TRABAJO FIN DE GRADO

INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

Algoritmos aplicados a la reserva de recursos para Network Slicing en Redes 5G

Autor

Manuel Jesús Justicia Alados

Directores

Pablo Muñoz Luengo

Óscar Adamuz Hinojosa

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, septiembre de 2020

Algoritmos aplicados a la reserva de recursos para Network Slicing en Redes 5G

Autor
Manuel Jesús Justicia Alados

Directores
Pablo Muñoz Luengo
Óscar Adamuz Hinojosa

Algoritmos aplicados a la reserva de recursos para network Slicing en redes 5G

Manuel Jesús Justicia Alados

Palabras clave: 5G, network slicing, red de acceso radio (RAN), cloud RAN (C-RAN), reserva de recursos.

Resumen

Las redes móviles han experimentado en los últimos años una evolución exponencial en cuanto a diversidad de contenido, usos y demandas. El siguiente paso dentro de esta evolución es la implantación de la red 5G en todo el mundo.

Las redes 5G pretenden actualizar la infraestructura de la red móvil en vista de la cantidad de contenidos y tipos de tráfico que pretenden ser introducidos y agregados en los próximos años. La gran cantidad de dispositivos que se conectan a diario a la red crece a ritmo vertiginoso y la red necesita adaptarse a ello. Surgen así los objetivos de la red de quinta generación: latencia extremadamente baja, un aumento de la eficiencia energética en la red, gran ancho de banda en la red, la integración de los vehículos en las comunicaciones, comunicaciones masivas entre maquinas y la posibilidad de tener una gran cantidad de sensores conectados a la red entre muchos otros mediante el paso del mundo físico a las funciones virtualizadas de la red entre otras.

De entre esas otras funciones que introduce 5G para conseguir esa adaptación de la infraestructura a los nuevos servicios que se quieren implementar, el network slicing busca aportar a la red la capacidad de definir dentro de una misma infraestructura física una gran diversidad de redes lógicas que atienden a la gran diversidad de los tipos de servicios a los que podrían conectarse todos los usuarios (e.j., personas, vehículos, máquinas, sensores, etc…) de la red.

El network slicing copa una gran cantidad de investigaciones pues se considera de una necesidad imperiosa lo descrito con anterioridad: satisfacer dentro de la misma red todos los servicios sea cual sea su procedencia y naturaleza, con requisitos muy diversos entre ellos.

El otro concepto que absorbe gran parte de las investigaciones actuales es el de cloud RAN (C-RAN). En la quinta generación se redefine lo que son las funciones de la red como hasta ahora las conocemos. De tener en 4G el eNB (es decir, el nodo de acceso a la red 4G) englobando las funciones de recepción (transmisión) de la señal radio, así como las funciones de procesamiento de esta señal, hasta la extracción (inserción) de los datos que retransmitirán a través del núcleo de la red (interfaz radio) , el C-RAN propone una separación de esas funciones, aportando para ello un sistema compuesto por unidad centralizada y unidad distribuida, las cuales se verán identificadas mediante estaciones base y centros de cómputo respectivamente, separando las funciones y distribuyéndolas entre las dos unidades. Tratando así de mover del mundo físico al virtual todas aquellas funciones que sea posible implementar mediante software siempre y cuando el procesamiento de dichas funciones pueda hacerse dentro de una frontera temporal determinada.

Las diversas naturalezas de contenido, la crecida en la demanda de la red, la eficiencia energética que se quiere conseguir en la red, la baja latencia, todo esto, hace que se necesiten algoritmos que puedan dimensionar el número de elementos destinados al procesamiento en los centros destinados a tal efecto, para que los distintos slices de la red puedan verse satisfechos en todos sus indicadores de QoS y QoE.

Este trabajo pretende sumar a las investigaciones ya realizadas, el uso de una serie de algoritmos

con sus implementaciones para determinar los recursos de cómputo necesarios en las unidades centralizadas encargadas del procesado de datos. Para ello se propone la implementación de un algoritmo que aproxima los cálculos a nivel de celda y otro que hace lo propio a nivel individual de usuario. Entre ellos se observará cuál de los dos puede determinar mejor el dimensionado de los recursos de cómputo, dado que los factores que tiene en cuenta generalizan menos el cálculo permitiendo así integrar datos que resultan más individuales a nivel de usuario.

Este proyecto tratará de proporcionar una solución al problema de determinar el número de procesadores necesarios en dos centros de cómputo para satisfacer en un escenario de red 5G, en el que se tiene un conjunto de estaciones base sirviendo de recursos radio a una serie de usuarios. Dichos usuarios generaran tráfico de una naturaleza distinta, aunque relacionada, teniendo así dos slices dentro de nuestra infraestructura.

Además, se incluirá una propuesta de criterio de conexión entre las estaciones base y los centros de cómputo atendiendo a un balanceo de carga equitativo.

Resources Allocation algorithms for Network Slicing in 5G

Manuel Jesús Justicia Alados

Keywords: 5G, network slicing, Radio Access Network (RAN), cloud RAN (C-RAN), resource allocation.

Abstract

From some years to now, mobile networks have been developing an exponential evolution when talking about demands, sorts of content and utilizations. The next step of this evolution is actually deploying the 5G network all over the globe.

5G networks pretend to update how the global network is understood, starting by the network infrastructure which may be adapted to a completely new number of contents and traffic kinds appearing in the next years. The fast-growing number of devices that everyday try to make a part of the network creates the need for adapting in the proper network. This gives birth to some new goals for the fifth generation of networks such as extremely low latency, an increase in the energetic efficiency, big band width, integration of vehicles in mobile communications, massive machine communications and having enormous amount of sensors that are able to be connected to the network by moving from the physical world to the virtualized environment among so many others.

Among all those introduced functions by 5G in order to achieve the adaptation, network slicing aims to give to the network the ability of defining a big amount of logic networks inside the same physical infrastructure, giving service to the colossal amount of different kinds of traffic that can be generated by all users.

Network slicing is the main research topic in a large number of articles due to be considered a big needing because of the previous statement: satisfying all the generated traffic inside the network despite what its nature is.

In the other hand, we have CLOUD RAN being the other 5G big feature. The Fifth Generation of networks will redefine what the networks functions as we know them are. From having the eNB (the access node in LTE) developing radio and computing functions, C-RAN proposes a separation of those functions using a centralized and distributed unit system by distributing these functions between both units in order to move as many functions as possible from the physic world to the virtualized one..

The several natures of traffic, the growing demand, the energetic efficiency that is pursued, the low latency, all this makes necessary the development of highly optimized resource allocation algorithms in centralized units which are in charge of that tasks so the different slices can be satisfied in terms of QoS and QoE.

This project aims to add to already made researches the utilization of algorithms and their implementation to be able to stablish the required Computational Resources in the centralized units in charge of data processing. In order to achieve this, the implementation of an algorithm that approaches the calculus at cell level and the implementation of another algorithm that approaches the calculus at user level are proposed. They will be compared and suddenly it will be observed which one approaches better given that one of them takes into account individual features and might give a most likely to be real approximation.

It will be tryed to give a solution to the processor number establishment in two different computational centers in order to satisfy inside a 5G scenario composed by a set of Base Stations which serve Radio resources to a set of users. These users will generate traffic of diverse natures even though this natures will be related having then two slices inside our network.

Furthermore, a criteria proposal to determine the computational center to wich may be conected a certain base station in terms of load balance will be introduced.

Yo, **Manuel Jesús Justicia Alados**, alumno de la titulación Ingeniería de Tecnologías de Telecomunicación de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 75936747Q, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Manuel Jesús Justicia Alados

Granada a 3 de septiembre de 2020.

D. **Pablo Muñoz Luengo**, Profesor del Área de Telemática del Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada.

D. **Óscar Adamuz Hinojosa**, Miembro del Área de Ingeniería Telemática del Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Algoritmos aplicados a la reserva de recursos para Network Slicing en redes 5G***, ha sido realizado bajo su supervisión por **Manuel Jesús Justicia Alados**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 3 de septiembre de 2020.

Los directores:

Pablo Muñoz Luengo **Óscar Adamuz Hinojosa**

Agradecimientos

Me gustaría empezar esta sección enmarcándonos en el año 2020, donde desgraciadamente en el mundo no han cesado las desgracias y desastres naturales. Por tanto, aunque se salga de lo corriente en un proyecto técnico, creo que no solo yo, si no todos aquellos que llegaren a leer este trabajo agradeciéramos al personal sanitario en primer lugar que ha luchado y sigue luchando contra la pandemia ocasionada por el COVID-19. Es por ellos que si a la fecha que el lector este contemplando estas líneas goza de buena salud se encuentre en ese estado. No solo a médicos, enfermeros y farmacéuticos, sino a todos esos virólogos e ingenieros genéticos que estudian en este momento la vacuna y, querido lector, quiero transmitirte mis infinitos deseos de que la vacuna este en circulación para cuando estés leyendo esto.

Seguidamente, agradezco a todos los profesores que han contribuido a mi formación en esta carrera universitaria, pues gracias a su trabajo mucha gente como yo ha conseguido una formación en esta materia.

A Miguel y Toñi, los padres del que escribe y que no podrían haberme dado una educación mejor y un cariño mayor, pues gracias a ellos soy la persona que a día de hoy soy, gracias a no dejarme nunca que me rinda y procurarme un futuro que se antoja próspero y que siempre cuidarán de mí.

A Claudia, por siempre estar a mi lado y apoyarme siempre en todo lo que hago en esta vida, pues ella es la que quiero que sea la mujer de mi vida.

A mis tutores, Pablo y Óscar, que han sabido tener una paciencia sin parangón con este trabajo, una actitud impecable hacia mí siempre que me he dirigido a ellos para consultar cualquier tipo de cuestión, sabiendo guiarme en todo momento. Os deseo lo mejor en vuestra vida laboral y personal, me habéis demostrado ser dos personas excepcionales y muy comprometidas con lo que hacéis.

A todas las personas con las que he compartido sesiones eternas de estudio, de agobios y dudas. Pues con ellos he adquirido la capacidad de dejar a un lado el interés personal y poner por encima el siempre conseguir sacar adelante las materias en conjunto.

Al personal no docente de la escuela, sin su trabajo la escuela no sería lo que es, no funcionaría al nivel que funciona ni tendríamos las condiciones apropiadas en las clases para poder recibir esta formación.

Por último, agradecer a todos los deportistas de éxito, principalmente al difunto Kobe Bryant, pues el saber de su vida me ha aportado una visión y una ética de trabajo aún en construcción que ojalá un día se parezca a la suya, con los valores de no rendirse nunca, de siempre dar hasta su último aliento. Y al que para mí es el padre de la tecnología actual, Nikola Tesla, pues él es un referente para todos los ingenieros sin excepción estoy seguro.

Y permitidme cerrar esta sección con una cita del mismo Kobe, pronunciada en su corto ganador de un premio Óscar a mejor corto de animación.

"You asked for my hustle,
I gave you my heart".
Rest easy Kobe.

Índice.

Índice de figuras.

Índice de tablas

Acrónimos

GSM - Global System for Mobile Communications.
EDGE - Enhanced Data Rates for GSM Evolution
MMS - Multimedia Messaging Service
UMTS - Universal Mobile Telecommunications System
HSDPA - High Speed Downlink Packet Access
GPS - Global Positioning System
LTE - Long Term Evolution
uRLLC - Ultra Reliable Low Latency Communication
IoT - Internet of things
V2X - vehicular to everything
VNF - Virtual Network Function
RU – Radio Unit
CU – Centralized Unit
DU – Distributed Unit
BBU – Base Band Unit
RRU – Radio Resource Unit
eNB – Evolved NodeB
QoS – Quality of Service
QoE – Quality of Experience
CPU – Central Processor Unit
mMTC – massive Machine Type Communication
BS – Base Station
SINR – Signal to Interference and Noise Ratio
PRB – Physical Resource Block
MCS – Modulation and Coding Scheme
TFH – Transport Fronthaul

Capítulo 1.-Introducción.

1.1.-Antecedentes.

Las distintas generaciones de redes telefónicas han surgido en virtud de cambios significativos en la manera de entender este tipo de comunicaciones. Su evolución se ha visto motivada en el mayor de los casos por el creciente número de terminales y de usuarios, así como de avances en la investigación del tipo de contenido que se podía integrar en la red.

Así pues la primera generación de redes, el 1G, introdujo la tecnología para las llamadas entre teléfonos móviles con tecnología de conmutación de circuitos durante los años 70. A esto le sucedió la red GSM, que se puede considerar como 2G, con la posterior aparición de la red EDGE aun dentro de la segunda generación de redes móviles, en la cual ya se introdujo el SMS, el roaming y los servicios de facturación entre otras muchas características. Por su parte el EDGE, que conforma el 2,5G introdujo servicios multimedia como es el envío de MMS y el acceso a correo electrónico como exponentes más importantes en sus servicios integrados.

A comienzos del siglo XX aparece la tercera generación de redes, conocida como 3G e introduce el estándar UMTS y la interfaz radio HSDPA y HSDPA+, esta última considerada el 3,5G, donde empieza a introducirse el acceso a internet de alta velocidad, las videollamadas, banca virtual, integración de servicios GPS mediante internet y el video a la carta entre otros.

Años más tarde, hacia el 2010 se despliega la red 4G con su estándar LTE, introduciendo los servicios de telefonía IP, TV de alta definición, cloud computing, videojuegos a tiempo real y velocidades de hasta 1Gbps.

Finalmente comienza la investigación y el diseño de soluciones orientadas a la red de quinta generación hacia 2015. Con 5G surge la posibilidad de estar conectados en todo momento mediante dispositivos móviles wearables, integración de vehículos en la red, la reducción del consumo de energía en un 90%, el aumento de la velocidad en el orden del Gigabit y la implementación de muchísimas mejoras en la infraestructura de la red, tales como la implementación del Cloud RAN, la definición de celdas extremadamente pequeñas, el network slicing y la virtualización de la red.

Si vemos la evolución desde la aparición del primer terminal móvil que se comercializase en masa allá por el año 1983, la investigación en tecnología de las telecomunicaciones ha conseguido avances que nunca se podrían haber imaginado en ese año.

La necesidad natural del ser humano por socializar se ve satisfecha en gran medida gracias a las redes de comunicación, especialmente la inalámbrica. Tanto es así que el número de terminales móviles utilizados en el mundo ha evolucionado exponencialmente. Desde la aparición del 3G y el invento de los teléfonos inteligentes, las ventas se han disparado en estos dispositivos.

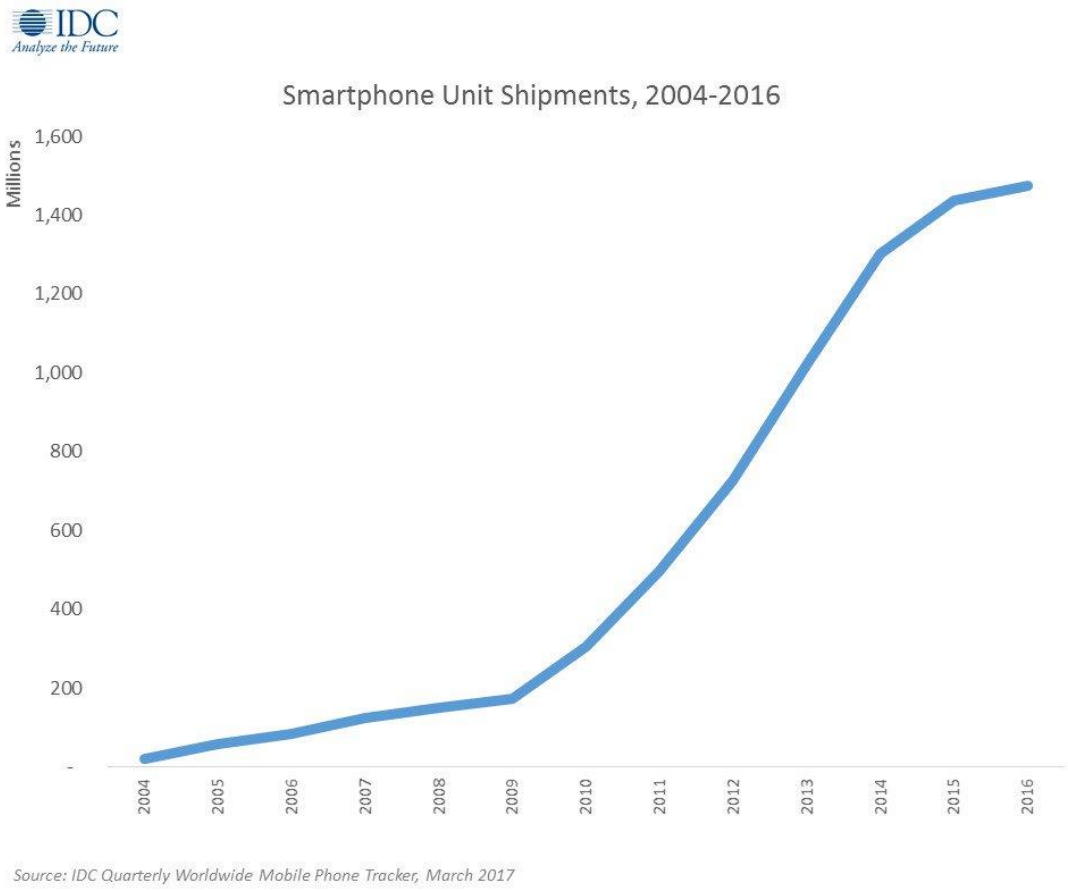


Figura 1.1: Número de smartphones vendidos.[1]

El grafico 1.1 muestra el número de smartphones vendidos desde el año 2004 hasta 2016 en millones de unidades, como puede verse la tendencia ascendente es una constante sin ritmo de parar.

Ante la gigantesca crecida del número de terminales inteligentes en el mundo, la infraestructura de la red debe evolucionar en consonancia el número de usuarios que hacen uso de esta red. Debido a ese factor tan humano de siempre querer más cosas y mejores y ese afán de superar todo lo conocido surgió 4G como la red más rápida conocida hasta el momento. Años más tarde, 4G ha quedado desbancado por su sucesor, aun en investigación y pruebas en muchos lugares: la quinta generación de redes, el 5G.

5G se implementa debido a que hace posible desplegar servicios que van más allá de Mobile broadband, 5G introduce lo que se conoce como extreme mobile broadband, un cambio en la red que busca una máxima velocidad de conexión y calidad de servicio, además de servicios mMTC que hacen posible la conexión de miles dispositivos IoT, servicios uRLLC relacionados con conseguir una baja latencia en la comunicación y servicios V2X enfocados a comunicaciones vehiculares.

La virtualización de las funciones de la red, conocidas como *virtualized network functions*, en adelante VNFs, permiten el aislamiento, la abstracción de la red y la facilidad de uso.

La virtualización está estrechamente relacionada con el C-RAN, paradigma introducido en 5G y que separará la red en unidades distribuidas y centralizadas. Así la unidad centralizada se compondrá de un conjunto de Base Band Units que se encargaran de implementar VNFs, de las cuales se hace uso para conseguir objetivos muy diversos como redes de baja latencia ultra fiables (URLLC) mediante el uso de computadores comerciales encargados de procesar los datos extraídos de la señal radio en banda base[2].

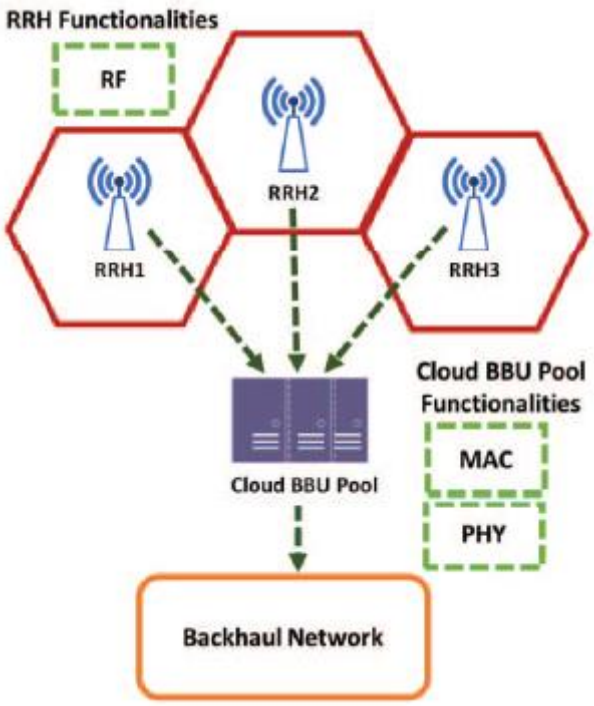


Figura 1.2: Unidades distribuidas y centralizadas.[3]

Así pues, surgen también nuevos conceptos en lo que es la infraestructura física y las divisiones que esta tiene, los eNB de 4G, dan paso a los gNB en 5G para la red de acceso.
En cuanto a las celdas, se introduce el concepto de small cell y macrocell, que se detallan a continuación.

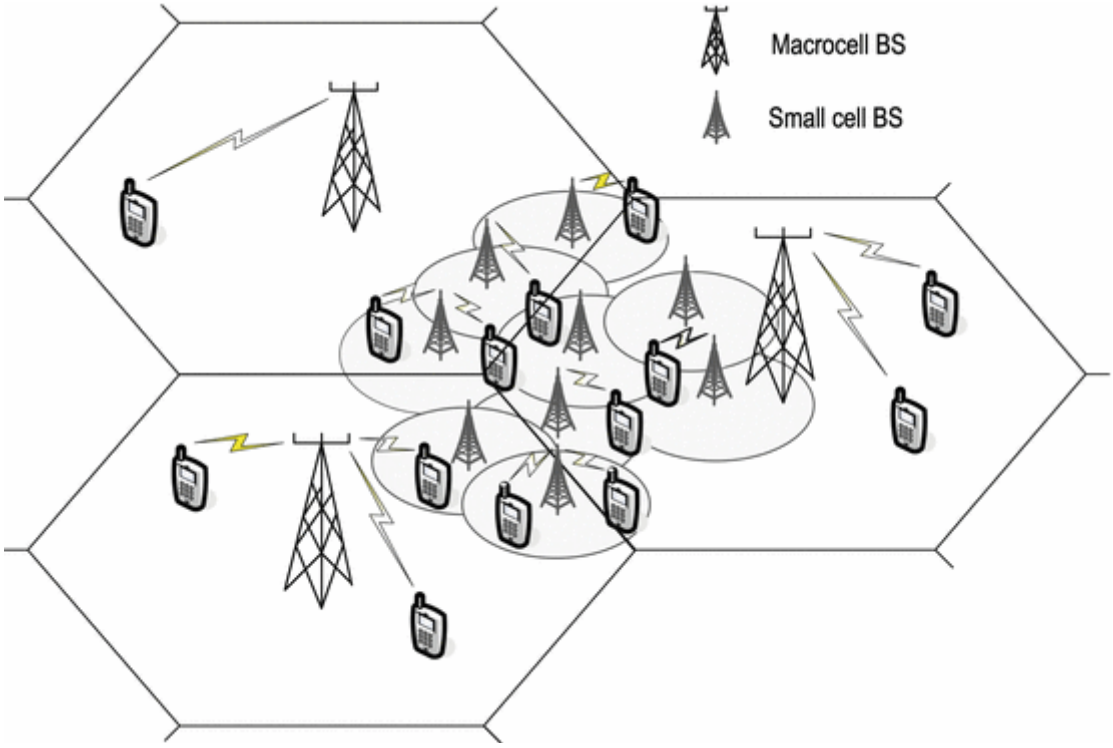


Figura 1.3: Macrocells y Small cells.[4]

Las macrocells pueden entenderse en este nuevo contexto como las celdas tradicionales de cobertura que se han desarrollado hasta en 4G. Las smallcell surgen en el contexto del 5G y dentro de estas se puede realizar una subdivisión en varios tipos: [5]

- Macrocells: Son el tipo de celda más parecido a las celdas convencionales, pueden definirse en radios de hasta 30 kilómetros.
- Microcells: Con radios de entre 2 y 4 kilómetros.
- Picocells: Radios de hasta 2 kilómetros.
- Femtocell: Su máxima dimensión es de un kilómetro y medio de radio.

Otro concepto muy importante dentro de las redes 5G y en concreto de este trabajo es network slicing. A modo de explicación sencilla, network slicing permite definir redes lógicas (denominadas *slices*) definidas dentro de una infraestructura física de red común que están adaptadas a los requisitos funcionales y de rendimiento de los servicios que acomodan.
Así pues, se pueden solicitar a la red slices para tráficos elásticos e inelásticos, destinados a jugar online, a consumir contenidos de video en cualquier plataforma, descarga de archivos y un largo etc.

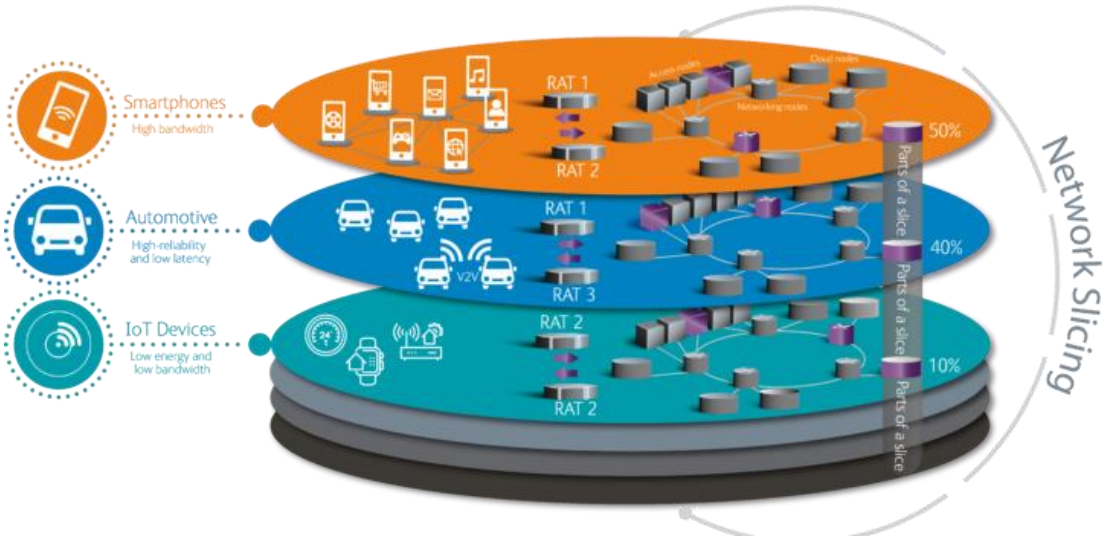


Figura 1.4: Concepto de Network Slicing. [6]

Como se ve en la figura 1.4 el slicing da lugar a la implementación de varias redes lógicas que se sustentan sobre una infraestructura física común. Gracias al network slicing conseguimos que esa infraestructura se pueda dividir en particiones lógicas cada una de ellas dedicadas a un servicio y tipo de tráfico concretos.
El hecho de que la red sea parcialmente imprevisible, pues aun existiendo distribuciones estadísticas que puedan modelar el comportamiento de la red y haya procesos estocásticos que ayuden a describir de forma matemática lo que ocurre en la red, introduce una nueva manera de localizar los recursos físicos para el procesado de la red. Esto es introducido con el Cloud RAN, una red de acceso a recursos radio que va a separar funciones de cómputo de la estación base física como hasta ahora se venía haciendo en 4G. De este modo todos los cálculos computacionales se pueden hacer en un centro de cómputo centralizado, o en un paradigma distribuido, el cual es el que se desarrolla en este trabajo teniendo en cuenta que se definirán dos centros de cómputo, lo que ahorra muchísimo dinero de la infraestructura e introduce una mayor eficiencia.

1.2.-Motivación y objetivos

La quinta generación de redes se presume como una revolución mundial en la materia de la

interconexión de todos los dispositivos móviles del planeta. Este trabajo comienza motivado por la necesidad de determinar el número de procesadores que se deben utilizar en un centro de cómputo que esté prestando ese servicio a un conjunto de celdas cercanas, implementando para ello un escenario de Cloud RAN separando las estaciones base en una unidad radio (RU), la unidad centralizada (CU) y la unidad distribuida (DU). Estando estas dos últimas dentro de la Base Band Unit (BBU).
Debido a la heterogeneidad de los usuarios de la red y de los tipos de tráfico que se ofertan en la red de redes, el estudiar el network slicing es una alternativa muy interesante de cara a incrementar al máximo la QoS de todos y cada uno de los usuarios.
Para la realización del trabajo, va a ser indispensable la utilización del concepto de C-RAN. Se plantea un escenario que bien puede emular una red 5G. Dispondremos de un conjunto de estaciones base conectadas a unos centros de cómputo, lo que visiblemente muestra la arquitectura que se utilizará en 5G, basada en gNB. Estos nodos se componen de una unidad distribuida (DU) que estaría conformada por las estaciones base cuyo principal objetivo es dar cobertura a una zona concreta y una unidad centralizada (CU) que correspondería a la unidad de donde se procesan todos los datos y mensajes de la unidad distribuida. Estas dos unidades son el ejemplo más claro de la construcción de un escenario en el que el C-RAN se pone en escena.
Los objetivos del proyecto se conforman así:

- El objetivo principal del presente trabajo no es otro que proporcionar un dimensionado de los recursos computacionales que se necesitan en un escenario 5G con una red de acceso que implementa Cloud RAN, con usuarios conectados a la RU y el procesado llevado a cabo por la BBU implementando esta la CU y DU.

Este objetivo principal vendrá de la consecución de los siguientes objetivos complementarios:

1. Describir el problema que se plantea y el escenario implementado para ello.
2. Análisis teórico de los algoritmos a considerar.
3. Derivación de los algoritmos para su aplicación al escenario implementado.

Asimismo, se le confieren los siguientes objetivos secundarios:

1. Revisar los temas de investigación actuales relacionados con el 5G.
2. Analizar propiamente el simulador antes de servirnos de sus datos de salida.
3. Seleccionar los datos de salida del simulador que nos conducen al objetivo principal.
4. Implementar mediante código MATLAB los algoritmos propuestos para el cálculo del número de CPUs necesarias en las unidades centralizadas para el procesado de los datos.

Una vez se consiga esa implementación y se obtengan los resultados de los algoritmos, se evaluará si estos son capaces de proporcionar un numero de CPUs válido para cumplir los requisitos de retardo que imponen los estándares de 5G.

1.3.-Organización de la memoria

Para proporcionar una idea sobre que se va a leer en la memoria de este proyecto se proporcionan aquí unas pinceladas sobre cómo se estructura esta memoria y sobre que versa cada capítulo de la misma.

1.-Introducción.

Este capítulo se dedica a poner en conocimiento del lector sobre que versa el proyecto de manera sencilla además de introducir conceptos importantes a lo largo del proyecto.

2.-Estado del arte.

Proporciona una visión de varios horizontes de investigación actuales relacionados con el network slicing y las redes 5G.

3.-Planificación.

Describe la historia de cómo se ha desarrollado el proyecto, el trabajo empleado, el tiempo dedicado a cada tarea, los costes y los requisitos de este trabajo

4.- **Estimación de recursos de computación en C-RAN: análisis teórico.**

En este capítulo se describe el funcionamiento del simulador utilizado para desplegar los escenarios de la red que se han utilizado para probar los algoritmos y desarrollar su implementación en función de los datos de salida del simulador. Asimismo, se presentan los algoritmos utilizados para el cálculo de la frecuencia, el cómo se han desarrollado y que requisitos y datos se estiman importantes para la implementación.

5.-Descripción e implementación de los algoritmo

Este capítulo está dedicado en su totalidad a explicar cómo se llevado a cabo la implementación de los algoritmos utilizados para el cálculo de la frecuencia computacional necesaria.

6.-Evaluación de los resultados obtenidos.

Aquí se explica cómo se ha realizado la batería de pruebas necesaria para ver si realmente funcionan los algoritmos desarrollados y cuáles son los resultados que arrojan los mismos en materia de numero de CPUs necesarias para cumplir los requisitos de retardo.

7.-Conclusiones

Por último, se hacen unas observaciones sobre los resultados obtenidos y sobre lo que significa en si el grado en el que se integra este proyecto.

Capítulo 2: Estado del arte

2.1.-Introducción.

La creciente demanda de servicios de red de diversa índole, así como un número de usuarios de la red que aumenta por momentos, hace que la importancia depositada en la velocidad de la transmisión de datos sea un factor crítico a la hora de planificar una red y dimensionar los recursos que se destinan a ella.

Nos encontramos en un punto donde no es un secreto que las redes móviles y la gestión de la movilidad de sus usuarios han cobrado una relevancia nunca vista para la interconexión de dispositivos móviles. Derivado de ese gran número de interacciones entre usuarios, equipos finales y equipos de red, nace la diversidad de contenido disponible para consumir y disfrutar.

Entre esos contenidos encontramos varios que son especialmente relevantes dada su creciente demanda por parte de los usuarios, como son el streaming multimedia y los videojuegos multijugador en tiempo real. Tal es su relevancia, que, de cara a la próxima generación de redes, uno de los temas que más aportan a la inmediatez de la satisfacción de la demanda de los usuarios, surge lo que se conoce como Network Slicing.

