profundizará en los detalles de la sección anterior.

Utils para la asignación a *slices*

Se puede establecer un paralelismo entre dispositivos y *slices*. La asignación de recursos entre dispositivos debe hacerse con los recursos de la *slice*. Los recursos de esta última se distribuyen desde la celda. Para decidir la asignación entre *slices*, es posible asignarles una métrica de la misma manera que en el caso de UEs. La métrica a seguir se calcula como una función de las estadísticas de la *slice*. Por lo tanto, es conveniente tener un mecanismo que facilite la obtención de la métrica y la asignación.

Se emplea un mecanismo similar al descrito en la Sección 5.2.1, en donde la asignación se realiza mediante el uso de matrices, tal como se propuso en dicha sección.

Las herramientas necesarias para llevar a cabo este proceso se encuentran definidas en el módulo *Scheduler Utils*, donde se hace uso de la clase framework_Slice para establecer el *scheduler* con sus respectivas capas de procesamiento.

En cuanto al proceso ilustrado en la Figura 5.7, se comienza creando el objeto de *scheduling* entre *slices*, que incluye sus correspondientes capas de procesamiento. A partir de estos, se obtienen las métricas asociadas a cada *slice*. Posteriormente, se realiza la asignación de recursos a partir de la(s) *slices* que presentan la mayor métrica obtenida.

Mediciones de las *slices*

Las mediciones y estadísticas son piezas esenciales para llevar a cabo el problema de asignación. El módulo *Slice Statistics* contiene un repertorio de funciones para obtenerlas. Estas medidas están directamente relacionadas con el rendimiento de la *slice* hacia los usuarios. Algunas medidas que proporciona el módulo son:

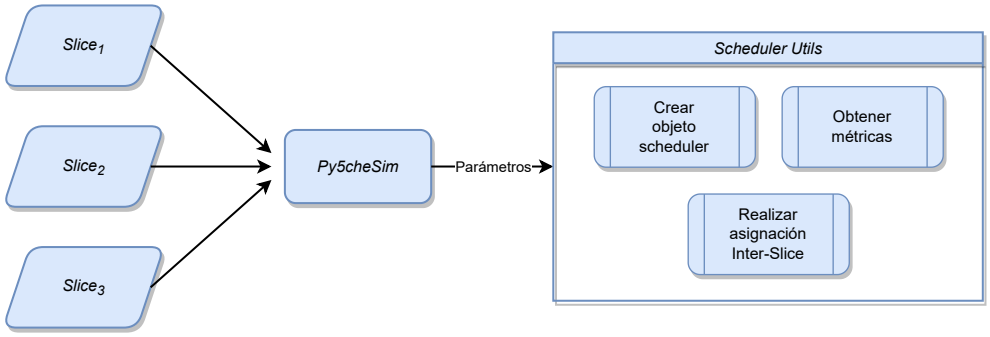


Figura 5.7: Asignación de recursos a *slices*.

- El número de paquetes recibidos dentro de una ventana de tiempo específica.
- El número de dispositivos en la *slice* que alcanzan un cierto nivel de servicio definido por el usuario.
- La eficiencia en la utilización de los recursos.
- El *throughput* total alcanzado por la *slice*.
- El *throughput* medio de la *slice* en una ventana de tiempo.

Las medidas mencionadas pueden ser combinadas para generar diversos algoritmos de asignación. Se tiene previsto seguir incorporando nuevas medidas en este módulo.

IA entre slices

Aprovechando el módulo de estadísticas de *slices*, es posible diseñar modelos que aprendan una estrategia de asignación óptima *inter-slice* basada en inteligencia artificial. Para brindar una plataforma de desarrollo, se creó el módulo *AI Slice Scheduler Utils*.

En principio, este módulo guarda similitudes con el presentado en la Sección 5.2.1; sin embargo, se decidió mantenerlo como un módulo independiente para permitir su utilización en futuras implementaciones de manera autónoma.

5.2.3. Pautas para el diseño de *schedulers*

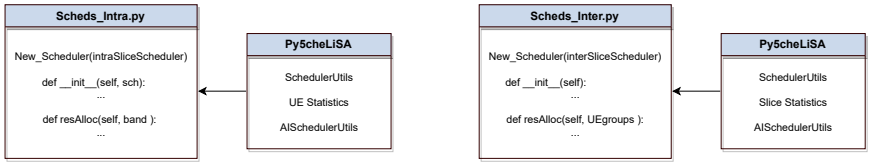
Cualquier nueva implementación de un *scheduler inter* o *intra-slice* debe realizarse como una nueva clase dentro del simulador. Estas clases deben mantener una estructura, siendo estas heredadas de una clase general (como se explica en la Sección 4.3.1). Para cada caso, se tiene la siguiente configuración:

- *Scheduler intra-slice*: cualquier nuevo algoritmo de asignación de recursos se aloja en el archivo "Scheds_Intra.py" de Py5cheSim. La clase correspondiente hereda de la clase padre IntraSliceScheduler. Finalmente, se debe implementar

una función "resAlloc" que sobrescribe una función de su clase padre y permite el desarrollo de algoritmos.

- *Scheduler inter-slice*: Cada nuevo *scheduler* debe implementarse en el archivo "Scheds_inter.py" del simulador y debe definirse como una clase que herede de la clase InterSliceScheduler. Luego, en la función "resAlloc" es donde se implementa el algoritmo, utilizando los módulos de la biblioteca según corresponda.

El vínculo entre estas nuevas clases y la biblioteca se describe en la Figura 5.8.



(a) Diseño de *scheduler Intra-Slice.* (b) Diseño de *scheduler Inter-Slice.*

Figura 5.8: Implementación de nuevos *schedulers*.

5.3. En suma

En este capítulo se han expuesto las principales modificaciones realizadas al simulador seleccionado, entre las que se encuentran la inclusión de nuevas características y ajustes al estándar. Asimismo, se ha presentado Py5cheLiSA como un conjunto de herramientas útiles para el desarrollo de nuevos *schedulers*, con el propósito de facilitar la experiencia de los usuarios que se acercan por primera vez a este simulador.

La mayor parte de las modificaciones implementadas fueron diseñadas con el propósito de aportar a la investigación en curso. En el siguiente capítulo, se presentarán los hallazgos más significativos de la presente investigación. Varios de los aportes presentados en este capítulo fueron empleados para obtener dichos resultados.

Capítulo 6

Algoritmos de asignación de recursos

Hasta este momento, se ha proporcionado al lector una introducción detallada sobre la tecnología 5G en el Capítulo 2. Se presentaron conceptos básicos tales como la estructura de trama, escenarios de uso, acceso al medio y características de la comunicación. En la Sección 2.8.1 se introdujeron los algoritmos de asignación de recursos más utilizados en redes celulares y, en general, en sistemas que requieren asignación de recursos entre dispositivos. Sin embargo, en 5G su elección juega un rol aún más crucial, pues debe tener la capacidad de soportar exigentes demandas de un tráfico heterogéneo.

En el Capítulo 3, se presentó el paradigma del aprendizaje por refuerzo, abarcando desde conceptos básicos hasta una comprensión detallada del algoritmo de *Deep Q-Learning*. Asimismo, en el Capítulo 4, se presentó Py5cheSim, un entorno de simulación de código abierto para 5G, en el cual se destacaron sus principales características. Finalmente, en el Capítulo 5, se explicaron los aportes al simulador como herramienta de investigación para el presente proyecto. Todos estos temas se unen para proporcionar una comprensión sólida de los resultados prácticos más relevantes de la investigación, que se presentan en este capítulo.

El problema objetivo de la investigación -como se mencionó previamente- es la implementación y evaluación de algoritmos del estado del arte en la gestión de recursos para redes 5G. En particular, aquellos que utilizan técnicas basadas en DRL.

Ahora bien, las redes 5G son un tema de mucha actualidad y su estandarización continúa más allá de los despliegues que se realizan en el mundo. Por tanto, la literatura e investigaciones sobre sus prestaciones, y en particular sobre algoritmos de *scheduling*, son también muy recientes. Sin embargo, se han realizado múltiples esfuerzos por parte de la industria y la academia en desarrollar y caracterizar a los algoritmos basados en aprendizaje por refuerzo aplicados a estas redes.

A efectos de la presente investigación, dos algoritmos fueron seleccionados en base a tres criterios: su potencial de crecimiento, cuán implementables son en el simulador y su capacidad de demostrar la aplicación de la inteligencia artificial en la distribución de recursos. Estos algoritmos son los presentados en [35] y [10].

El presente capítulo se estructura de la siguiente manera: en primer lugar, se introduce el escenario de evaluación en tanto resulta compartido para ambos

algoritmos; luego, se presenta el primer algoritmo implementado, junto con sus características, análisis del entrenamiento, resultados de su aplicación tanto en el entorno de evaluación original como en nuevos escenarios; posteriormente, se introduce el segundo algoritmo siguiendo el mismo esquema del anterior; finalmente se realiza un análisis comparativo entre ambos algoritmos.

6.1. Escenario de evaluación

Para comparar y evaluar correctamente ambos modelos es importante la consideración del escenario de funcionamiento. Para ello, existen dos posibles caminos:

- Evaluar los algoritmos en el mismo escenario planteado del trabajo original.
- Utilizar un escenario común para la experimentación de los algoritmos.

El primero tiene como ventaja poder comparar los resultados con los descritos en el trabajo original. No obstante, para lograrlo es necesario contar con idénticas condiciones de evaluación. Tener como base diferentes simuladores implica que los resultados no solo dependen de la implementación del algoritmo sino de otras hipótesis que escapan a nuestra investigación.

El segundo camino, por su parte, tiene la riqueza de poder comparar los algoritmos en igualdad de condiciones. De esta manera es posible visualizar las principales diferencias de desempeño sin depender del simulador y su implementación a bajo nivel. Por esta razón se decidió seguir este camino.

El escenario en cuestión fue definido de forma arbitraría para contemplar la diversidad de los casos de uso. Incluye a tres grupos de usuarios alojados en tres *slices* diferentes, etiquetadas como eMBB-0, eMBB-1 y eMBB-2.

Tabla 6.1: Perfiles de tráfico del escenario de evaluación.

	eMBB-0	eMBB-1	eMBB-2
Cantidad de usuarios	4	4	5
Tamaño de paquetes	Constante 300 bytes	Pareto [Media = 410 bytes]	Lognormal [Media = 800 bytes, Máx =900 bytes]
Tiempo de arribo entre paquetes	Uniforme en el intervalo [0, 0.6] ms	Uniforme en el intervalo [0,1.2] ms	Uniforme en el intervalo [0,1] ms
SLA: Tasa de datos	7.5 Mbps	4.5 Mbps	11 Mbps

La idea principal de esta configuración es la de contar con tráfico heterogéneo en cada *slice*. Siendo eMBB-2 la más exigente de ellas, seguido por eMBB-0 y luego eMBB-1.

Es importante recalcar que si se compara contra los trabajos originales, es posible observar una gran diferencia en los perfiles de tráfico con una cantidad de usuarios significativamente mayor y tasas de datos por usuario más pequeñas.

La decisión de diseño adoptada se fundamenta en el objetivo principal de la presente investigación, el cual es demostrar el potencial del aprendizaje por refuerzo en entornos de redes móviles mediante una prueba de concepto. Sin perjuicio de lo cual, dichas implementaciones pueden ser extendidas a redes con mayor cantidad de usuarios. Además, durante el desarrollo de la investigación, se enfrentó la limitación de no contar con la tecnología utilizada en los trabajos originales. Como consecuencia, se tuvo que adaptar a un escenario que pudiera ser manejado por el hardware disponible. Asimismo, cada usuario conectado puede representar bloques de usuario que tienen una demanda común.

Por otra parte, los SLA observados en la tabla surgen del desempeño de los usuarios en condiciones óptimas, destinando todos los recursos de la radiobase a cumplir con sus necesidades. Más adelante se presentarán gráficas que demuestran dichos valores.

Resulta pertinente señalar que, para agregar dinamismo al sistema, se hizo que las *slices* mencionadas se encuentren activas en distintos momentos durante la simulación. La *slice* eMBB-0 se puede observar en la Figura 6.1.

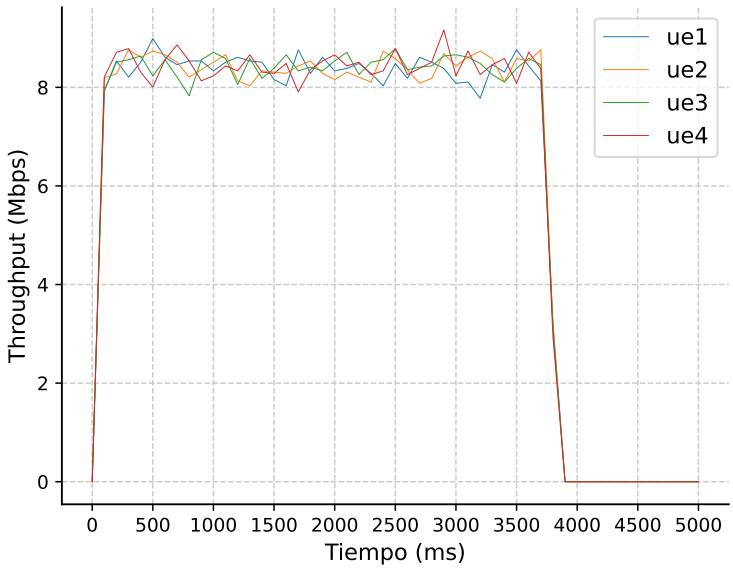


Figura 6.1: Tráfico original de la *slice* eMBB-0.