Dentro de la investigación de las redes 5G, el Network Slicing es uno de los conceptos que más interés por parte de los investigadores capta, si bien es cierto que es uno de los que menos determinados están.

A continuación, se procuran unas pinceladas sobre los temas que junto con el network slicing copan las prioridades investigativas de los ingenieros y/o complementan y sirven de base para el slicing.

2.2.-Network Function Virtualization, NFV.

Las funciones de red virtualizadas son un nuevo paradigma en lo que se refiere al cómputo en la nube para sustento de las comunicaciones. Dichas funciones virtualizadas cobran su nombre del hecho de estar implementadas en máquinas virtuales localizadas en los distintos centros de cómputo y procesado de datos. Su instalación dentro de la red permite tener una ubicuidad de las funciones y seguir un proceso bajo demanda. Un punto fuerte de estas funciones virtualizadas es que los operadores pueden implementarlas en su red y ofrecer redes especializadas en ciertas necesidades sin tener que invertir cantidades ingentes de dinero. [7]

Las funciones virtualizadas tienen multitud de aplicaciones y beneficios que van desde la reducción del capital y overhead de operaciones hasta la reducción del tiempo que tardan los productos en salir al mercado.[7]

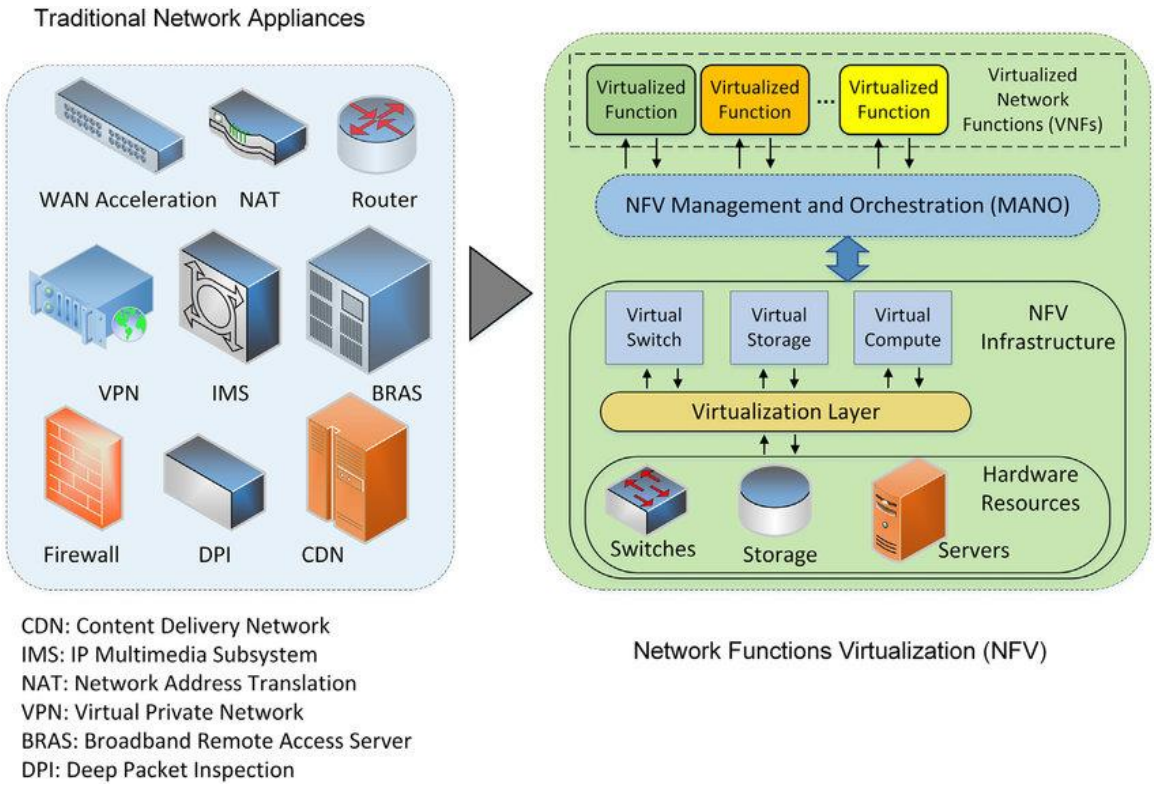


FIGURA 2.1: VNFs.[8]

Como se aprecia en la figura 2.1, la arquitectura del sistema de máquinas virtuales que definen las VNFs se compone de un gestor de la infraestructura virtual que controla tanto la red interna virtual como el almacenamiento. Por su parte el gestor de las VNF junto con el planificador de las VNF controla desde los permisos para ejecutar una función concreta hasta la reserva de recursos dentro de las máquinas virtuales.

La virtualización cobra una importancia crítica dentro del panorama futuro en las redes de comunicación. Como bien se viene dejando claro, la generación futura se caracteriza por la diversidad y por la heterogeneidad, lo cual hace que el coste de la red se reduzca enormemente si es una red virtualizada, capaz de desempeñar funciones para agregar esa heterogeneidad. Es donde la red definida por software es capaz de crear pequeñas subredes lógicas que hacen a cada una satisfacer las necesidades de un tipo determinado de tráfico.

2.3.-Network Slicing

La definición del network slicing puede expresarse como la forma de definir sobre la misma infraestructura física de la red, pues de otra manera este concepto no podría existir, de una serie de redes lógicas (denominadas slices), que se especializan en prestar un servicio concreto, reservando para ello una serie de recursos dentro de los compartidos por toda la infraestructura física.

El principal cometido del network slicing es ofrecer una solución escalable y sencilla de segmentar la red para ofrecer servicios a demanda. Pueden ser desplegados en función de unas características específicas de QoS como latencia, seguridad, disponibilidad y un largo etcétera. El slicing es un concepto que se complementa con el de Cloud RAN [9], expuesto en detalle en la **sección 2.4**.

Introducir el network slicing es el siguiente paso en la virtualización de la red haciendo posible la existencia simultanea entre redes lógicas de naturalezas totalmente diferentes pero aisladas dentro de la misma red.[2]

En [10] se detallan los requisitos de los slices desde la perspectiva de la gestión de recursos radio, dicha investigación está relacionada con el problema que pretende resolver este proyecto, sin

embargo, la diferencia reside en que en este proyecto se quiere abordar el dimensionado de recursos computacionales en lugar de recursos radio.
Los requisitos propuestos son:

- Deben ser dinámicamente creados y modificados.
- En función de sus requisitos, el área cubierta por el despliegue de un slice puede cambiar. Es decir, un slice dedicado a la ultra baja latencia estará definido en un área menor que uno dedicado a la comunicación masiva de máquinas (mMTC).
- En función de a que este dedicado el slice, habrá servicios cuya ejecución esté garantizada y otros que no. Por ejemplo, en un slice dedicado a la reproducción de video en streaming bajo demanda, podemos imaginar una suscripción de pago a una plataforma, en este caso tendríamos una conexión con parámetros garantizados para el streaming, pero sin embargo si tratáramos de hacer uso de la navegación web no tendríamos esas garantías.
- Los requisitos de throughput, latencia y estabilidad serán distintos según el tipo de slice.
- El buen funcionamiento de un slice no se debe ver comprometido por otros de menor prioridad (es decir, los slices están aislados entre sí).
- Cada tenant, que es la entidad que solicita a la red la creación del slice, puede aplicar políticas de admisión y planificación en su slice cumpliendo con los recursos que se le garantizan.
- La utilización global de recursos debe ser maximizada desde la perspectiva del operador de red.

[10].

2.4.-Cloud RAN.

A modo de primera definición, podemos designar al Cloud RAN como el hecho de virtualizar parcialmente las funciones de red usadas en los nodos de acceso radio. El Cloud RAN engloba a muchas de estas funciones virtualizadas que se aplican a la parte radio.
Como extraemos en [9], el Cloud RAN, o C-RAN, introduce un diseño revolucionario hasta la fecha de la arquitectura de las redes móviles. Introduce una capacidad de incrementar el tráfico de datos reduciendo el coste monetario y operacional.

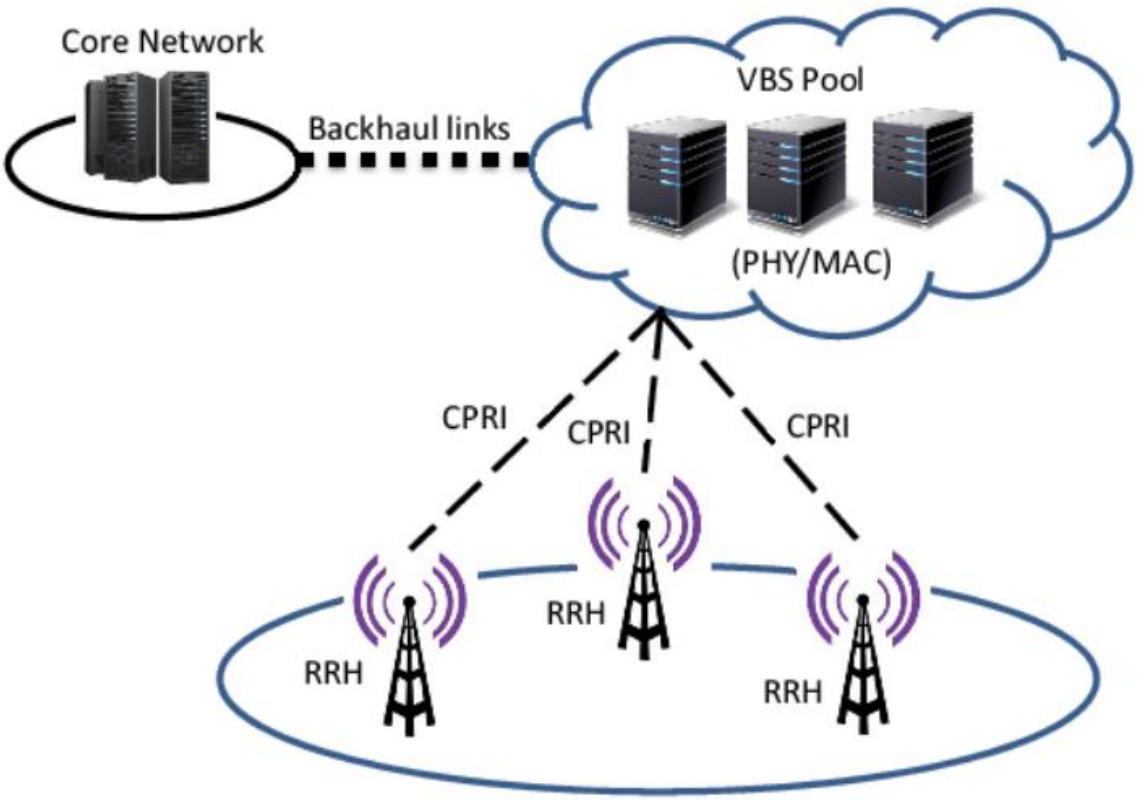


FIGURA 2.2: Arquitectura Cloud-RAN.[11]

El gNB es el nodo equivalente en 5G al eNB en 4G. En C-RAN este nodo se verá diseccionado en tres unidades, la RU, la CU y la DU.

El C-RAN se basa en desacoplar las funciones de cómputo del gNB y las reúne en un centro de procesado común para todas. De esta manera la red queda descompuesta en centros de recursos radio con antenas y centros de cómputo albergando procesadores de tiempo real de alto rendimiento que controlarían una serie de lo que antes se conocía como estaciones base, cuyas funciones de cómputo quedarían extirpadas en virtud de las Base Band Units (BBU), que constituirían los centros de cómputo.

En C-RAN existen una serie de requisitos que se pueden denominar críticos:

1- La capacidad requerida en el FrontHaul.
2- El requisito de latencia para la BBU.
3- Requisitos de tiempo real para el sistema operativo y el entorno de virtualización.[2].

Dentro del estudio del C-RAN, donde la virtualización es una de las herramientas principales se presentan tres arquitecturas virtualizadas principalmente, todas ellas constituidas por Hardware, sistema operativo y funciones virtualizadas. Dichas arquitecturas son las máquinas virtuales, los containers y los unikernels.

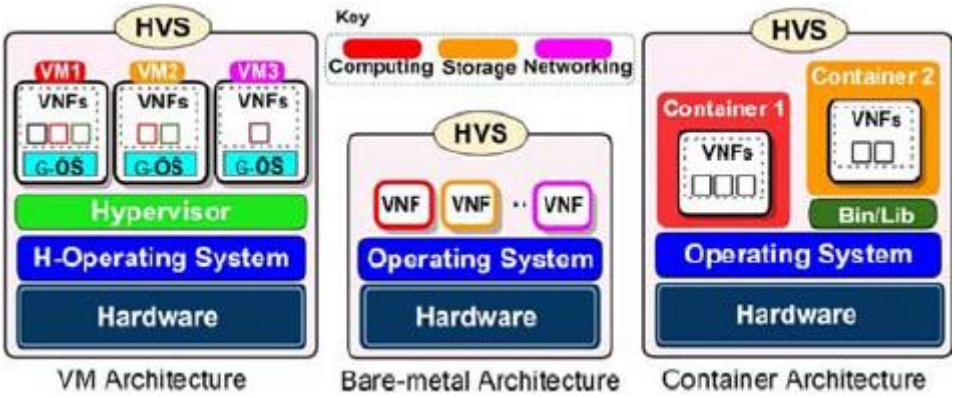


FIGURA 2.3: Arquitecturas de sistemas C-RAN [2]

En la arquitectura de **máquinas virtuales** además del hardware y del sistema operativo se incluye una tercera capa que proporciona recursos hardware virtualizados, con un sistema operativo invitado. Sin embargo, esta adición de una capa más nos incluye un overhead mayor si bien es cierto que el nivel de aislamiento de las funciones que se virtualizan es mayor mediante esta solución.[2]

Los **containers** proporcionan una solución basada en un sistema operativo ligero que está virtualizado. Es capaz de agrupar los recursos y procesos además de otros contenedores. Presenta una ventaja en cuanto a rapidez, portabilidad e impacto pues son soluciones muy ligeras, sin embargo, tienen un inconveniente en cuanto al aislamiento del hardware. Cabe destacar dentro de su portabilidad que si los corriéramos en VMs podríamos incorporarles el hipervisor, que es la capa antes mencionada en las Máquinas Virtuales para añadir un aislamiento hardware, lo cual paliaría el inconveniente que esta arquitectura puede presentar.[2]

A medio camino entre las dos arquitecturas previas, se presentan los **unikernerls**. Se componen de un sistema operativo especializado muy ligero que contiene las librerías mínimas y código de aplicación. Presentan una opción muy buena en cuanto a peso y seguridad, sin embargo, no son aún realizables en la práctica.[2]

La red de Backhauling se aprecia en la figura 2.2 y la podemos entender como la que conecta los distintos centros de cómputo que se disponen en nuestra red. De esta manera inmediatamente se obtiene que la red de Backhauling va a estar basada en una alta heterogeneidad debido a las infinitas naturalezas del tráfico que debe ser capaz de gestionar.

Dentro de esta heterogeneidad, en [12], se mencionan distintos tipos de gestión que esta red debe de satisfacer y dar soporte, entre estos está el gestionar recursos radio tales como la asociación de las celdas.

2.5.-Protocolo HARQ.

Hybrid Automatic Repeat Request, es una técnica empleada para la detección y corrección de errores, para ello se incluyen bits llamados bits de detección de errores.

Su papel en C-RAN resulta de una importancia crítica cuando toda la pila de protocolos 5G-NR (new radio) está virtualizada, pues es el encargado de imponer un límite superior al tiempo máximo de procesamiento que tiene que cumplir una BBU para poder garantizar la rapidez que exige una red 5G.

Dicho limite se trata de un Round Trip Time (RTT) cuyo valor es de 8ms y que se divide en los siguientes subretardos [2]:

1- Retardo de propagación, entendido como el tiempo que tarda la señal en viajar desde la DU hasta el usuario.
2- Retardo de adquisición, que refiere al tiempo en el que la unidad distribuida procesa la llegada de un mensaje.
3- Retardo de transporte FrontHaul. Este retardo hace referencia al tiempo que emplea el

mensaje en viajar desde el centro de cómputo hasta la estación de radio o RU.

4- Un offset, utilizado para sincronización.

El protocolo HARQ se articula en torno a una serie de Frames y deadlines para la recepción y el *acknowledgement* de los distintos mensajes que se intercambian, imponiendo así fronteras muy restrictivas a la hora de desarrollar retardos en dicho intercambio.

Para ayudar a la comprensión de cómo funciona HARQ, se espera que la siguiente figura ilustre satisfactoriamente dicho funcionamiento.

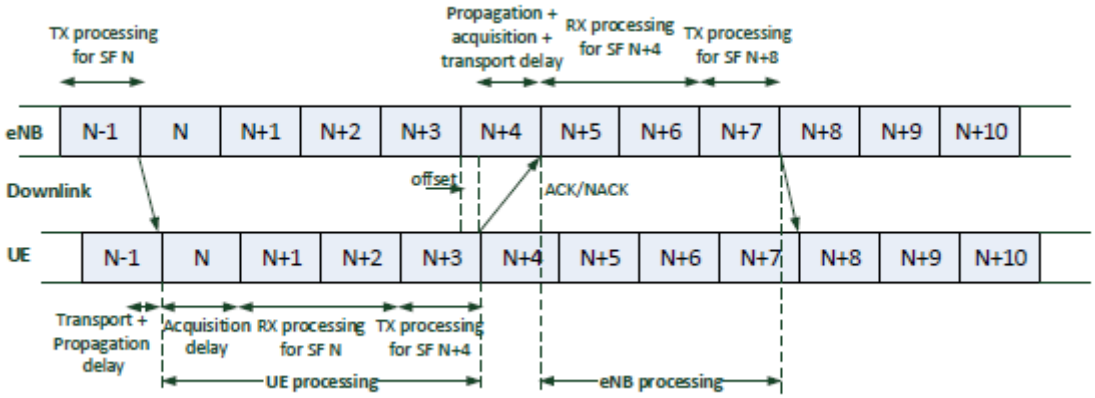


Figura 2.4: Diagrama de Frames HARQ.[12].

Pondremos el foco como en el diagrama específico para DOWNLINK. Supongamos que en el subframe N se envía una PDU, Protocol Data Unit, que viene a referirse a un mensaje con datos, desde el centro de cómputo a la estación base a la que un determinado usuario está conectado. Dicho mensaje se adquiere durante el frame N+1 y se demora un tiempo conocido como **adquisition delay**. Dicho mensaje que se ha adquirido debe de procesarse tanto en la cadena de recepción como la de transmisión en un tiempo menor de los 3ms que se imponen, esto es, debe de estar procesado, a lo sumo, cuando termine el subframe N+3, para que en el subframe N+4 su ACK/NACK sea transmitido.

En FDD LTE, el Round-Trip Time (RTT) para HARQ es de 8ms, y de esos 8, 3 son para el procesado con la división para enlace ascendente y descendente que ya se ha mencionado. [12]

Como está expuesto en [2] el base-band processing time, que no es otra cosa que el tiempo de procesamiento que estamos fijando para calcular la frecuencia, se subdivide a su vez en tiempo de codificación, transformada de Fourier y modulación además del propio tiempo de procesado. Además, cabe destacar que independientemente de la forma en la que se implemente cualquier algoritmo y sea la aplicación de este cual sea, el tiempo dedicado al procesado siempre tendrá la misma frontera superior.

2.7.-Trabajos previos

En relación a estos términos y conceptos existe una gran variedad de trabajos realizados: el Cloud RAN se desarrolla en artículos como [2], en el cual se introduce como se lleva a cabo el cambio del eNB en LTE hacia las arquitecturas C-RAN que se han presentado anteriormente, además presenta unos resultados numéricos que relacionan el MCS utilizado con el tiempo de procesamiento que se necesita para realizar ciertas funciones, ahora virtualizadas.

Los autores de [9] proponen una implementacion de un planificador destinado a la transmisión del trafico de distintos slices en función de una serie de recursos asignados a cada uno de los slices que modelan en el artículo, aportando unos resultados numéricos que prueban la efectividad de la planificación desarrollada.

La referencia [12] evalúa las últimas investigaciones en Cloud RAN y aporta mediante un escenario desplegado sobre OpenAir Interface, para el cual proponen una reserva de recursos eficiente

minimizando el consumo de energía sobre un escenario C-RAN, para ello incluyen el consumo eléctrico tanto de la BBU como de la RRU.
Los autores de [14] proponen un modelo para la planificación y aprovisionamiento de la red que optimiza el coste del despliegue de un escenario C-RAN para redes 5G. Proponen también una optimización del despliegue de las funciones virtualizadas garantizando cumplir la demanda de los usuarios y los requisitos de rendimiento.
En [15] proporciona una arquitectura de red 5G que adapta la evolución de los tipos de comunicación, las tendencias de los usuarios y la tecnología existente y discute temas que toman parte en la construcción real de una red 5G.
En [12] se analizan temas críticos en la virtualización al mismo tiempo que proporciona un modelo de la carga de procesamiento para procesado en banda base de LTE operando de nuevo sobre OpenAir Interface.
El protocolo HARQ se desarrolla y explica principalmente en [2] donde se integra dentro de la arquitectura Cloud RAN y en [15] donde además se le confiere otro retardo más a tener en cuenta dentro del cómputo total para cumplir con los tiempos de procesado impuestos. En [14] los autores introducen los principales motivos de implementar slices con las tecnologías que pueden hacerlo y discute sobre la implementación y estandarización del 5G.
Por parte del network slicing en [16] se discute como la red de acceso puede seccionarse para satisfacer requisitos heterogéneos mientras se comparten los mismos requisitos de radio y de procesamiento mediante la configuración de las capas más bajas de la red.

Capítulo 3.-Planificación.

3.1.-Introducción

A la hora de la realización de un proyecto de investigación y que sirve como punto final a un grado tan excelso en contenido, el resultado del mismo tiene que estar a la altura. Es por ello que el planificar este proyecto dentro de lo que supone una pandemia sin precedentes resulta crucial aún más si cabe.

Sin embargo, la disposición de los tutores a la hora de la resolución de cuestiones planteadas por mí ha sido impecable y llevada a cabo con una celeridad sobresaliente, haciendo así posible determinar la planificación de manera satisfactoria

A continuación, se detalla una división de las tareas junto con una descripción de las mismas que en conjunto darán forma a este trabajo, una descripción de los recursos de los que se dispone para su realización y finalmente una estimación de los costes monetarios que supone este proyecto.

3.2.-Cronología

A continuación, se desglosan las tareas en las que se dividirá este proyecto por orden cronológico, desde la primera idea hasta la última de las tareas que sería el plasmo de todo el proyecto en este documento.

Bloque de trabajo 1:Revisión del Estado del Arte.

T1.1.-Documentación sobre conceptos relacionados con el 5G

El primer paso antes de ni siquiera elegir el tema en el que queremos enfocar el trabajo es hacer lectura comprensiva de los diversos conceptos que están relacionados con la quinta generación de redes. Pretende ser la aportación de una base sobre las ideas que se manejan y cuáles de los temas están más o menos investigados y en cual podría encajar este trabajo para tratar de aportar nuevas ideas.

T1.2.-Elección del tema del trabajo

Una vez que la lectura comprensiva del conjunto de artículos e investigaciones convenientes, llega el momento de valorar cuales de todas las opciones y ramas de investigación cumplen en mayor medida con los gustos propios y cual desarrolla mayor interés para su investigación. Sobre esta decisión hubo poca duda en que el trabajo se quería enfocar al network slicing pues la naturaleza de este tema dejaba muchas ideas sobre las que poder hacer un trabajo.

T1.3-Adquisición de conceptos relacionados con el Network Slicing

De nuevo, después de elegir el tema del trabajo, el siguiente paso debe ser la ampliación del conocimiento en conceptos relacionados con el slicing. Centrando la lectura en conceptos más enfocados al slicing tales como la forma de reservar recursos físicos dentro de un centro de cómputo, definir una barrera imaginaria entre lo que queremos dejar a cargo de la unidad distribuida del gNB y las funciones que llevará a cabo la unidad centralizada.

Bloque de trabajo 2: Estudio del simulador.

T2.1.-Familiarización con el simulador

Para el desarrollo de este trabajo se utilizará un simulador desarrollado sobre el software MATLAB. Es por ello que antes de construir ningún código, es muy importante el familiarizarse con el simulador del que partimos. A fin de saber cuáles son los datos de los que este simulador se nutre para sacar unos resultados que posteriormente deberán ser evaluados a fin de determinar cuáles de las variables resueltas en el serán importantes para continuar con el trabajo.

T2.2.-Evaluación de datos relevantes para construir el modelo

El hecho de partir con el simulador ya funcionando, implica que inmediatamente después de conocer lo que ocurre durante la ejecución del mismo, en el cual se construye un escenario que podríamos considerar real, es conveniente que los datos de salida del mismo se revisen en función de la utilidad que nos aportarán. Por tanto, del simulador hay que evaluar cuales de los datos arrojados se proceden a utilizar como base para los cálculos en los que se enfoca este trabajo.

Bloque de trabajo 3: Implementación de los algoritmos propuestos.

T3.1.-Eleccion de la modulación asignada a cada celda y usuario

Al principio, para dar el primer paso, debemos analizar cada celda de cobertura y cada usuario. Se proponen dos aproximaciones para el cálculo de la frecuencia: la primera consiste en promediar para una celda en función de la eficiencia espectral de sus usuarios. La segunda, particularizar para cada uno de los usuarios en el sistema y ver como impactará eso posteriormente en la capacidad de calculo que va a ser necesaria en la unidad centralizada.

T3.2.-Elección del algoritmo para el cálculo de la frecuencia requerida

Esta tarea se centra en evaluar diversos algoritmos enfocados al cálculo de la frecuencia de cómputo necesaria para procesar todos los mensajes dentro de las fronteras temporales requeridas. Teniendo en cuenta los datos que el simulador pueda ofrecer se deberá determinar cuál es el que mejor ajusta con la salida de este y por tanto es más adecuado para nuestro cometido.

T3.3.-Cálculo del requisito de frecuencia de cómputo por celda

Aquí es donde entra la modificación comentada al algoritmo que se elige. De nuevo primeramente se utilizará para calcular la frecuencia que requeriría una celda si asignáramos valor medio a la eficiencia espectral de la celda en función de las distintas eficiencias espectrales de los usuarios. Posteriormente el cálculo se volverá a desarrollar para particularizar para cada usuario las modificaciones que se estimen oportunas en la implementación del algoritmo que se elija.

T3.4.-Cálculo de los tiempos de procesamiento

Inmediatamente después del cálculo de la frecuencia requerida, surge la idea de hacer los cálculos justo al revés, es decir calcular que tiempo de procesamiento es capaz de darnos cada frecuencia en cada celda para poder elegir fácilmente la frecuencia de cómputo que requiere dicha celda.
De nuevo este cálculo viene pareado, se realizará tanto por celda como viendo la contribución

individual de los usuarios.

T3.5.-Definición de un criterio para comparar celdas de cobertura

A la hora de calcular los requisitos, con vistas de costes dedicados a la construcción de un escenario en la vida real es interesante ver una comparativa que nos indique cuanta cpu llegaría a consumir una celda si disminuimos su área de cobertura, incrementándose por tanto el número de celdas que se deberían de montar, o si por el contrario el tener celdas de tamaños grandes en vista de poder tener un número reducido de celdas puede ser de mayor utilidad a la hora de dimensionar los recursos computacionales para el procesado de los mensajes de todos los usuarios.

T3.6.-Designación de los centros de cómputo

El siguiente paso para continuar determinando donde se encontrarán los recursos físicos y los virtualizados es determinar donde se encuentran los centros donde se va a procesar el tráfico de los usuarios y las celdas.

T3.7.-Elección de un criterio para la conexión RRU-BBU

Se desarrollará un criterio para asignar cada unidad de recursos radio con un centro de cómputo que la maneje. Para ello finalmente se optará más por una solución lógica que física, haciendo que cada estación radio se conecte a una unidad de banda base en función de un algoritmo que balancea la carga, buscando conseguir resultados muy parejos en lo que respecte a la carga soportada por cada una de las estaciones de banda base.

T3.8.-Aplicación del algoritmo para determinar el número de CPUs necesarias

Tras evaluar cada estación base y obtener sus límites de tiempo para el procesado del tráfico de sus usuarios, se abre la puerta de aplicando un criterio obtenido de [2], se calcula finalmente un numero de Cores necesarios solamente para procesar el tráfico generado en la red.

T3.9.-Búsqueda e implementación de mejoras al algoritmo

Después de un análisis pertinente de los resultados arrojados por la tarea anterior se hará una pequeña búsqueda de mejoras al código y al algoritmo, para optimizar el cálculo de las CPUs necesarias. Pasando así a dar un resultado mucho más acorde con lo que sería un escenario real.

T3.10.-Evaluación del sistema en función de la carga en el mismo

Para concluir la parte de elaboración de código y trabajo con el simulador se hace una prueba al sistema. Consistente en una prueba evolutiva en función de la carga, la cual aumentamos añadiendo usuarios al sistema y evaluando si con los recursos físicos que tenemos, aun sería capaz la BBU de procesar todo el tráfico de todos los usuarios en el tiempo estipulado.

Bloque de trabajo 4: Redacción de la memoria.

Como punto final a este proyecto y titulación, se plasma en un documento esta descripción del mismo.

En el siguiente diagrama de Gantt se recoge el lapso temporal transcurrido entre cada tarea desde su inicio hasta el final de su realización.

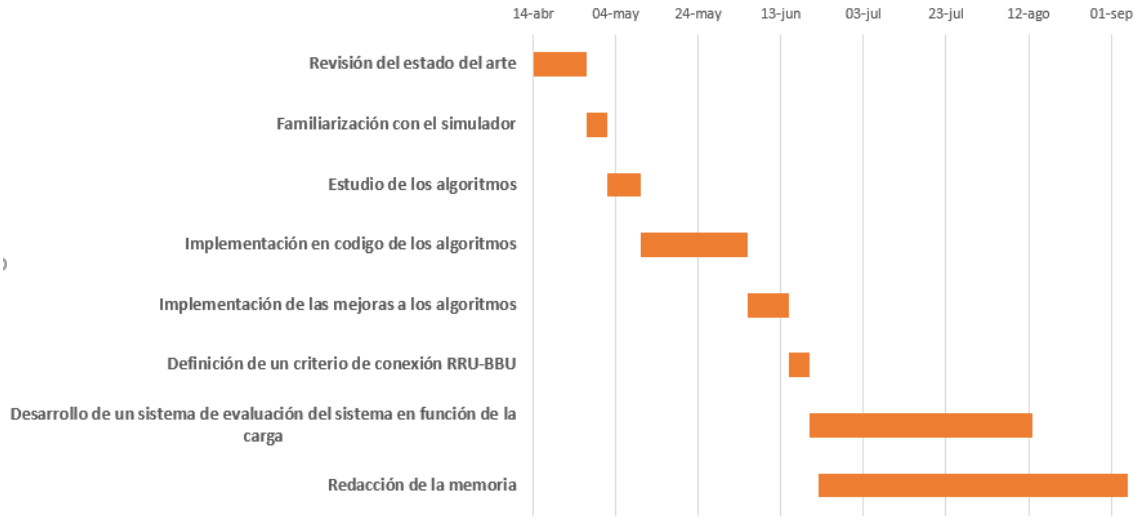


Figura 3.1: Diagrama de Gantt para la planificación de la realización del proyecto.

3.3.-Planificación de recursos

En esta parte se hace una estimación del costo del proyecto. No queriendo decir exclusivamente monetario, sino también costo en relación a tiempo invertido, horas de trabajo realizado, de investigación, de reunión con los tutores del trabajo y de hardware.
Un proyecto así, aunque sea propiamente una manera por parte del alumno de mostrar los conocimientos que ha adquirido durante los años que demora el Grado en esta ingeniería, no sería extraño verlo como encargo a un grupo consultor o incluso a los mismos investigadores que han realizado los artículos que se usan aquí como referencia para hacer nuestro estudio particular.
Pues si bien en estructura y directrices podría fácilmente compararse con los proyectos encargados a una empresa, siempre salvando las distancias, con una analogía bastante clara, donde el tribunal evaluatorio sería la empresa que encarga el proyecto, aunque en este caso lo elige el alumno. Los tutores en este caso, serían los jefes de equipo y la comisión interna evaluadora del resultado del proyecto, y el alumno compondría la unidad de trabajadores que colaboran en la realización del grueso del proyecto.
Partiendo de esta analogía sencilla y clara, se procede a hacer un desglose en coste humano, hardware y software de este proyecto, concluyendo con su estimación monetaria.