Esta *slice* eMBB-0, se caracteriza por presentar un tráfico altamente exigente. Se espera que cada uno de los seis usuarios asociados a esta *slice* alcance tasas de transferencia de datos mayores a 7.5 Mbps, aproximadamente.

Para poder simular el comportamiento de esta *slice* de manera efectiva y realista, se decidió activar el grupo de usuarios desde el inicio de la simulación y hasta el

75 % de la misma. Esta decisión se basó en la necesidad de obtener una visión completa de la interacción entre los usuarios y la red en distintos momentos de la simulación. De esta manera, se puede evaluar el comportamiento de la *slice* en distintas condiciones y ajustar el sistema en consecuencia para maximizar su rendimiento.

Por otro lado, en la Figura 6.2 se presenta el comportamiento óptimo de *throughput* del grupo de usuarios de la *slice* eMBB-1.

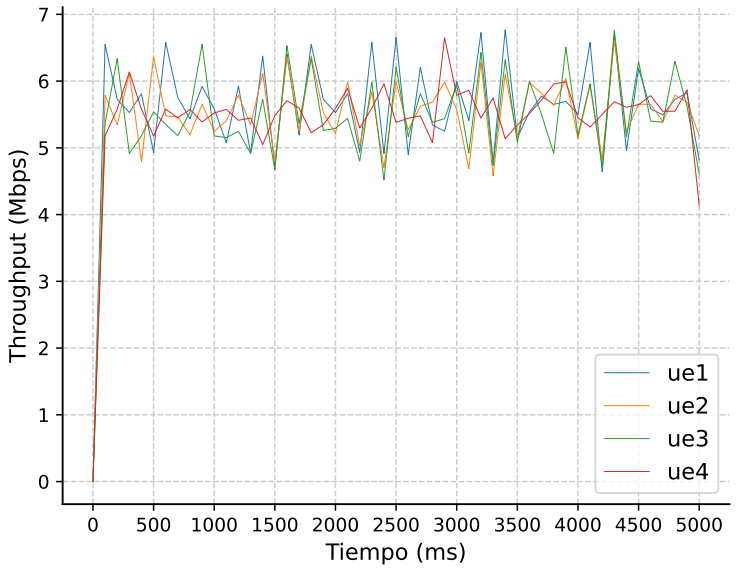


Figura 6.2: Tráfico original de la *slice* eMBB-1.

Durante el transcurso del desarrollo de la simulación, el tráfico de la presente *slice* se mantiene activa en todo momento. No obstante, es relevante destacar que esta *slice* en particular se caracteriza por presentar condiciones de tráfico sumamente variables, lo cual se atribuye a la distribución de Pareto que sigue el tamaño de sus paquetes. Incorporar este tipo de tráfico permite comprender el impacto de tráfico impredecible en el desempeño del sistema. A pesar de esta variabilidad se determina que en condiciones ideales el tráfico de la *slice* eMBB-1 puede superar los 4.5 Mbps.

Por último, en la Figura 6.3 se presenta el *throughput* en condiciones ideales para la *slice* eMBB-2.

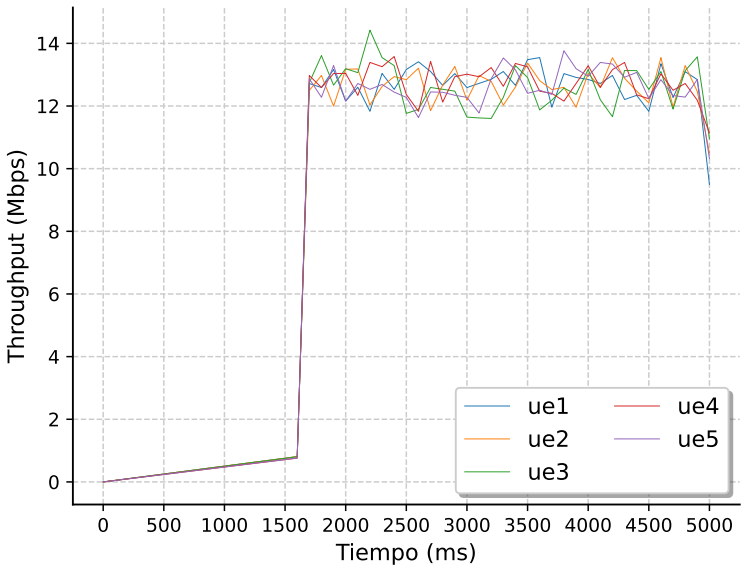


Figura 6.3: Tráfico original de la *slice* eMBB-2.

Se observa que el tráfico de sus usuarios está presente desde el 25 % del tiempo total de simulación y se extiende hasta el final de la misma.

Su tráfico, al igual que el anteriormente descrito, se caracteriza por contar con cierta variabilidad. Siendo que sus usuarios superan con facilidad un *throughput* de 11 Mbps.

En otro orden, en cuanto a la configuración general de la radiobase, se tuvo las consideraciones que se observan en la Tabla 6.2.

Tabla 6.2: Configuración de la radiobase.

Ancho de banda	10 MHz
Scheduler Intra-Slice	Round Robin
Tiempo entre asignaciones Inter-Slice	60 ms

La elección del tiempo entre asignaciones *Inter-Slice* se debe a que, al ser más breve, permite una mayor cantidad de asignaciones y, por tanto, una convergencia de la solución más eficiente. Por otro lado, el tiempo debe ser suficientemente grande como para garantizar que el *scheduler intra-slice* pueda realizar su asignación correctamente distribuyendo los recursos que le fueron otorgados.

La decisión de utilizar algoritmo el *intra-slice* Round Robin se fundamenta en la necesidad de disponer de un *scheduler* confiable y funcional, que, no obstante, sea

lo más simple posible. De este modo, se logra focalizar la complejidad del sistema en la parte *inter-slice*.

Por último, se optó por utilizar un ancho de banda de 10 MHz dado el bajo número de usuarios conectados a la red.

En lo que resta del capítulo se presentará la implementación de los dos algoritmos mencionados en Py5cheSim. Además, se realizará un análisis tanto sobre el diseño como de su desempeño.

6.2. Primer algoritmo

El primer algoritmo estudiado fue el propuesto en [35]. Allí, Rongpeng Li et al. presentan una implementación basada en aprendizaje por refuerzo para el *scheduling* en escenarios de *network slicing* brindando segmentación de recursos basada en prioridades. Por un lado, el trabajo en cuestión tiene una gran relevancia e impacto en el área. Pero además es sencillo en su construcción y sirve como base para futuros desarrollos. Estas, entre otras características la hicieron en un inicio la motivación de la presente investigación.

6.2.1. Formulación

El objetivo del algoritmo es resolver el desafío de asignación de recursos entre diferentes *slices* en redes 5G, mediante la implementación de un modelo de aprendizaje por refuerzo. La solución implica realizar una asignación óptima de recursos a cada *slice* con el fin de maximizar una función de recompensa (*reward*) R, que considera tanto la satisfacción de los usuarios como la utilización efectiva de la red.

De manera más específica, el algoritmo busca maximizar R, teniendo en cuenta factores como la demanda de servicio d_i para cada *slice* individual, la cantidad de recursos asignados r_i a cada una de ellas, la cantidad total de *slices* N y la cantidad de recursos disponibles en la celda R_T. La formulación original del problema se expresa de la siguiente manera:

$$
\begin{aligned}
&\arg_r\max \mathbb{E}\left[R(r,d)\right] \\
&s.t.: \\
&r = (r_1, r_2, ..., r_N) \\
&r_1 + r_2 + ... + r_N = R_T \\
&d = (d_1, d_2, ..., d_N)
\end{aligned}
$$

Esto implica que la asignación de recursos estará dada por un vector w, cuya i-esima componente corresponde con la cantidad de recursos asignados a la i-esima *slice*.

La Figura 6.4 muestra la dinámica de aprendizaje implementada. En la parte izquierda se representa el entorno en un tiempo t arbitrario, que proporciona información sobre el estado y la recompensa de la acción realizada en $t-1$ al

agente. Es importante destacar que, en el algoritmo presentado, el entorno captura las características de todas las *slices* en forma de una n-upla de tantos elementos como el número de *slices*. Es decir, el estado se define como $s = [s_1, s_2, .., s_N]$, donde cada s_i representa el estado de la i-ésima *slice*. Posteriormente, se calcula la recompensa como un escalar, en función del estado del entorno. La información del estado y la recompensa se utilizan para actualizar al agente de *Deep Reinforcement Learning* (DRL), el cual selecciona una acción a realizar (en tiempo t) entre las *slices*. Finalmente, el ciclo se completa un tiempo de asignación *inter-slice* después (en $t + 1$), cuando se calcula el nuevo estado y la recompensa (flecha inferior).

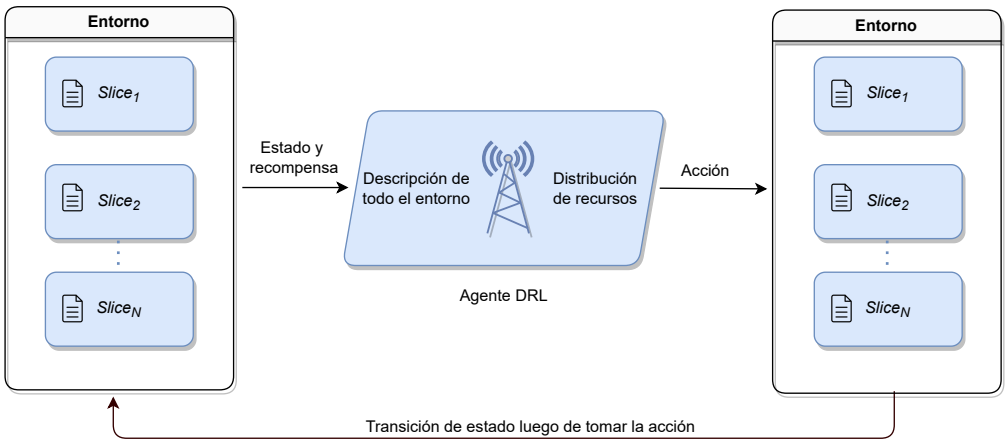


Figura 6.4: Diagrama de la dinámica del algoritmo uno.

6.2.2. Características del modelo

El trabajo original aborda la técnica de *slicing* tanto en la red de acceso como en el núcleo de la red. Sin embargo, en nuestra investigación, debido a las limitaciones del simulador Py5cheSim, nos hemos enfocado únicamente en el *scheduling* a nivel de radio. Por tanto, se han establecido las siguientes definiciones en función de la adaptación del trabajo original:

Estado

En la implementación actual, el estado se define como la cantidad de paquetes recibidos por cada *slice* durante una ventana de tiempo. Para ello, se ha implementado el estado en forma de vector de n elementos, donde cada componente corresponde al estado de la n-ésima *slice*. En el escenario expuesto -donde conviven tres *slices*-, el estado será de la forma $S = [S_{eMBB-0}, S_{eMBB-1}, S_{eMBB-2}]$. Se debe tener en cuenta que cada componente del estado dependerá del perfil de tráfico seleccionado; aquellos con mayores demandas generarán más paquetes y, por lo tanto, el valor de su componente en el vector estado será mayor.

En el trabajo original no se mencionan aspectos técnicos sobre el modelado de la solución. Por lo tanto, en la presente investigación fue necesario tomar

nuevas decisiones para adecuar el diseño. Entre ellas, tomar una granularidad de 10 paquetes a considerar en el vector estado, así como limitar la cantidad máxima de paquetes.

Acción

La acción se define como la cantidad de recursos alojados para cada *slice*. Esta asignación se dará en forma porcentual sobre la cantidad de recursos totales. Agregando a lo anterior, la asignación debe contar con una granularidad que permita el equilibrio entre precisión y tiempo de aprendizaje.

Como en el trabajo original no se detalla la configuración de las acciones, en el presente diseño se contempla una granularidad porcentual del 20 % de los recursos. Es decir, una *slice* puede ser asignada con 0, 20 %, 40 %, 60 %, 80 %, y 100 % de los recursos disponibles. Esta decisión, si bien es de corte general, permite tener una primera aproximación del funcionamiento de este algoritmo de aprendizaje. En términos prácticos, en la implementación el algoritmo devuelve un índice que corresponde a un elemento del conjunto de distribuciones total. Estas distribuciones son de la forma $[0, 20, 80]$, $[0, 40, 60]$, $[0, 60, 40]$, y así sucesivamente. Donde cada elemento del vector corresponde a eMBB-0, eMBB-1 y eMBB-2, respectivamente.