3.3.1.-Coste de tiempo y coste humano

Pártase de la estructura propuesta anteriormente para la realización de las tareas que conforman el proyecto. Las horas que se determinen para dichas tareas se verá incrementada también por el tiempo de reuniones telemáticas con los profesores y otras fuentes de tiempo.
Atendiendo a la metodología ensayo-error-mejora, se estima que en total la cantidad de horas dedicadas por parte del alumno al proyecto supera las 300 horas con creces y por tanto los 12 créditos ECTS asignados a la asignatura Trabajo de Fin de Grado cumplen las estimaciones horarias por crédito que se estipulan.
Por parte de los tutores, su trabajo ha sido encomiable mostrando siempre una gran disponibilidad y celeridad para la atención al alumno, viniendo de ellos un gran número de horas trabajadas en

relación a este proyecto en sus tareas de asesoramiento, investigación, entrevistas y desarrollo del simulador utilizado. Teniendo todo esto en cuenta, la cantidad de horas trabajadas por ambos, sin tener en cuenta la realización y construcción del simulador, cuyo desarrollo es de una dedicación incalculable, se puede estimar en unas 60 horas por parte de cada uno de ellos.
A todo esto, hay que sumarle el tiempo que la comisión evaluadora será público de la exposición de este trabajo y el tiempo invertido en deliberar la calificación del mismo. Estimado en una hora aproximadamente por profesor de la comisión evaluadora. Constituida por tres profesores como se detalla en la normativa expuesta en [27].
Con todo esto puesto sobre la mesa, se detalla a continuación la estimación en una pequeña tabla.

Entidad trabajadora	Personas involucradas	Horas de trabajo
Alumno	1	320
Tutores	2	120
Comisión evaluadora	3	3
Totales	6	443

Tabla 3.1: Desglose de horas dedicadas al proyecto.

3.3.2.-Coste hardware

Para la elaboración del Proyecto se utiliza un ordenador portátil personal. Se trata de un ordenador ASUS ROG Notebook PC GL752V. El equipo tiene las siguientes características relativas a la potencia y a su rendimiento:

-Procesador Intel Core i7 6700HQ Quad Core @2.6 GHz con ultraboost a 3.6GHz.
-1TB HDD y 240GB SSD.
-2x8GB DDR4 2133MHz RAM
-Gráficos NVIDIA GTX 960M 2GB dedicados.

En el momento de la compra, el ordenador no incluía el disco de estado sólido ni la segunda tarjeta de memoria RAM. [17]

3.3.3.-Coste software

En este proyecto se ha utilizado el software MATLAB 2018A para la realización del código referente a la programación para la simulación.
El sistema operativo del ordenador es WINDOWS 10 Professional.

3.3.4.-Coste monetario

Ahora que se ha descrito cada coste del proyecto, vamos a proceder a estimar lo que el mismo hubiera costado de haberse desarrollado fuera del marco de un TFG.
Para los costes software tendremos el siguiente monto:

Producto	Coste
Licencia MATLAB estándar de por vida	2000€ [18]
S.O WINDOWS 10 Professional	259€ [19]
Totales	***2259€***

Tabla 3.2: coste total software.

De igual forma se detalla el coste hardware atendiendo a la observación que se menciona en la sección de coste hardware:

Producto	Coste
Asus ROG GL752V	1300€ [20]
Tarjeta RAM SODIMM 8GB QUMOX	36,09€ [17]
SSD M.2 WESTERN DIGITAL GREEN 240GB	42,74€ [21]
Totales	***1378,83€***

Tabla 3.3: coste total hardware.

Por último, se expone ahora una estimación del coste humano, para lo cual se detalla el criterio a modo de entender mejor la tabla de coste monetario para el trabajo humano de acuerdo con [22]. La comisión evaluadora, compuesta por tres profesores, asumidos como doctores, en calidad de Profesores Titulares de Universidad, se les asume un coste monetario de 34172,54€ al año, asumiendo 40 horas semanales y 52 semanas por año, el coste de una hora de su trabajo se estima en 16,43€ por hora trabajada.
En el caso del tutor Pablo Muñoz Luengo, cuyo puesto es Profesor Ayudante Doctor, se le asume una retribución anual de 28924,03€, lo cual de la misma manera que antes nos deja una estimación de 13,9€ la hora.[23]
Para el tutor Óscar Adamuz, ejerciente como Contratado Predoctoral de tercer año, se le asumirá un sueldo de 17321,18€ anuales, lo que hace un total de 8,32€ la hora.[24]
Por último, en el caso del alumno se asumirá un sueldo de ingeniero Junior en contrato modalidad de prácticas, de 14000€ anuales, dando lugar a 6,73€ por hora trabajada.
Detallando todo en una tabla:

Persona	Horas trabajo	Precio por hora	Personas	Total, de puesto
Profesor evaluador	1	16,43€	3	49,29€
Tutor Pablo Muñoz	60	13,9€	1	834€
Tutor Óscar Adamuz	60	8,32€	1	499,2€
Alumno	320	6,73€	1	2153,6€
Totales	***443***	***7,98€***	***6***	***3535,09€***

Tabla 3.4: coste total humano.

Poniendo todo en conjunto, el coste total del proyecto se muestra en la siguiente tabla:

Tipo de coste	Coste total
Coste humano	3535,09€
Coste software	2259€
Coste hardware	1378,83€
Totales	***7172,92€***

Tabla 3.5: coste total humano.

Por tanto, el proyecto fuera de un marco de TFG, encargado a una empresa ascendería a 7172,92€.

Capítulo 4.-Estimación de recursos de computación en C-RAN: análisis teórico.

4.1.-Simulador de escenario 5G, Network Slicing

Para entender bien lo que se ha desarrollado en este trabajo es de imperiosa necesidad conocer cómo funciona el simulador del que se parte para la realización del mismo. En este capítulo se detallará cual es la estructura del simulador para comprender mejor de dónde vienen los resultados obtenidos antes de aplicar ninguna de las líneas de código del algoritmo.
Principalmente el cometido del simulador es definir en base a unos parametros de entrada que el usuario puede introducir, un escenario de una red 5G en la que se implementa el network slicing. En funcion de los parametros de configuración que se introduzcan el escenario variará en numero de estaciones base, macroceldas, usuarios, slices implementados, correlación del tráfico de los distintos slices y demas parametros importantes de la red.
Para entender el funcionamiento del simulador al completo se van a proporcionar explicaciones de cuáles son las principales tareas que se implementan en el simulador, detallando de que se trata cada una.
La primera tarea se puede definir como la **definición de los parámetros de configuración** del escenario, la cual tiene como principal cometido definir los parámetros generales de los que va a partir nuestra simulación como es el tráfico que genera cada usuario, el número de usuarios que vamos a introducir en el sistema, el número de iteraciones que va a durar la simulación, los parámetros referentes a las antenas que forman parte de la red de acceso radio (RAN), distinguiendo entre varios tipos de ellas y datos sobre cómo se reparte el tráfico en los distintos slices que intervienen, tales como la proporción de usuarios que toman el servicio que implementa cada uno de los slices.
La tarea que le sigue a la definición de la configuración es la **generación de la demanda del tráfico** se determinan todos los parámetros relativos a los usuarios, empezando por ubicarlos dentro del escenario para posteriormente ver cuál va a ser la BS a la que se conecten, se determina también el slice del que va a formar parte el tráfico que genere cada usuario. También puede determinarse si los usuarios pueden moverse, algo que por simplificar la tarea no se realiza en este trabajo.
Continua el trabajo desarrollado por el simulador con la **planificación de las celdas**, pues si bien la anterior trataba todos los parámetros relativos a los usuarios, esta hace lo propio con las estaciones base. Proporciona valores para modelar el comportamiento de las estaciones base tales como su localización y el número de canales que es capaz de montar cada estación base y se puede definir una estrategia para distribuirlas en el espacio en el que se desplegaría el escenario.
Para terminar el triunvirato de tareas dedicadas a un elemento en concreto de los que intervienen activamente en la simulación surge la **planificación de tenants**, que, por supuesto, determina los parámetros para los slices, principalmente la distribución de canales que tendrá cada uno y la correlación entre el tráfico de ambos.
Mencionar que el simulador da la opción de cara a hacer cálculos sobre los resultados de una simulación en un momento posterior o sobre una salida concreta del simulador de guardar en un archivo todas las variables generadas, crearlas de nuevo, o cargar una simulación previamente guardada a interés de cada usuario y de cuál sea su investigación.
Una vez la determinación de todos los parámetros relativos a usuarios, estaciones base y slices, empieza el despliegue del escenario generado.
Desplegar la red de acceso es la tarea que sigue, el script que la implementa es el encargado de

calcular y almacenar prácticamente todo lo que ocurre en nuestro escenario, calculará y guardará entre otros parámetros la BS a la que cada usuario se conecta en cada iteración de la ejecución, la eficiencia espectral de los mismos, la potencia recibida, el uso de los canales por parte de cada celda, si se satisfacen las necesidades de cada usuario y la carga ofrecida a cada slice y Estación Base entre otras muchas variables.
Una vez se ha desplegado la red de acceso, todos los indicadores sobre el rendimiento de la red quedan recogidos por un script que almacenará en las variables pertinentes los datos.
Para finalizar las tareas desarrolladas por el simulador, este proporciona como salida una serie de gráficos que se pueden recoger todos en el siguiente.

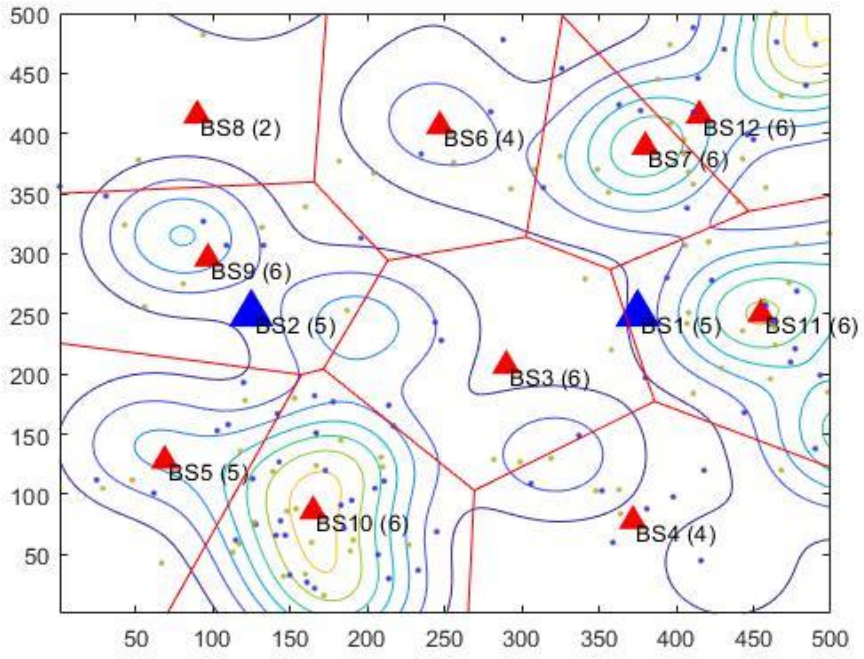


Figura 4.1: Escenario simulado con los dos slices.

Como puede ser observado en la figura 4.1 tenemos un escenario desplegado que consta de una serie de estaciones base denotadas por un triángulo rojo, las cuales corresponden a las estaciones base de las microceldas. Se observa también que el escenario tiene dos estaciones base de macroceldas que se designan mediante un triángulo azul.
Además de estos símbolos, se distinguen una serie de puntos en el escenario que representan la localización de los distintos usuarios en una instantánea. Se puede comprobar que hay dos tipos de usuario distinguidos por el color del punto que los representa. Esta separación se hace en base al slice al que pertenecen.
Se observa también un conjunto de líneas de nivel. Estas líneas se encargan de representar la concentración de los usuarios, siendo menor la separación entre líneas donde se tiene una concentración mayor y más laxa donde la concentración de los usuarios disminuye.
Finalmente distinguimos también una división del escenario en varios polígonos cuyo contorno se traza en color rojo. Estos polígonos representan las áreas de Voronoi del escenario, dando lugar a polígonos los cuales circunscriben a los puntos más cercanos a cada una de las estaciones base de microceldas.

4.2.-Introducción y descripción del algoritmo.

Este capítulo pretende describir primeramente cual es el problema que buscamos resolver mediante los algoritmos.

El objetivo último que se persigue es el conocer el número de procesadores que se necesitan en los centros de cómputo que designemos con el fin de poder procesar el tráfico generado en el tiempo que el protocolo HARQ nos marca.

Para el desarrollo de estos algoritmos se partirá de un escenario generado en el simulador en el cual una serie de 10 estaciones base, llamadas C3 a C12, cuyos recursos de cómputo se albergarán en dos centros de cómputo distinto llamados C1 y C2, darán servicio a 150 usuarios de dos tipos de slice distintos en una muestra de 10000 instantáneas. Adicionalmente a esto, se antoja oportuno esclarecer que los usuarios tendrán un throughput variable, no será el mismo para todos debido a que en uno de los algoritmos los usuarios se tratarán asignándoles un MCS individual. Para nuestros cálculos es necesario decir que, aunque tengamos dos centros de cómputo en las estaciones base de las macroceldas, estas no dejan de ser estaciones base y se estudiará también su comportamiento debido a su disposición para dar cobertura a los usuarios que se encuentren en unas condiciones de SINR relativa a estas estaciones base favorables para ser servidos por ellas, la SINR, es un parámetro que mide la calidad de la señal que se está recibiendo y que se puede calcular de la siguiente forma extraída de [29]:

$$SINR(u,r) = \frac{P_b^{RX}(d_{b,u})}{(\sum_{j\in B\setminus\{b\}} L_j \cdot \pi_j(r) \cdot P_j^{RX}(d_{j,u})) + P_N}$$

ECUACIÓN 4.1: SINR.

Donde:

- $P_b^{RX}(d_{b,u})$ es la potencia recibida a una distancia d de la small cell b.
- $d_{b,u}$ es la distancia entre las smallcell b y el usuario u.
- L_j es el factor de carga de la celda j.
- $\pi_j(r)$ es una función que toma el valor 1 o 0 en función de la smallcell a la que pertenece el trozo de espectro por el que se transmite la señal.
- P_N es la potencia de ruido recibida.

Este escenario puede considerarse real y por tanto los cálculos que se realicen sobre él deben ajustar a la realidad también.

Estos usuarios se distribuirán aleatoriamente por la superficie del escenario.

Figura 4.2: Diagrama de flujo del algoritmo.

Como se ve en la figura 4.2 para llegar a la frecuencia de cómputo hay varios pasos importantes antes. Se debe primeramente a fin de obtener el esquema de codificación y modulación. Este paso es muy importante, pues con este dato veremos cómo contribuye cada usuario y/o celda a la frecuencia total requerida.

Tras esto, se debe calcular el número de recursos físicos con los que cuenta cada celda. Para ello necesitamos el número de canales disponibles en cada una de las celdas y el número de PRB disponibles en cada canal de cada celda. Un PRB es cada uno de los bloques en los que se dividen los recursos físicos que hay disponibles en el sistema. Estos se distribuyen en canales con un ancho de banda determinado y dependiendo de ese ancho de banda y la numerología utilizada tendremos una cantidad u otra de PRBs.

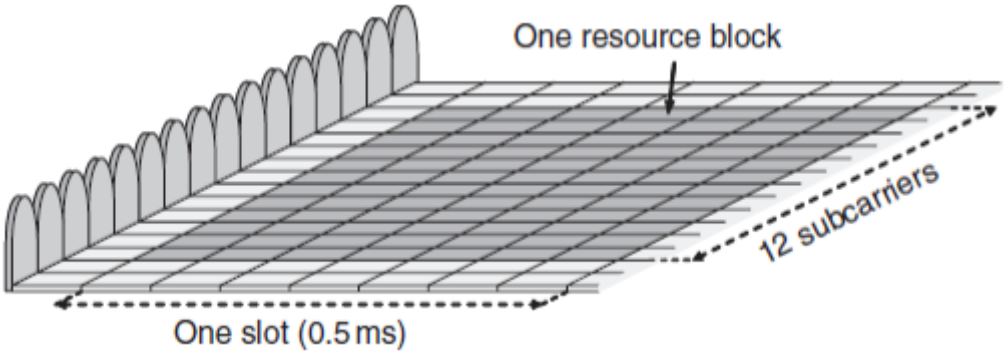


Figura 4.3: Definición de PRB.[25]

Si atendemos a la figura 4.3, la cual representa las subportadoras que componen cada PRB, podemos ver como cada bloque de recursos viene de utilizar durante una división de un slot una de estas portadoras. Además, se utilizan 7 símbolos dando lugar a 84 recursos donde se pueden introducir los símbolos de las distintas constelaciones.

El algoritmo utilizado se introduce en [2]. Dichos PRB están distribuidos en canales con un ancho de banda determinado y en función de ese ancho de banda tendremos una cantidad determinada de bloques. El PRB es la mínima unidad de información, la mínima división dentro de los recursos físicos que una estación radio puede asignar a un usuario. Esto es, el ancho de banda mínimo que se puede asignar a un usuario. Cada PRB tiene una asignación de 180 kHz en frecuencia. Y están distanciados 15 kHz entre cada uno si la numerología es 0. Además de esto debemos tener en cuenta también la existencia de la banda de guarda, la cual se va a considerar simétrica. Así pues, para realizar el cálculo de los PRB por canal seguiremos la siguiente formula:

$$N_{PRB} = \frac{BW_{Channel} - 2 * BW_{GuardBand}}{BW_{PRB}}$$

ECUACIÓN 4.2: NUMERO DE PRB.[26]

Donde:

- $BW_{Channel}$ es el ancho de banda del canal para el cual estamos calculando los PRB.
- $BW_{GuardBand}$ es el ancho de banda de la banda de guarda.
- BW_{PRB} es la asignación en frecuencia a un PRB.

En nuestro escenario ya se ha especificado que al usar numerología 0 el ancho de banda de un PRB es de 180kHz. LA banda de guarda mínima con esta configuración es de 692.5kHz[26]. Tomando estos datos se obtiene que el ancho de banda posible para cada canal es de 50MHz[26] y por tanto el número asignable de PRB en cada canal es de:

$$N_{PRB} = \frac{50MHz - 2 * 692.5kHz}{180kHz} = 270\ PRB\ por\ canal.$$

ECUACIÓN 4.3: NUMERO DE PRB POR CANAL.

Otro aspecto que cobra vital importancia a la hora de determinar la frecuencia requerida para procesar todo el tráfico a tiempo es el índice de modulación y codificación de un usuario, su MCS. El MCS es utilizado a fin de aumentar la eficiencia espectral bajo la condición de cumplir que el Block Error Rate (BLER) sea inferior al 10%. Define el esquema de codificación y modulación que utiliza cada usuario. Si ese usuario utiliza una constelación de orden mayor, su eficiencia espectral aumentará debido a que está enviando más datos con el mismo ancho de banda, sin embargo, la cercanía entre los puntos de la constelación hará que sean más sensibles a ruidos e interferencias las constelaciones más grandes. A fin de evitar esto se selecciona el MCS que garantice que el BLER esté por debajo del 10% y proporcione al usuario la máxima eficiencia espectral. La influencia del MCS en el tiempo de procesado es directa, cuanto mayor sea el MCS que tiene asignado un usuario

mayor será el tiempo que se emplee para procesar sus mensajes.

MCS Index I_{MCS}	Modulation Order Q_m	Target code Rate x [1024] R	Spectral efficiency
0	2	120	0.2344
1	2	193	0.377
2	2	308	0.6016
3	2	449	0.877
4	2	602	1.1758
5	4	378	1.4766
6	4	434	1.6953
7	4	490	1.9141
8	4	553	2.1602
9	4	616	2.4063
10	4	658	2.5703
11	6	466	2.7305
12	6	517	3.0293
13	6	567	3.3223
14	6	616	3.6094
15	6	666	3.9023
16	6	719	4.2129
17	6	772	4.5234
18	6	822	4.8164
19	6	873	5.1152
20	8	682.5	5.332
21	8	711	5.5547
22	8	754	5.8906
23	8	797	6.2266
24	8	841	6.5703
25	8	885	6.9141
26	8	916.5	7.1602
27	8	948	7.4063
28	2	reserved	
29	4	reserved	
30	6	reserved	
31	8	reserved	

Tabla 4.1: Tabla para mapping entre eficiencia espectral y mcs.[27]

En la tabla 4.1 podemos ver detallado como podemos llegar a determinar la constelación asignada a un usuario en función de su eficiencia espectral conseguida a través del orden de modulación que puede utilizar.

La determinación de ese orden de modulación viene mapeada atendiendo a entre que valores se sitúan los órdenes de modulación en la tabla 4.1, valiéndonos de que, si hay varios valores a los que se le asigna un determinado orden de modulación, si el valor que se evalúa esta entre el más bajo y el más alto pertenecientes a un mismo orden de modulación, se le asignará ese orden.

Para verlo más claro pongamos un ejemplo: supongamos que la celda 2 tiene una eficiencia

espectral media de 1,8. Si miramos la tabla 4.1, le correspondería el orden 4 de modulación pues el valor 1,8 se encuentra dentro de las fronteras inferior y superior asignadas a dicho orden de modulación, que en este caso serían los valores comprendidos entre 1,4766 y 2,7305, pues a partir de este se les asignaría el orden 6 de modulación.
Una vez se averigua cual es el orden de modulación, resulta casi trivial descubrir cual sería el esquema de modulación utilizado para la celda. Tratándose de un sistema cuyos símbolos están compuestos por n datos binarios, siendo n el número de orden de modulación, basta con elevar 2 a la n-esima potencia para descubrir el tipo de modulación que se obtiene para la celda.
De ese modo, atendiendo a la tabla 4.1, las distintas celdas usaran un esquema de modulación para sus mensajes atendiendo al criterio que se expone en la siguiente tabla:

Orden de modulación de la celda	Esquema de modulación asignado
2	QPSK
4	16-QAM
6	64QAM
8	256QAM

Tabla 4.2: Equivalencias orden-esquema de modulación.

Por descontado sabemos, que en aquellas celdas donde la eficiencia espectral sea de un nivel más alto será de recibo tener una SINR más alta, dando esto lugar a una mayor calidad de señal y por tanto esa transmisión goce de mayor seguridad que una en la que la relación señal a interferencia y ruido sea menor. Así pues, sabemos que cuanto mayor sea el orden de modulación, más símbolos vamos a poder representar y mayor va a ser la propia seguridad de nuestra red, que empieza aquí, en el esquema de modulación escogido. Aportará por tanto una mayor robustez a la transmisión de datos una modulación de orden mayor y será en ellas donde podremos también asignar un MCS mayor.
Esta asignación del MCS se traducirá a posteriori en una necesidad de frecuencia mayor para el procesado de los datos a la hora de tener que cumplir una frontera en el campo temporal.
En el mapeado de la eficiencia al MCS, el cual es el más importante a la hora de hacer el cálculo de la frecuencia de procesado necesaria, vamos a utilizar de nuevo la tabla 5.1, pues con ella tenemos un mapeo claro y conciso de que MCS se debe asignar a una eficiencia espectral dada.
Traducido al código del proyecto, esto se desarrolla mediante un bucle que llama recurrentemente a la función encargada de mapear el valor de la eficiencia espectral a un MCS, la cual atendiendo al valor que se le indique como eficiencia espectral devuelve el MCS que le corresponde atendiendo al siguiente criterio:
Se asignará a una celda aquel MCS que en la tabla tenga asignada una eficiencia espectral que de entre todas las albergadas en dicha tabla, sea la inmediata inferior a la que se indique.
Esto es, poniendo un ejemplo, si la celda tiene una eficiencia de 3,4785, mirando la tabla buscamos el MCS cuya eficiencia sea inmediatamente inferior a la que tenemos. En este caso corresponde con el MCS 13, por tanto, se le asignaría este MCS a la celda.
La función encargada de este mapeo tiene como único parámetro de entrada el dato de la eficiencia espectral y mediante una simple comprobación de en qué intervalo se encuentra la eficiencia que se le está pasando como parámetro, devuelve el MCS que le correspondería.

Por supuesto la importancia de estos factores queda eclipsada para el cálculo del tiempo por la frecuencia de cómputo que podemos llegar a lograr en los centros de cómputo. Aun siendo un factor clave para el cálculo del tiempo que tardaríamos en procesar el tráfico de los usuarios que tenemos en el sistema, no es la meta utilizar esta como una variable en un principio, sino que más bien el objetivo que perseguimos es el cálculo de ésta para poder garantizar una velocidad de cálculo suficiente para agregar todo el tráfico dentro de los límites que impone el HARQ. En concreto para nuestro escenario, y teniendo en cuenta las características del simulador, el cual

está preparado para modelar con fidelidad lo que es el tráfico de enlace descendente (DL), los cálculos ofrecidos y expuestos se centrarán en esta parte del tráfico.

El algoritmo puede utilizarse no solo para el cálculo del tiempo de procesamiento de tráfico, sino también para calcular otros retardos como son el de codificación y acomodación de los datos a transmitir en la señal banda base. Estos dos elementos son los que se ha demostrado en distintas investigaciones que comprometen más el sistema en lo que a tiempo se refiere, pues son las dos operaciones que requieren de más cómputo y las que más tienen que decir por tanto en lo que se refiere a recursos para el sistema en global. No es ese el punto en este trabajo, pero se menciona con carácter informativo para el lector.
El algoritmo, de acuerdo con los autores de [2], es una aproximación lineal de los cálculos reales del tiempo para el procesado, los cuales son logarítmicos y complejos en el tiempo, asimismo afirman que la aproximación lineal propuesta presenta diferencias despreciables respecto de ese algoritmo original.
Dada la versatilidad de aproximación de este algoritmo, se le incorpora también un coeficiente polinómico que cambiará en función del tiempo que queramos calcular, si es el de procesamiento como persigue este trabajo o si por ejemplo es el tiempo de codificación y transformación en frecuencia como menciona el párrafo anterior.

La siguiente ecuación presenta dicha aproximación lineal.

$$\tau_{[\mu s]} = \frac{N_{PRB}}{f^2_{[GHz]}} \sum_{i=0}^{2} \alpha_i \, I^i_{MCS}$$

ECUACIÓN 4.4: TIEMPO DE PROCESADO.

En la ecuación anterior vemos los diferentes elementos que intervienen en el algoritmo utilizado:

- $\mathbf{N_{PRB}}$ representa el número de PRB.
- f(GHz) es la frecuencia de cómputo que tenemos en giga Hertzios.
- I_{MCS} es el índice de modulación y codificación asignado.
- α_i es el coeficiente polinómico que variará su rango de valores en función del tipo de procesado, DL o UL.

Como la gran mayoría de los descubrimientos que se puedan hacer hoy en día, este trabajo ya contaba con antecedentes en la investigación de temas muy similares. De hecho, artículos de esta índole son los que han servido de referencia a la hora de extraer maneras de calcular, información útil para construir el modelo y ver dónde están los puntos clave a la hora de poner limitaciones al mismo.
Para nuestros cálculos, así como para todos aquellos a los que es capaz de dar resultado el algoritmo podemos ver cuál es el valor de esos coeficientes polinómicos en la tabla a continuación.

	α_0	α_1	α_2
T_{proc}^{DL}	32.583	1.072	0.03
T_{proc}^{UL}	35.545	1.623	0.086

Tabla 4.3: Coeficientes polinómicos del tiempo de procesado.[2]

De dicha tabla se puede obtener la inmediata conclusión de que el proceso que contribuirá en mayor medida al cálculo de la frecuencia es el tiempo para el procesamiento en canal ascendente. Esto se sustenta y corrobora su sentido si atendemos al límite HARQ donde el límite es más flexible con el procesamiento del canal descendente que con el del canal ascendente pues el tiempo límite

asignado al canal descendente asciende al doble que el asignado al ascendente.
Es aquí donde se estima oportuno detallar cuales han sido esos puntos clave donde se han fijado los requisitos a cumplir por el modelo.
Como antes se ha mencionado en varias ocasiones, el límite más insalvable es el que establece el retardo **HARQ.**
Para hacer una simplificación, dado que nuestro simulador está fuertemente preparado para el tráfico descendente, vamos a focalizar este punto hacia ese tráfico DOWNLINK.
En esta sección se llevará a cabo el desarrollo matemático del algoritmo para derivarlo y adaptarlo a lo que este trabajo investiga.
La restricción que HARQ impone consiste en que una vez es enviado, el mensaje debe ser contestado con un ACK/NACK dentro de la restricción temporal que HARQ nos impone y que se explicó en la **sección 2.5**.
No obstante, a estos 3 ms que tenemos para procesar, 2 en el caso de DOWNLINK, debemos sustraerle otro retardo, o añadir otra condición más a cumplir.
Dicha condición es el **retardo de transporte FrontHaul**, en adelante T_{fh} que no es más que el tiempo que tardan los mensajes en viajar por el enlace físico entre el centro de cómputo y la estación base.
Tomando como base lo que se menciona en [15], donde se trata de explicar todos los retardos existentes en el HARQ, introduciendo el transporte FrontHaul, vemos como a este tiempo de procesamiento debemos sustraerle el doble del tfh. La respuesta al por qué de sustraer el doble reside en el camino recorrido por el mensaje enviado desde la BBU al usuario y por el ACK/NACK devuelto a la BBU.

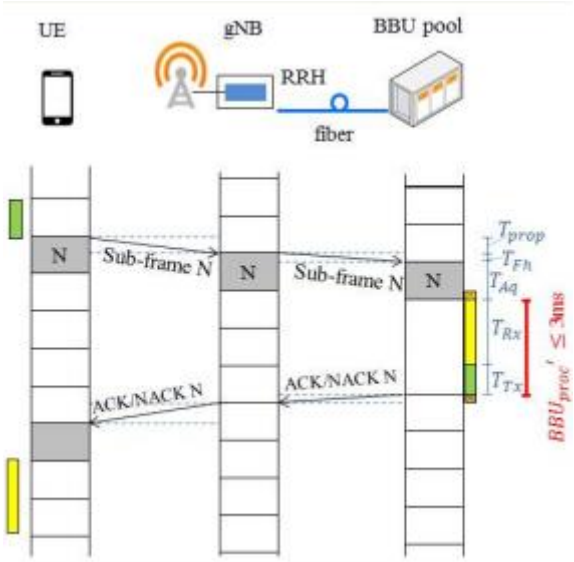


Figura 4.4: Retardo de transporte FrontHaul.[15]

En la figura 4.4 se aprecia claramente lo mencionado en el párrafo anterior. De esta forma el tiempo restante para procesar puede calcularse de forma sencilla haciendo uso de la siguiente ecuación:

$$T_{Proc\ restante} = T_{Proc\ HARQ} - 2 * Tfh$$

ECUACIÓN 4.5: CÁLCULO DEL TIEMPO REAL DE PROCESAMIENTO.

Asimismo, para calcular T_{fh} se hace uso de la siguiente ecuación:

$$Tfh = \frac{D}{c}$$

ECUACIÓN 4.6: CÁLCULO DEL RETARDO DE FRONTHAUL.

Donde:

- D representa la distancia que separa el centro de cómputo de la estación base.
- c representa la velocidad de la luz en el medio físico que interconecta la estación base y el centro de cómputo.

Posteriormente se evaluará el comportamiento y la influencia del Tfh en la frecuencia requerida y en los tiempos reales de procesamiento.
Dado que se antoja de una importancia mayor y puesto que los resultados que arroja son más veraces y representativos, los cálculos que se exponen en este capítulo se utilizan también para realizar los cálculos relativos a las estaciones base de las macroceldas. Pues no considerarlas sería un error catastrófico y probablemente haría que los cálculos fueran erróneos si no se las tuviera en cuenta a la hora de contabilizar la contribución de cada una de las celdas que componen el escenario simulado.
Primeramente, se estima oportuno informar de algunas consideraciones que se han tomado a la hora de construir el modelo. Tales consideraciones son:

- Todos los usuarios serán tratados por igual y los recursos radio se distribuirán sin otorgar privilegios a ninguno de ellos.
- El medio físico existente entre el usuario y la estación base es el aire.
- Por su parte el medio físico entre la estación base y el centro de cómputo es fibra óptica.
- Las distancias calculadas se expresan en metros, así como las coordenadas de los ejes.

4.3.-Algoritmo para el MCS medio de las celdas

Para tener una aproximación inicial al resultado de la frecuencia requerida para procesar el tráfico de los dos slices sin distinguir entre ellos se extrajeron varios datos útiles del simulador. En primera instancia se toma la eficiencia espectral de cada celda, calculada como la media de entre todas las eficiencias espectrales asociadas a cada uno de los usuarios conectados a cada BS.
A partir de este dato de la eficiencia espectral haciendo uso de la tabla 4.1, se realiza la función $f_{\overline{SE}\to\bar{I}_{MCS}}()$ con lo que se obtiene lo que se haya llamado **MCS medio de la celda**, y a partir del cual se asigna una modulación única para todos los usuarios conectados a una BS.
Para el cálculo de los PRB se toman los PRB disponibles en cada canal de acuerdo a la ecuación 4.2. A partir de ahí, tomamos otro valor de salida del simulador que es los canales que utiliza cada unidad de radio. Se quiere aclarar que en la simulación se toma el ancho de banda de cada canal en 50 MHz y por tanto lógicamente atendiendo a la ecuación que asigna un numero de PRB a cada ancho de banda de canal tendremos 270 PRB por canal. Por tanto, el número de PRB por celda disponibles para asignar será:

$$N_{PRB-celda} = \sum_{i=n}^{i} PRB_n$$

ECUACIÓN 4.7: CÁLCULO DE LOS PRB DISPONIBLES EN UNA CELDA.

Donde:

- i representa el número del canal activo de la celda, para una celda tendremos varios canales activos.
- n representa el primero de los canales activos de la celda.
- PRB_n hace referencia a los PRB que tiene cada canal, en nuestro caso todos tienen 270 y por tanto podemos escribir la ecuación 5.4 como sigue:

$$N_{PRB-celda} = 270 * N_C$$

ECUACIÓN 4.8: CÁLCULO DE LOS PRB DISPONIBLES EN UNA CELDA SIMPLIFICADO.

Donde solo tenemos una nueva variable que es N_c que representa el número de canales activos en la celda.

Continuando con el detalle de cómo se extrae cada valor relevante para el cálculo que nos concierne, que es el de la frecuencia, necesitamos determinar un tiempo a cumplir para que nos dé una barrera inferior la cual no podremos rebasar a la baja si queremos que el tiempo determinado por HARQ se cumpla. En esta primera implementación del algoritmo se toma el valor de ese tiempo en 3ms basándonos en lo descrito en [2].

Sin embargo, ese tiempo de 3ms está contemplado para tramitar todo el procesado del tráfico tanto descendente como ascendente. Si miramos detalladamente, encontramos en [4] que esos 3ms que se mencionan están divididos en 2ms para el proceso del tráfico DL y 1ms para el tráfico UL. Por tanto, encontramos que el algoritmo se divide en dos partes como sigue:

$$\tau_{[\mu s]} = \frac{N_{PRB}}{f^2_{[GHz]}} \sum_{i=0}^{2} (\alpha_{i-DL} + \alpha_{i-UL})\, I^i_{MCS}$$

ECUACIÓN 4.9: TIEMPO DE PROCESADO EN AMBOS SENTIDOS.

Y dado que para el enlace descendente tendremos 2 ms y para el ascendente solo dispondremos de 1ms para procesar el tráfico podemos escribir:

$$\tau_{DL[\mu s]} = 2 * 10^3[\mu s] - 2 * Tfh = \frac{N_{PRB}}{f^2_{[GHz]}} \sum_{i=0}^{2} \alpha_{i-DL} * I^i_{MCS}$$

ECUACIÓN 4.10: FRONTERA PARA LA FRECUENCIA DOWN LINK.

$$\tau_{UL[\mu s]} = 1 * 10^3[\mu s] - 2 * Tfh = \frac{N_{PRB}}{f^2_{[GHz]}} \sum_{i=0}^{2} \alpha_{i-UL} * I^i_{MCS}$$

ECUACIÓN 4.11: FRONTERA PARA LA FRECUENCIA UPLINK.

Haciendo entonces uso de 4.5 podemos despejar la frecuencia que vamos a calcular de una manera sencilla.

$$f_{[GHz]} = \sqrt{\frac{N_{PRB}}{\tau_{[\mu s]}} \sum_{i=0}^{2} (\alpha_{i-DL} + \alpha_{i-UL})\, I^i_{MCS}}$$

ECUACIÓN 4.12: CALCULO DE LA FRECUENCIA DE CÓMPUTO.

Con lo cual es fácil calcular la frecuencia para un tiempo dado. Dado que las restricciones son distintas para el UL y el DL se opta por llevar a cabo los cálculos separadamente.

Se cree conveniente en este punto explicar que el simulador implementado para este trabajo está

diseñado para calcular parámetros de Down link únicamente. Así pues, los cálculos que se implementan en este trabajo se centraran en calcular resultados para este algoritmo únicamente en el enlace descendente.
De este modo podemos seguir con la transformación del algoritmo amoldándolo a nuestro requerimiento.
Si tenemos en cuenta lo comentado en la introducción acerca del simulador y su predisposición para el tráfico descendente:

$$f_{c[GHz]} = \sqrt{\frac{N_{PRB}}{\tau_{DL[\mu s]}} \left(\sum_{i=0}^{2} ((\alpha_{iT-DL}) + (\alpha_{iFFT-DL}) + (\alpha_{iEnc-DL}) + (\alpha_{iMod-DL})) * I_{MCS}^{i} \right)}$$

ECUACIÓN 4.13: CALCULO DE LA FRECUENCIA DE CÓMPUTO PARA DOWN LINK.

Donde:

- α_{iT-DL} es el coeficiente aplicado al tiempo puro de procesado del trafico.
- $\alpha_{iFFT-DL}$ es el coeficiente aplicado para el tiempo invertido en la transformada en frecuencia.
- $\alpha_{iEnc-DL}$ se trata del coeficiente que aplica para el tiempo de codificación.
- $\alpha_{iMod-DL}$ por ultimo, es el coeficiente que interviene en el tiempo de modular la señal utilizada para transmitir el tráfico.

Estas descomposiciones se extraen de [2], concretamente en la sección IV: Model For Computational Resources, que pueden articularse como se ha hecho, descomponiendo el coeficiente polinómico de la ecuación 5.3 en los que se han desarrollado sucesivamente.

Por último, para tener en cuenta el número de usuarios que están conectados a cada estación base, se introduce un factor igual al número de usuarios conectados a la estación base. Por tanto, de forma definitiva para el cálculo de la frecuencia requerida para cada celda tendremos:

$$f_{c[GHz]} = \sqrt{\frac{N_{PRB} * N_{Usuarios-bs}}{\tau_{DL[\mu s]}} \left(\begin{array}{l} \sum_{i=0}^{2} (\alpha_{iT-DL}) + \sum_{i=0}^{2} (\alpha_{iFFT-DL}) \\ + \sum_{i=0}^{2} (\alpha_{iEnc-DL}) + \sum_{i=0}^{2} (\alpha_{iMod-DL}) \end{array} \right) * I_{MCS}^{i}}$$

ECUACIÓN 4.14: CALCULO FINAL DE LA FRECUENCIA DE CÓMPUTO PARA DOWN LINK.

4.4.-Algoritmo para el MCS de los usuarios.

Implementado el cálculo que considera el MCS medio de la celda, se pensó en sustituir este cálculo por un agregado total que tenga en cuenta a todos los usuarios individualmente.
De nuevo para este cálculo partimos de nuevo de la ecuación 4.1, a partir de ella podemos seguir las mismas transformaciones hasta la ecuación 4.10. Partiendo de ahí, a continuación, se muestran las modificaciones introducidas buscando que el cálculo sea efectivo también cuando se considera individualmente a cada usuario.
Al individualizar para cada usuario necesitamos que, en lugar de multiplicar por el número de usuarios, veamos la contribución individual de cada usuario, por tanto, resulta conveniente introducir un nuevo sumatorio englobando aquello que es individual para cada usuario, que en nuestro caso es el número de PRB que se le asigna a cada usuario y su MCS. Aplicando esto podemos modificar la ecuación y llegar a lo siguiente:

$$f_{[GHz]} = \sqrt{\frac{1}{\tau_{DL[\mu s]}} \sum_{u=0}^{N_{Usuarios-bs}-1} \left(PRB_u \left(\sum_{i=0}^{2} \begin{pmatrix} (\alpha_{iT-DL}) + (\alpha_{iFFT-DL}) + \\ (\alpha_{iEnc-DL}) + (\alpha_{iMod-DL}) \end{pmatrix} * I^i_{MCSu} \right) \right)}$$

ECUACIÓN 4.15: INDIVIDUALIZACIÓN DEL CÁLCULO.

Donde:

- PRB_u son los PRB asignados al usuario u dentro de la celda en la que se encuentra.

Como vemos ahora tenemos I_{MCSu}, que representa el índice de modulación y codificación asignado a cada usuario mediante el mapeo explicado con anterioridad en función de la eficiencia espectral individual.
Tenemos entonces un cálculo donde se consigue un aislamiento entre usuarios para posteriormente agregar el tráfico que todos producen para ver el requerimiento de frecuencia para satisfacer por separado a todos y cada uno de los usuarios que tenemos en cada celda.
Se propone aplicar una propiedad de los sumatorios, la cual podemos enunciar como *"la suma de todas las partes en las que se divide una cantidad determinada da como resultado esa misma cantidad"*, esto es aplicable siempre que la cantidad no dependa de la iteración del sumatorio.

Traducido al lenguaje matemático:

$$\sum_{i=1}^{n} x = n * x$$

ECUACIÓN 4.16: PROPIEDAD DEL SUMATORIO.

Una vez tenemos esto en cuenta, podemos sacar del primer sumatorio el número de PRB asignados a cada usuario, puesto que el reparto de PRB se hace uniformemente entre los usuarios que se conectan a la celda. Resultando esto en la ecuación siguiente, y final, para este segundo calculo.

$$f_{c[GHz]} = \sqrt{\frac{N_{PRB-celda}}{\tau_{DL[\mu s]}} * \sum_{u=0}^{N_{Usuarios-bs}-1} \left(\left(\sum_{i=0}^{2} \begin{pmatrix} (\alpha_{iT-DL}) + (\alpha_{iFFT-DL}) + \\ (\alpha_{iEnc-DL}) + (\alpha_{iMod-DL}) \end{pmatrix} * I^i_{MCSu} \right) \right)}$$

ECUACIÓN 4.17: ECUACIÓN FINAL PARA EL CÁLCULO INDIVIDUALIZADO EN DOWN LINK.

En el siguiente epígrafe se procede a una comparación cualitativa entre las dos formas de calcular la frecuencia de cómputo que requerirá una celda.

4.5.-Configuración del método de estimación de la frecuencia de cómputo.

Para terminar este quinto capítulo de este TFG se va a proceder a esclarecer el criterio de asignación de un valor a cada variable de las que interfieren en nuestros cálculos independientemente de si nos referimos al cálculo con el MCS medio o si nos referimos al cálculo individualizado.

Empezaremos por el **tiempo total de procesado**. Dado que estamos hablando que nuestro simulador está preparado para cálculos de enlace descendente, atendiendo a lo que HARQ impone tendremos un límite superior de 2ms.
El **MCS** se plantea de dos formas distintas dependiendo del tipo de calculo que implementemos.

Así pues, en el cálculo general se tendrán en cuenta todos los usuarios sin excepción conectados a la estación base. En contraposición encontramos el caso de tener en cuenta individualmente el MCS de los usuarios. En ese caso se tratará de forma especial a aquellos que dada su baja eficiencia espectral se les asigne el MCS 0. Estos usuarios no serán tenidos en cuenta a la hora de hacer el cálculo pues no se admitirán en la celda y por tanto su tráfico ni siquiera se procesará.
El número de **PRB** viene calculado teniendo en cuenta el número de canales disponible para cada celda como se explicó en la **sección 4.2**. No obstante se debe de tomar en cuenta que los PRB deben ser asignados en fracción entera y dicho número de PRB asignado a un usuario debe ser múltiplo de dos [28] debido a que este criterio viene por la configuración de reserva de recursos usando como referencia 3GPP TS 38.214.
Consecuencia inmediata de esto es el hecho de que probablemente no se asignen todos los PRB disponibles en una celda. Para el caso en el que calculamos utilizando el MCS medio de la celda, también se calcularan los PRB que realmente podemos utilizar para cumplir el criterio antes expuesto. En caso de optar por el cálculo individualizado encontraremos que resulta innecesario asignar recursos a un usuario que tiene un MCS asignado igual a 0, pues dado que su tráfico no se va a procesar ni cursar no se estarían utilizando eficientemente los recursos que tenemos en la red, algo inadmisible en el contexto en el que nos encontramos donde necesitamos la mayor rapidez posible de procesado de cara a dar una QoS propia de una red 5G.
Finalmente, los coeficientes polinómicos que intervienen en cada una de las iteraciones de los sumatorios son valores constantes y que complementarán a los de la tabla 4.3.

	α_0	α_1	α_2
T_{FFT}^{DL}	7.609	0	0
T_{ENC}^{DL}	3.907	0.773	0.027
T_{MOD}^{DL}	13.851	0.133	0.002
T_{FFT}^{UL}	6.957	0	0
T_{ENC}^{UL}	10.512	1.631	0.083
T_{MOD}^{UL}	17.494	-0.08	0.006

Tabla 4.4: Valores de los coeficientes polinómicos, complementaria a 4.3. [2]

Para entender cómo se utilizan estos valores dentro de la ecuación 4.17, el subíndice que acompaña al coeficiente, hace referencia a la iteración del sumatorio. Dichos valores son constantes para cualquiera de los cálculos, ya sea el cálculo que utiliza el MCS medio de cada celda o el que utiliza el MCS individual de cada usuario. Huelga decir que, aunque en la tabla se presenten los valores de los coeficientes para el enlace uplink, estos no se utilizarán.

Capítulo 5.- Descripción e implementación de los algoritmos.

En este capítulo del trabajo, se va a exponer de manera detallada como están implementadas las dos versiones del algoritmo explicadas anteriormente.

5.1.-Estructura de ficheros

Como se detalló anteriormente, para este trabajo partimos de que el simulador de la red, es decir, el código que simula el despliegue de red viene ya dado, y ha sido proporcionado por los tutores. Sin embargo, conviene hacer inventario de las variables que cobran vital importancia de cara a construir en este trabajo un código que construya resultados a partir del despliegue que genera el simulador. Podríamos abstraer el simulador a una caja negra que nos proporciona unas herramientas, las variables, las cuales extraemos selectamente para el algoritmo que implementamos:

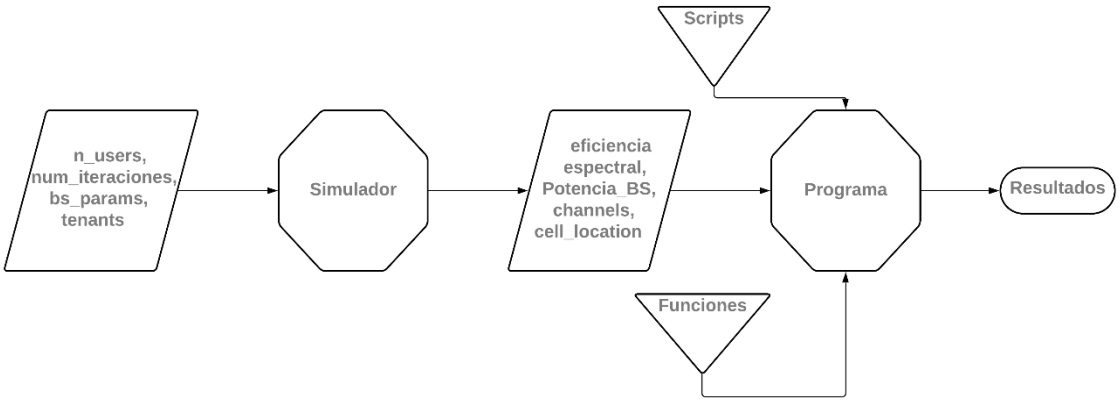


Figura 5.1: Diagrama de Bloques de la implementación.

El sistema de la implementación se compone de cinco bloques básicos. El primer y segundo bloque se han descrito en el capítulo anterior y se han detallado cuales son las tareas que cumple el segundo bloque concretamente, el del simulador.
El tercer bloque que observamos son las variables de salida del simulador que consideramos útiles en la implementación de este proyecto. Tales variables se agrupan principalmente en parámetros de las estaciones base tales como la potencia, los canales que montan, su localización y el tipo de estación base, así como la estación base que en cada una de las instantáneas está dando servicio a cada uno de los usuarios que hay en el sistema.
El cuarto bloque es el que se ha llamado Programa, y como vemos tiene dos entradas adicionales que son los scripts y las funciones que se han desarrollado para que en conjunto formen ese bloque. En este capítulo se va a describir el funcionamiento de dicho bloque, explicando al detalle las tareas que desempeñan los scripts y funciones que lo componen.
El quinto bloque es el de resultados, al que más adelante se le dedicará un capítulo en exclusividad.

5.2.-Desarrollo del cálculo.

Esta sección se centrará en explicar detalladamente que es lo que desarrolla el cuarto bloque de la figura 5.1. Dicho bloque a su vez vemos como se ha formado de la integración de otros dos que le sirven como entrada que son las funciones y los scripts que componen la implementación de

dichos cálculos. En el siguiente diagrama se muestra el desglose de los cometidos principales de este bloque.

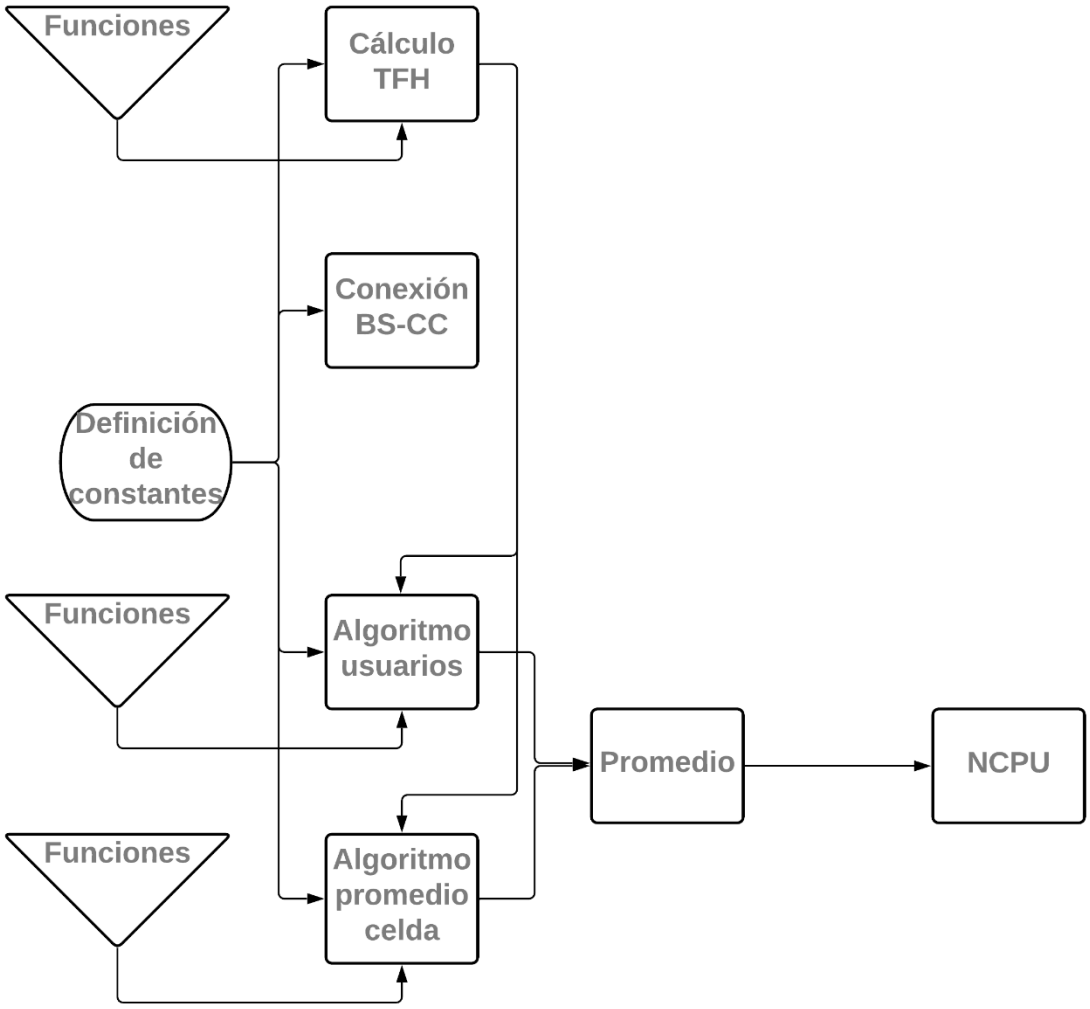


Figura 5.2: Diagrama de Bloques de la implementación.

Como vemos tenemos una serie de bloques cada uno de los cuales representa una tarea llevada a cabo por el programa que implementa el cálculo.
El primer bloque es el **Definición de constantes**, este bloque se encarga de definir en un script todas las variables que van a ser comunes en las dos implementaciones realizadas del algoritmo. Desde inicializar vectores con datos comunes hasta definir los coeficientes polinómicos que forman parte de la sumatoria presente en el algoritmo, pasando por el cálculo del número de PRB que hay disponibles entre todos los canales que tienen disponibles cada una de las celdas.
La siguiente tarea que desarrolla el bloque es la de **designar a que centro de cómputo se va a conectar cada estación base**, esto es, donde vamos a colocar la unidad radio (RU) y donde vamos a colocar las BBU donde albergaremos las CU y DU. El criterio al que se atiende para esta conexión esta desarrollado en profundidad en el **Anexo I** de esta memoria.
La ejecución del programa continua con el desarrollo del **cálculo del Tfh**. Se quiere aclarar que debido a que las estaciones base, sean de microceldas como de macroceldas, mantienen su posición en todas las instantáneas en las que se ha ejecutado el bloque del simulador y por tanto esta tarea solo requiere de una única ejecución pues los resultados arrojados una vez concluida siempre serán idénticos. Para esta tarea, el programa hace uso de una función sencilla para calcular la distancia entre las estaciones base y los centros de cómputo, calculando a partir de ahí el retardo hasta el centro de cómputo desde cada una de las estaciones base conectadas a cada

uno de estos centros.
Para llevar a cabo la tarea del **cálculo para usuarios**, se definió previamente en la tarea de definir los datos constantes un par de matrices tridimensionales en las que se van a guardar los datos de frecuencia de cómputo y tiempo de procesado requerido en función de cuál sea la variable sobre la que realizamos un barrido para sacar el máximo de datos útiles en relación a estas dos variables.
Tal y como ocurre para esta tarea, ocurre para el **cálculo para el promedio de las celdas**, estas tareas difieren en la implementación, pero el cometido es muy similar pues ambas buscan aportar un dato de la frecuencia requerida para conseguir un determinado tiempo de procesado y otro para el tiempo que conseguimos con cada frecuencia sobre la que hacemos el barrido.
La penúltima de las tareas que se realizan dentro del programa es el **promedio entre todas las instantáneas** en las que se ha ejecutado el bloque del simulador para posteriormente obtener un dato relativo al número de procesadores que necesitaríamos en cada uno de los centros de cómputo, atendiendo al criterio de conexión que se ha definido en el ya mencionado Anexo I.

5.3.-Consideraciones previas a las implementaciones.

Llegados a este punto, la importancia recae en exponer y esclarecer al lector como ha sido implementado este proyecto. Para ello se explica en este capítulo de manera detallada y paso por paso como intervienen los distintos factores en el código escrito, mostrando como se ha calculado cada parámetro y qué lugar ocupa en la cadena de cálculo cada elemento.
Este capítulo se divide en tres secciones principales. Como se viene dejando claro el estudio se ha hecho de las dos formas que se mencionan, con un cálculo generalizado para la celda y con un cálculo específico para cada usuario, además a estas dos implementaciones se les suma la implementación de mejoras consideradas para unos resultados más exactos por parte del algoritmo.
Definir los parámetros constantes es una tarea de gran importancia de cara a la eficiencia de los cálculos, pues dicha definición va a ahorrar tiempo de ejecución a los scripts principales que darán un resultado representativo y significativo. El script encargado de la definición de los datos constantes define una serie de datos que son comunes a ambos algoritmos, con esto conseguimos aislar aun más el desarrollo de cada uno de los algoritmos.
En estas líneas se creará un vector de tres posiciones que contiene el valor de cada uno de los coeficientes polinómicos que se utilizan en el sumatorio.

A continuación, se define la frecuencia mínima que debe tener cada CPU:

$$f_{min-CPU} = 2.7GHz$$

ECUACIÓN 5.1: CÁLCULO DE LOS PRB DE CADA CELDA.

El valor de 2.7GHz se extrae de [2], el hecho de que se creen dos vectores equiespaciados, uno para frecuencia y otro para tiempo se debe a que posteriormente se hará un barrido y se calculará tanto las frecuencias con las que se obtienen determinados tiempos de procesamiento como los tiempos que se consiguen con cada frecuencia de las incluidas en el barrido.
Otra definición que encontramos es la de dos vectores de nuevo uno de tiempo y otro de frecuencia, encargados ambos de seleccionar un conjunto de frecuencias y de tiempos respectivamente, para posteriormente mostrar resultados de forma gráfica.
El script continua con un fragmento de código donde se deben inicializar las matrices donde se contarán los canales disponibles en cada celda, así como los PRB de cada celda. Para ello con dos bucles se calcularán los totales de canales, y posteriormente se multiplicarán por el número de PRB que tiene cada canal.
Como se introdujo en la **sección 4.2** el número de PRB es de 270 por cada canal de cada celda pues estos canales son de 50MHz cada uno.
Este es número de PRB brutos por celda, este dato se dividirá entre el número de usuarios de cada

celda en cada instantánea y se hará cumplir al número de PRB las condiciones que se expusieron en la **sección 4.6** y este pasará a ser el valor asignable de PRB en las celdas.

5.4.-Algoritmo de promedio de celda.

A modo de pequeño resumen, los pasos que se siguen para calcular el número de CPUs necesarias para procesar los datos dentro del límite de tiempo que tenemos es el siguiente:

1. Definir los datos constantes.
2. Definir las reglas de mapeo de eficiencia espectral a un orden de modulación.
3. Determinar la modulación que utilizará cada celda.
4. Mapear la eficiencia espectral a un MCS de acuerdo a las reglas definidas en el paso 2.
5. Finalmente, con el algoritmo para el MCS medio, calculamos la frecuencia requerida.

La consecución de estos cinco puntos es de gran importancia, pues todos dependen del anterior, la definición de datos constantes es común tanto a la implementación generalizada para las celdas como a la implementación individualizada, al haberse explicado antes la implementación de ese script es innecesario repetirla aquí.
La implementación del mapeo de la eficiencia espectral a un esquema de modulación previa determinación del orden de modulación se expresa en código con un bucle que comprueba para cada celda en que rango se encuentra el valor de su eficiencia espectral media, a lo que se le asigna un orden de modulación.
En este punto de la implementación, hemos conseguido calcular todos los parámetros necesarios para poder proceder con la aplicación del algoritmo para celdas, se ha calculado tanto los bloques de recursos físicos como el MCS de la celda y además se conoce el valor de los coeficientes α. Si recordamos la fórmula a aplicar para desarrollar el algoritmo, la ecuación 4.4, en este punto hay dos datos que se desconocen y que son el tiempo de procesado y la frecuencia de cómputo.
Para el cálculo de la frecuencia necesaria para conseguir un tiempo tope determinado, se ha construido una función capaz de hacer un barrido recorriendo todas las posiciones del vector equiespaciado de tiempos, el cual puede albergar tantos datos de tiempo como el usuario desee, pudiendo modificar este sencillamente con tan solo cambiar el valor de un parámetro.
A continuación, se muestra en pseudocódigo como se hace dicho barrido.

Función: Cálculo frecuencias requeridas para la celda		
Entrada	:	PRB_celda, time_vector, α,MCS_celda,n_users_celda
Salida	:	Matriz con todas las frecuencias requeridas.
Paso 1	:	Inicializar la matriz de salida
Paso 2	:	Total += α * MCS_celda^(i)
Paso 3	:	Aplicar raíz cuadrada a (Total * PRB_celda / tiempo)
Paso 4	:	Repetir para todas las celdas
Paso 5	:	Devolver la matriz de frecuencias

Como se aprecia, se construye una matriz de tantas filas como posiciones tenga el vector donde se tienen los tiempos tantas columnas como numero de celdas haya en el sistema.
Con esto conseguimos tener para cada celda la frecuencia, siempre dada en Giga Hertzios, de cómputo necesaria para obtener un tiempo determinado de procesamiento.
A tiempo de ejecución:

- el código selecciona cada una de las celdas y hace la operación tantas veces como posiciones hay en el vector de tiempos.
- Dentro de cada operación, un bucle itera sobre los coeficientes polinómicos, almacenando el valor del acumulado en una variable auxiliar que se reinicia a 0 en cada instante de tiempo.
- Dentro de este bucle, cada coeficiente se multiplica por el MCS de la celda elevado al número de iteración del sumatorio, desde cero hasta dos.
- Cuando se tiene calculado el resultado del sumatorio se multiplica por los PRB de la celda y el número de usuarios de la misma.
- Por último, se divide entre el tiempo correspondiente en cada iteración y se obtiene la raíz cuadrada de este número, resultando en la frecuencia que se necesitaría para obtener el tiempo de procesamiento que se fija en cada iteración.

En la matriz retornada por la función tendremos un volumen grande de datos que se corresponden con las frecuencias necesarias de cada celda por columnas, para un tiempo concreto de procesamiento que se muestra por filas.
Asimismo, igual que se hace el cálculo para la frecuencia necesaria para un determinado tiempo, también se ha construido una función cuyo objetivo es ver que tiempo conseguimos con cada frecuencia. Su cálculo es análogo al del tiempo, pero con livianas diferencias en el barrido, para ello se llama a otra función encargada de hacer lo propio, que podemos ver cómo funciona a continuación:

Función: Cálculo tiempos de procesamiento conseguidos para la celda		
Entrada	:	PRB_celda, frecuency_vector, α,MCS_celda,n_users_celda
Salida	:	Matriz con todos los tiempos de procesamiento conseguido.
Paso 1	:	Inicializar la matriz de salida
Paso 2	:	Total += α * MCS_celda^(i)
Paso 3	:	Dividir (Total * PRB_celda) entre f^2
Paso 4	:	Repetir para todas las celdas
Paso 5	:	Devolver la matriz de tiempos

Resulta sencillo ver las analogías entre funciones. Sin embargo, las sutiles diferencias se pueden comprobar fácilmente:

- La matriz donde se almacena el resultado tiene una dimensión diferente, en este caso tendrá tantas filas como frecuencias hayamos seleccionado para el cálculo.
- Ahora no se realiza la raíz cuadrada si no que se eleva al cuadrado la frecuencia en GHz para obtener el tiempo de procesamiento que se consigue, en milisegundos.

Salvando estas dos diferencias, el procedimiento del cálculo es idéntico en ambas funciones. La salida de la función para **calcular el tiempo de procesamiento conseguido en cada celda** devolverá una matriz cuyos datos nos aportaran los milisegundos que se tardaría en cada celda en hacer el procesado si la frecuencia de cómputo que tenemos disponible fuera la que se preselecciona.
Un esquema sencillo, a modo de resumen de lo que se implementa aquí viendo cómo influyen los distintos archivos sería el siguiente:

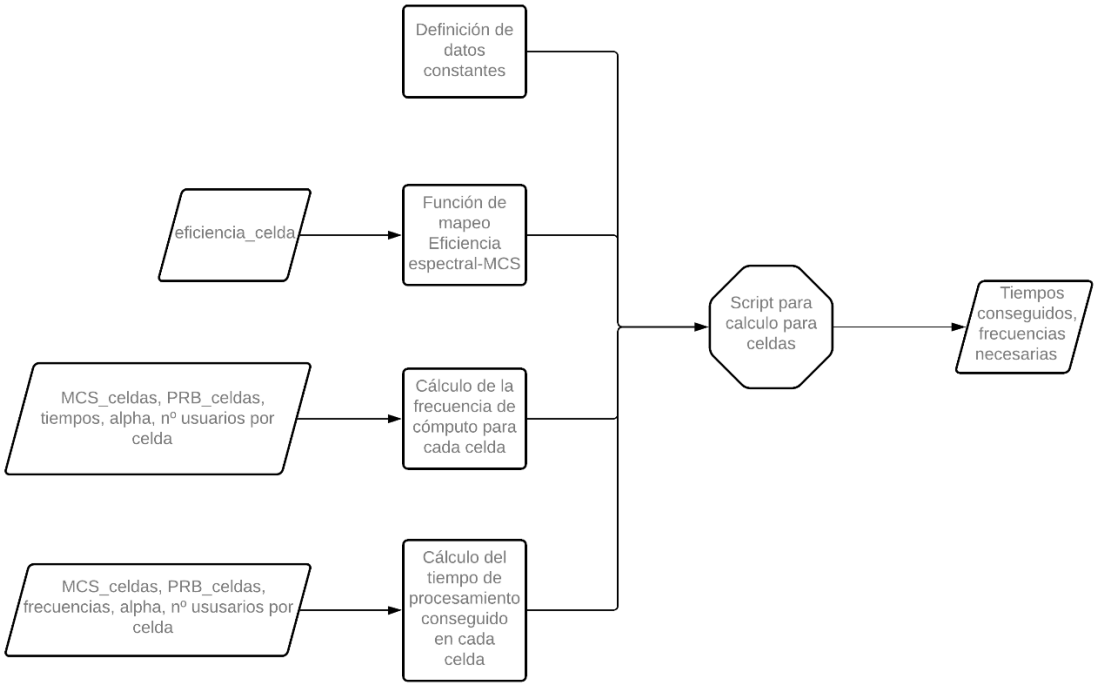


Figura 5.3: Diagrama De Implementación Celdas.

5.5.-Algoritmo individualizado a usuarios.