Recompensa

El propósito de la recompensa es doble: en primer lugar, evitar el desperdicio de recursos mediante la eliminación de la sobreasignación, y en segundo lugar, asignar una cantidad adecuada de recursos a cada *slice* para garantizar el cumplimiento del nivel de servicio acordado (SLA). Considerando lo anterior, se puede expresar la recompensa R como una función de varias variables. En particular, se utiliza la suma de la eficiencia de los recursos asignados de cada *slice* SE (*resource block usage ratio*), así como la cantidad de usuarios que cumplen con el nivel de experiencia acordado QoE. Además, se emplean dos constantes ψ y β, que ponderan el peso de SE y QoE, respectivamente.

$$R = \psi \cdot SE + \beta \cdot QoE \tag{6.1}$$

Es importante resaltar que la elección de los valores de ψ y β se realiza comúnmente de manera empírica, tomando en cuenta las necesidades particulares de la red. En el caso específico de esta implementación, se ha establecido un valor de 1 para ambas constantes.

En el estudio original, se propone una definición de eficiencia que se diferencia de la utilizada en esta investigación. El concepto de eficiencia espectral se empleó en dicho estudio, el cual se calcula como la cantidad de bits transmitidos por segundo por unidad de ancho de banda. Por otro lado, en la presente adaptación se tomó como medida de eficiencia la relación entre la cantidad de bits transmitidos y la cantidad de bits disponibles en el *transport block*.

Si bien ambas medidas persiguen el objetivo de evaluar la eficiencia de los recursos, se consideró que la metodología empleada en esta investigación resulta más directa y adecuada para los objetivos planteados en este estudio.

Entrenamiento

Para el entrenamiento del algoritmo presentado, se consideraron los perfiles de tráfico planteados en la sección 6.1. Durante el proceso de entrenamiento, se consideró el mismo escenario de evaluación; pero además, se realizaron múltiples variaciones en la cantidad de usuarios conectados, con el fin de abarcar una mayor cantidad de estados en el algoritmo.

La obtención de los hiper parámetros se llevó a cabo mediante la ejecución de varias instancias del algoritmo con diferentes configuraciones, las cuales se entrenaron en paralelo durante aproximadamente 8 horas.

La configuración para el algoritmo para el cual se obtuvo mejor rendimiento fue la siguiente:

- Tasa de aprendizaje (*Learning rate*) = 0.001
- Factor de descuento = 0.9
- Red neuronal de tres capas completamente conectadas. Las primeras dos de 32 neuronas, la tercera capa -correspondiente a la de salida-, tiene tantas neuronas como la cardinalidad del espacio de acciones.

6.2.3. Evaluación

Se llevó a cabo la simulación con el algoritmo implementado utilizando el escenario descrito en la sección anterior. En la Figura 6.5 se puede observar los resultados de la asignación de recursos.

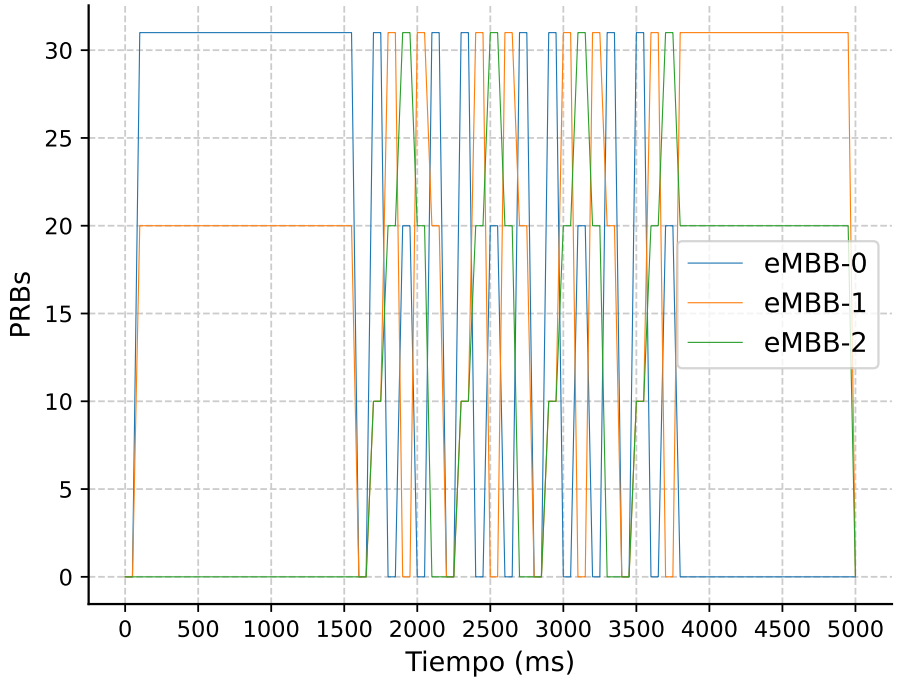


Figura 6.5: Asignación de recursos del algoritmo uno.

Se pueden diferenciar las tres etapas del escenario. Del instante 0 a 1500 ms aproximadamente, se le asignan recursos a emBB-0 y eMBB-1. Dandole al primero el 60 % de los recursos mientras que para el segundo, el 40 %.

Durante el periodo en el que los tres grupos intercambian datos (entre aproximadamente 1500 ms y 3700 ms), se produce un fenómeno oscilatorio producto del diseño del algoritmo. La asignación se realiza únicamente observando la cantidad de paquetes en *buffer* y esta última varía constantemente. Por ejemplo, es muy probable que en un instante de tiempo eMBB-0 tenga más paquetes para transmitir que eMBB-1. Como resultado, la distribución de los recursos fluctúa en esta ventana de tiempo.

Posteriormente, de 3700 ms en adelante, se distribuye un 60 % de los recursos a eMBB-1 y un 40 % a eMBB-2, dejando sin recursos a eMBB-0. Durante esta fase, se ha detectado un uso inadecuado de la asignación, a pesar de haber alcanzado el nivel de servicio establecido se esperaría una mayor asignación para eMBB-2 que para eMBB-1.

En la figura 6.5 se puede observar la recompensa obtenida durante la simulación de evaluación.

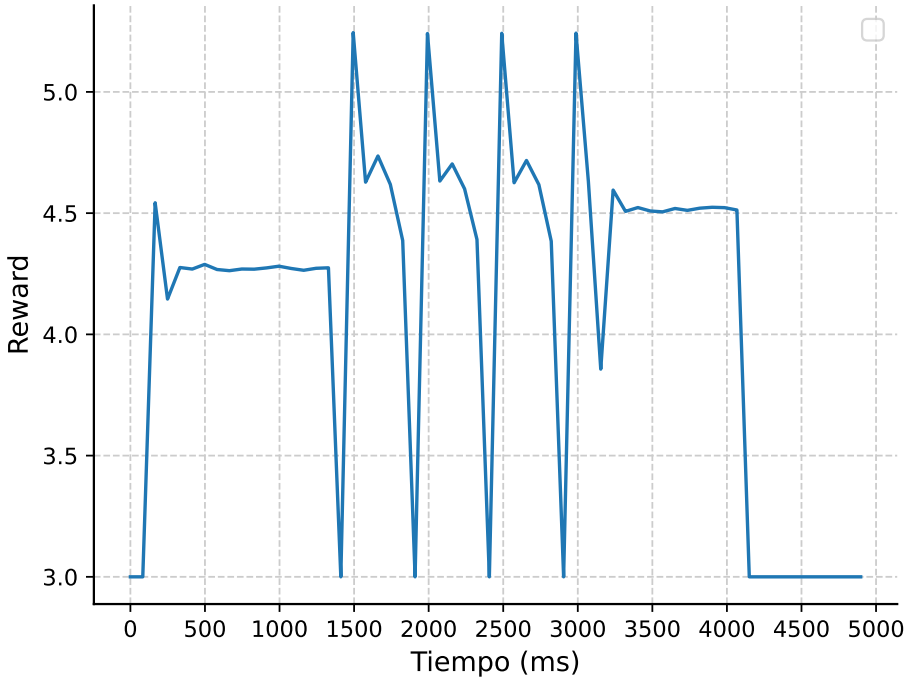


Figura 6.6: Recompensa de la simulación del algoritmo uno.

La recompensa oscila entre 3 y 5.5 unidades aproximadamente. Es relevante tener en cuenta que esta medida se determina mediante la suma del índice de satisfacción y la utilización de las tres *slices*. Aunque la primera *slice* se alcanza con facilidad debido al diseño de las acciones, la segunda *slice* plantea mayores obstáculos debido a que los recursos son frecuentemente sobreasignados.

A continuación se puede observar el *throughput* alcanzado para cada *slice*.

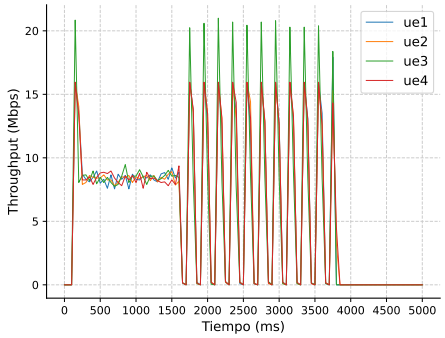


Figura 6.7: *Throguhput* en eMBB-0.

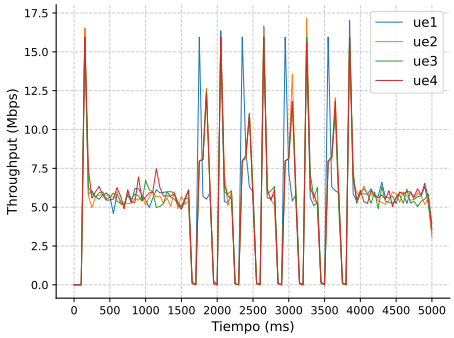


Figura 6.8: *Throguhput* en eMBB-1.

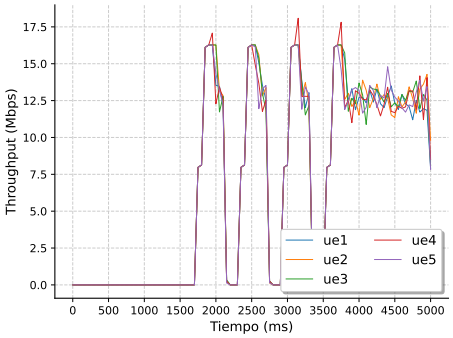


Figura 6.9: *Throguhput* en eMBB-2.

Se puede evidenciar que durante los períodos en los que solo se encuentran dos *slices* en actividad, el algoritmo es capaz de proveer los recursos necesarios para el funcionamiento. No obstante, al activarse los tres grupos, se observan fluctuaciones y picos de *throughput*, afectando negativamente el rendimiento del sistema.

6.3. Segundo algoritmo

¿Cuál sería el impacto de abordar la asignación para cada *slice* de manera independiente? ¿Es posible que un modelo aprenda de manera constructiva entre *slices*? Yu Abiko et al. en [10] presenta un algoritmo que toma en cuenta estas características con el fin de resolver el problema de la asignación de recursos.

El modelo en cuestión se basa en la técnica de aprendizaje distribuido Ape-X, que fue presentada por Horgan en [29]. Al abordar la asignación para cada fragmento de manera independiente y permitir que el modelo aprenda de manera constructiva entre fragmentos, el enfoque propuesto se presenta como una alternativa flexible, adaptable y robusta al enfoque tradicional. Además, la adopción de la técnica Ape-X puede reducir significativamente el tiempo necesario para llegar a una solución convergente.

Dinámica

En primer lugar, es importante mencionar que el algoritmo implementado es una simplificación del algoritmo de Ape-X. Pues en su versión original cada actor tiene diferentes políticas de exploración, mientras que en la presente implementación se siguió la misma para todos. La dinámica debe tener en cuenta su escenario de aplicación y, en esta implementación, cada *slice* se corresponde con un actor.

En la dinámica del algoritmo original, cada *slice* le informa su estado a la radiobase en tiempo t. En función de ella, la recompensa de la acción en t-1 es calculada y almacenada en la memoria de experiencia. Este procedimiento se repite para cada *slice* en el mismo tiempo t. Luego, en función del estado y del modelo realiza una asignación a cada *slice* en el tiempo t. La Figura 6.10 ilustra esta dinámica. Cada *slice* le transfiere el estado y recompensa de manera independiente a la radiobase y, de igual manera, la radiobase toma una acción diferente por *slice*.

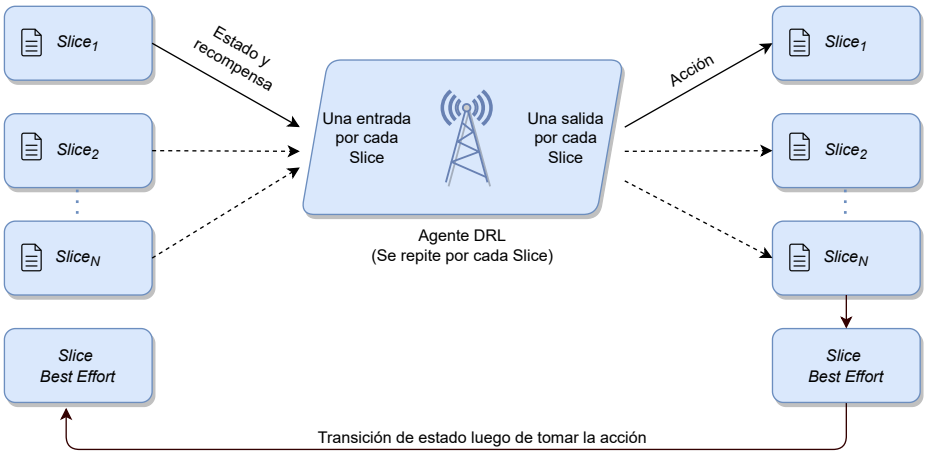


Figura 6.10: Diagrama de la dinámica del algoritmo dos.

6.3.1. Características del modelo

En la presente implementación, se tomó un conjunto de características que apuntan a un modelo que resuelve la asignación *slice* por *slice* de manera autónoma. También es pertinente mencionar que la implementación presentada es una simplificación y adaptación del trabajo original, a efectos de demostrar su capacidad de funcionamiento. A continuación, se introducen sus características y definiciones asociadas.

Estado

El estado contempla bastante más información que el primer algoritmo. En esta ocasión el estado está constituido por una tupla de ocho elementos que describe a la situación de una *slice* en particular. Siendo cada componente de la tupla:

- Su indice de satisfacción,
- su relación entre RB asignados y utilizados,
- la cantidad de RB asignados,
- requerimiento de *throughput*,
- requerimiento de *delay*,
- cantidad de UEs de la *slice* y
- el tráfico acumulado en el *buffer*.

Los primeros tres están directamente relacionados con la recompensa. El índice de satisfacción lo abreviaremos como NSRS (*network slice requierements satisfaction*) y lo implementamos como la cantidad de usuarios que superan los requerimientos de *throughput* sobre la cantidad de usuarios totales. A medida que los usuarios cumplen con sus requerimientos acordados, el indicador incrementará hasta el valor unitario. Por otro lado, a medida que los usuarios no alcanzan el nivel de servicio acordado, NSRS se aproximará a cero.

Por otro lado, abreviaremos la relación entre recursos asignados y utilizados como RBUR (por su descripción en inglés: *RB usage ratio*). RBUR se define como el cociente entre los recursos utilizados y los recursos disponibles. Este indicador evita el sobre dimensionamiento de recursos. A medida que este valor se acerca a uno, significa que la *slice* está utilizando efectivamente los recursos asignados. Por otro lado, si se asignan más recursos de los necesarios, este indicador se acercará a cero.

El tercer elemento que conforma el estado de la *slice* es la cantidad de recursos de bloques de radio (RBs) asignados en un instante de tiempo t. Si en dicho instante de tiempo, la *slice* cuenta con N_{RBs} RBs, este es el valor que se toma para esta componente del estado. Es importante destacar que la cantidad de RBs asignados es un factor crítico en la evaluación de la situación de la *slice* y tiene un significativo impacto en la acción a seleccionar.

Por otro lado, tanto el *throughput* como el *delay* son requerimientos que caracterizan a cada *slice*, permitiendo diferenciar una de otra. Es importante destacar que, en esta adaptación en particular, el requerimiento de *delay* no es relevante, pero es una característica que puede ser útil en futuras adaptaciones. Por lo tanto, el enfoque en este momento se centra en el *throughput*, como uno de los requerimientos clave para caracterizar a las *slices*.

Además de lo mencionado anteriormente, es importante considerar el nivel de exigencia de la *slice* al definir las dos últimas componentes del estado. Es esencial tener en cuenta que se espera un comportamiento diferente para *slices* con un bajo número de usuarios en comparación con aquellas con una gran cantidad de ellos. Además, el tráfico acumulado proporciona información valiosa sobre la situación de la *slice* en términos de la recompensa. Por lo tanto, al diseñar una estrategia de asignación de recursos, estas dos componentes del estado deben ser

cuidadosamente consideradas para garantizar un rendimiento óptimo de la *slice* en diferentes escenarios.

Incluir un estado que contemple tantos atributos puede ser beneficioso para capturar con mayor precisión la información del entorno. Sin embargo, a medida que se agregan más elementos, aumenta la complejidad del modelo y, por lo tanto, el costo del aprendizaje. Es importante destacar que el estado se adoptó del trabajo original [10], pero se adaptó de acuerdo con los criterios de esta investigación y las condiciones del simulador utilizado. Es fundamental considerar estos ajustes al momento de evaluar el modelo para garantizar que sus resultados sean consistentes con el trabajo original.

Acción

Dentro del modelo implementado, la acción se define como el aumento o la disminución de los recursos asignados a la *slice*. Este cambio se lleva a cabo mediante el elemento IDRB (*increase or decrease in the number of RB*) que se define de la siguiente manera:

$$IDRB = \lfloor -1^{act} \cdot 2^{\lfloor act/2 \rfloor - 1} \rfloor \tag{6.2}$$

Siendo *act* un valor entero, salida del modelo de aprendizaje. En nuestra versión implementada, limitamos la cantidad de acciones en ocho. Este fue un criterio de diseño que surge de la necesidad de tener cambios controlables en el sistema y no tener saltos significativos. A su vez, se puede observar de la Ecuación 6.2 que se toma el valor entero inferior de la expresión. Esto se debe a que los RB asignables son una cantidad entera.

Si llamamos ARB a la cantidad de RBs asignados en tiempo t, utilizando el concepto anterior queda formulado de la siguiente manera:

$$ARB_t = ARB_{t-1} + IDRB \tag{6.3}$$

El concepto de ARB no considera otras *slices*, sino que una sola a la vez. Esto tiene muchos beneficios en la convergencia y autonomía pero cuenta con algunas consideraciones a tener en cuenta. Pues no tendrá noción del estado general de la red y, por tanto, su asignación podrá afectar a las demás.

Con el fin de garantizar un equilibrio adecuado entre la asignación y la utilización, el trabajo [10] propone la inclusión de una *slice* auxiliar, denominada "Best Effort". Esta *slice* permite la convivencia de grupos de usuarios que no tienen acuerdos de servicio y, por lo tanto, dependen del estado de la red. Además, puede considerarse como un *buffer* de recursos disponibles en términos de RB.

Es deseable dar prioridad a los grupos de usuarios con requerimientos de servicio más bajos. Con este objetivo en mente, se propone un algoritmo que ordena las *slices* según su nivel de demanda, comenzando por asignar recursos a aquellas que tienen menor requerimiento. En caso de que los recursos de radio RB disponibles se agoten, se recortarán primero los grupos de usuarios que requieren mayor *throughput*. Este enfoque garantiza que las *slices* con menor demanda tengan prioridad, evitando que los recursos sean ocupados por las *slices* más exigentes.

Recompensa

Para la recompensa se basa en una noción similar a la presentada en el primer algoritmo. En este caso, se presenta de la siguiente manera:

$$R = NSRS \cdot RBUR \tag{6.4}$$

Los indicadores previamente definidos, NSRS y RBUR, son considerados en la presente noción de recompensa con el objetivo de lograr un equilibrio adecuado entre la cantidad de recursos asignados a *slices* y el cumplimiento de los requerimientos acordados con los usuarios. A diferencia del primer algoritmo, en esta instancia, no se añaden ambos elementos, sino que se toma su producto. Esta elección tiene un impacto directo en el proceso de aprendizaje, ya que influye en la evaluación de las decisiones tomadas. Al utilizar el producto de los indicadores, se busca maximizar ambos y, por lo tanto, no se requiere el uso de constantes para ponderar cada elemento.

Tanto NSRS como RBUR toman valores entre 0 y 1, lo que significa que R también toma valores en ese rango. Cuando R es 1, tanto NSRS como RBUR son 1, lo que satisface el requisito de la *slice* con la asignación mínima de recursos de radio (RB).

En el trabajo original se define una cantidad de RB máxima por *slice*. En la presente implementación se sorteó dicha hipótesis como criterio de diseño.

No obstante se tuvo nuevas consideraciones de casos limite para la recompensa. Estos se definen de la siguiente manera:

$$R = \begin{cases} 1 & \text{si tráfico} = 0 \text{ y ARB} = 0 \\ 0 & \text{si tráfico} > 0 \text{ y ARB} = 0 \\ 0 & \text{si tráfico} = 0 \text{ y ARB} > 0 \\ NSRS \cdot RBUR & \text{si tráfico} > 0 \text{ y ARB} > 0 \end{cases}$$

Siendo “tráfico” y ARB, la cantidad de paquetes en el buffer de la *slice* y la cantidad de RB asignados a ella en el instante previo, respectivamente.

Entrenamiento

Para el entrenamiento, al igual que en el primer algoritmo se tuvo en cuenta el escenario planteado en la Sección 6.1 con algunas modificaciones. Se entrenó el modelo con diferentes conjuntos de hiper parámetros hasta la obtención de un algoritmo que cumpla con los objetivos planteados.

Cada modelo se entrenó durante aproximadamente 12 horas, modificando momentos de activación de los usuarios y la cantidad de usuarios en las mismas. El modelo que mejor se adecuó, fue aquel con la siguiente configuración:

- Learning rate = 0.001
- Factor de descuento = 0.3

- Red neuronal de tres capas completamente conectadas. Las primeras dos de 64 neuronas, mientras que la tercera cuenta con tantas neuronas como la cardinalidad del espacio de acciones.

6.3.2. Evaluación

El escenario de evaluación fue común para ambos algoritmos. Se recuerda al lector que en la Sección 6.1 se advierte la existencia de las tres *slices* con diferentes requerimientos de servicio y tiempos de activación. No obstante, además de las consideradas, se adicionó por consideraciones de diseño la *slice* "BestEffort". Esta última se utilizó simplemente como un *buffer* de recursos, sin tener efecto alguno en las decisiones del algoritmo.

En la figura 6.11 se presenta el desempeño de este algoritmo de asignación.

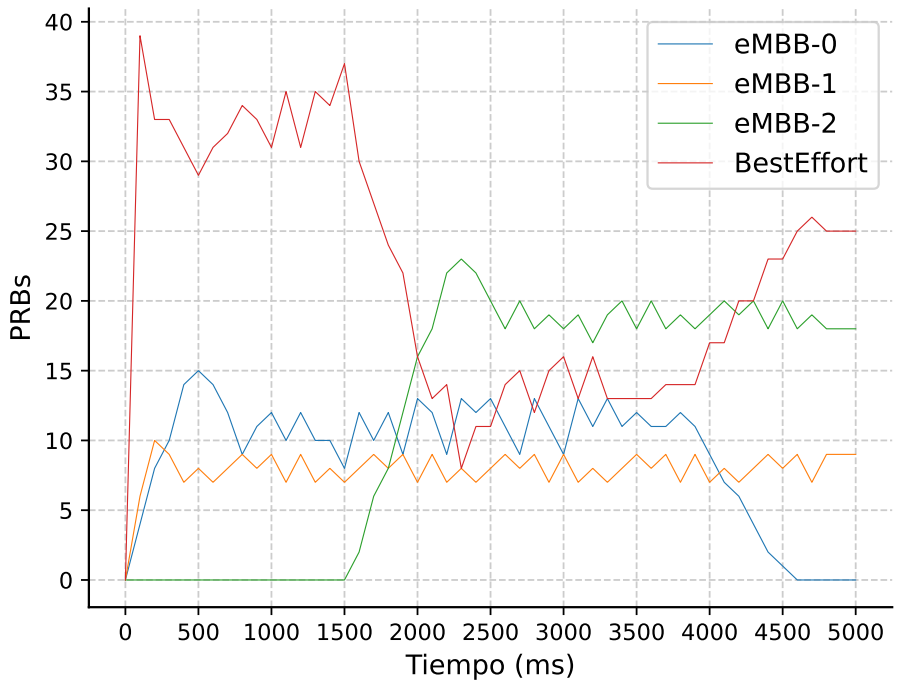


Figura 6.11: Recursos asignados por *slice* del algoritmo dos.

Al analizar la simulación, se puede constatar que la asignación de recursos se realiza de manera gradual y sin cambios significativos, lo cual es coherente con la definición de las acciones correspondientes. Esto se debe a que se toma en consideración el IDRB para llevar a cabo la asignación de recursos.

Asimismo, es pertinente resaltar que la simulación se ejecutó en una escala temporal de 5000 ms, lo que se considera una ventana de tiempo estrecha. En una simulación más extensa, se podría apreciar con mayor claridad una pendiente más pronunciada en las situaciones de activación o desactivación de las *slices*.

En cuanto al comportamiento cualitativo, se puede observar que se cumple con las expectativas previstas, según las cuales:

- La *slice* eMBB-0 recibe recursos desde el inicio de la simulación hasta el 75 % de la misma.
- La *slice* eMBB-1 recibe recursos durante todo el transcurso de la simulación.
- La *slice* eMBB-2 recibe recursos a partir del 25 % de la simulación y hasta su finalización.

Además, la asignación de recursos se ajusta adecuadamente a los requisitos de cada *slice*. En este sentido, se observa que la *slice* eMBB-2 cuenta con la mayor cantidad de recursos asignados, seguida por eMBB-0 y, finalmente, eMBB-1.

La recompensa para cada instante de tiempo puede observarse en la figura 6.12.