Ya desde muy pronto se intuyen grandes diferencias a la hora de la implementación del algoritmo individualizando para usuarios con el generalizado para las celdas. En esta sección se va a estudiar como esta implementado el algoritmo para los usuarios tratados de manera individual. El objetivo principal de llevar a cabo dos implementaciones no es más que ver posteriormente cuál de los dos es más exigente y si podrían llegar a ser equivalentes, todo esto se desarrollará en la sección de resultados extraídos.
Los pasos que se siguen en esta implementación son distintos y más extensos que en la implementación anterior donde se resumió en cinco pasos lo que desarrollaba ese script. En este de nuevo se va a dar primero una versión escueta para entender cómo se implementa este algoritmo y posteriormente se entrará en extenso detalle de que es lo que se hace. Por pasos, el script desarrolla lo siguiente:

1. Determinar a qué BS está conectado cada usuario en cada instantánea.
2. Mapear su eficiencia a MCS.
3. Determinar cuántos usuarios tiene cada estación base conectados.
4. Ver el número de PRB que se le asignan a cada usuario.
5. Calcular tiempos y frecuencias.

Resulta muy importante determinar bien a que estación base se conecta cada usuario en cada instantánea por la razón de que con ello podremos tener en cada instante la eficiencia espectral que tiene cada usuario. Adicionalmente, resulta un factor crítico a la hora de posteriormente determinar la frecuencia que necesitará una celda y el tiempo que conseguirá con cada frecuencia.
La manera en la que se realiza esta operación, se basa en extraer de un dato que arroja el simulador la estación base que sirve a cada usuario, no hay más que coger todas las filas de la matriz y seleccionar la columna igual al número de iteración.
De esta manera tendríamos dentro de una matriz el número de estación base al que se ha

conectado cada usuario. La matriz de la que se extrae la eficiencia espectral para cada instantánea es un dato que el simulador proporciona. De ahí que la extracción sea una operación realmente sencilla, pues solo habría que seleccionar la columna correspondiente al número de iteración para el que se está calculando.
Siguiendo con la búsqueda de completar los valores de todos los parámetros que necesitamos el segundo paso es análogo a lo que se hace en la otra implementación, simplemente en este caso el bucle será más largo para determinar el MCS de todos y cada uno de los usuarios.

Función: Determinar el MCS de cada usuario		
Entrada	:	Eficiencia espectral del usuario
Salida	:	MCS asignado a cada usuario
Paso 1	:	Inicializar la matriz de salida
Paso 2	:	Aplicar el mapeo de la tabla 4.1
Paso 3	:	Repetir para todos los usuarios
Paso 4	:	Devolver la matriz que contiene el MCS de cada usuario

En este caso el bucle recorre todos los usuarios, que es otro parámetro de entrada al simulador y puede ser modificado fácilmente antes de ordenar una simulación al software donde se implemente. A la función de mapeo de eficiencia a MCS se le pasa esta vez un dato extraído de nuevo de una matriz que es un dato de salida del simulador, la eficiencia espectral de la que goza cada usuario en cada instantánea. Por tanto, para el cálculo que se necesita hacer será necesario pasar como parámetro dicha matriz, seleccionando el usuario y la instantánea de la simulación. Las reglas de mapeo eficiencia-MCS son exactamente iguales que en la implementación generalizada por celdas, de nuevo extraídas de la tabla 4.1.
Seguidamente se utiliza una función construida para contabilizar cuantos usuarios tiene cada estación base conectados a ella. La utilidad de esta función reside en que posteriormente necesitaremos el número de usuarios que tiene cada celda para poder completar el sumatorio que involucra a todos los usuarios del algoritmo.

Función: Calcular el número de usuarios conectados a cada celda		
Entrada	:	Estación base a la que se conecta el usuario, nº de usuarios en el sistema, nº de estaciones base en el sistema
Salida	:	Número de usuarios conectados a cada base
Paso 1	:	Inicializar la matriz de salida
Paso 2	:	Contabilizar el número de usuarios conectados a la primera estación base
Paso 3	:	Repetir para todas las estaciones
Paso 4	:	Devolver la matriz que contiene el número de usuarios de cada BS

La función arriba expresada recibe como parámetros la estación base de los usuarios, el número de usuarios en el sistema y el número de estaciones base que hay en el escenario.
A partir de ahí simplemente realiza una comparación entre la estación base a la que está conectado cada usuario y el número de estación base a la que se le están contabilizando los usuarios conectados. Cada vez que la comparación resulte verdadera, incrementará la posición del vector donde se contabilizan los usuarios correspondiente a la iteración del bucle para al final

del bucle que recorre todos los números de estación base tengamos en la posición correspondiente del vector el número de usuarios que hay conectados a cada BS.
El siguiente paso es asignar los PRB a cada usuario. Como se ha venido diciendo y dejando claro, la asignación de PRB se asume sin prioridades e igualitaria entre todos los usuarios de la celda. Es menester recordar que los PRB que se asignan a un usuario deben de constituirse como un número entero positivo y además este número debe ser par. Para este cálculo se hace una división entera entre todos los PRB de la celda y los usuarios que están conectados a la estación base. Posteriormente si el número de PRB que se han asignado a un usuario es impar, se le sustraerá uno para hacer par dicho número de PRB asignados. El motivo de la sustracción no es otro que evitar que se asignen más PRB de los que realmente hay disponibles. La adición en lugar de la sustracción también haría par el número de PRB asignados a un usuario, pero esto haría que fuera posible rebasar el número que una celda puede asignar de PRB a sus usuarios y por tanto se cometería un grave error.
De nuevo nos encontramos en el punto donde tenemos todos los datos de entrada del algoritmo perfectamente determinados y calculados para cada usuario. Esto significa que el siguiente paso, análogamente a la implementación anterior pasa por construir una función que haga un barrido recorriendo el vector de tiempos y otra función que se encargue de lo mismo con la diferencia de que el barrido se haga en los valores de la frecuencia.

Función: Calcular la frecuencia requerida por celda con MCS de usuarios		
Entrada	:	PRB usados de cada celda, time_vector, α, MCS_usuario, nº usuarios de cada celda, BS_usuario, nº BS en el escenario
Salida	:	Frecuencias de cómputo requeridas en cada celda
Paso 1	:	Inicializar la matriz de salida
Paso 2	:	Contribucion_usuario += α * MCS_usuario^(i)
Paso 3	:	Aplicar raíz cuadrada a (Contribucion_usuario * PRB_celda / tiempo)
Paso 3	:	Repetir para todas las estaciones y usuarios
Paso 4	:	Devolver la matriz que contiene las frecuencias requeridas

Esta función resulta a simple vista sensiblemente más compleja que la análoga en el algoritmo de celdas.
Como parámetros de entrada se incluyen además de los que ya tenía la función de las celdas, la estación base a la que pertenece cada usuario y el MCS de cada usuario.
La implementación difiere principalmente en que ahora tenemos dos variables donde se almacenan los acumulados. Una la suma del usuario que si recordamos la ecuación 5.12 debemos sumar la contribución de cada usuario dentro de la celda y la otra es la suma de la celda, que es donde se almacena la suma de acumulados de los usuarios, dando así lugar a la contribución total de todos los usuarios de cada celda. Para el total de cada usuario se utiliza el MCS individual de cada usuario. Una vez tenemos la contribución de todos los usuarios al tráfico de la celda, multiplicamos por los PRB utilizados en la celda. Esto es debido a que después de la operación para aplicarles paridad a los PRB, al momento de la sustracción sabemos que probablemente vayan a quedar PRB sin asignarse a ningún usuario.
Una vez se ha obtenido este último dato, se procede al barrido haciendo un cálculo final para determinar la frecuencia necesaria para conseguir cada uno de los tiempos que figuran en el vector de tiempos sobre el que se realiza el barrido.
Y finalmente la función retornará una matriz con tantas filas como posiciones tenga el vector de tiempos y tantas columnas como estaciones base haya en el escenario, quedando así en cada posición de la matriz la frecuencia necesaria en cada celda para conseguir cada tiempo de

procesamiento.

Por su parte la otra función, que nos proporciona que tiempo de procesamiento se obtiene de cada frecuencia, viene construida como sigue:

Función: Calcular el tiempo de procesamiento por celda con MCS de usuarios		
Entrada	:	PRB usados de cada celda, frecuency_vector, α, MCS_usuario, nº usuarios de cada celda, BS_usuario, nº BS en el escenario
Salida	:	Frecuencias de cómputo requeridas en cada celda
Paso 1	:	Inicializar la matriz de salida
Paso 2	:	Contribucion_usuario += α * MCS_usuario^(i)
Paso 3	:	Dividir (Contribucion_usuario * PRB_celda / f^2)
Paso 3	:	Repetir para todas las estaciones y usuarios
Paso 4	:	Devolver la matriz que contiene los tiempos de procesamiento conseguidos

Para esta función el código es ligeramente diferente al de la anterior, sin embargo, se parecen bastante ya que la primera parte de la función, hasta donde se hace el barrido es idéntica. Las diferencias vienen cuando se debe hacer el barrido, en este caso el barrido recorre el vector de frecuencias equiespaciadas entre unos límites que se pueden elegir antes de la ejecución del programa.
A la salida de la función obtenemos una matriz que tendrá tantas filas como frecuencias haya en el vector sobre el que hacemos el barrido y tantas columnas de nuevo como estaciones base tengamos en el escenario. Cada posición de la matriz contendrá un dato expresado en microsegundos que corresponderá con el tiempo de procesado alcanzado para cada celda con cada frecuencia.

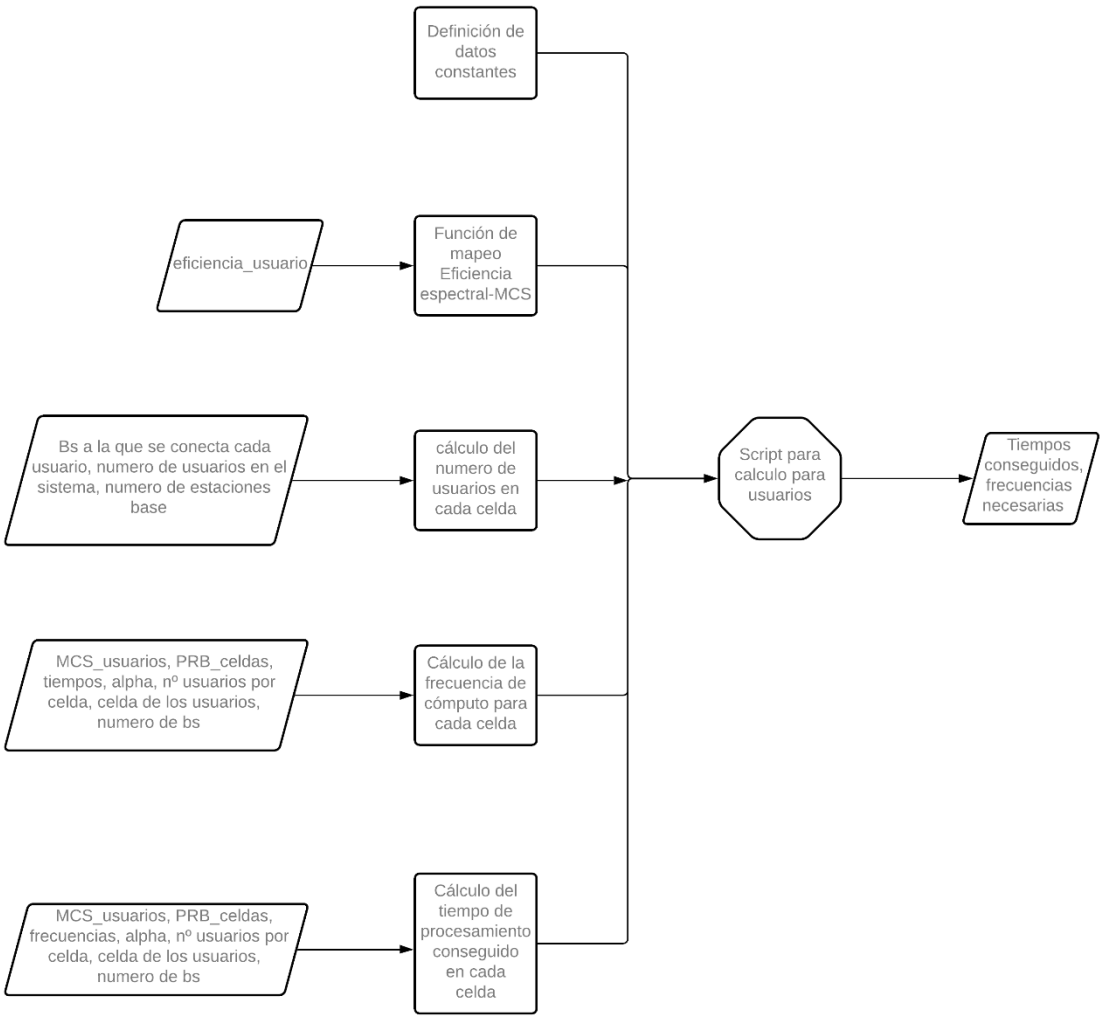


Figura 5.4: Diagrama de implementación usuarios.

En la figura 5.4 vemos de nuevo en un esquema de que archivos se vale el script destinado a implementar el algoritmo para los usuarios individualmente.

5.6.-Mejoras propuestas.

Hasta ahora, se han expuesto las dos implementaciones principales en su primera forma. Esto es, la idea original con la que se emprendió la realización de este trabajo. Sin embargo, las ideas primigenias no suelen ser definitivas y a medida que los proyectos avanzan suelen verse modificadas. No iba a ser menos este trabajo pues a medida que se iba completando y realizando surgían varias alternativas que se pensó podían mejorar el trabajo que realizaban las implementaciones expuestas en el anterior epígrafe. No solo aportan una mejoría, sino también una visión más fiel de lo que se refiere a la realidad de las redes.
Principalmente, las mejoras se pueden resumir en dos, de las cuales una estará divida a su vez en dos funcionalidades.

5.6.1.-Mejora en la planificación de los usuarios.

En lo relativo a la planificación de los usuarios, hemos visto en la implementación del algoritmo individualizado que todos los usuarios contribuyen ponderadamente por su MCS en el tráfico generado en cada celda. Se ha mostrado también como no se dan prioridades ni favoritismos a ninguno de estos usuarios a la hora de repartir los PRB.

Partiendo de estas dos bases, viene implementada esta mejora partiendo del siguiente razonamiento: Los usuarios cuya eficiencia espectral sea especialmente baja, tanto así que el MCS que se les asigne a dichos usuarios sea el cero, serán considerados como usuarios cuyo tráfico no se va a cursar. Por tanto, su tráfico no se procesará de ninguna manera y la asignación de PRB a dichos usuarios no será sino un detrimento del número de PRB que se puede asignar a usuarios cuyo esquema de codificación y modulación sea de 1 en adelante y por tanto el tráfico de estos si vaya a ser cursado.
Aplicando este pensamiento, se va a estimar la mínima frecuencia de cómputo dentro de los centros de calculo que puede satisfacer las necesidades de la red teniendo en cuenta que aquellos usuarios cuya SINR sea tan baja que quede por debajo del umbral permitido serán bloqueados y su tráfico no se cursará
No solo es la frecuencia lo que recoge el impacto de esta mejora, también los demás usuarios se ven beneficiados debido a que habrá menos usuarios entre los que se deben repartir los recursos físicos, por tanto, su tráfico se ve procesado más rápidamente y la calidad del servicio que pueden llegar a experimentar, escalará en cuanto al monto de las métricas que la miden, principalmente la velocidad de la red.
En materia de código, eliminar de la planificación y el procesado a los usuarios con $I_{MCS} = 0$, se lleva a cabo mediante la inserción en el código de una sencilla comprobación antes de entrar al bucle del sumatorio, dicha condición se recoge en el siguiente diagrama:

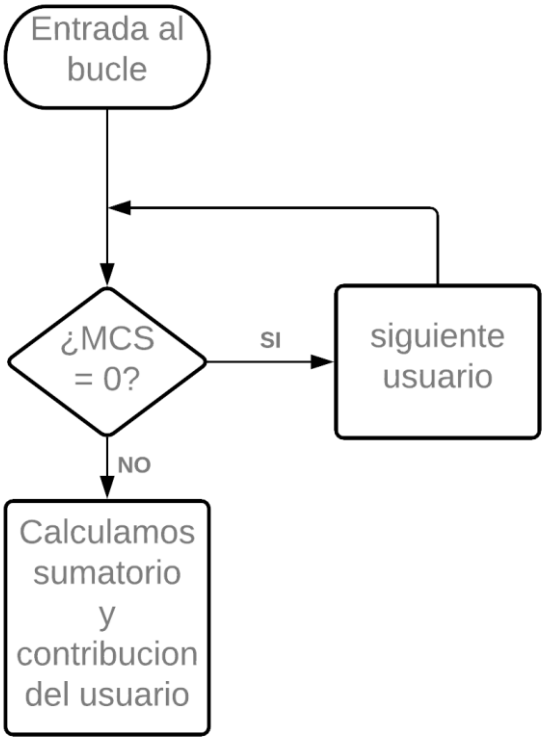


Figura 5.5: Comprobación de MCS para usuario.

En la figura anterior podemos ver como difiere claramente el cálculo con el que se presentó en la sección 5.5. Vemos como se ha introducido una condición que comprueba el mcs del usuario al cual se le va a calcular su contribución al tiempo de procesado de la celda, verificando si es distinto de cero, lo cual hace de llave para que continúen los cálculos relativos a ese usuario. En el caso de que el MCS del usuario sea cero, simplemente se obviará y no se le aplicará el algoritmo. Pues como hemos dicho previamente, resulta en un lastre procesar el tráfico de estos usuarios.
El código para tener en cuenta esta mejora no se reduce al que se acaba de mostrar, como se ha

expuesto previamente, el caso de un MCS igual a cero influye también en los PRB que se asignan a cada usuario. Recordemos la función que contabilizaba el número de usuarios que había conectados a cada BS.

5.6.2.-Número de CPUs por centro de cómputo.

Como dato, la frecuencia de cómputo tiene una importancia y representa una información muy valiosa. Sin embargo, este trabajo trata de ser lo más verosímil posible y busca dar una respuesta real a un tema en investigación. Esto hace que el dato de la frecuencia sea ilustrativo y relevante, pero no útil. El tener una frecuencia de cómputo no sirve de nada si no podemos sacarla del mercado, de componentes, de procesadores existentes en el mercado actual. De aquí nace la segunda mejora implementada.
Esta segunda mejora podría dividirse en dos, no obstante, por su similitud se englobarán ambas en una y se expondrá de tal forma.
En el artículo [2] podemos leer que la frecuencia mínima que debe de tener cada una de las CPUs debe tener una frecuencia de al menos 2.7GHz. A día de hoy, los procesadores que se encuentran en el mercado rondan esta frecuencia, superada con creces en muchas ocasiones debido procesadores que admiten overclocking. Así, por ejemplo, en el equipo utilizado para la realización de este trabajo se dispone de un procesador Intel i7-6750HQ, el cual tiene una frecuencia de trabajo de 2.6GHz pudiendo llegar en modo boost a 3.6GHz. Por tanto, la frecuencia mínima de cada CPU es una frecuencia muy realista, incluso podría decirse que muy asequible, en cuanto a lo que el mercado ofrece y no sería difícil encontrar procesadores que hicieran cumplir con este requisito al proyecto. Huelga decir que esta no es más que la frecuencia mínima, por tanto, no habría inconveniente en hacer uso de CPUs con mayor velocidad de cómputo.
Esta mejora empieza a implementarse desde que se definen los datos constantes en el cual se define la variable *fmin*, al valor que se muestra en la ecuación 7.1**.**
Este valor no se vuelve a modificar en toda la ejecución, por tanto, puede cambiarse solo antes de la ejecución del programa.
El parámetro aparece de nuevo en el script en el cual se calculan los procesadores necesarios para procesar el tráfico a tiempo en diversas situaciones. Principalmente este parámetro actúa como divisor, pues para encontrar el número de CPUs que se necesitan, basta con dividir el dato de la frecuencia total requerida entre la frecuencia de CPU que estamos utilizando y redondear al siguiente entero a fin de tener un número suficiente de procesadores y que además sea entero.

Capítulo 6.- Evaluación de los resultados obtenidos

Una vez que se ha completado el séptimo capítulo donde se ha explicado cómo se han realizado las bases de las implementaciones del algoritmo, entramos en esta sección donde además de exponerse la implementación de la extracción de resultados se discutirán y explicarán los resultados obtenidos, se compararán las distintas implementaciones y se evaluará el impacto de ciertos parámetros.

6.1.-Evaluación del impacto del Transporte FrontHaul sobre la frecuencia de cómputo

Si recordamos, y en busca de buenas prácticas, la evaluación de requisitos fue lo primero que se explicó en la sección de las implementaciones. Es por ello que, antes de exponer ni discutir ningún resultado, lo primero que debemos evaluar es el impacto de introducir esos requisitos.
Dado que el retardo HARQ es algo que no puede ser obviado y que es de donde todos los problemas que se buscan resolver en este trabajo nacen, no podemos siquiera cuestionar su importancia para lo que estamos desarrollando. Sin embargo, si hablamos del Tfh, que recordemos, era el retardo de propagación de los datos en la parte FrontHaul del sistema, este parámetro sí que puede ser sometido a evaluación debido a su dependencia directa de la distancia a la que se encuentran las estaciones base de los centros de cómputo.
En esta primera subsección del análisis de resultados vamos a poner de manifiesto la importancia, o no, de incluir el Tfh en nuestra implementación.
Para ello es importante recordar la ecuación 4.6 donde se expuso como se iba a calcular este retardo. Sencillamente este retardo se calcula como la distancia entre la velocidad de transmisión del medio físico. Por tanto, desde ya se informa que el medio físico que se ha considerado entre las estaciones base y los centros de cómputo es la **fibra óptica**, mientras que, con carácter meramente informativo, el medio físico que se supone entre los usuarios y su estación base será espacio libre, aire. Debido a que se ha tomado la fibra óptica como medio de transmisión, se ha tomado como referencia que la velocidad de la luz en el medio sea:

$$C_{fo} = \frac{2}{3} * c\ m/s$$

ECUACIÓN 6.1: DEFINICIÓN DE LA VELOCIDAD DE TRANSMISIÓN TOMADA PARA FIBRA ÓPTICA.

Donde sabemos que *c* es la velocidad de la luz en el vacío, o sea:

$$c = 3 * 10^8\ m/s$$

ECUACIÓN 6.2: VELOCIDAD DE TRANSMISIÓN EN EL VACÍO.

Para el cálculo del Tfh se tiene en cuenta la heterogeneidad del escenario. En un primer momento se consideró hacer un cálculo de peor escenario, para dar simplicidad a la implementación y considerar solo la estación base más alejada de cada centro de cómputo, pero fue rápidamente desechada esta idea, el objetivo de esta sección es clarificar el porqué de desechar la integración de este concepto en los cálculos. Si recordamos, se dedica un script en su totalidad a la implementación de todo el cálculo del Tfh para cada una de las estaciones base.
El siguiente fragmento de seudocódigo trata de mostrar la idea plasmada para calcular ese Tfh

Función: Calcular el tiempo de transporte FrontHaul		
Entrada	:	D_BBU_RRU
Salida	:	T_{fh} de cada RU
Paso 1	:	Inicializar la matriz de salida
Paso 2	:	Dividir distancia / C_{fo}
Paso 3	:	Convertir en microsegundos
Paso 3	:	Repetir para todas las estaciones y usuarios
Paso 4	:	Devolver la matriz que contiene los T_{fh}

Salta a la vista la simplicidad de la implementación de este cálculo. El script almacena en un vector el retardo de cada estación base a su centro de cómputo asignado. Finalmente, dado que los resultados están expresados en segundos se les aplica un factor de escala para poder tener ese tiempo en microsegundos, que resulta una magnitud más útil que el segundo en vista de las magnitudes que maneja el algoritmo.
Para ver los resultados de forma gráfica e ilustrativa para el ojo humano, vamos a representar gráficamente para cada celda la primera frecuencia de cómputo que nos proporciona un tiempo de procesamiento por debajo del límite sin considerar el Tfh y considerando el Tfh. Este resultado se extrae individualmente de cada instantánea de la simulación, por tanto, puede comprobarse separadamente para cada una de las instantáneas.

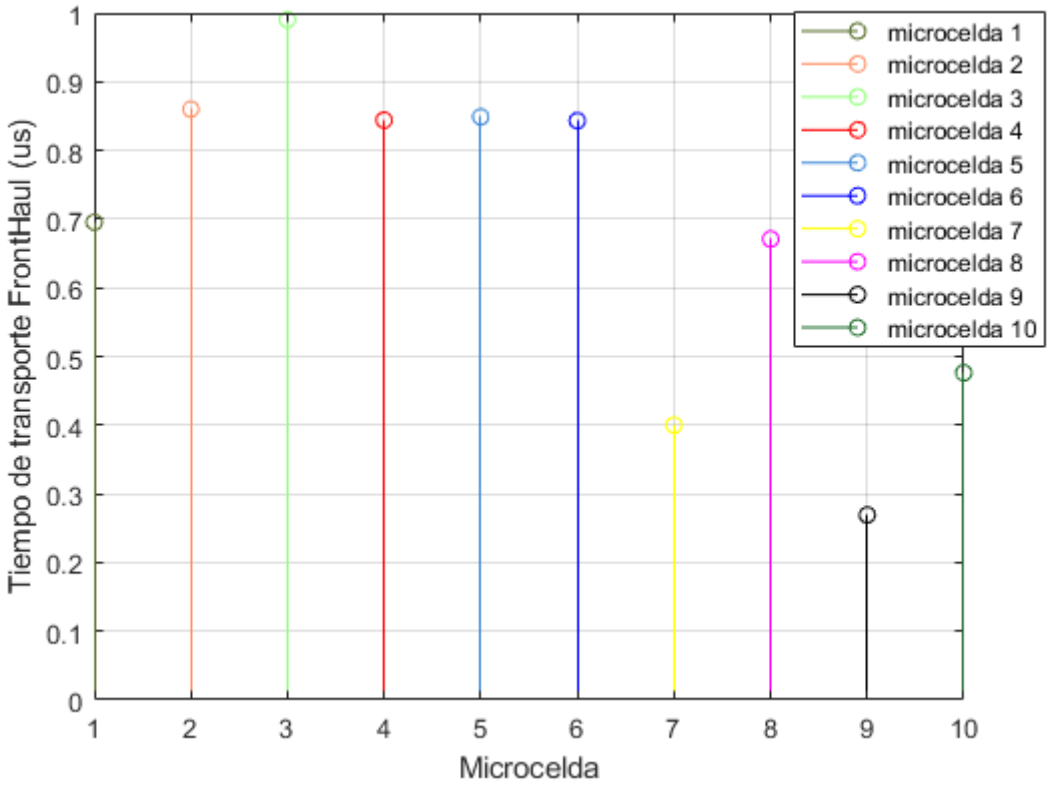


Figura 6.1: TFH para cada una de las microceldas

La figura 6.1 ilustra para cada una de las microceldas, el tiempo empleado por la señal en viajar

desde éstas hasta el centro de cómputo que le corresponde a cada una. La diferencia de tiempo introducida por el T_{fh} tiene un valor maximo para la microcelda 3, sin embargo este valor no es en ningún caso mayor de un microsegundo. Con estos datos evaluados, la conclusión a la que llegamos es que, debido a la gran velocidad de transmisión en el medio, sumado a la distancia a la que se encuentran las estaciones base y los centros de cómputo, la introducción del Tfh no produce una diferencia significativa en la frecuencia de cómputo que se requiere para cumplir los requisitos de tiempo de procesado impuestos por los límites. Podemos decir a raíz de esto que la diferencia entre las frecuencias requeridas es insignificante y despreciable. En adelante, viendo la conclusión que se extrae, no se tendrá en cuenta el Tfh debido a su papel despreciable en el requisito de la frecuencia.

6.2.-Análisis del resultado obtenido en las implementaciones

A continuación, vamos a describir los resultados obtenidos en cuanto a la frecuencia de CPU necesaria para procesar los datos dentro del tiempo del cual se dispone. Tras el análisis, en otra subsección se comprarán los resultados entre ambas implementaciones.

6.2.1.-Análisis de la implementación del algoritmo por celdas

Para la implementación por celdas se ha visto en secciones anteriores que se ha tomado un valor de MCS medio para toda la celda.
Para ver de manera gráfica el resultado de frecuencia obtenido, se desarrolla en dos scripts la realización de sendos gráficos con los resultados almacenados en las matrices donde se guarda el tiempo conseguido con cada frecuencia de cómputo y la frecuencia necesaria para conseguir cada tiempo respectivamente. El código que implementa dicho plot es de naturaleza sencilla: Se selecciona una muestra representativa de todos los datos calculados para no introducir datos de variaciones casi despreciables dentro del vector que actúa como variable independiente en cada una de las gráficas y posteriormente se seleccionan los valores correspondientes de tiempo y frecuencia respectivamente para realizar el dibujo de ambos gráficos.
Para que los datos que se muestran en las gráficas sean más representativos, se promedian todas las instantáneas que se simulan. Esto se hace para buscar un dato que nos asegure escalabilidad en el sistema y este dimensionado en función de los requerimientos de todas las instantáneas. Permitiendo así que el modelo construido pueda funcionar en la práctica totalidad de los casos sin hacer un sobredimensionado demasiado abrupto, para poder tener un factor de utilización lo más alto posible sin llegar a la situación de que no se pudiera procesar el tráfico de ninguna estación base.

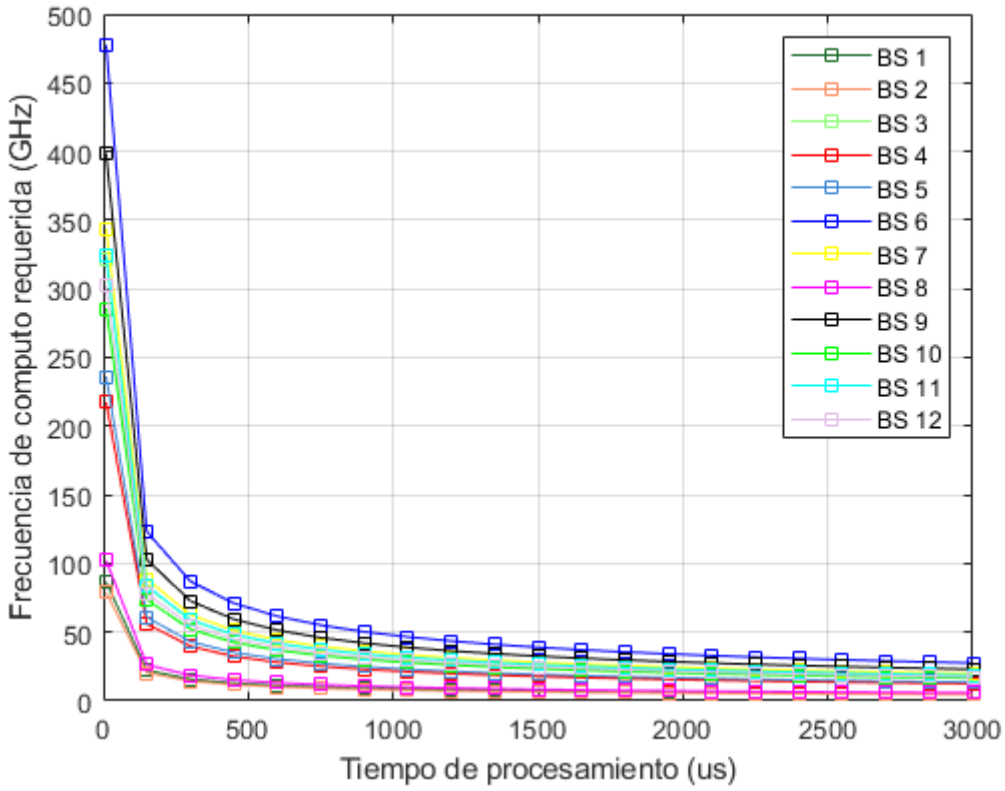


Figura 6.2: Frecuencias medias por celda para conseguir el tiempo de procesado deseado.

En la figura 6.2 vemos cómo evoluciona la frecuencia requerida para procesar a tiempo los mensajes. En la gráfica se muestra la frecuencia media requerida por cada estación base para procesar a sus usuarios mediante el criterio de la primera implementación del algoritmo. Debido a las condiciones individuales de cada celda como es el número de usuarios conectados, los canales disponibles y por tanto los PRB, las curvas de las estaciones base difieren unas de otras. Además, se observa una bajada exponencial de la frecuencia requerida a medida que aumenta el tiempo disponible para procesar, esto se debe a que la función que se está representando tiene la variable independiente, el tiempo en este caso, en el denominador.