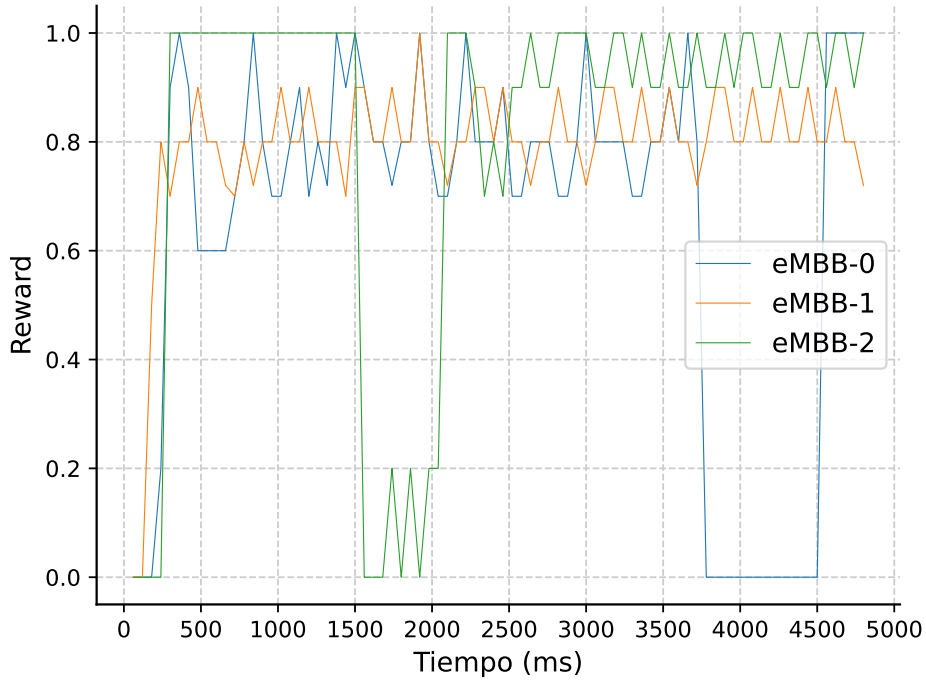


Figura 6.12: Recompensa en la simulación del segundo algoritmo.

En el caso inicial, se puede constatar que tanto eMBB-0 como eMBB-1 están en el período de transición, caracterizado por el inicio de la activación del tráfico. La *slice* eMBB-2 tiene recompensa constante de 1 entre 0 y 1500 ms aproximadamente. Esto se ajusta a la definición de casos límite que se presenta en la Sección 6.3.1. En ella, si la *slice* no cuenta con tráfico y no se le asignan recursos, tendrá recompensa 1, como se presenta en la gráfica. Luego de pasados los 1500 ms, la recompensa

inicia desde 0 seguido por un período de transición hasta llegar a su valor de régimen. Después de aproximidamente 3700 ms, eMBB-0 deja de tener tráfico para transmitir. No obstante, debido a la granularidad de las acciones, no es posible decrecer el número de RB directamente a cero, sino que debe hacerlo de manera gradual. Debido a ello, su recompensa es cero desde 3700 ms a 4500 ms, pues no hay datos para transmitir pero sí tiene recursos asignados. En régimen, eMBB-2 es la *slice* con mayor recompensa -valores entre 0,9 y 1- en tanto su tráfico es menos variable que eMBB-0 y eMBB-1 y, por consiguiente, resulta más predecible para el algoritmo. Por su parte, el grupo de usuarios perteneciente a eMBB-0 es el más impredecible, como se observó en la Sección 6.1. Es por tanto coherente que su recompensa sea la más baja de las tres. Pues no existe una única asignación de recursos para ella y, al ser tan variable, la utilización de recursos también lo será, deteriorando la recompensa. En el caso de este grupo oscila entre 0,7 y 1. Por su parte el grupo eMBB-1 tiene una recompensa que oscila entre 0,7 y 0,9.

Asimismo, el deterioro en la recompensa se debe principalmente al índice de utilización de los recursos RBUR. Como se mencionó anteriormente, el RBUR fue implementado como la cantidad de datos transmitidos, sobre la capacidad de datos asignados (tamaño del bloque de transporte). Se debe dar por tanto condiciones particulares para que los datos llenen el bloque de transporte y se tenga un RBUR igual a uno. Más adelante se presentará un escenario en el que esto sí sucede.

En la figura 6.13 se puede observar el *throughput* logrado a través del algoritmo implementado.

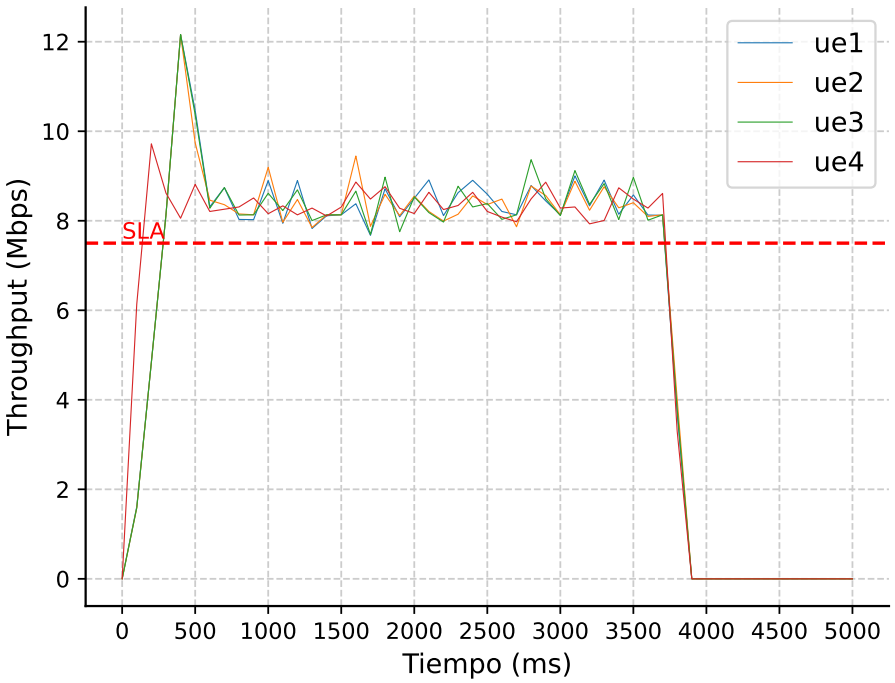


Figura 6.13: Throughput de la *slice* eMBB-0.

A partir de la gráfica presentada, se puede inferir que se ha alcanzado el nivel de servicio acordado de 7,5 Mbps. Cabe destacar que se observa un incremento abrupto en el *throughput* alrededor del instante de tiempo de 500 ms. Este comportamiento se debe a que los usuarios comienzan a generar paquetes en el buffer de transmisión desde los primeros milisegundos, y debido a que las acciones del algoritmo se llevan a cabo de manera gradual, la transmisión del buffer se produce en dicho instante de tiempo. Este evento también se puede observar en la Figura 6.11, donde se aprecia que en ese momento se asignan más recursos que en régimen normal, lo que explica el pico en el *throughput*.

Se muestra en la Figura 6.14 el *throughput* obtenido mediante la implementación del algoritmo.

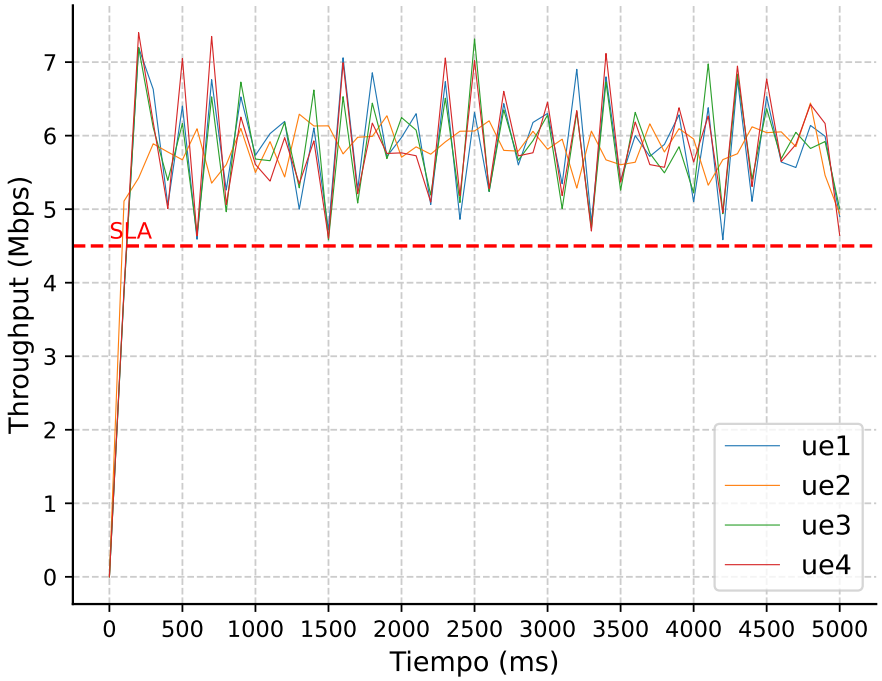


Figura 6.14: Throughput de la *slice* eMBB-1.

Es relevante destacar que en este escenario se estableció una restricción menos rigurosa en relación al nivel de servicio, debido principalmente a la naturaleza variable del tráfico. A pesar de ello, se puede observar que se logra superar el nivel de servicio establecido, aunque con un margen reducido.

La tercera *slice*, eMBB-2, utilizada en la simulación se puede observar en la figura 6.15.

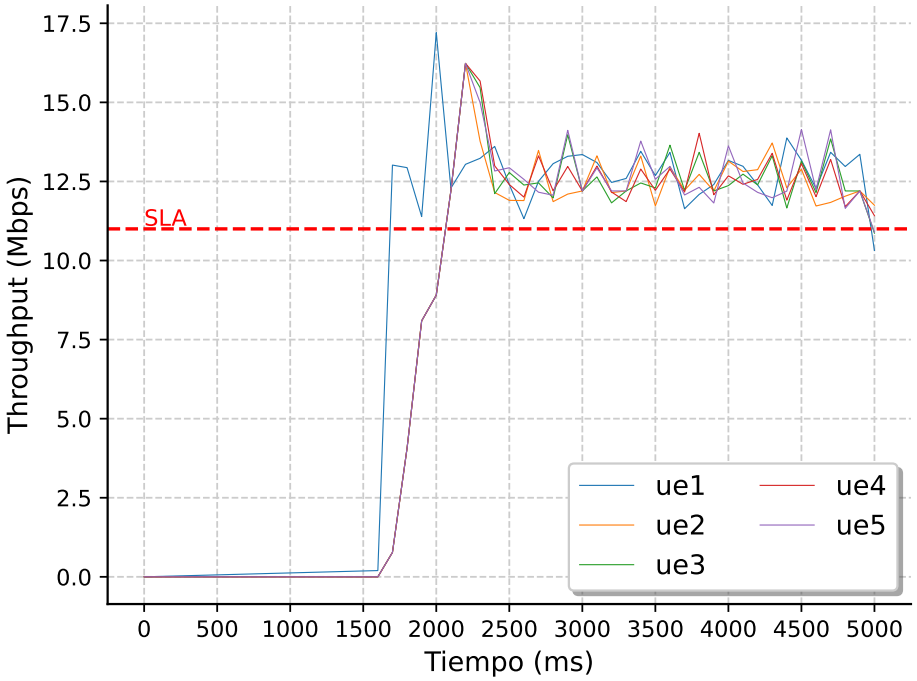


Figura 6.15: Throughput de la *slice* eMBB-2.

Al igual que en el grupo eMBB-0, se observa en la Figura 6.11 un pico en el comienzo del intercambio debido a la acumulación de paquetes en el instante previo. De la gráfica se puede observar que en esta *slice* se logra superar el nivel de servicio establecido.

6.3.3. Un nuevo escenario

Con el fin de demostrar el alcance de este algoritmo se diseñó un nuevo escenario de simulación. En la Tabla 6.3 se puede visualizar los nuevos perfiles de tráfico.

Tabla 6.3: Perfiles de tráfico para el nuevo escenario de evaluación.

	eMBB-0	eMBB-1	eMBB-2
Cantidad de usuarios	6	4	3
Tamaño de paquetes	Lognormal truncado [Media = 800 bytes, Máx =900 bytes]	Pareto [Media = 400 bytes]	Constante 200 bytes
Tiempo de arribo entre paquetes	Uniforme en el intervalo [0,1] ms	Constante 1 ms	Uniforme en el intervalo [0,1] ms
SLA: Tasa de datos	11 Mbps	3.8 Mbps	1.1 Mbps

La configuración adoptada tiene como idea principal la generación de un tráfico heterogéneo en cada *slice* que difiere del presentado en la Sección 6.1. En esta nueva configuración, el grupo de usuarios con mayor demanda se encuentra en la eMBB-0, seguido por eMBB-1, y en tercer lugar se encuentra eMBB-2, cuyos requerimientos son menos exigentes. Es importante destacar que, al igual que en el escenario anterior, la cantidad de usuarios se ha limitado significativamente para permitir la simulación con el equipamiento disponible.

En cuanto a los tiempos de activación de las *slices*, estos son similares al primer escenario. Los requerimientos de cada *slice* se obtuvieron a partir de su comportamiento en condiciones ideales.

La Figura 6.16 ilustra el comportamiento de la *slice* eMBB-0 en condiciones ideales.

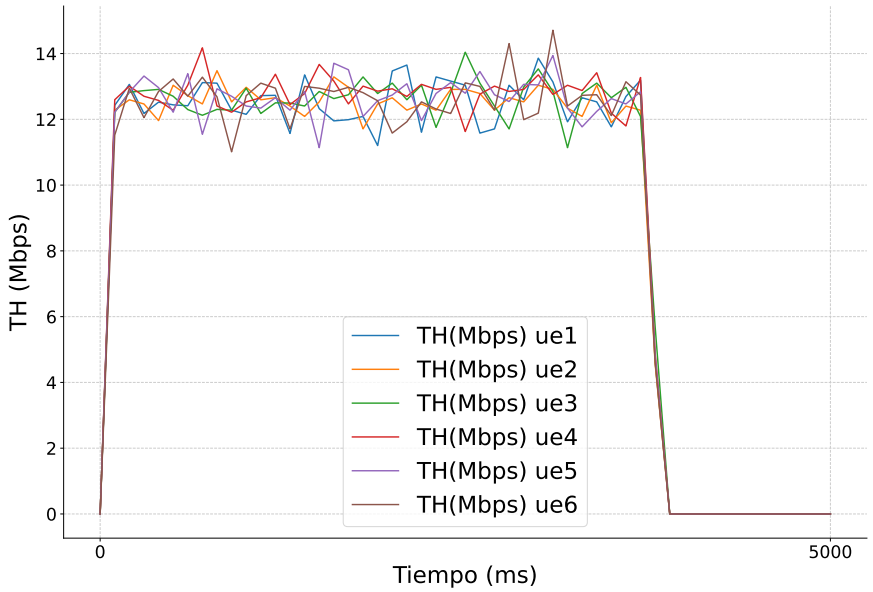


Figura 6.16: Tráfico original de la nueva *slice* eMBB-0.

La *slice* eMBB-0 se distingue por presentar un tráfico altamente demandante. Se prevé que cada uno de los seis usuarios asociados a esta *slice* alcance tasas de transferencia de datos superiores a los 11 Mbps, aproximadamente. Dicha *slice* estará activa desde el inicio de la simulación y hasta un 75 % de la misma.

Por otro lado, se muestra en la Figura 6.17 el comportamiento óptimo del *throughput* del grupo de usuarios pertenecientes a la *slice* eMBB-1.

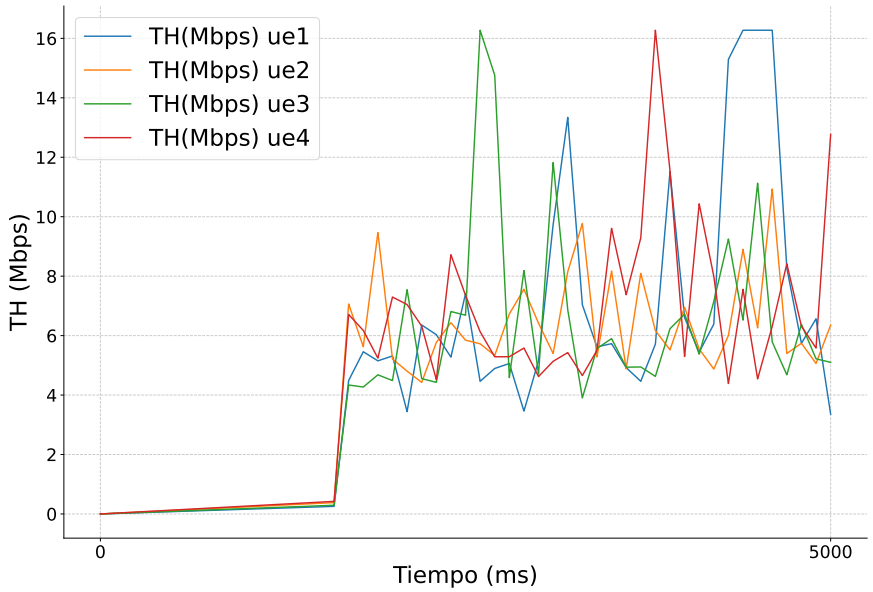


Figura 6.17: Tráfico original de la nueva *slice* eMBB-1.

La *slice* eMBB-1 se caracteriza por presentar un tráfico con un comportamiento altamente variable, lo cual dificulta su predicción. Cabe destacar que esta *slice* se encuentra activa durante el 25 % de la simulación y hasta su culminación. A pesar de esta variabilidad, se ha determinado que en condiciones ideales, el tráfico de la *slice* eMBB-1 puede superar los 3.8 Mbps.

Por último, en la Figura 6.18 se presenta el *throughput* en condiciones ideales para la *slice* eMBB-2.

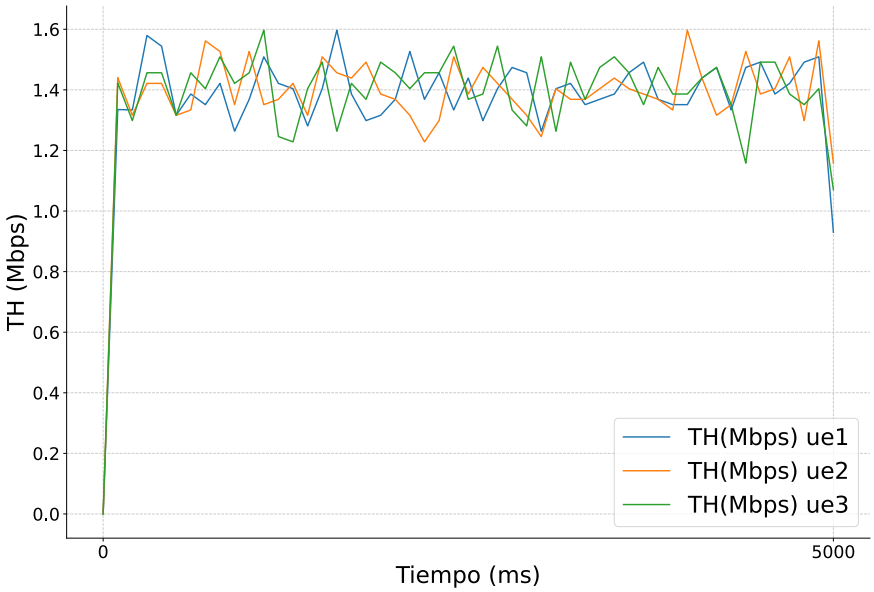


Figura 6.18: Tráfico original de la nueva *slice* eMBB-2.

Su tráfico se caracteríza por ser mucho menos exigente en términos de throughput. Siendo que sus usuarios superan con facilidad un *throughput* de 1.1 Mbps.

Dado el escenario descrito, el algoritmo implementado realiza la asignación de recursos como se observa en la Figura 6.19.

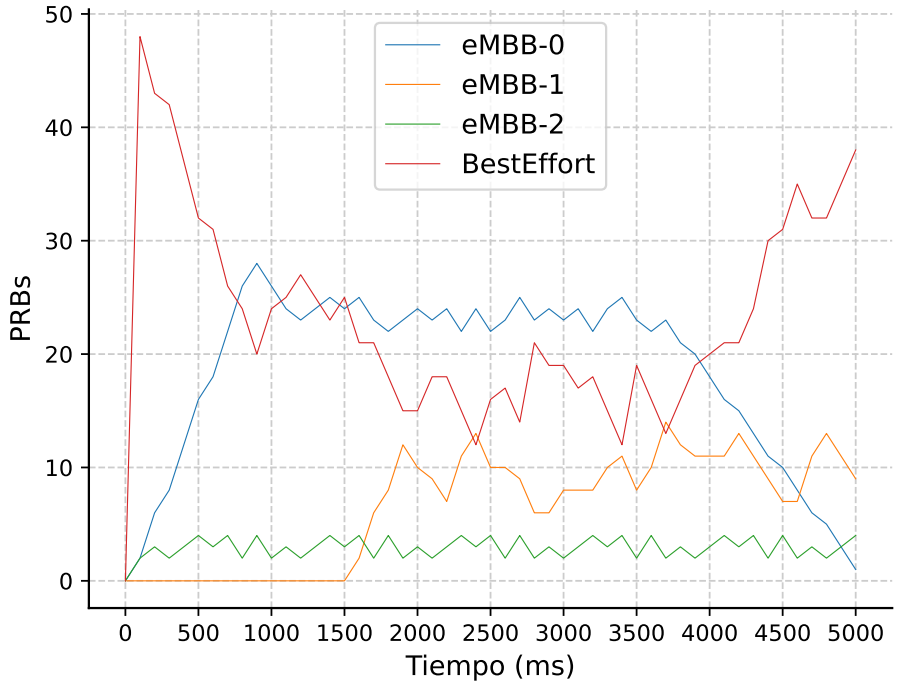


Figura 6.19: Recursos asignados por *slice*.

La figura en cuestión presenta una estructura similar a la planteada en el escenario previo. Se puede observar que la respuesta del algoritmo es coherente, ya que cada instancia de activación se corresponde con un aumento en la asignación de recursos.

En este caso, debido al modelado del tráfico, se puede notar una distinción significativa en cuanto a la asignación de recursos entre los diferentes grupos de usuarios. Además, a partir de la gráfica se puede apreciar nuevamente que la asignación de recursos se realiza de manera gradual.

En relación a la Figura 6.20, se muestra una gráfica que representa la evolución de la recompensa obtenida en cada momento durante la simulación de evaluación.

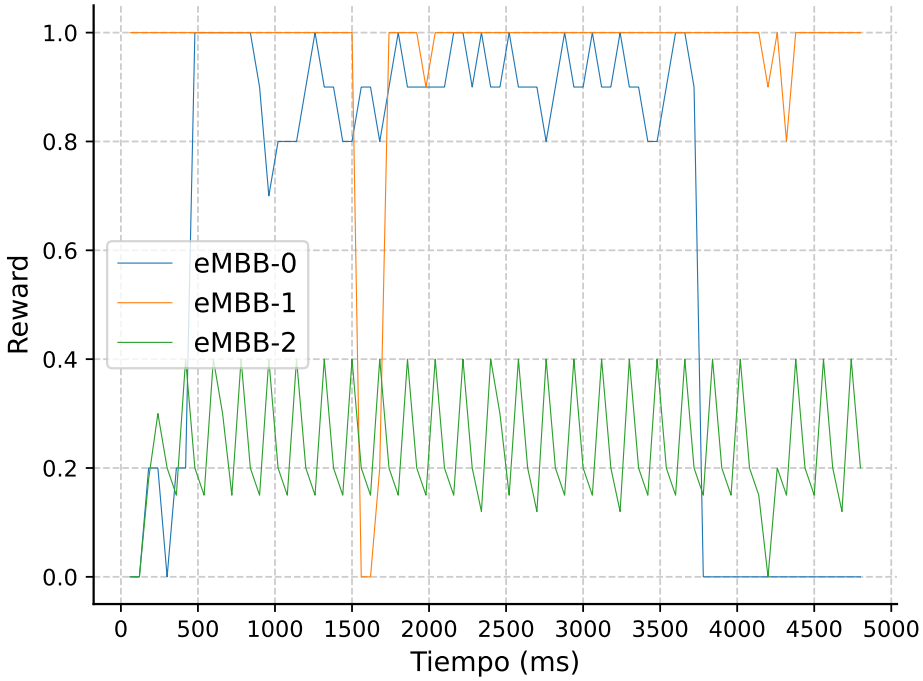


Figura 6.20: Recompensa de cada *slice* para cada instante de tiempo.

En este contexto, eMBB-0 logra obtener una recompensa que varía entre 0,8 y 1 durante la mayoría de su ejecución. A partir del 75 % de la simulación, los recursos destinados a eMBB-0 tienden a cero, pues la *slice* tiene RB asignados pero no tiene tráfico lo cual es acertado. En cambio, para eMBB-1 se observa una recompensa de 1 entre 0 a 1500 ms aproximadamente. Ajustándose a la definición de casos límite que se presenta en la Sección 6.3.1. Luego de pasados los 1500 ms, la recompensa inicia desde 0 en la transición, hasta obtener su valor en régimen.

En relación a eMBB-2, se observa una situación que difiere de la anterior, dado que durante todo el proceso de evaluación, esta instancia no consigue superar una recompensa de 0,4. Este desempeño se asocia fundamentalmente a la baja demanda de tráfico que caracteriza a este caso, lo cual genera dificultades para el algoritmo en el equilibrio entre el uso eficiente de los recursos y el cumplimiento del nivel de servicio requerido.

Además, al tratarse de una *slice* con un número reducido de usuarios, el incumplimiento de los requisitos de uno de ellos implica una disminución significativa en la recompensa obtenida.