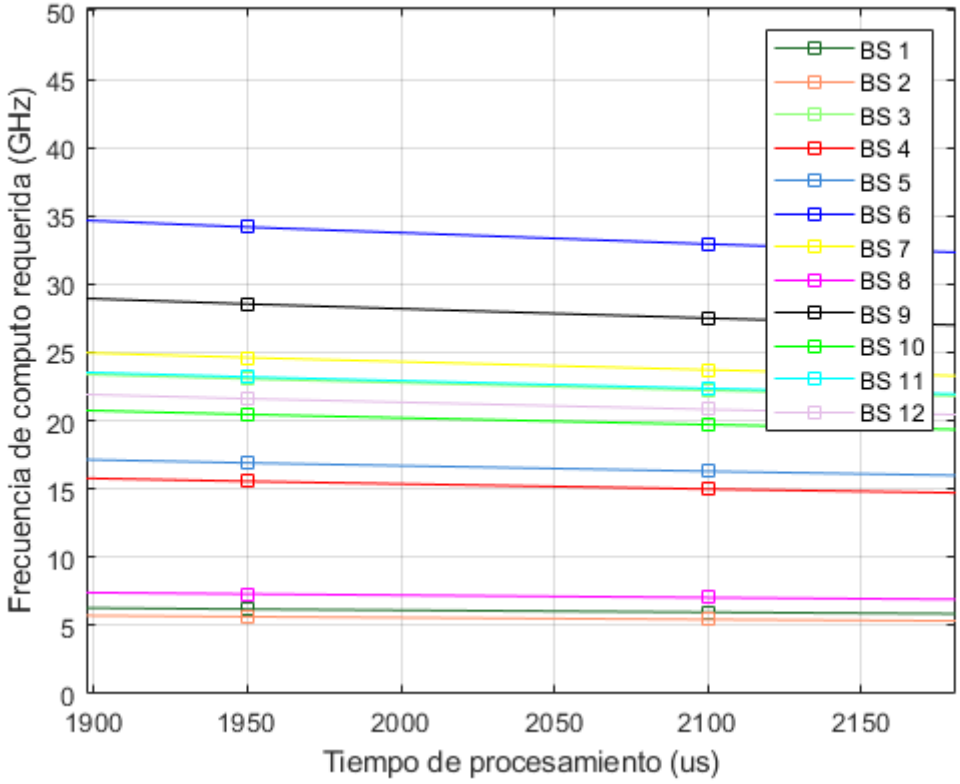


Figura 6.3: Frecuencias medias por celda para cumplir los requisitos harq.

En esta otra figura se muestra un zoom en el cual se puede ver cuál es la frecuencia necesaria en cada celda para cumplir el tiempo de procesado de 2ms impuesto por el HARQ. A partir de esta grafica podemos ver como los requerimientos para el promedio de todas las iteraciones de la simulación se puede estimar entre los 5GHz y los 35GHz.
Por mera comprobación, a continuación, se mostrará la gráfica con los ejes cambiados para comprobar la veracidad de la implementación.

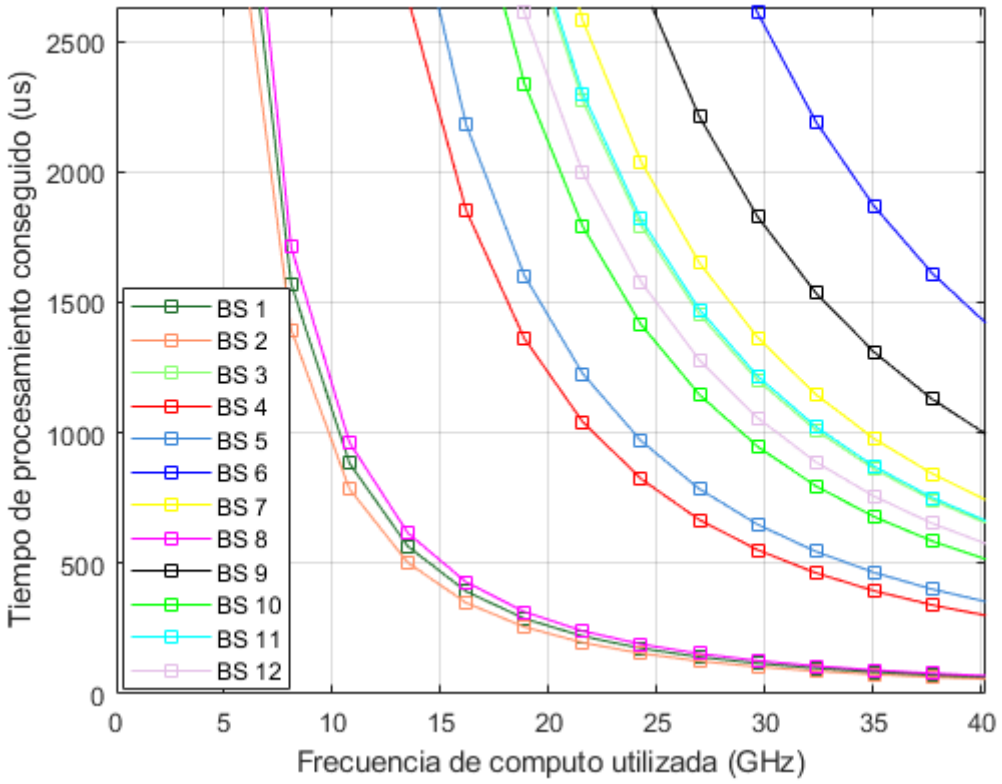


Figura 6.4: Tiempo de procesamiento conseguido en cada celda con una frecuencia determinada.

Puede compararse en sendos gráficos que las curvas adoptan iguales valores en una y en otra, o sea, los valores del eje x en la figura 6.2 tienen asociado un valor en el eje y de la misma grafica que corresponde con el valor del eje x de la figura 6.4 que tiene asignado un valor en el eje y, que es igual al valor del eje x de la primera grafica.

6.2.2-Analisis de la implementación del algoritmo individualizado

Esta sección se presenta análoga a la justo anterior, los métodos de análisis y extracción de resultados han sido muy similares para ambas.

Para esta implementación ya se ha visto con anterioridad como se ha conseguido implementar el cálculo individualizando a cada usuario conectado a la red e incluso dejando de tener en cuenta a aquellos usuarios a los cuales el tráfico no se les iba a procesar.

La analogía entre extracciones de resultados es tal que podemos apreciar su mínima diferencia simplemente con las líneas en las que llamamos a las funciones que dan una muestra grafica de los resultados que se obtienen.

Del nombre de las funciones se distingue que en este caso vamos a prestarle especial atención a cada usuario. Respectivamente las funciones plotean de nuevo el tiempo que se consigue con cada frecuencia de cómputo para este algoritmo y la frecuencia necesaria para conseguir cada tiempo determinado.

Con objeto de determinar de una forma más fiel a la realidad y que nos permita un dimensionamiento correcto y que no llegue a provocar una saturación, de nuevo a la hora de representar los datos se han promediado en todas las instantáneas tanto la frecuencia requerida como el tiempo conseguido en cada una de las instantáneas en cada una de las celdas que forman parte del escenario desplegado en el simulador.

Nuevamente se van a mostrar los resultados gráficamente donde vamos a comprobar que requisitos tendríamos en el sistema de dimensionado a la hora de decidir cuantas CPU

necesitaríamos en cada centro de cómputo.

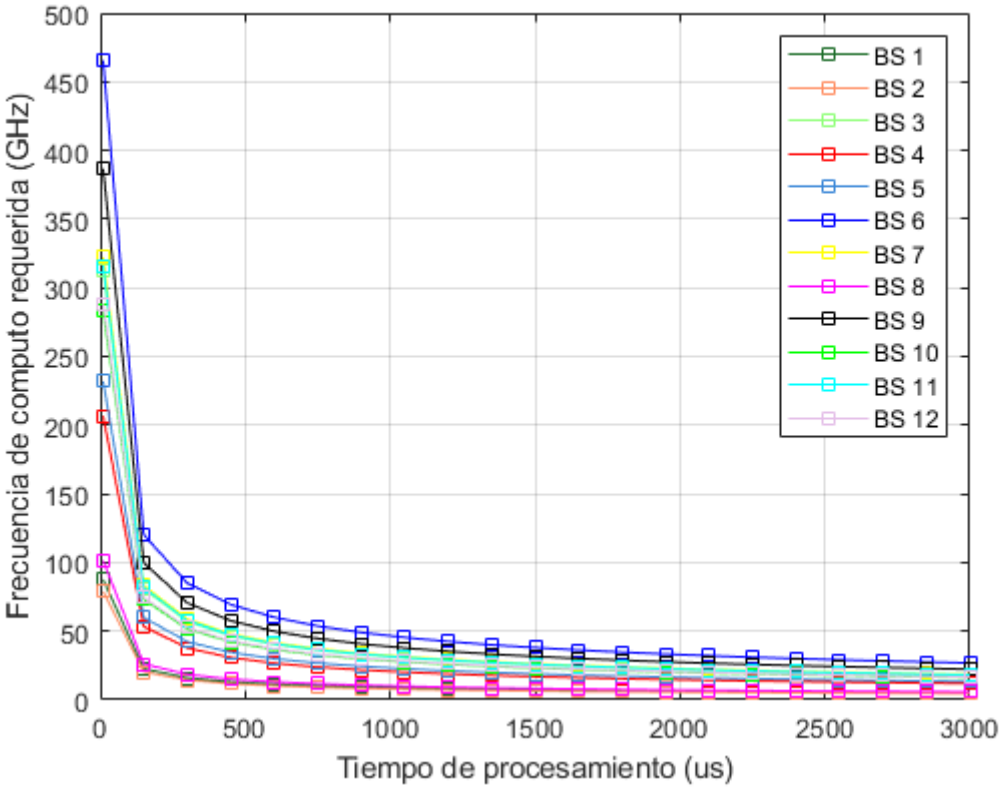


Figura 6.5: Frecuencias medias por celda para conseguir el tiempo de procesado deseado con implementación individualizada.

De nuevo vemos en una gráfica la comparativa entre todas las celdas en lo que se refiere a la frecuencia que se necesita para conseguir un tiempo determinado en lo que sería la demora del procesado de los mensajes.

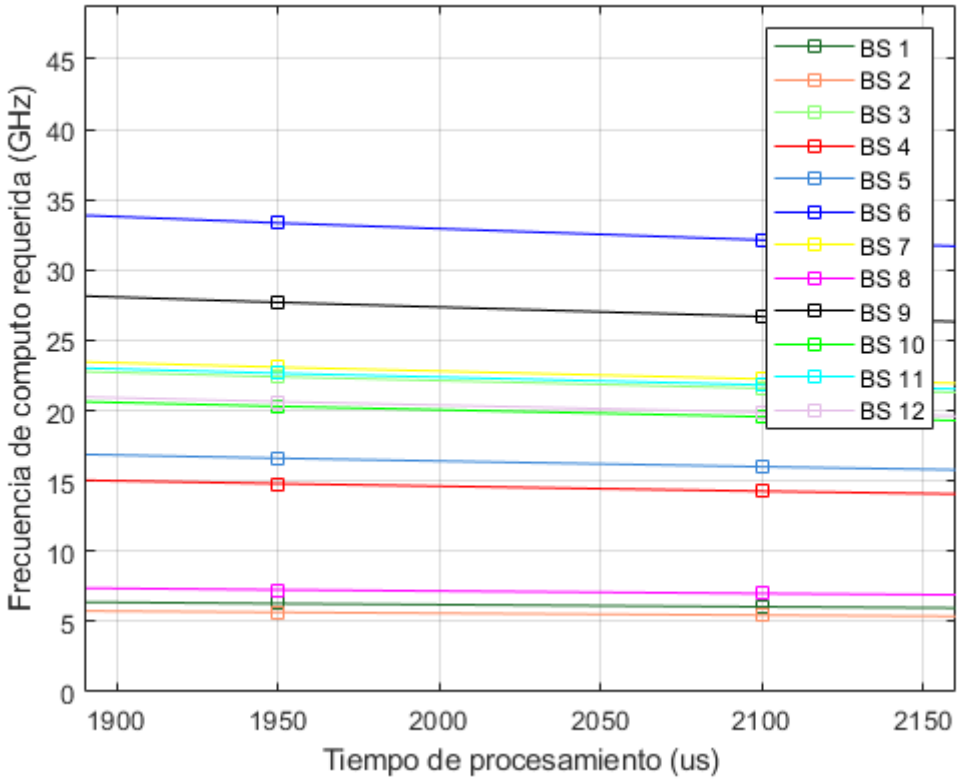


Figura 6.6: Frecuencias medias por celda para cumplir los requisitos harq con implementación individualizada.

Vemos como el umbral en el que se mueven las frecuencias requeridas es altamente parecido el que se obtenía en la implementación considerando el MCS medio de cada celda, sin embargo, esa comparación no ocupa en esta sección y se discutirá más adelante en una sección dedicada exclusivamente a discutir la comparativa entre implementaciones y proporcionar un veredicto sobre cual está más correspondida con la realidad.

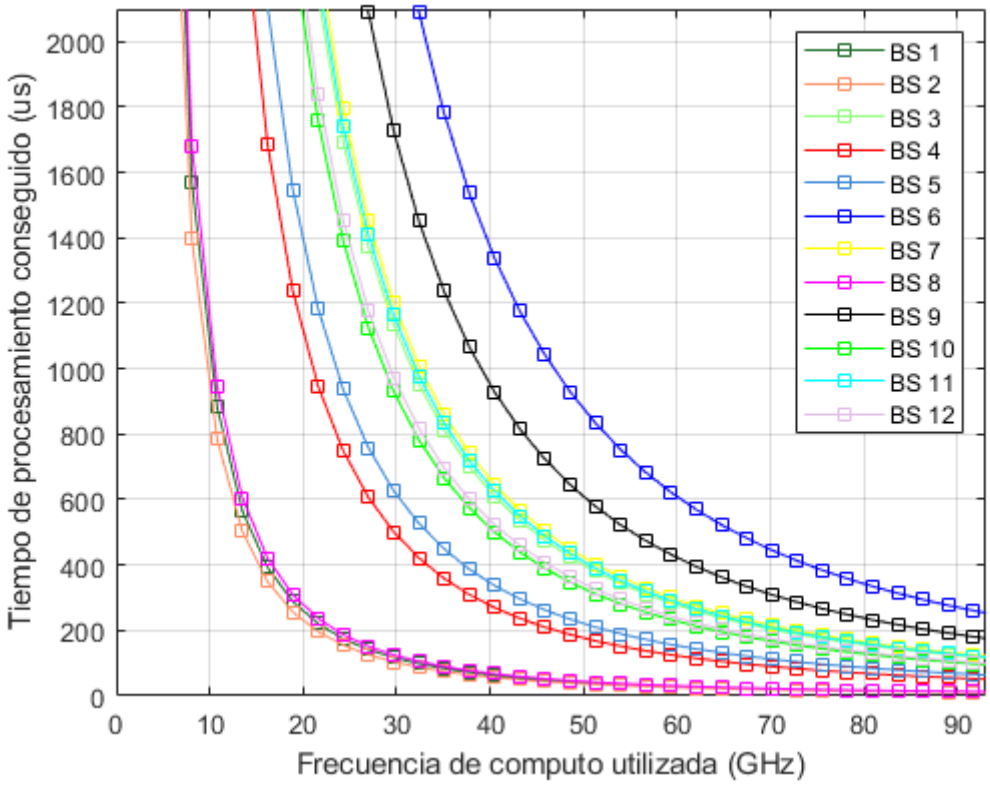


Figura 6.7: Tiempo de procesamiento conseguido en cada celda con una frecuencia determinada con implementación individualizada.

En vista del resultado de la figura 6.7, se puede concluir que los cálculos están bien implementados a raíz del siguiente criterio: si comparamos los valores de tiempo obtenidos en la figura 6.7 para una frecuencia y estación base dadas con la frecuencia requerida para un determinado tiempo de la figura 6.5 o 6.6, igual al que conseguimos en la figura 6.7 según lo recién explicado, obtenemos un dato que si bien puede diferir en cifras poco significativas debido a la granularidad con la que se dibujan ambas gráficas, se pueden considerar iguales.
Esto lo podemos esclarecer con un ejemplo: si nos fijamos en la figura 8.8 en la curva para la BS6, obtendremos el punto (35,07GHz, 1865us), si ahora hacemos lo propio en la figura 6.6, obtendremos un punto de coordenadas muy similares, el (1800us, 34,62GHz). Debido a la granularidad ya comentada, no son puntos iguales, pero puede darse validez al cálculo implementado.

6.3.-Comparación de las implementaciones

Una vez se han visto los resultados de manera gráfica, en esta sección se pretende llevar a cabo una comparación minuciosa y exhaustiva de las implementaciones cuyos resultados se han presentado en la sección anterior.
Para ello se pretende elaborar varias métricas, presentadas en orden de importancia para valorar cuál de las dos implementaciones nos permitiría fabricar un dimensionado mejor, de mayor rendimiento y más ajustado a la realidad.
Primeramente, vamos a realizar una comparación entre las frecuencias de cómputo medias que se obtienen en cada una de las celdas con ambas implementaciones a fin de proporcionar una idea de la métrica que se intuye más importante a la hora del diseño del sistema de cómputo que es por supuesto la frecuencia de cómputo que nos hace falta.

En la siguiente tabla se disponen los datos obtenidos en el cálculo de ambas implementaciones para el promedio de la frecuencia necesaria para procesar los mensajes en enlace descendente, Down-Link, para cada una de las celdas, considerando también las macroceldas como estación base que puede dar cobertura a usuarios que así lo requieran.

Celda\Implementación	**Algoritmo 1**	**Algoritmo 2**
C1	**6,1238 GHz**	**6,1984 GHz**
C2	**5,5784 GHz**	**5,5858 GHz**
C3	**22,7614 GHz**	**22,1202 GHz**
C4	**15,3656 GHz**	**14,6171 GHz**
C5	**16,6974 GHz**	**16,4020 GHz**
C6	**33,7495 GHz**	**32,9197 GHz**
C7	**24,3119 GHz**	**22,7940 GHz**
C8	**7,2206 GHz**	**7,1653 GHz**
C9	**28,1870 GHz**	**27,3448 GHz**
C10	**20,2015 GHz**	**20,0443 GHz**
C11	**22,9198 GHz**	**22,3674 GHz**
C12	**24,3425 GHz**	**20,3753 GHz**

Tabla 6.1: Frecuencias medias necesarias por celda en cada implementación.

Los resultados que recoge la tabla, arrojados por el sistema de cálculo esclarecen que en siete de las doce celdas que conforman el escenario tenemos un requerimiento mayor en la frecuencia de cómputo cuando utilizamos el algoritmo generalizado, con el MCS promedio de la celda. Si bien es cierto que la diferencia de frecuencias en todas las celdas se reporta mínima, llegando como máximo a ser de 2,96GHz, esta diferencia puede resultar critica a la hora de elegir el número de CPUs a integrar en cada centro de cómputo para poder hacer un procesamiento paralelo de todas las celdas conectadas a cada centro de cómputo.
Con el fin de aportar más datos a estas mediciones, vamos a comprobar cuál es el MCS promedio que se le asigna a cada celda durante todo el cálculo, esto es, promediar los MCS medios de cada celda en cada instantánea. Realizando este cálculo se obtiene que el MCS medio total para cada celda es cada uno de los que se recogen en la siguiente tabla:

Celda	MCS medio por celda
C1	**9,4956**
C2	**9,2451**
C3	**10,0560**
C4	**8,1535**
C5	**10,6544**
C6	**10,4718**
C7	**8,0241**
C8	**9,9779**
C9	**11,0787**
C10	**11,4999**
C11	**11,8253**
C12	**8,4179**

Tabla 6.2: MCS medio entre todas las instantáneas para cada celda.

Así pues, vamos también a visionar el número medio de usuarios que tiene cada celda a lo largo de todas las instantáneas para las que se calculan estos resultados.

Celda	Número de usuarios medio por celda
C1	**1,45**
C2	**1,85**
C3	**13,55**
C4	**10,75**
C5	**13,45**
C6	**8,75**
C7	**14,8**
C8	**4,05**
C9	**14,45**
C10	**30,4**
C11	**20,6**
C12	**15,9**

Tabla 6.3: Número de usuarios medio entre todas las instantáneas para cada celda.

Con los datos que hemos extraído, podríamos pensar que las dos implementaciones podrían llegar a poderse considerar equivalentes, sin embargo, para poder demostrar esto aún se necesitan pruebas contundentes de que el dimensionado, entendido como el número de CPUs que necesitamos implantar en cada centro de cómputo, es salvo mínimas diferencias el mismo para las dos IMPLEMENTACIONES.

La complejidad de las implementaciones de cada algoritmo es otra de las métricas que se pretende utilizar. Por ello es que vamos a medir esta métrica mediante el uso del tiempo de ejecución que necesita el equipo sobre el que se desarrolla el trabajo para cada una de las implementaciones. En un primer momento se quiere puntualizar que el sentido común incita a pensar que la implementación más compleja en base a esta métrica va a ser la que individualiza a cada usuario para los cálculos.

El tiempo de ejecución de cada uno de estos bucles es el que se recoge en la tabla mostrada a continuación:

	Algoritmo generalizado	Algoritmo individualizado
Tiempo de ejecución medido	**96,39s**	**108,90s**

Tabla 6.4: Tiempo de ejecución de cada implementación.

A la vista queda con estos datos que la ejecución del algoritmo individualizado consume más ciclos de reloj en el procesador del equipo en el que se ejecuta que el algoritmo generalizado. Esto se debe a que dentro de los bucles que calculan todos los datos de salida, los que lo hacen para el individualizado deben ejecutar más vueltas del bucle debido a que no son solo una vuelta por celda, sino que es una vuelta por usuario de cada microcelda.

Hasta ahora se han comparado varias métricas de carácter importante para medir la veracidad y eficiencia de los algoritmos. Sin embargo, la métrica que definitivamente dará una respuesta a cuál es la implementación que da una imagen más fiel de lo que realmente necesitaríamos desplegar será la del número de CPUs necesarias trabajando paralelamente para poder procesar todos los mensajes a tiempo de cumplir el requisito que impone HARQ.

Con el fin de poder dilucidar esta cuestión, en la subsección donde se discutió el impacto del *TFH* en la frecuencia de cómputo que se requiere se determinó que los centros de cálculo se albergan en las estaciones base 1 y 2, que son las de las macroceldas y en ellas se determina que cada una constituye una BBU independiente una de otra. Atañendo el cálculo de CPU al criterio que se utiliza para determinar a que BBU se conecta cada BS definido en el anexo 1, vamos a designar

que la frecuencia de las CPUs que se utilizan es la frecuencia mínima posible, definida anteriormente en 2,7GHz.
Para llevar a cabo el cálculo la forma de proceder es la siguiente:

- Sumar las frecuencias medias requeridas por cada una de las celdas en cada BBU.
- Dividir entre la frecuencia mínima de CPU designada.
- Redondear al entero inmediatamente superior.

El redondeo introduce un sobredimensionamiento en el sistema. Realmente no tendrá ningún impacto negativo ya que al tratarse a lo sumo de una CPU adicional resultado de un cálculo con decimales en el que no podemos introducir CPU parciales o de frecuencia inferior, nos reducirá el tiempo de procesamiento sin introducir en el sistema un deterioro del rendimiento, ya que el factor de utilización seguirá siendo muy alto.

Algoritmo\BBU	BBU1	BBU2
MCS MEDIO	**47**	**43**
MCS INDIVIDUAL	**46**	**43**

Tabla 6.5: Numero de CPUs necesarias en cada implementación.

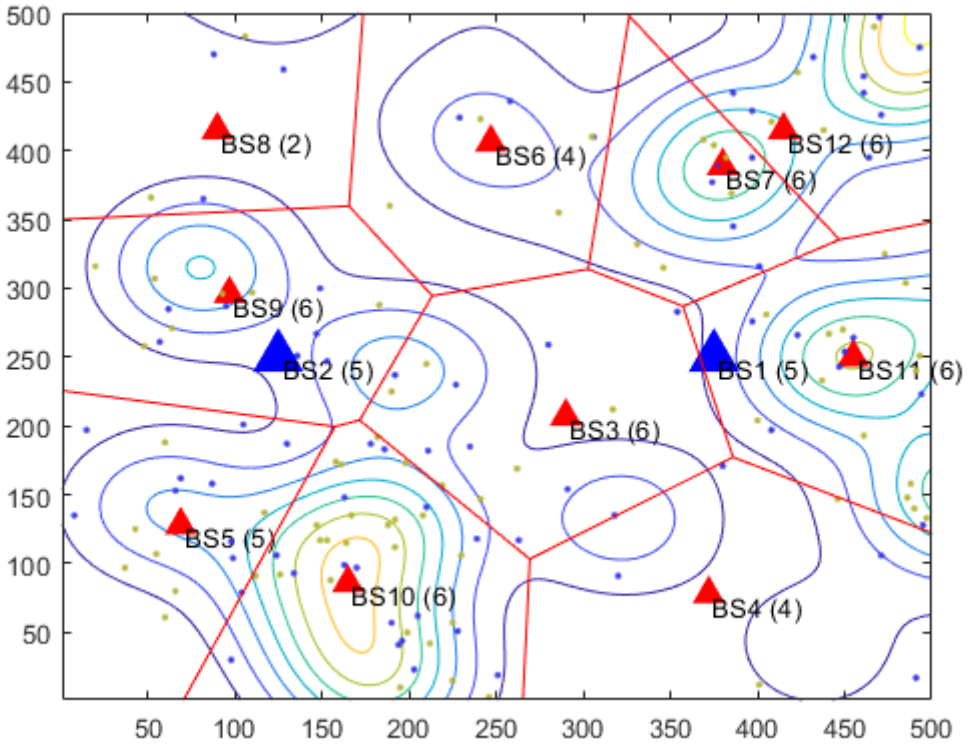
La tabla 6.5 muestra el número de CPUs necesarias a fin de que el procesado DOWNLINK se realice a tiempo. En este caso la diferencia entre CPUs es determinante para lo que es un factor tan crítico como la velocidad de procesado. Viendo la diferencia no se puede garantizar que las implementaciones sean equivalentes. En un escenario en el que la heterogeneidad del tráfico generado por sus usuarios sea de un orden del que actualmente puede suponerse en vista de la cantidad de contenidos que hay en la red, utilizar un MCS medio para todos los usuarios de la celda sería algo que no podría dar una imagen verosímil de la realidad y provocaría una bajada en la QoS de usuarios que requieran un tipo de tráfico más demandante en virtud de aquellos cuya demanda sea menos exigente.
Sin embargo, el dimensionamiento con el MCS individual de cada usuario sí que nos proporciona una visión real de cómo se comporta el escenario, pues la mencionada heterogeneidad del tráfico demandado y la eficiencia espectral de cada usuario hacen que el tratamiento individual de cada usuario sea no solo necesario si no imperativo.
Cabe destacar que el dimensionado solamente difiere en una CPU que estaría ligada al centro de cómputo 1, dicha CPU se observa en la celda C7, coincidiendo con la celda donde la diferencia de frecuencia en las implementaciones es máxima.

Conclusión final de la comparativa: en adelante se considerará solamente la implementación teniendo en cuenta el MCS individual de cada usuario, pues se considera una imagen más fiel de la realidad, así como una implementación más eficiente en vista de los tiempos de ejecución para el desarrollo de los cálculos necesarios. Como en un principio se pensó, esta es la implementación correcta y la que nos proporcionará los mejores resultados para calcular correctamente los recursos que se deben atribuir a la red.

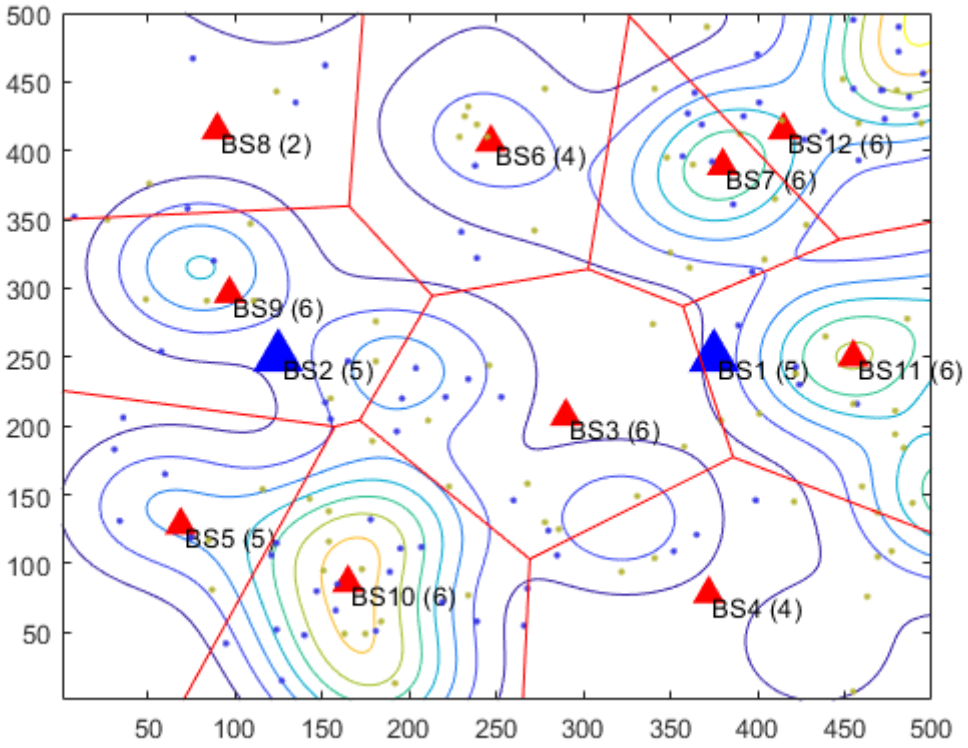
6.4.-Análisis del umbral de requisitos del sistema

Continuando con la búsqueda del mejor resultado de dimensionamiento para nuestro escenario, el siguiente paso se detalla en esta subsección. El objetivo principal de la misma es determinar cuál es el requerimiento de pico del sistema a lo largo de las instantáneas y el rendimiento mínimo. A lo largo de todas las instantáneas en las que se ejecuta el simulador, los usuarios representados en el sistema cambian de posición, consiguiéndose que no haya correlación entre la posición en la que se encuentran en una instantánea y cualquier otra. Dada esta aleatoriedad, resulta imposible predecir en cual de esas instantáneas vamos a tener mayor concentración de usuarios en una estación base concreta. Este problema se arregla a posteriori, cuando el simulador termina

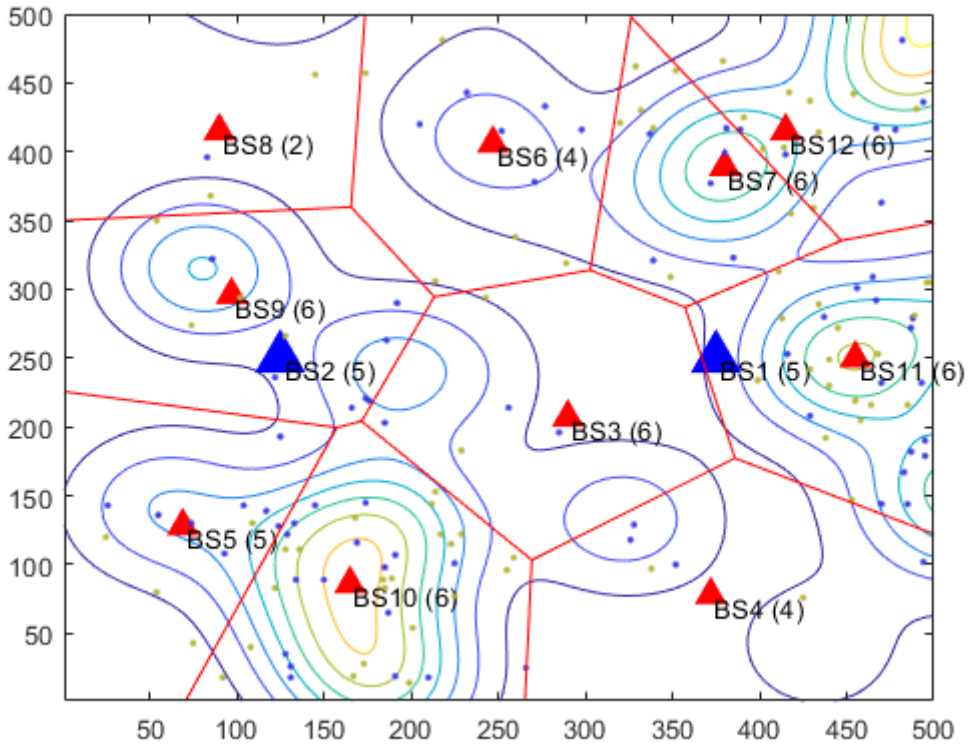
su ejecución y nos proporciona sus parámetros de salida en los que descubrimos la posición en cada una de las instantáneas de cada uno de los usuarios.



(A)



(B)



(c)

Figura 6.8: Disposición de los usuarios en distintas instantáneas.