Se ha llevado a cabo un análisis de este comportamiento. De forma cualitativa, se ha determinado que, en estados transitorios, si se asignan 3 RB a la *slice* eMBB-2, se obtiene un valor de NSRS igual a 1 y un RBUR igual a 0.4. Sin embargo, si se le asignan 2 RB, la *slice* tendrá un NSRS igual a 0.3 y un RBUR igual a 0.3. Estos valores de recompensa oscilan dependiendo del estado del sistema pero tienden

a seguir este comportamiento. En consecuencia, para el tráfico en cuestión, el algoritmo implementado no responde adecuadamente.

En este sentido, una posible alternativa para abordar este comportamiento podría ser formular una recompensa que priorice el nivel de servicio acordado, con el fin de lograr un equilibrio más adecuado en la asignación de recursos. Por otro lado, relajar la restricción del nivel de servicio no sería una opción adecuada, ya que si bien resultaría en mejores niveles de recompensa, llevaría a una asignación deficiente de recursos en la práctica.

A continuación se puede observar los elementos que componen a la expresión de la recompensa. En la figura 6.21 se puede observar el comportamiento del indicador NSRS. Mientras que en la figura 6.22 se puede observar el indicador de utilidad RBUR.

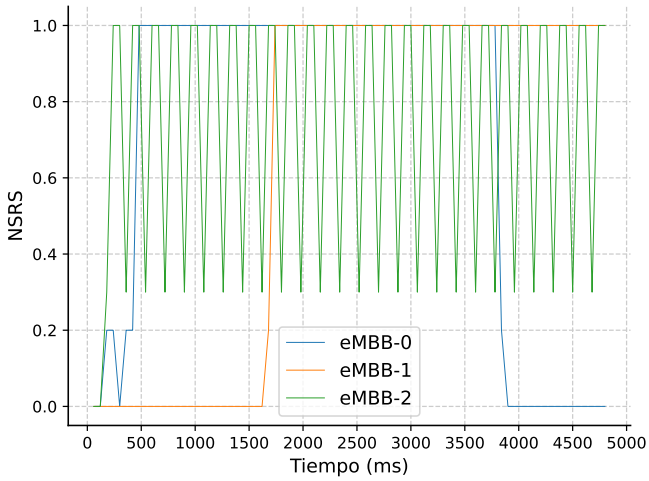


Figura 6.21: NSRS para las tres *slices*.

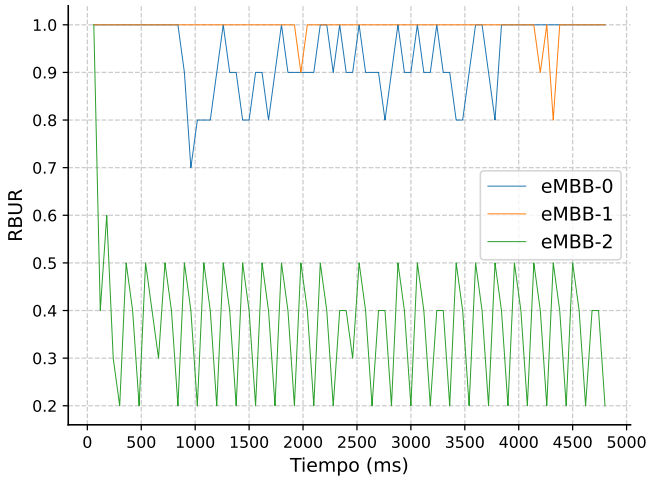


Figura 6.22: RBUR para las tres *slices*.

En la figura 6.22, se puede observar lo mencionado anteriormente acerca del comportamiento de la *slice* eMBB-2. Además, se puede apreciar que la *slice* eMBB-1 alcanza el valor 1 en RBUR, mientras que eMBB-0, cuyo tráfico es más impredecible, no lo hace.

6.4. Comparativa

Una vez ya introducidos y evaluados los algoritmos en cuestión, se pueden observar significativas diferencias entre ambos.

Estados

En primer lugar, resulta evidente que el primer algoritmo basado en la cantidad de paquetes recibidos por *slice* no es óptimo para un sistema tan complejo como las redes celulares. En lugar de ello, resulta conveniente contar con un estado que presente alguna correlación con la recompensa deseada. En este sentido, se podría asumir que el aumento en la cantidad de paquetes en buffer requiere de mayores recursos, lo que se traduce en una recompensa significativa. No obstante, este enfoque puede resultar ineficiente en régimen y, en cierta medida, presupone un comportamiento de tráfico constante. Además, la cantidad de paquetes en *buffer* puede oscilar en un rango muy amplio de valores, y en casos extremos, puede divergir si no se le asignaron los recursos adecuados en los pasados *slots*.

En línea con lo anterior, no existe una forma clara para acotar el conjunto de estados. Más aún, en el trabajo original lo intentan hacer al considerar la cantidad de paquetes recibidos considerando una ventana de tiempo. Esto presenta diversos problemas durante el entrenamiento pues el modelo aprende la asignación en base a esa ventana de tiempo, sin considerar la cantidad de paquetes almacenados esperando para enviar. Esto es particularmente grave en situaciones de *Q-Learning* donde para actualizar la política utilizo el algoritmo de ϵ-greedy en el período de aprendizaje.

Por otra parte, el segundo algoritmo, al considerar una cantidad sustancialmente mayor de características en su estado, permite desarrollar una solución de asignación más precisa. La definición del estado toma en cuenta aspectos relacionados a las restricciones de la *slice*, su estado, y nociones relacionados directamente con la recompensa. Este último explica una significativa mejora en la convergencia respecto al primer algoritmo. Además, el hecho de que considere las restricciones le permite al agente poder decidir directamente en base a ellos.

Sin embargo, la complejidad del aprendizaje también aumenta con la incorporación de más componentes. Aun así, esta complejidad se contrarresta por el propio diseño del algoritmo, que se enfoca únicamente en una *slice*, evitando la necesidad de considerar nuevas combinaciones, como ocurre en el primer algoritmo.

Acciones

En lo que respecta a las acciones, se puede observar que el primer algoritmo es muy limitado, ya que solo puede asignar recursos a partir de un conjunto dado. Por otro lado, el segundo algoritmo puede asignar cualquier cantidad posible a la *slice*. Además, el segundo algoritmo simplifica el aprendizaje al tener una menor cantidad de acciones posibles.

Recompensa

En relación a la definición de recompensa, se observan similitudes entre ambos algoritmos en lo que respecta a la consideración de un indicador de satisfacción del servicio y otro de utilización de la red. Cabe destacar que en ambas implementaciones se empleó una adaptación derivada del trabajo original del algoritmo dos.

En este contexto, se ha determinado que la eficacia de la recompensa del primer algoritmo es inferior a la del segundo.

Reflexión

El primer algoritmo ayuda a comprender la aplicación del aprendizaje por refuerzo en escenarios de redes móviles. No obstante, como resultado de nuestra implementación se determinó que tiene características que no la hacen una solución completa ni eficiente. En particular, el hecho de que las acciones sean demasiado generales afecta el desempeño de la red en varios aspectos, ya que no tiene en cuenta las pequeñas variaciones del tráfico. A su vez, el espacio de acciones y estados no es fácilmente modificable, dado que añadir un particionamiento más pequeño implica agregar todas las combinaciones posibles de la distribución de esos elementos. Esto enlentece notoriamente el tiempo de aprendizaje.

Por otro lado, el segundo algoritmo presenta un desempeño notablemente superior al primero y se adapta fácilmente a nuevos escenarios de tráfico. Gracias a su diseño, es posible ampliar su alcance a diferentes escenarios con un número variable de *slices*, lo que lo convierte en una solución flexible y versátil. Además, el entrenamiento a través de *slices* independientes acelera el proceso de aprendizaje y permite una captura más precisa de las características del entorno.

Es fundamental destacar que, a pesar de haberse utilizado versiones simplificadas de los trabajos originales, los resultados obtenidos reflejan de manera evidente la disparidad en la naturaleza de ambos algoritmos. Estos resultados indican una tendencia y sugieren una posible dirección a seguir. Es importante tener en cuenta estas observaciones al considerar futuros desarrollos en el área.

Capítulo 7

Conclusiones

La necesidad de esta investigación surge de las demandas del contexto actual en torno a las redes móviles de quinta generación. Como se mencionó en la introducción, constituye un desafío para la asignación de recursos dada la coexistencia de dispositivos con distintos requerimientos de servicio. Debido a su complejidad, es imperativo contar con técnicas de aprendizaje profundo para abordarlo. Pues, es necesario contar con mecanismos adaptables y eficientes para estos entornos dinámicos.

El presente trabajo final de maestría se desarrolló en base a la temática de la asignación de recursos en redes 5G mediante aprendizaje por refuerzo. Como principal objetivo se planteó estudiar dos importantes trabajos en los cuales se implementan algoritmos de *scheduling* basados en aprendizaje profundo, específicamente DRL.

Como herramienta principal de la presente investigación se utilizó un simulador de código abierto de redes 5G, llamado Py5cheSim y desarrollado en el Instituto de Ingeniería Eléctrica de UdelaR. Se le realizaron modificaciones en pos de obtener resultados más robustos y realistas, tales como la generación de distintos perfiles de tráfico y ajustes para la adaptación al estándar. Adicionalmente, se creó una biblioteca asociada para favorecer el desarrollo de *schedulers*. Estos aportes configuran el preludio de los resultados centrales.

En este simulador se evaluaron ambos algoritmos mencionados, a través de su implementación, en un mismo escenario de tráfico. Dicho escenario consistió en tres grupos de usuarios con distintos requerimientos de servicio y momentos de activación, con el propósito de emular un entorno dinámico. El objetivo principal de la evaluación de los algoritmos era garantizar el cumplimiento del nivel de servicio acordado, mediante una asignación eficiente de los recursos.

El primer algoritmo contempla tanto al estado como a las acciones de forma global. Este acercamiento demostró varias desventajas como algoritmo de asignación. En primer lugar, su baja granularidad de acciones y estados llevó a un comportamiento errático en la asignación de recursos. Además de una constante sobreasignación -debido a la naturaleza de su diseño-, no cumplió con las expectativas de los diferentes grupos de usuario. Sin embargo, este abordaje es la forma más simple de diseñar un algoritmo de aprendizaje y, como tal, da lugar a otro tipo de

implementaciones más complejas.

Por su parte, el segundo algoritmo toma estados y acciones individuales para cada *slice*. En particular, el estado comprende una mayor cantidad de características de la red y la acción tiene mayor granularidad. El análisis efectuado evidencia claramente que este algoritmo supera en rendimiento al primero. La definición precisa de los estados, acciones y recompensa permite contar con un modelo robusto y adaptable a diferentes situaciones. Es particularmente destacable su desempeño en entornos dinámicos, donde logra una utilización óptima de los recursos, cumpliendo con los requerimientos de servicio.

Asimismo, se evaluó el segundo algoritmo en un escenario separado, donde cumplió con creces las expectativas. No obstante, se identificaron casos en los que la respuesta del sistema no fue óptima. En particular, se observaron problemas en simulaciones con grupos de usuarios de bajos niveles de servicio y, en menor medida, en aquellos con patrones de utilización de la red muy variables. En cuanto al primer caso, se realizó un análisis y se justificó su comportamiento, que se debe a un problema en el balance entre nivel de servicio y utilización de recursos. En cuanto al segundo, el problema es natural para estos sistemas de asignación. Un algoritmo entrenado más exhaustivamente lo podría sortear.

Es preciso aclarar que este estudio se basa en pruebas de concepto -no reales sino simuladas-, con las correspondientes limitaciones de una herramienta de simulación, así como costos de tiempo. Al extrapolarlas a un escenario real se deberán tener ciertas consideraciones, tales como el período de aprendizaje -donde la respuesta aún no es eficaz y puede conllevar problemas- y la correcta configuración de los hiperparámetros del algoritmo de DRL.

Por último, es pertinente señalar que los resultados preliminares se publicaron en [46] y se presentaron en las conferencias en [14], [51], [32]. Tanto el simulador como los algoritmos implementados pueden ser encontrados en el repositorio de Github en [4].

7.1. Hacia adelante

El presente trabajo constituye un insumo más al gran esfuerzo por la incorporación de la tecnología de quinta generación. No obstante, esta tecnología no se ha establecido aún en Uruguay, lo cual torna significativos los aportes en la temática. Más aún, en el mundo se está discutiendo cómo serán las redes 6G, donde ya se puede vislumbrar la convergencia entre *Network Slicing* e IA. Los mismos algoritmos vistos en el presente documento -o muy similares- serán igualmente aplicables a esta nueva tecnología.

Como trabajo a futuro, se exhorta a continuar con el desarrollo del simulador Py5cheSim: adicionar nuevas características y lograr un funcionamiento más eficiente para grandes cantidades de dispositivos.

Por otra parte, se sugiere el entrenamiento de algoritmos basados en aprendizaje distribuido, en línea con el segundo algoritmo presentado, pero a partir de escenarios más complejos y diversos, restricciones de *delay*, mediante diferentes arquitecturas y tiempos de aprendizaje más extensos.

Finalmente, en vista de las limitaciones señaladas, propias de un entorno de simulación, se promueve continuar el camino con investigación aplicada.

Esta página ha sido intencionalmente dejada en blanco.

Referencias

[1] 5g toolbox from matlab. https://la.mathworks.com/products/5g.html. Accessed: 2023-01-04.

[2] Function approximation in reinforcement learning (and deep reinforcement learning). https://storage.googleapis.com/deepmind-media/UCL%20x%20DeepMind%202021/Lecture%207-%20Function%20approximation%20in%20reinforcement%20learning%20.pdf. Accessed: 2023-01-26.

[3] pydoctor. https://github.com/twisted/pydoctor. Accessed: 2023-01-10.

[4] Repositorio de github. https://github.com/linglesloggia/py5chesim. Accessed: 2023-01-04.

[5] srsran. https://www.srslte.com/. Accessed: 2023-01-08.

[6] ursec. https://www.gub.uy/unidad-reguladora-servicios-comunicaciones/. Accessed: 2023-01-08.

[7] 3GPP. NR; Radio Resource Control (RRC); Protocol specification. Technical Specification (TS) 38.331, 3rd Generation Partnership Project (3GPP), 10 2022. Version 17.3.0.

[8] 3GPP. NR; Physical channels and modulation. Technical Specification (TS) 38.211, 3rd Generation Partnership Project (3GPP), 09 2022. Version 17.3.0.

[9] 3GPP. NR; Physical layer procedures for data. Technical Specification (TS) 38.214, 3rd Generation Partnership Project (3GPP), 09 2022. Version 17.3.0.

[10] Yu Abiko, Takato Saito, Daizo Ikeda, Ken Ohta, Tadanori Mizuno, and Hiroshi Mineno. Radio resource allocation method for network slicing using deep reinforcement learning. In *2020 International Conference on Information Networking (ICOIN)*, pages 420–425. IEEE, 2020.

[11] Adedotun T. Ajibare and Olabisi E. Falowo. Resource allocation and admission control strategy for 5g networks using slices and users priorities. In *2019 IEEE AFRICON*, pages 1–6, 2019.

[12] Anna Brunstrom. Experimentation and measurement methodologies for next generation mobile networks. URL: https://www.dropbox.com/s/o3qua2hemsztpar/slide-anna.pdf?dl=1. Accessed: 2023-01-08.

[13] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[14] AUGM. Jornadas de jóvenes investigadores. `http://grupomontevideo.org/sitio/jornada-de-jovenes-investigadores/`, 2022. Accessed: 2023-02-02.

[15] Pablo Belzarena Claudina Rattaro, Lucas Inglés. Herramientas de simulación de redes móviles de futura generación. `https://iie.fing.edu.uy/investigacion/grupos/artes/es/proyectos/herramientas-de-simulacion-de-redes-moviles-de-futura-generacion/`. Accessed: 2023-01-08.

[16] UNIDAD REGULADORA DE SERVICIOS DE COMUNICACIONES. Decreto 113. `https://www.gub.uy/unidad-reguladora-servicios-comunicaciones/institucional/normativa/resolucion-n-113022-autorizacion-pruebas-5g-antel`, 2022. Accessed: 2023-01-08.

[17] UNIDAD REGULADORA DE SERVICIOS DE COMUNICACIONES. Decreto 114. https://www.gub.uy/unidad-reguladora-servicios-comunicaciones/institucional/normativa/resolucion-n-114022-autorizacion-pruebas-5g-am-wireless-uruguay-sa, 2022. Accessed: 2023-01-08.

[18] UNIDAD REGULADORA DE SERVICIOS DE COMUNICACIONES. Decreto 115. https://www.gub.uy/unidad-reguladora-servicios-comunicaciones/institucional/normativa/resolucion-n-115022-autorizacion-pruebas-5g-telefonica-moviles-del-uruguay, 2022. Accessed: 2023-01-08.

[19] Erik Dahlman, Stefan Parkvall, and Johan Skold. *5G NR: The next generation wireless access technology*. Academic Press, 2020.

[20] Enzo Davyt, Alexis Muzante, and Martín Rizzo. A5ignator: Framework para la implementación de algoritmos de asignación de recursos en 5g. 2021.

[21] Deniz, C., Uyan, O. G., and & Gungor. On the performance of lte downlink scheduling algorithms: A case study on edge throughput. *Computer Standards & Interfaces*, 59:96–108, 2018.

[22] Paula Varela Diego Sanchez, Mateo Trujillo. Py5chesim v2.0: Extension de funcionalidades de un simulador de redes 5g para ensayo de asignacion de recursos. `https://github.com/charluy/PFC`, 2022.

[23] Tomás Domínguez-Bolaño, José Rodríguez-Piñeiro, José A García-Naya, and Luis Castedo. The gtec 5g link-level simulator. In *2016 1st International Workshop on Link-and System Level Simulations (IWSLS)*, pages 1–6. IEEE, 2016.

[24] Allen B Downey. Lognormal and pareto distributions in the internet. *Computer Communications*, 28(7):790–801, 2005.

[25] Pablo Belzarena et al. Inteligencia artificial aplicada a redes 5g. https://iie.fing.edu.uy/investigacion/grupos/artes/es/proyectos/inteligencia-artificial-aplicada-a-redes-5g/. Accessed: 2023-01-08.

[26] Jianqing Fan, Zhaoran Wang, Yuchen Xie, and Zhuoran Yang. A theoretical analysis of deep q-learning. In *Learning for Dynamics and Control*, pages 486–489. PMLR, 2020.

[27] Panagiotis K Gkonis, Panagiotis T Trakadas, and Dimitra I Kaklamani. A comprehensive study on simulation techniques for 5g networks: State of the art results, analysis, and future challenges. *Electronics*, 9(3):468, 2020.

[28] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[29] Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado Van Hasselt, and David Silver. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*, 2018.

[30] Chin-Kuo Jao, Chun-Yen Wang, Ting-Yu Yeh, Chun-Chia Tsai, Li-Chung Lo, Jen-Hsien Chen, Wei-Chen Pao, and Wern-Ho Sheen. Wise: A system-level simulator for 5g mobile networks. *IEEE Wireless Communications*, 25(2):4–7, 2018.

[31] Shihao Ju, Ojas Kanhere, Yunchou Xing, and Theodore S Rappaport. A millimeter-wave channel simulator nyusim with spatial consistency and human blockage. In *2019 IEEE global communications conference (GLOBECOM)*, pages 1–6. IEEE, 2019.

[32] khipu. Latin american meeting in artificial intelligence. https://khipu.ai/, 2022. Accessed: 2023-02-02.

[33] Praveenkumar Korrai, Eva Lagunas, Shree Krishna Sharma, Symeon Chatzinotas, Ashok Bandi, and Björn Ottersten. A ran resource slicing mechanism for multiplexing of embb and urllc services in ofdma based 5g wireless networks. *IEEE Access*, 8:45674–45688, 2020.

[34] Ying Loong Lee and Donghong Qin. A survey on applications of deep reinforcement learning in resource management for 5g heterogeneous networks. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1856–1862. IEEE, 2019.

[35] Rongpeng Li, Zhifeng Zhao, Qi Sun, I Chih-Lin, Chenyang Yang, Xianfu Chen, Minjian Zhao, and Honggang Zhang. Deep reinforcement learning for resource management in network slicing. *IEEE Access*, 6:74429–74441, 2018.

[36] Nguyen Cong Luong, Dinh Thai Hoang, Shimin Gong, Dusit Niyato, Ping Wang, Ying-Chang Liang, and Dong In Kim. Applications of deep reinforcement learning in communications and networking: A survey. *IEEE Communications Surveys & Tutorials*, 21(4):3133–3174, 2019.

[37] Federico Mason, Gianfranco Nencioni, and Andrea Zanella. A multi-agent reinforcement learning architecture for network slicing orchestration. In *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*, pages 1–8. IEEE, 2021.

[38] Energía y minería Ministerio de industria. Decreto. `https://medios.presidencia.gub.uy/legal/2022/decretos/12/miem_347.pdf`, 2022. Accessed: 2023-01-08.

[39] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[40] Werner Mohr. The 5g infrastructure association. *ITU-R.[Online]. Available: https://www. itu. int/en/ITU-R/study-groups/rsg5/rwp5d/imt-2020/Documents/S03-1 5GPPP [Accessed: Mar. 29, 2019]*, 2017.

[41] Masahiko Nabe, Masayuki Murata, and Hideo Miyahara. Analysis and modeling of world wide web traffic for capacity dimensioning of internet access lines. *Performance evaluation*, 34(4):249–271, 1998.

[42] Navid Nikaein, Mahesh K Marina, Saravana Manickam, Alex Dawson, Raymond Knopp, and Christian Bonnet. Openairinterface: A flexible platform for 5g research. *ACM SIGCOMM Computer Communication Review*, 44(5):33–38, 2014.

[43] EL PAIS. ¿en qué situación está el 5g en uruguay? `https://www.elpais.com.uy/vida-actual/en-que-situacion-esta-el-5g-en-uruguay`. Accessed: 2023-01-08.

[44] Natale Patriciello, Sandra Lagen, Biljana Bojovic, and Lorenza Giupponi. An e2e simulator for 5g nr networks. *Simulation Modelling Practice and Theory*, 96:101933, 2019.

[45] Gabriela Pereyra. Scheduling in 5g networks: Developing a 5g cell capacity simulator. 2021.

[46] Gabriela Pereyra, Lucas Inglés, Claudina Rattaro, and Pablo Belzarena. An open source multi-slice cell capacity framework. *CLEI Electronic Journal, vol. 25, no 2, may. 2022, pp. 1-21.*, 2022.

[47] Stefan Pratschner, Bashar Tahir, Ljiljana Marijanovic, Mariam Mussbah, Kiril Kirev, Ronald Nissel, Stefan Schwarz, and Markus Rupp. Versatile mobile communications simulation: The vienna 5g link level simulator. *EURASIP Journal on Wireless Communications and Networking*, 2018(1):1–17, 2018.

[48] Yichen Qian, Jun Wu, Rui Wang, Fusheng Zhu, and Wei Zhang. Survey on reinforcement learning applications in communication networks. *Journal of Communications and Information Networks*, 4(2):30–39, 2019.

[49] Vincenzo Sciancalepore, Xavier Costa-Perez, and Albert Banchs. Rl-nsb: Reinforcement learning-based 5g network slice broker. *IEEE/ACM Transactions on Networking*, 27(4):1543–1557, 2019.

[50] M Series. Imt vision–framework and overall objectives of the future development of imt for 2020 and beyond. *Recommendation ITU*, 2083:0, 2015.

[51] SIGCOMM. Tma phd school. `https://tma.ifip.org/2022/`, 2022. Accessed: 2023-02-02.

[52] Richard S Sutton and Andrew G Barto. The reinforcement learning problem. *Reinforcement learning: An introduction*, pages 1–26, 1998.

[53] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[54] Marcos Toscano, Federico Grunwald, Matías Richart, Javier Baliosian, Eduardo Grampín, and Alberto Castro. Machine learning aided network slicing. In *2019 21st International Conference on Transparent Optical Networks (ICTON)*, pages 1–4. IEEE, 2019.

[55] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

[56] Bin Xiang, Jocelyne Elias, Fabio Martignon, and Elisabetta Di Nitto. Joint network slicing and mobile edge computing in 5g networks. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–7, 2019.

[57] Xu Yang, Yapeng Wang, Ieok Cheng Wong, Yue Liu, and Laurie Cuthbert. Genetic algorithm in resource allocation of ran slicing with qos isolation and fairness. In *2020 IEEE Latin-American Conference on Communications (LATINCOM)*, pages 1–6, 2020.

Esta página ha sido intencionalmente dejada en blanco.

Índice de tablas

Esta página ha sido intencionalmente dejada en blanco.

Índice de figuras

Índice de figuras

Tablas

Tabla 7.1: Tabla 5.1.3.2-1 extraída de [9].

Index	**TBS**	**Index**	**TBS**	**Index**	**TBS**	**Index**	**TBS**
1	24	31	336	61	1288	91	3624
2	32	32	352	62	1320	92	3752
3	40	33	368	63	1352	93	3824
4	48	34	384	64	1416		
5	56	35	408	65	1480		
6	64	36	432	66	1544		
7	72	37	456	67	1608		
8	80	38	480	68	1672		
9	88	39	504	69	1736		
10	96	40	528	70	1800		
11	104	41	552	71	1864		
12	112	42	576	72	1928		
13	120	43	608	73	2024		
14	128	44	640	74	2088		
15	136	45	672	75	2152		
16	144	46	704	76	2216		
17	152	47	736	77	2280		
18	160	48	768	78	2408		
19	168	49	808	79	2472		
20	176	50	848	80	2536		
21	184	51	888	81	2600		
22	192	52	928	82	2664		
23	208	53	984	83	2728		
24	224	54	1032	84	2792		
25	240	55	1064	85	2856		
26	256	56	1128	86	2976		
27	272	57	1160	87	3104		
28	288	58	1192	88	3240		
29	304	59	1224	89	3368		
30	320	60	1256	90	3496		

Tabla 7.2: Tabla 5.1.2.2.1-1 de [9]

Bandwidth Part Size	Configuration 1	Configuration 2
1 - 36	2	4
37 72	4	8
73 144	8	16
145 275	16	16

Esta es la última página.
Compilado el martes 28 marzo, 2023.
http://iie.fing.edu.uy/