En la figura 6.8 se ven tres instantáneas que ilustran lo que se expone en el párrafo anterior. Por simplicidad, se puede observar claramente en la BS4, que queda en la esquina inferior derecha concentra un mayor número de usuarios en *(B)*.
Son estas instantáneas las que buscamos. Se van a realizar cálculos el escenario de tal manera que se van a evaluar todas las instantáneas buscando cual es la demanda máxima y mínima por parte de cada una de las estaciones base. Una vez reunidas todas las demandas máximas y mínimas, trataremos los datos como si estuvieran simultáneamente demandando el máximo tráfico a lo largo de todas las instantáneas en una sola.
Este escenario no es un escenario realista, pues el número de usuarios hipotético seria mayor que el total de usuarios en el sistema en el caso de búsqueda de instantáneas máximas, y menor en el caso de buscar instantáneas que presenten demandas mínimas para cada estación base, pero se busca el factor de sobredimensionado que hemos introducido con el redondeo del número de CPUs calculadas.
Para calcular este número el procedimiento es idéntico al que se ha seguido en el cálculo de la subsección anterior. Aplicando de nuevo el mismo criterio para la conexión entre estaciones base y BBU.

Celda\Requisitos	MINIMOS	MEDIOS	MAXIMOS
C1	**0 GHz**	**6,1984 GHz**	**16,8282 GHz**
C2	**0 GHz**	**5,5858 GHz**	**16,5321 GHz**
C3	**6,2357 GHz**	**22,1202 GHz**	**34,0175 GHz**
C4	**0 GHz**	**14,6171 GHz**	**23,5381 GHz**
C5	**0 GHz**	**16,4020 GHz**	**26,5622 GHz**
C6	**14,7973 GHz**	**32,9197 GHz**	**44,4555 GHz**
C7	**9,1706 GHz**	**22,7940 GHz**	**35,1162 GHz**
C8	**0 GHz**	**7,1653 GHz**	**13,9418 GHz**
C9	**15,25 GHz**	**27,3448 GHz**	**39,6330 GHz**
C10	**6,9033 GHz**	**20,0443 GHz**	**31,1052 GHz**
C11	**5,2235 GHz**	**22,3674 GHz**	**33,6723 GHz**
C12	**5,2964 GHz**	**20,3753 GHz**	**32,4811 GHz**

Tabla 6.6: Umbral de requisitos del sistema.

Vemos en la tabla como hay algunas celdas, en las que cabe destacar las macroceldas, en las que la frecuencia requerida es de 0GHz para la situación de mínimo requisito. Esto puede ser debido a dos razones: la primera es que no haya ningún usuario conectado a la estación base de la macrocelda porque este siendo servido por una estación base de microcelda. La otra razón, también valida al igual que la anterior para la celda C8, es que el o los usuarios conectados a esta celda en esa precisa instantánea tengan asignado el MCS = 0 y por lo tanto no sean cursados, lo cual hace que la contribución de la celda al cálculo de la frecuencia necesaria sea nula.
Puede apreciarse también que los valores medios se acercan generalmente más a los máximos que a los mínimos. Este indicador resulta relevante a la hora de evaluar si el escoger la frecuencia media calculada es una buena opción, ya que al estar cerca de la frecuencia máxima que llega a necesitar la celda, el sistema difícilmente colapsaría. Para el escenario concreto que se está simulando en este trabajo se han tomado 10000 instantáneas para la simulación, a fin de que los datos representen una reserva de recursos justa, eficiente y con un factor de utilización lo más alto posible.
Estos datos recogidos en la tabla 6.6, cobran mayor representación si en lugar de recogerlos así vemos el rango de CPUs que son necesarias para obtener esa frecuencia.

NCPUS\REQUISITOS	MINIMOS	MEDIOS	MAXIMOS
BBU 1	**17**	**46**	**67**
BBU 2	**14**	**43**	**63**

Tabla 6.7: Umbral de CPUs requeridas en el sistema.

Con estos resultados vemos como de nuevo los valores medios están más cerca de los máximos que de los mínimos, esto es un indicador bueno a la hora de saber si el dimensionado es suficiente, debido a que se aproxima al máximo requerimiento del sistema a lo largo de las 10000 instantáneas.

6.5.-Impacto de los usuarios con baja eficiencia espectral.

Esta quinta sección de los resultados estará dedicada a comprobar el impacto que los usuarios a los que se le asignó el MCS = 0 tendrían en el dimensionado del sistema.
Si nos remontamos a la ecuación 4.17, vemos que el MCS es un factor determinante a la hora de calcular la contribución de cada usuario al cálculo de la frecuencia de cómputo requerida. Es el factor más importante dentro del sumatorio que se debe multiplicar por el cociente de los PRB y

el cuadrado de la frecuencia.
Podemos esperar que al tener los usuarios asignados el MCS = 0, su contribución va a ser nula pues el resultado del sumatorio debe ser cero. Esto debería traducirse en la no afectación por parte de ese usuario a los cálculos, solamente afectaría en el tiempo de ejecución, donde introduciría más ciclos de reloj que son innecesarios.
Sin embargo, mirando de nuevo la ecuación 4.17, hay que hacer una potencia dentro del sumatorio, en este caso la base de la potencia es 0 y el exponente es 0 para la primera iteración en su cálculo. Esto llevaría a una indeterminación que el software MATLAB resuelve de la siguiente manera:

$$0^0 = 1$$

ECUACIÓN 6.3: RESOLUCIÓN DE INDETERMINACIÓN.

Siendo la indeterminación resuelta de esa forma, la contribución de los usuarios con MCS = 0 sí que va a tener un impacto en la frecuencia de cómputo que necesitamos, pues aun siendo usuarios que no se les dará servicio, hacen que la frecuencia que requerimos aumente sin necesidad, solamente porque se planifican sin que su tráfico sea procesado y se ha visto anteriormente el significado de tener asignado un MCS = 0. Lo cual resulta contradictorio y erróneo, que es lo que se quiere demostrar en esta sección.
Lo expuesto en el anterior párrafo se traduciría en un error de dimensionamiento, pues no haría otra cosa que introducirnos una subida en la frecuencia calculada haciendo el dimensionado ineficiente y con un plus en los costes del sistema. Para comprobar la veracidad de esto se procede a mostrar el resultado de una batería de pruebas relativas a esta comprobación.
Para entender lo que se va a mostrar en los próximos párrafos, es necesario saber cuántos usuarios con MCS = 0 tiene cada celda.

CELDA	USUARIOS MCS = 0
C1	**0**
C2	**0**
C3	**2**
C4	**0**
C5	**1**
C6	**1**
C7	**3**
C8	**0**
C9	**0**
C10	**0**
C11	**3**
C12	**2**

Tabla 6.8: Usuarios con MCS = 0 en cada celda.

Para verlo con datos analíticos realizamos la comparación en la siguiente tabla, donde se muestran las frecuencias para cumplir el retardo con 2ms.

CELDA	Frecuencia sin MCS=0	Frecuencia con MCS=0
C1	**6,1984 GHz**	**6,125GHz**
C2	**5,5858 GHz**	**5,5815 GHz**
C3	**22,1202 GHz**	**22,9259 GHz**
C4	**14,6171 GHz**	**15,5911 GHz**
C5	**16,4020 GHz**	**16,8632 GHz**
C6	**32,9197 GHz**	**33,5179 GHz**
C7	**22,7940 GHz**	**24,5522 GHz**
C8	**7,1653 GHz**	**7,2753 GHz**
C9	**27,3448 GHz**	**28,1644 GHz**
C10	**20,0443 GHz**	**20,1917 GHz**
C11	**22,3674 GHz**	**23,0513 GHz**
C12	**20,3753 GHz**	**21,6021 GHz**

Tabla 6.9: Diferencias de frecuencias teniendo en cuenta usuarios con mcs = 0.

En la tabla se aprecia como las frecuencias varían muy poco, pese a que en algunas es más acentuada la variación. Esto es debido a la baja cantidad de usuarios en el sistema que tendrán MCS = 0 en distintas instantáneas.
Podemos llegar a la conclusión de que finalmente no introducirá una diferencia realmente significativa el considerar estos usuarios, sin embargo, en ocasiones puntuales sí que puede ser un problema considerarlos, sobre todo cuando la celda tenga pocos usuarios conectados y estén muy alejados o en zonas de baja cobertura.

Por último, vamos a ver el impacto en el número de CPUs por centro de cómputo que producen estos usuarios al ser considerados, aunque de la tabla 8.10 podemos deducir que la diferencia puede ser no mínima, sino incluso nula.

NCPUS\REQUISITOS	MINIMOS	MEDIOS	MAXIMOS
BBU 1 NO MCS=0	**17**	**46**	**67**
BBU 2 NO MCS=0	**14**	**43**	**63**
BBU 1 MCS = 0	**20**	**47**	**68**
BBU 2 MCS = 0	**16**	**44**	**64**

Tabla 6.10: Comparación Umbral de CPUs requeridas en el sistema.

El número de CPUs necesarias para hacer los cálculos varía en todos los casos debido a la contribución de los usuarios que no se planifican, porque, aunque su contribución sea poca, la resolución que aplica el software a la indeterminación presente en la ecuación 6.3 hace que se tomen en cuenta. Esto pone de manifiesto que el cálculo con los usuarios que tienen asignado el MCS=0 puede introducir un error de dimensionado bajo ciertos parámetros aleatorios como la ubicación de usuarios y la potencia de señal que es capaz de hacer llegar la antena de la estación base a la posición de estos usuarios.

6.6.-Comparacion de las necesidades de cada celda

Continuando con la extracción de resultados, el siguiente punto al que se dirige el trabajo es a comparar los requerimientos de cada celda. Esto es importante en el sentido de que con esta comparativa podemos ver la procedencia de la demanda más alta de trafico de entre todas las celdas. Resulta útil para determinar donde se podría construir una microcelda aún más pequeña,

donde se necesitaría un despliegue provisional por un pico de tráfico debido a una concentración alta puntual y muchas más aplicaciones.
Partiendo del simulador, este nos proporciona una división trazando las áreas de Voronoi. Estas áreas se construyen en base a un número determinado de puntos en el mapa, en nuestro caso el escenario del despliegue de la infraestructura simulada. A partir de ahí, se trazan en el mapa una serie de áreas que cumplen el criterio de reunir los puntos del mapa más cercanos a un determinado punto. Traduciendo esto al escenario que tenemos desplegado, los puntos clave serían las microceldas, no se incluyen las macroceldas en el trazado de Voronoi. Con esto, una función incluida en MATLAB nos ayuda a solucionar este trazado a partir de las coordenadas de las estaciones base de microceldas. La función calcula la distancia de cada punto del mapa a cada una de las microceldas y divide mediante polígonos el mapa. En esos polígonos se encuentran confinados los puntos más cercanos a cada estación base. Para esclarecerlo, veámoslo sobre una de las figuras.

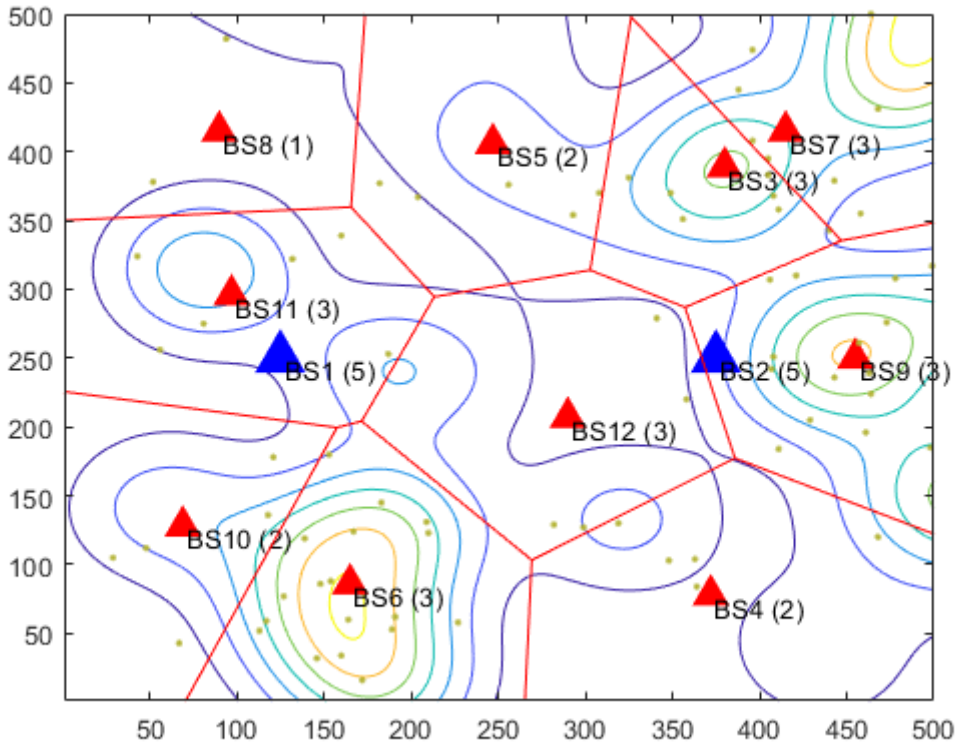


Figura 6.9: Áreas de Voronoi.

Fijémonos por ejemplo en la BS12. Como se puede observar, su área de Voronoi es un hexágono no necesariamente centrado en la estación base. Sin embargo, cualquier punto inscrito en dicho hexágono guarda una distancia menor con la BS12 que con cualquier otra en el mapa sin incluir las estaciones de macroceldas. Los puntos de frontera, delineados en rojo podemos decir que están a una distancia idéntica entre las estaciones base contiguas.
En un primer momento resulta obvio pensar que estas áreas de Voronoi pueden coincidir con las áreas de cobertura de cada celda. Sin embargo, aunque esto pudiera ser cierto, se ha desarrollado un script para delimitar las áreas de cobertura de cada celda en base a otro criterio que se ha estimado de mayor importancia. Este es la potencia recibida por cada antena de la estación base. Este cálculo de las áreas de cobertura se realizará evaluando en cada punto del mapa cual es la estación base de la que procede la mayor potencia de señal recibida. Dicha potencia se mantiene a lo largo de todas las instantáneas y la podemos extraer del parámetro de salida del simulador. A esta potencia se le resta un factor de amortiguación en decibelios.

Una vez calculada esta potencia recibida se dibujan las áreas de cobertura.

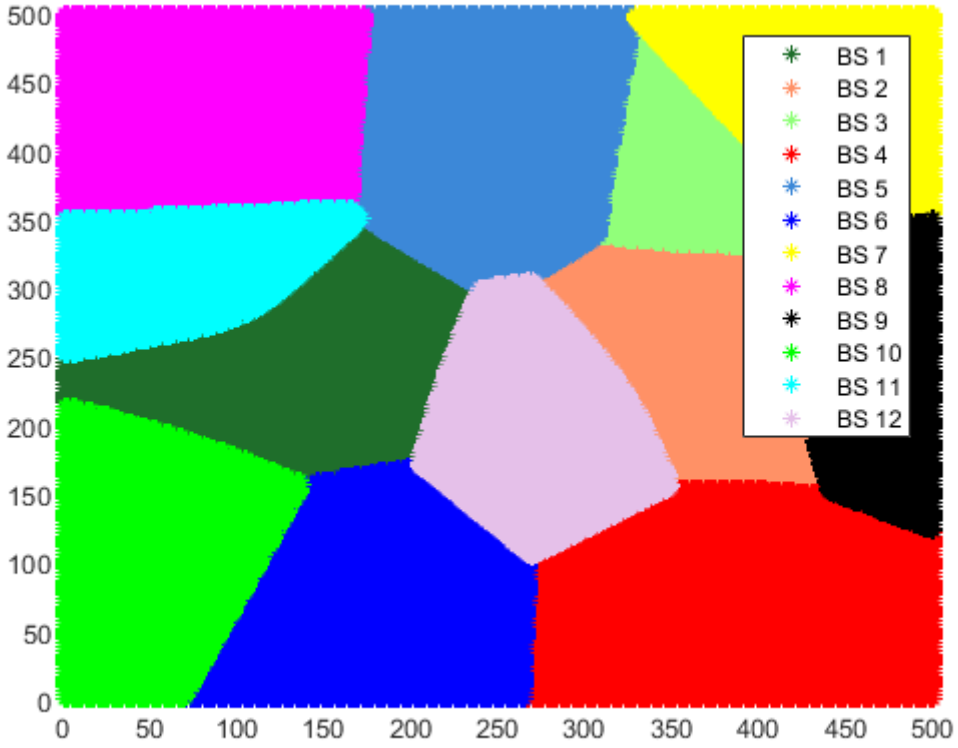


Figura 6.10: Áreas de Cobertura.

Las áreas de Voronoi no tenían en cuenta las zonas de cobertura de las estaciones base de macroceldas, sin embargo, al haber comprobado con anterioridad que estas sí que dan servicio a usuarios, es importante incluirlas en el cálculo de las zonas de cobertura.
Para dar una idea de a cuanta área dan cobertura las distintas estaciones base, se muestra en la siguiente tabla el número de puntos en el mapa servidos por cada estación base.

CELDA	m^2 servidos
C1	98604
C2	82668
C3	46528
C4	130512
C5	106948
C6	99080
C7	73992
C8	98440
C9	57632
C10	83148
C11	50020
C12	72440

Tabla 6.11: Superficie servida por cada estación base.

El número medio de usuarios que toman cobertura de cada celda se recoge como sigue:

CELDA	NUMERO MEDIO DE USUARIOS
C1	**1,5112**
C2	**1,317**
C3	**14,027**
C4	**10,321**
C5	**11,106**
C6	**30,046**
C7	**17,145**
C8	**4,4258**
C9	**20,585**
C10	**12,607**
C11	**13,824**
C12	**13,085**

Tabla 6.12: Número de puntos servidos por cada estación base.

Con ayuda del coeficiente de correlación de Pearson, vamos a ver la relación que existe entre el área de cobertura de una celda y el número de usuarios que tiene. Para calcular este coeficiente se hará uso de la ecuación estadística que permite calcularlo:

$$r_{xy} = \frac{\frac{\sum XY}{N} - \bar{X}\bar{Y}}{S_x S_y}$$

ECUACIÓN 6.4: COEFICIENTE DE CORRELACIÓN DE PEARSON.

Aplicando la ecuación 6.4 a las variables que se están manejando aquí:

- X representaría el área de la celda.
- Y el número de usuarios conectados en cada celda.

Tras realizar el cálculo, resulta en una relación entre el número medio de usuarios y el área de la celda de ***-0,22589***, por lo que podemos convenir que en este escenario no existe una correlación representativa entre las variables evaluadas, que son el número de usuarios medios que sirve cada celda y el área de cobertura que alberga la misma.

Por su parte si lo calculamos para la frecuencia de cómputo necesaria para satisfacer las necesidades de cada celda y el área de la celda, el valor obtenido es de ***-0,3897*** lo cual no nos sirve para decir que la correlación sea fuerte como para considerar dependientes las variables.

En cualquier caso, para nuestro escenario la relación puede establecerse ligeramente inversa entre el número de usuarios, la frecuencia y el área de la celda, sin llegar a tomar esta relación como un dato totalmente fiable.

Finalmente, la comprobación que queda por hacer es si en las celdas con mayor número de usuarios se necesitan frecuencias de cómputo mayores. Se puede pensar que estas variables tendrán una correlación fuertemente positiva, pero la intervención del MCS en este cálculo puede hacer que ese primer pensamiento sea erróneo. En cualquier caso, para nuestro escenario la relación entre el número de usuarios medio por celda y la frecuencia necesaria por celda para cumplir el HARQ es de ***+0,9729***, lo cual confirma la teoría de la estrecha relación entre ambas, pues por poca que sea la aportación de un usuario, siempre añadirá frecuencia necesaria para su procesado.

A la vista de los resultados no se puede concluir que las celdas de área mayor necesiten una frecuencia de cómputo más elevada para satisfacer las fronteras de retardo que estamos estableciendo, pues más que de la superficie va a depender en mayor medida de la distribución de los usuarios dentro de la celda, así como de la concentración de los mismos.

6.7.-Cálculo de la demanda de recursos de cada slice

El simulador del cual parten los cálculos en este trabajo está diseñado de tal manera que, al

ejecutarlo para diseñar un escenario concreto, conforme a las directrices que se le introducen en los distintos scripts de configuración que ya se detallaron en la sección correspondiente al simulador, implementa una serie de slices para los cuales se pueden variar también los parámetros.
En el caso de la simulación de la que se parte para todo el desarrollo del trabajo se implementan dos slices, llamados a lo largo de toda la ejecución del simulador tenants debido a que los tenants son las entidades que solicitan el slice a la red. A cada tenant se le asigna un porcentaje de usuarios cuya naturaleza del tráfico será la misma que el slice al que pertenecen está preparado para dar servicio.
El objetivo de esta sección es mostrar los resultados obtenidos del estudio por separado de cada slice a fin de compararlos y ver lo que distintas naturalezas en el tráfico pueden impactar en la frecuencia de calculo que necesitamos.
Hasta ahora no se había tenido realmente en cuenta al tenant que pertenecen los usuarios, al menos no explícitamente. Sin embargo, hay más factores que intervienen indirectamente en la frecuencia de cómputo que sí que están relacionados con el slice al que pertenecen.
Dentro de lo que se puede considerar puramente configuración de los slices, hay una serie de parámetros que debemos tener en cuenta especialmente:

- El número de tenants que queremos implementar, lo cual influirá en la diversidad del tráfico que se cursará en los distintos centros de cómputo.
- El porcentaje de los usuarios que pertenecen a cada tenant. En nuestro caso y por una cuestión equitativa se va a determinar en un cincuenta por ciento para cada tenant. Esto nos dará también una comparación más justa y menos sesgada.
- El número de canales que se le asigna a cada tenant dentro de una celda. En función de cuantos usuarios haya en cada celda pertenecientes a cada tenant se le asignará un numero de canales a cada tenant.
- El factor de correlación entre la distribución del tráfico que modelan ambos tenants, que en este caso se ha seteado al mínimo nivel posible, por tanto, el tráfico tendrá características aleatorias entre ambos tenants.

Con estas características podremos ver una comparativa en la que no se otorga ningún tipo de prioridad ni privilegio a ningún tipo de tráfico y con el que se conseguirá un reparto más igualado de los recursos.
Los usuarios pertenecientes a cada slice siguen una distribución aleatoria y por tanto tienen la misma probabilidad de pertenecer tanto a uno como a otro, con lo cual el primer tenant tendrá un cincuenta por ciento de los usuarios y el otro tenant tendrá el cincuenta por ciento restante.

6.7.1.-Procedimiento para el cálculo de las necesidades de cada tenant

El proceso seguido para calcular en cada tenant los recursos relativos al cómputo que necesita es muy similar al que se sigue para calcular la frecuencia mediante el algoritmo individualizado a cada usuario.
En esta ocasión, al no tener todas las celdas los mismos PRB disponibles, se hace también una distinción por celdas y tenant.
En una matriz tridimensional se guarda para cada instantánea cuanta de la frecuencia necesaria para cada celda está relacionada con cada tenant. Para ello se hace una comprobación de a que tenant pertenece cada usuario en el instante antes de acumular su aportación al total de la celda. Así pues, se separa por celdas y tenant como antes se advirtió.
Para obtener un cálculo con sentido, esto se hace para todas las instantáneas en las que se simula, con objeto de una vez terminados todos los cálculos promediar exactamente igual que se hizo cuando se trataba de la frecuencia de cada celda.

Por tanto, una vez finalizados los cálculos tendremos un dato que nos informara de la frecuencia necesaria para realizar todo el procesado del tráfico de cada tenant cumpliendo los requisitos temporales que impone HARQ.

6.7.2.-Resultados del estudio

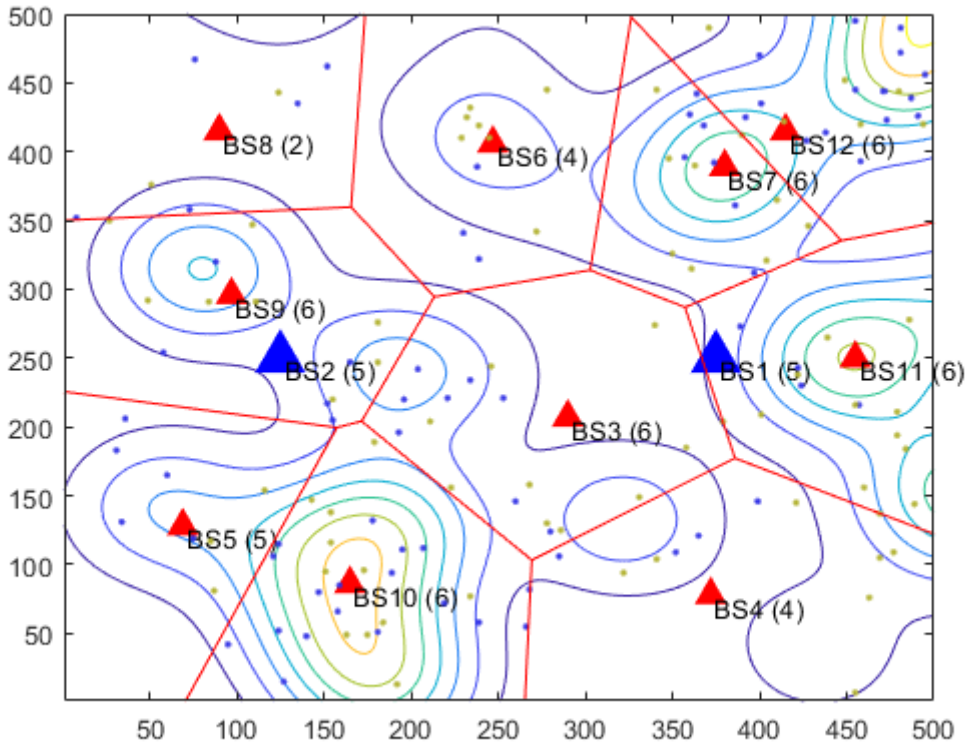
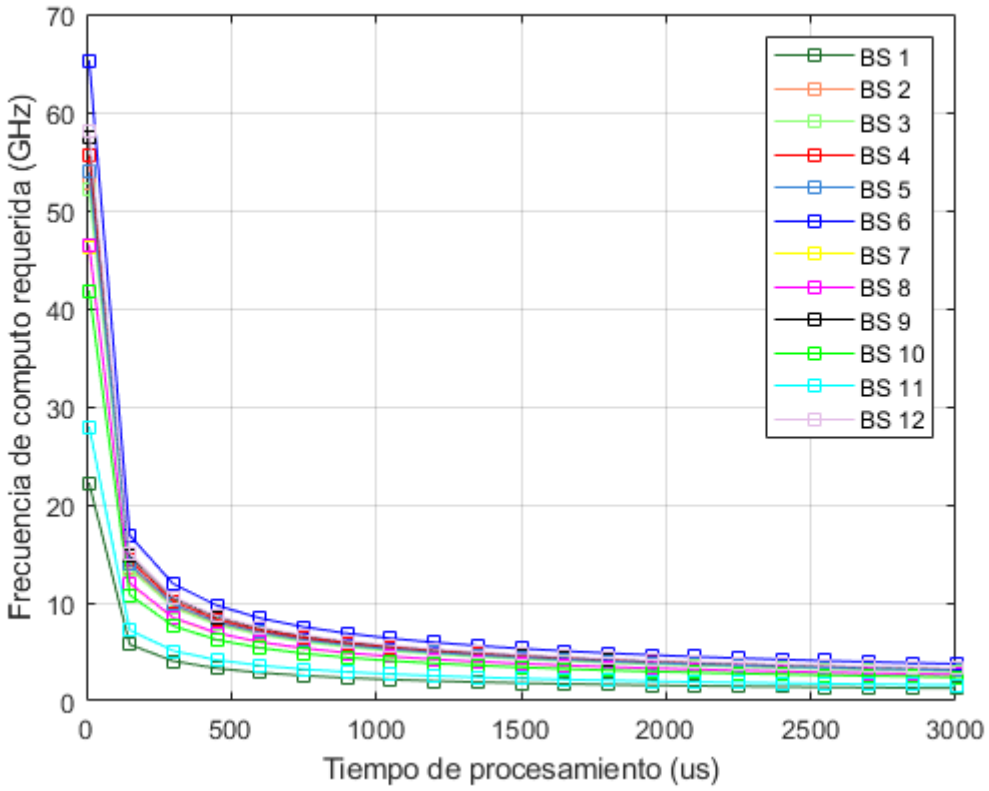
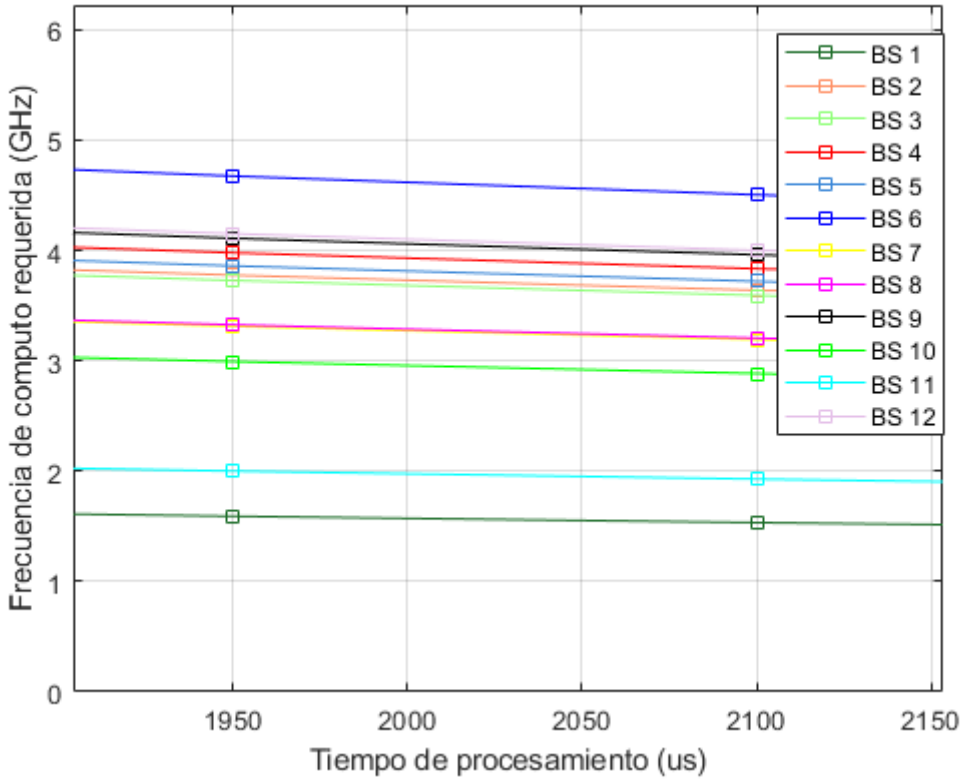


Figura 6.11: Mapa de los usuarios pertenecientes a cada tenant.

En la figura 6.11 podemos ver representado a cada usuario con un punto en el mapa cuyo color puede variar en función del tenant al que pertenecen. Los de color verdoso están generando tráfico perteneciente a un slice y los de color azul generan datos que se cursan como otro tipo de tráfico.
Como puede observarse la distribución es aleatoria e independiente de un usuario a otro. Por tanto, una vez más se busca la máxima uniformidad en la distribución sin sesgar nada.
Los resultados obtenidos para la frecuencia que separadamente demanda cada tenant para cumplir la frontera que impone el retardo HARQ se pueden ver en la figura que se muestra a continuación.

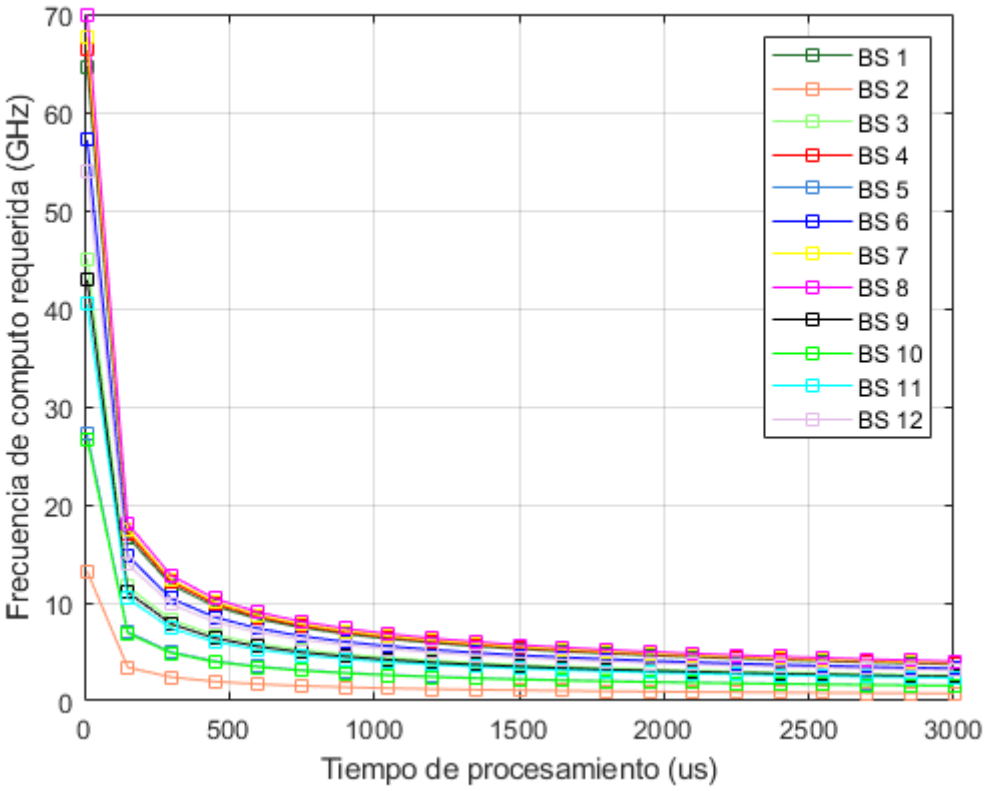


(A)

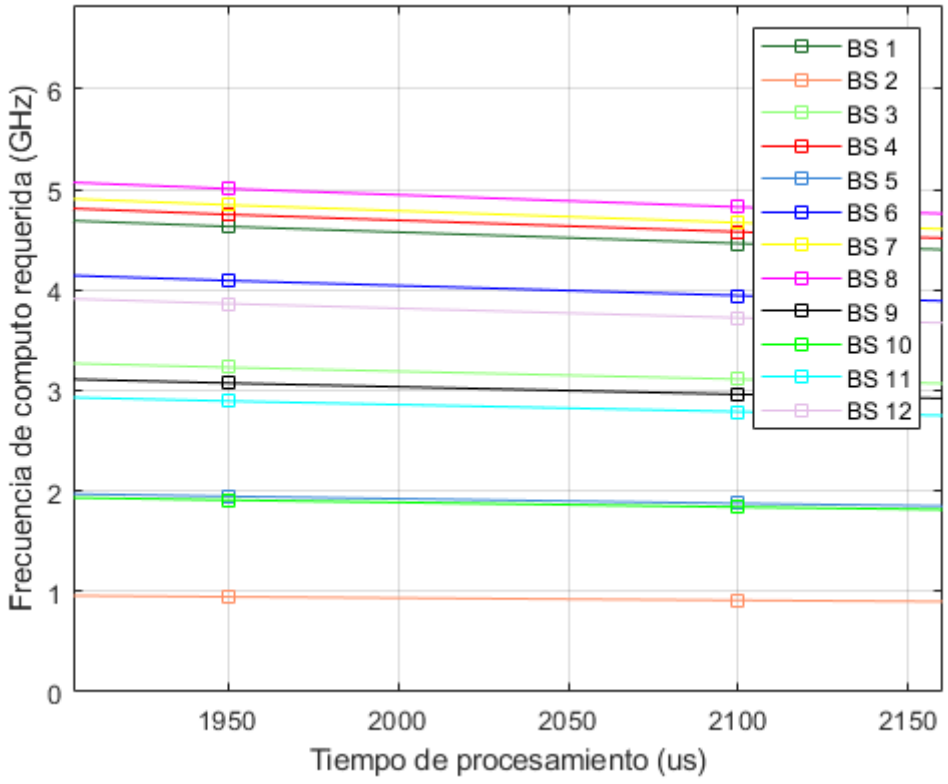


(B)

Figura 6.12: Frecuencia requerida para cumplir HARQ para el primer slice.



(A)



(B)

Figura 6.13: FRECUENCIA requerida para cumplir HARQ para el segundo slice.

Se nota a simple vista que la frecuencia necesaria para cada slice por separado es sensiblemente menor que la que se requiere cuando se hace el cálculo para toda la celda, ya sea teniendo en cuenta el MCS de cada usuario por separado o promediando el MCS para la celda.
No obstante, el tráfico que hay que procesar es todo, tanto del primer como del segundo tenant y para todas las estaciones base. A continuación, se muestran las frecuencias medias que requieren los dos slices juntos frente a la que hay disponible en las celdas.

CELDA\CALCULO	SLICES	CELDAS
C1	**6,1457 GHz**	**6,1984 GHz**
C2	**4,6634 GHz**	**5,5858 GHz**
C3	**6,8708 GHz**	**22,1202 GHz**
C4	**8,6218 GHz**	**14,6171 GHz**
C5	**5,7340 GHz**	**16,4020 GHz**
C6	**8,6585 GHz**	**32,9197 GHz**
C7	**8,0586 GHz**	**22,7940 GHz**
C8	**8,2308 GHz**	**7,1653 GHz**
C9	**7,0938 GHz**	**27,3448 GHz**
C10	**4,8412 GHz**	**20,0443 GHz**
C11	**4,8336 GHz**	**22,3674 GHz**
C12	**7,9161 GHz**	**20,3753 GHz**

Tabla 6.13: Comparativa frecuencia requerida calculando por slice y por celda.

La tabla 6.13 nos muestra que las frecuencias que se requieren por parte de los slices son menores que las que se dimensionó cuando se aplicó el algoritmo individualizando a cada usuario. Esto nos lleva a comprobar que efectivamente los centros de cómputo tienen un pertrecho mayor para poder satisfacer las necesidades de cada slice.

TENANT\BBU	BBU 1	BBU 2
TENANT 1	**10**	**12**
TENANT 2	**11**	**10**
TOTAL	**21**	**22**

Tabla 6.14: CPUs necesarias en cada BBU al calcular por tenant.

Valorando los valores mostrados en la tabla anterior, podemos concluir que cada tenant por separado requerirá un numero de CPUs muy cercano al mínimo que se calculó para las celdas en la sección 6.3. Teniendo en cuenta que se necesita procesar todo el tráfico que se genere, el número de CPUs necesarias encumbra en 21 y 22 para el centro de cómputo 1 y 2 respectivamente, lo cual es sensiblemente menor que lo que se requiere cuando el cálculo se hace para cada celda aplicando compartición de recursos entre slices, pues al separar recursos por slice, baja el número de CPUs que necesitaremos en lo centros de cómputo esto nos lleva a pensar que el dimensionado de los recursos se ha hecho correctamente pues se pueden albergar ambos slices sin poner en jaque la estabilidad de la red pues los recursos disponibles son mayores que los que se demandan.

6.8.-Análisis de la variación dinámica de la demanda.

Finalmente, la última cuestión que se pretende resolver es el cómo afecta que gradualmente se añadan usuarios al sistema. Es por ello que se ha querido nombrar así a esta última sección de resultados en la que se quiere recoger y describir el comportamiento del escenario y los elementos que lo conforman a medida que van creciendo las necesidades.

Esta sección podría servir como propuesta base para tomar ideas y poder aplicar algo que absorbe la dedicación de muchas de las investigaciones en informática y computación en este tiempo, el machine learning.
Lo que se va a tratar de descubrir sigue un proceso minucioso, arduo y con una complejidad elevada en su programación debido a la gran cantidad de parámetros que gobiernan los elementos que se van a ver envueltos en este proceso de evaluación de la evolución de lo que necesita nuestra red a nivel computacional.
Vamos a conseguir describir cómo se comporta el sistema en materia de requisitos abstrayéndonos a la posibilidad de que no haya ningún usuario en el sistema y todo este en reposo. Partiendo de esa base se introducirán periódicamente usuarios en el sistema, que de forma aleatoria aterrizan en el sistema y por su acción las necesidades de la celda en la que toman servicio al entrar en el escenario cambian en consonancia con los parámetros que se asignan al usuario, concretamente su eficiencia espectral y el MCS que le asignamos en función de ésta. Una vez introducidos en el sistema se evaluará si al entrar en escena los nuevos usuarios el sistema es capaz de darles servicio y procesar sus mensajes en el tiempo que establece el HARQ.
A partir de aquí aparece la disyuntiva clave en el comportamiento que tendrá la evolución de los requisitos computacionales y como si se tratase de una clave dicotómica se abren dos vías posibles: que el tiempo de procesado esté dentro del umbral permitido o que este tiempo haya sobrepasado el límite superior del umbral.

6.8.1.-Tiempo de procesado mayor al permitido

En el caso de que una vez el sistema recalcule el rendimiento del sistema una vez añadidos los nuevos usuarios y el tiempo de procesado obtenido con la frecuencia de la que se dispone en la BBU supere el umbral de HARQ, el dimensionado seria inservible puesto que no estaríamos proporcionando la QoS que se espera de un sistema 5G.
A raíz de este hecho, se añade una CPU de la frecuencia que se haya determinado al centro de cómputo.
Esto permite que de nuevo el umbral de tiempo de procesado baje del requisito HARQ y el procesado se complete en un tiempo menor de la frontera que se impone.
Así, una vez se tenga añadida esa nueva CPU en el sistema, integrada en el centro de cómputo correspondiente, de nuevo el tiempo que se tarda en procesar experimentará una bajada considerable por debajo del tiempo que se ha estipulado para el enlace descendente, que es de dos milisegundos.
Al conseguir un tiempo de procesado menor al límite, podremos aplicar el mismo procedimiento que se sigue en la **sección 6.8.2**.

6.8.2.-Tiempo de procesado dentro del permitido

Este procedimiento será el que se siga siempre que las condiciones sean favorables. Entendiéndose por favorables aquellas condiciones en las que el tiempo de procesado sea menor que el umbral.
De nuevo se remonta el planteamiento de este procedimiento al cálculo del tiempo que se tardaría en procesar los mensajes una vez se ha producido un incremento en el número de usuarios. Contrario a lo que versa en la sección 6.8.1, se obtiene que el tiempo de procesado es menor que el umbral de HARQ y por tanto se cumplen las condiciones favorables en el dimensionado. Éste seguiría funcionando con total normalidad pues aún sería capaz de prestar un servicio acorde a la QoS que se espera del sistema 5G.
Por tanto, al tener las condiciones a favor, se continua con el mismo número de CPUs integradas en los centros de cálculo y se espera a la llegada de nuevos usuarios para reevaluar la calidad de las condiciones.
En el caso de que las condiciones cambiaran a no favorables, se seguiría el procedimiento descrito

en la **sección 6.8.1**.

Para la implementación, el siguiente organigrama es de gran ayuda para visualmente esquematizar como debe de funcionar el programa encargado a tal efecto:

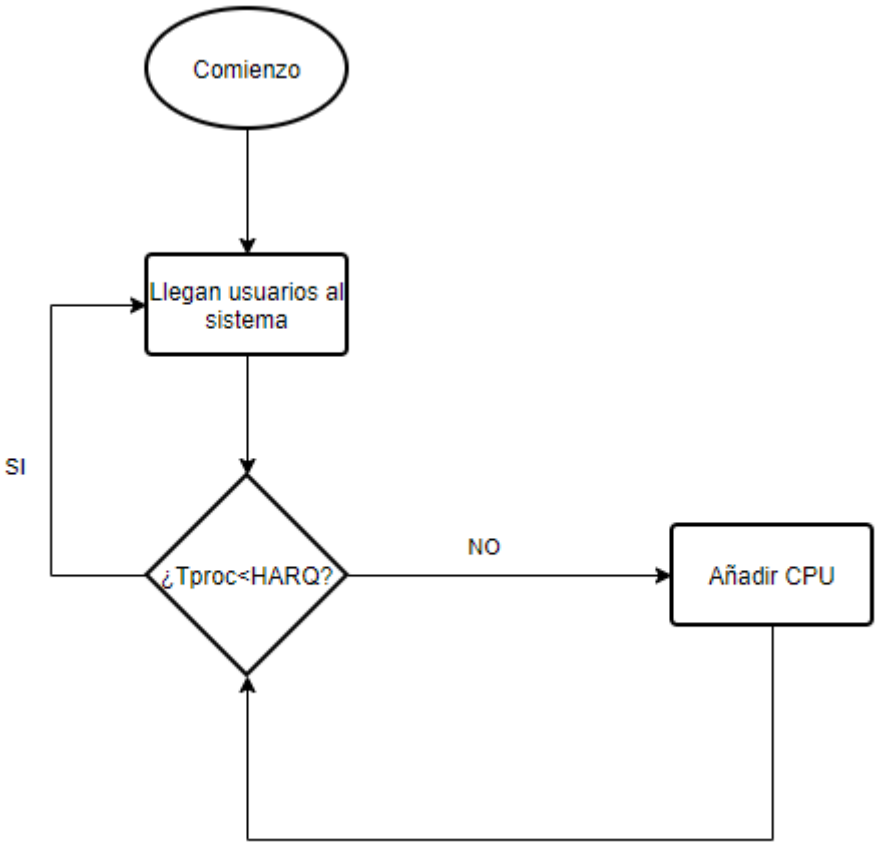


Figura 6.14: Organigrama para la evolución del sistema.

Siguiendo este sencillo organigrama en el que se omiten muchos detalles profundos, la evolución se implementara siguiendo una serie de pasos insalvables:

1. Partiremos para los cálculos del dimensionado mínimo por celda con la implementación del algoritmo individualizado.
2. Se parte de un numero de cuatro usuarios en el sistema.
3. En condiciones favorables se añadirán dos usuarios al sistema.
4. En condiciones desfavorables, se añade una CPU para el control de la estación base en la que más se demore el procesado aun no siendo superior en esta a los dos milisegundos, pero si en la suma de todas las estaciones base conectadas a un determinado centro de cómputo.

La clave del proceso reside en que cada vez que se añaden usuarios al sistema, proveniente este cambio de condiciones favorables, se simula para todas las iteraciones que se le dieron como parámetro de entrada al simulador y se promedia, obteniendo un tiempo de procesado para los usuarios presentes en el sistema en el momento actual.
Ese valor medio crecerá a medida que vaya creciendo el monto de usuarios en el sistema, por lo que cada vez se acercará más a la frontera para el tiempo de procesado.
Cuando ocurra esto, se añadirá una CPU relativa a la estación base cuyo tiempo de procesamiento sea mayor de entre todas las conectadas a la BBU donde el tiempo de procesado haya excedido lo permitido.

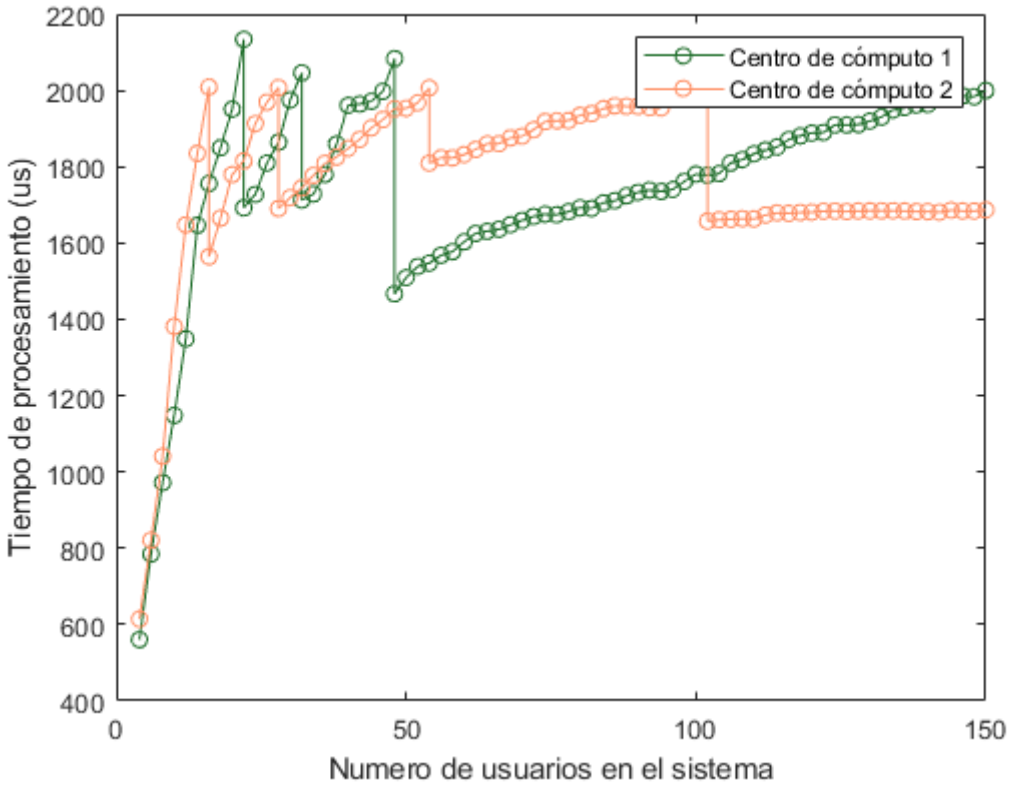


Figura 6.15: Evolución del tiempo de procesado en cada centro de cómputo.

La curiosa forma de sierra de la gráfica viene dada por los cortes cuando las condiciones no son favorables, en los que recordemos se añade una CPU en la BBU correspondiente y se recalcula todo el tiempo que se tarda en procesar, obteniéndose así esos "dientes de sierra".
Si analizamos bien la forma de la gráfica surgen varias cuestiones a las que se antoja de importancia considerable.
La cantidad de usuarios que entran en el sistema entre pico y pico no siempre es la misma, o sea, no se aprecia periodicidad en la aparición de los picos. Esto es debido a la aleatoriedad del lugar de aterrizaje en el sistema de los usuarios que entran en él. Cuando entran, al aterrizar en el escenario en una ubicación aleatoria, puede darse el caso de que ambos se conecten a estaciones base procesadas por la misma BBU o pueden ser distintas. En función de donde caiga el par de usuarios, la diferencia de espacio entre picos será mayor o menor.
Esto se ve especialmente recalcado a partir de los 48 usuarios en el sistema, instante del ultimo pico en el centro de cómputo 1. Se observa como la pendiente es más o menos constante en el crecimiento en este centro de cómputo y por el contrario en el segundo centro de cómputo se observan dos picos más, que dan paso a una curva casi plana. Esto se debe a lo recién comentado, por méritos de la aleatoriedad en la ubicación de los usuarios, se ha producido la caída únicamente en áreas cuya estación base está conectada al primer centro de cómputo y por tanto la pendiente se mantiene.
Lo mismo puede aplicarse a la demora en aparecer del primer pico en ambas curvas, sin embargo, aquí afectan más factores como puede ser el MCS de cada usuario.

Capítulo 7.- Conclusiones

7.1.-Conclusiones

Todos los resultados de las implementaciones se han hecho modelando la red sobre el paradigma de CLOUD RAN, pues hemos conseguido que las estaciones base consten de unidad distribuida y una unidad centralizada implementadas en la BBU y un unidad radio.
Esta memoria finaliza con la extracción de conclusiones sobre los resultados obtenidos y la valoración sobre los mismos.
A lo largo de la memoria se ha podido ver que las tareas desarrolladas buscaban principalmente realizar un dimensionado de recursos en la red, que se ven traducidos en el número de CPUs albergados en dos centros de cómputo, lugar donde se ven implementadas la DU y CU virtualizadas de cada gNB, los cuales fueran capaces de satisfacer las necesidades temporales del procesado. Se ha conseguido realizar la implementación de dos algoritmos que pueden dimensionar dichos recursos en centros de cómputo comunes. Gracias a los resultados que han arrojado cada uno de los algoritmos podemos concluir que el algoritmo individualizado por usuario, proporcionará una mayor eficiencia en su desarrollo y además un numero de procesadores más ajustado a la realidad de los requisitos que pude demandar una red, pues ha tenido en cuenta la contribución particular de cada usuario.
Se propusieron dos mejoras principales a la aplicación del algoritmo, una de ellas versaba sobre el procesado de los usuarios cuya calidad de señal es muy baja y por tanto su tráfico se opta por no procesarlo, pues lo único que aportarían a la red sería un sobredimensionamiento innecesario. La otra mejora dota al programa que desarrolla al algoritmo la capacidad de dar como resultado el dato de procesadores necesarios a implantar en cada centro de cómputo.
Se ha propuesto una forma de designar la ubicación de la unidad centralizada del gNB mediante un algoritmo de balanceo de carga tras comprobar que el transporte de FrontHaul no tenía un impacto apreciable en el sistema como para hacer desarrollar otro criterio.
Finalmente se ha conseguido demostrar que el dimensionado realizado por el algoritmo consigue con creces dar un dato relativo a la cantidad de CPUs integradas en cada centro de cómputo a fin de satisfacer las necesidades computacionales de dos slices con una naturaleza arbitraria y sin ninguna relación entre los tipos de tráfico a los que están dedicados cada uno de ellos

7.2.-Propuestas futuras

Antes se mencionó el machine learning como un posible trabajo derivado de la evaluación de la evolución del sistema. Para aclarar e ilustrar estas palabras, volvamos a la sección donde teníamos un escenario virgen, con un dimensionado pequeño.
Gracias al machine learning y la inteligencia artificial se podría desarrollar un sistema que se integraría fácilmente en escenarios como el descrito en este trabajo, encargado principalmente de administrar la eficiencia eléctrica y el consumo de potencia en los centros de cómputo.
La primera idea que surge es la de desarrollar un software capaz de evaluar en cada momento lo que la red demanda y ser capaz de una vez se ha determinado si las condiciones son favorables o no, dar permiso para utilizar una CPU que previamente no se utilizaba debido al incremento de la demanda y de apagar CPUs cuyo factor de utilización sea muy bajo a fin de cargar de eficiencia a la red, integrarla en un sistema ecológico restrictivo y minimizar el impacto en el consumo de potencia eléctrica de la central de cómputo.

Anexo 1.- Conexión de estaciones base con centros de cómputo.

Con el objetivo de hacer una asignación equitativa de recursos de cómputo y de localizarlos propiamente, en este anexo se pretende proponer un criterio que, si bien no es el más complejo, puede ser perfectamente válido en materia de reparto equitativo y localización de recursos de cómputo.
Para determinar con contundencia cuales son los mensajes y las estaciones base que se procesaran en cada uno de los centros de cómputo vamos a partir del hecho comprobado de que hay usuarios que se van a conectar a la estación base integrada en el centro de cómputo, sirviendo cobertura y conexión a un número que se ha comprobado es reducido pero existente de usuarios.
Por simplicidad y por conveniencia, los usuarios conectados a la cobertura de la estación base integrada en un centro de cómputo se procesarán en dicho centro de cómputo.
A partir de ahí, en vista y comprobación de que el impacto del Tfh en el retardo de transporte de FrontHaul en la red es despreciable, la distancia entre la estación base y el centro de cómputo no va a ser un factor a considerar para determinar esta conexión.

Como lo que buscamos es hacer un reparto equitativo, de manera ordenada se conectará una estación base a una BBU en función de cuál de las BBU es la que procesa menos usuarios. Esto produce un reparto equitativo de la carga entre los centros de cómputo.
Para determinar esta conexión se ha desarrollado un script que se encarga antes de realizar ningún cálculo de frecuencia de esta tarea.
El pseudocódigo que se muestra a continuación realiza esta función de conectar cada estación base a un centro de cómputo:

Función: Designar un centro de cómputo para una microcelda		
Entrada	:	Número de usuarios de cada microcelda
Salida	:	BBU para cada RRU
Paso 1	:	Inicializar la matriz de salida
Paso 2	:	Buscar la microcelda con más usuarios
Paso 3	:	Asignar al centro de cómputo con menos usuarios
Paso 3	:	Repetir para todas las estaciones
Paso 4	:	Devolver la matriz que contiene a que centro de cómputo se conecta cada microcelda

- Primeramente, se encarga de calcular el número medio de usuarios que tiene cada celda a lo largo de todas las instantáneas simuladas.
- Una vez calculada esa media empieza asignando a los centros de cómputo los usuarios servidos por su estación base.
- A partir de esta media, se busca aquella celda que tiene el máximo número de usuarios medios conectados y se asigna en la BBU que menos usuarios este procesando hasta el momento.
- En caso de haber la misma cantidad de usuarios procesados en ambos centros de cómputo, por defecto se asignan al segundo, esta medida se ha tomado arbitrariamente.

El resultado conseguido con este algoritmo de asociación se puede ver fácilmente mostrando las variables que almacenan el número de usuarios que se procesan en cada BBU y mostrando al lector una imagen que ilustre la conexión entre las BS y las BBU.

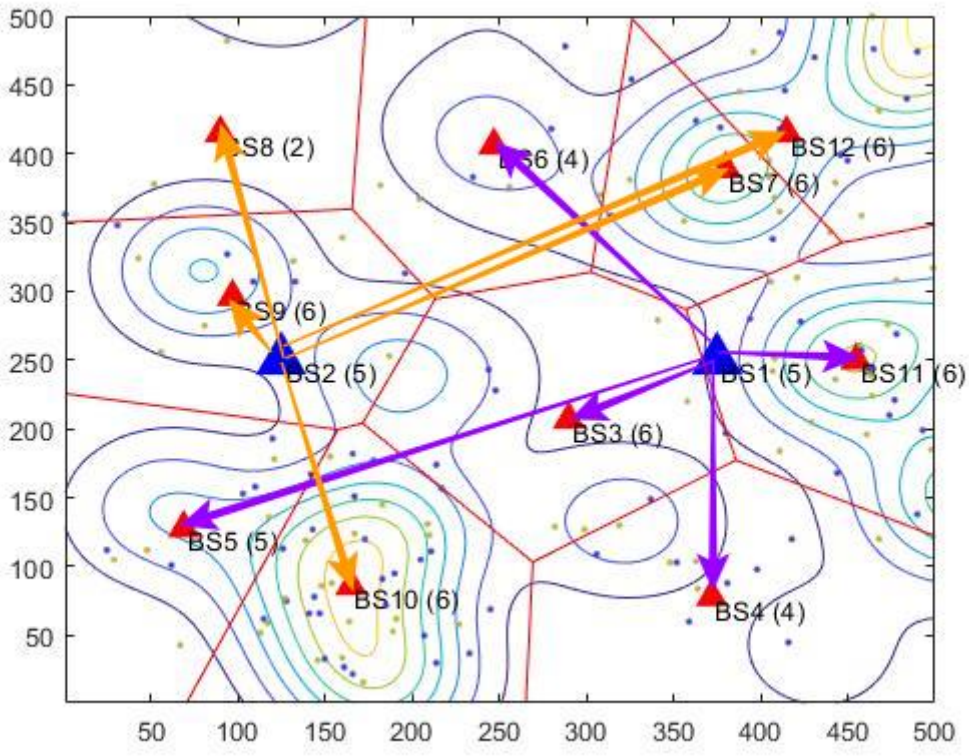


Figura A.1: Conexión entre estaciones base y centros de cómputo.

En la figura A.1 se muestra mediante una flecha el viaje que realiza el tráfico de cada estación base hasta su centro de cómputo designado, los cuales están alojados en las estaciones base de las macroceldas, conectadas estas últimas al centro de cómputo que albergan. Para el caso de nuestro escenario simulado el número de usuarios es de 150 en total, y están repartidos así:

CENTRO DE COMPUTO	CENTRO DE COMPUTO 1	CENTRO DE COMPUTO 2
USUARIOS PROCESADOS	**73,64**	**76,36**

Tabla A.1: Número medio de usuarios procesados en cada centro de cómputo.

Como vemos en la tabla, la diferencia con la mitad de los usuarios en cada centro de cómputo, que sería la opción ideal, es de apenas 1,36 usuarios, por lo que podemos asumir que el algoritmo es suficientemente bueno como para hacer un reparto equitativo de usuarios entre los centros de cómputo.

Capítulo 8.- Bibliografía.

[1] Cómo ha cambiado la venta de smartphones en los últimos años. elandroidelibre.elespanol.com/2017/03/cambiado-ventas-moviles-smartphones-ultimos-anos.html, 2017

[2] KHATIBI, Sina; SHAH, Kunjan; ROSHDI, Mustafa. Modelling of computational resources for 5G RAN. En 2018 European Conference on Networks and Communications (EuCNC). IEEE, 2018. p. 1-5.

[3] TRAN, Tuyen X.; YOUNIS, Ayman; POMPILI, Dario. Understanding the computational requirements of virtualized baseband units using a programmable cloud radio access network testbed. En 2017 IEEE International Conference on Autonomic Computing (ICAC). IEEE, 2017. p. 221-226.

[4] GAMAGE, Amila Tharaperiya; SHEN, Xuemin Sherman. Resource Allocation for Interworking Macro Cell and Hyper-Dense Small Cell Networks. En Resource Management for Heterogeneous Wireless Networks. Springer, Cham, 2018. p. 65-80.

[5] SMALL CELL. www.desentnetworks.com/technologies/small-cell, 2015

[6] Viavi Solutions. Network Slicing. www.viavisolutions.com/5g-network-slincing

[7] ERAMO, Vincenzo; LAVACCA, Francesco Giacinto. Optimizing the cloud resources, bandwidth and deployment costs in multi-providers network function virtualization environment. IEEE Access, 2019, vol. 7, p. 46898-46916.

[8] D. Firoozjaei, Mahdi & Jeong, Jaehoon & Ko, Hoon & Kim, Hyoungshick. Security challenges with network functions virtualization: Future Generation Computer Systems. 67. 10.1016/j.future.2016.07.002.

[9] RODRIGUEZ, Veronica Quintuna; GUILLEMIN, Fabrice. Cloud-RAN modeling based on parallel processing. IEEE Journal on Selected Areas in Communications, 2018, vol. 36, no 3, p. 457-468.

[10] T. Guo, A. Suárez. Enabling 5G RAN Slicing With EDF Slice Scheduling. 2019 TRAN, Tuyen X.; POMPILI, Dario. Dynamic radio cooperation for downlink cloud-RANs with computing resource sharing. En 2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems. IEEE, 2015. p. 118-126.

[11] WANG, Ning; HOSSAIN, Ekram; BHARGAVA, Vijay K. Backhauling 5G small cells: A radio resource management perspective. IEEE Wireless Communications, 2015, vol. 22, no 5, p. 41-49

[12] NIKAEIN, Navid. Processing radio access network functions in the cloud: Critical issues and modeling. En Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services. 2015. p. 36-43.

[13] YOUNIS, Ayman; TRAN, Tuyen X.; POMPILI, Dario. Bandwidth and energy-aware resource allocation for cloud radio access networks. IEEE Transactions on Wireless Communications, 2018, vol. 17, no 10, p. 6487-6500.

[14] AROUK, Osama, et al. Cost optimization of cloud-RAN planning and provisioning for 5G networks. En 2018 IEEE International Conference on Communications (ICC). IEEE, 2018. p. 1-6.

[15] V.K. QUINTUNA RODRIGUEZ. NEW NETWORK / IT COMMAND: Virtualized Function Performance For A Programmable Infrastructure. Networking And Internet Architecture

[CS.NI]. SORBONNE UNIVERSITÉ, 2018. ENGLISH. FFTEL-01884431F.

[16] ZHANG, Shunliang. An overview of network slicing for 5G. *IEEE Wireless Communications*, 2019, vol. 26, no 3, p. 111-117

[17] QUMOX 8GB DDR4 2133 2133MHZ PC4-17000 PC-17000 (260 PIN) MEMORIA SODIMM 8GB. WWW.AMAZON.ES/GP/PRODUCT/B07GPBK5YK/REF=PPX_YO_DT_B_ASIN_TITLE_O00_S00?IE=UUT8&PSC=1.

[18] MATLAB pricing licensing. es.mathworks.com/pricing-licensing.html

[19] Windows. www.microsoft.com/es-es/store/b/windows

[20] Asus ROG GL752VW, características, precio y especificaciones. planetared.com/2016/07/asus-rog-gl752vw-caracteristicas-precio-especificaciones/#:~:text=Precio%20del%20Asus%20ROG%20GL752VW,diversas%20tiendas%20online%20y%20f%C3%ADsicas.

[21] Western Digital WD Verde Internal SSD M.2 SATA, Verde, 240 GB. www.amazon.es/gp/product/B078WYS5K6/ref=ppx_yo_dt_b_asin_title_o08_s00?ie=UTF8&psc=1.

[22] ETS Ingeniería informática y telecomunicaciones, TFG. etsiit.ugr.es/pages/trabajos_fin_grado

[23] Retribuciones PDI UGR, gerencia.ugr.es/habilitacion/pages/retribuciones/retribucionespdi2019web.

[24] Retribuciones contrato predoctoral. ccoo.ugr.es/2019/11/18/aclaraciones-sobre-los-contratos-predoctorales/

[25] G. López Pradas. Predistorsion Digital de un Sistema de Comunicaciones OFDM: Estudio Mediante Simulación. Capítulo 2- LTE O LA CUARTA GENERACIÓN (4G) DE COMUNICACIONES MÓVILES. http://bibing.us.es/proyectos/abreproy/11983/fichero/Cap%C3%ADtulo+2+-+LTE.pdf

[26] 5G NR Resource Block Definition and RBs Calculation. Techplayon.com/nr-resource-block-definition-and-rbs-calculation/#:~:text=In%205G%2C%20One%20NR%20Resource,depend%20on%20sub-carrier%20spacing.

[27] 5G/NR - MCS/TBS/Code Rate. https://www.sharetechnote.com/html/5G/5G_MCS_TBS_CodeRate.html

[28] 5G/NR – Resource Allocation Type. https://www.sharetechnote.com/html/5G/5G_ResourceAllocationType.html

[29] NAVARRO-ORTIZ, Jorge, et al. Radio Access Network Slicing Strategies at Spectrum Planning Level in 5G and Beyond. IEEE Access, 2020, vol. 8, p. 79604-79618.