ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

SIMULACIÓN EN MATLAB DE UN ALGORITMO BASADO EN *MACHINE LEARNING* Y *HARD HANDOVER* PARA EL BALANCEO DE CARGA EN UNA RED CELULAR LTE

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

CARLOS ALBERTO PARREÑO MUIRRAGUI

DIRECTOR: PhD. PABLO ANIBAL LUPERA MORILLO

Quito, marzo 2021

AVAL

Certifico que el presente trabajo fue desarrollado por Carlos Alberto Parreño Muirragui, bajo mi supervisión.

_____

PhD. Pablo Anibal Lupera Morillo
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo, Carlos Alberto Parreño Muirragui, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

Carlos Alberto Parreño Muirragui

DEDICATORIA

Dedico este proyecto de titulación a mi mamá Mónica, mis hermanas Chris y Andrea, mis sobrinos Emilia, Elian y Mía, porque además de ser mi familia son el motor de mi vida.

A todas las personas que han creído en mí, porque siempre me han tendido la mano y me han ayudado a ser una persona más fuerte cada día.

AGRADECIMIENTO

Agradezco en primer lugar a mi mamá Mónica Muirragui, por darme su amor y fuerza necesaria para poder cumplir cada una de las metas que he alcanzado, por ser la persona más ejemplar y siempre estar a mi lado en los momentos más difíciles y duros.

A mi hermana Chris, por siempre apoyarme y darme una mano cuando más lo necesitaba sin importar las circunstancias, porque además de mi hermana es como una madre para mí.

A Allison, Frank, Freddy, Hugo, Ítalo, Javier, Jonathan y demás amigos que me han acompañado durante este largo camino, por siempre estar conmigo en los buenos y malos momentos.

A Carolina, por siempre motivarme a conseguir mis objetivos y a nunca rendirme.

Agradezco al Dr. Pablo Lupera por todo su apoyo en la realización de este proyecto de titulación, ya que a pesar de las circunstancias siempre estuvo presente guiándome con su conocimiento y compromiso.

ÍNDICE DE CONTENIDO

AVAL I
DECLARACIÓN DE AUTORÍA II
DEDICATORIA III
AGRADECIMIENTO IV
ÍNDICE DE CONTENIDO V
RESUMEN IX
ABSTRACT X
1. INTRODUCCIÓN 1
1.1 OBJETIVOS 2
1.2 ALCANCE 2
1.3. MARCO TEÓRICO 5
1.3.1. *HANDOVER* EN LTE 5
1.3.1.1. Técnicas de *handover* 5
1.3.1.1.1. Soft Handover 5
1.3.1.1.2. Hard Handover 6
1.3.1.2. Parámetros de *handover* 7
1.3.1.2.1. Margen de handover 7
1.3.1.2.2. Time To Trigger (TTT) o Tiempo de disparo 8
1.3.1.2.3. RSRP Potencia recibida de la señal de referencia 8
1.3.1.2.4. RSRQ Calidad recibida de la señal de referencia 8
1.3.1.3. Algoritmo de *hard-handover* en LTE 9
1.3.1.4. Proceso de *handover* en LTE 10
1.3.1.5. Ganancia de *handover* 12
1.3.1.6. Medidas de *handover* 13
1.3.1.6.1. Promedio de handovers por unidad móvil por segundo 13
1.3.1.7. Sobrecarga de usuarios (Problema de *handover*) 14
1.3.1.8. Capacidad máxima de usuarios activos en una celda LTE con ancho de banda de 5 MHz. 15
1.3.2. LTE SON (*LONG TERM EVOLUTION SELF-ORGANIZED NETWORKS*) 16
1.3.2.1. Características de las SON 16
1.3.2.1.1. Auto-configuración 17
1.3.2.1.2. Auto-optimización 17
1.3.2.1.3. Auto-corrección 18
1.3.3. PRESUPUESTO DE ENLACE 18
1.3.3.1. Modelo de espacio libre 19

1.3.3.1.1. Pérdida en trayectoria de espacio libre 20
1.3.3.2. Margen de desvanecimiento 20
1.3.3.2.1. Distribución probabilística de Rayleigh 21
1.3.3.3. Sensibilidad del receptor en redes LTE 22
1.3.3.4. Variación del Margen de *Handover* 23
1.3.4. *MACHINE LEARNING* 24
1.3.4.1. Clases de *Machine Learning* 25
1.3.4.1.1. Aprendizaje supervisado 25
1.3.4.1.2. Aprendizaje no supervisado 25
1.3.4.1.3. Aprendizaje por refuerzo 26
1.3.4.2. Programación dinámica (DP *Dynamic Programming*) 31
1.3.4.3. Algoritmo *Q-Learning* 32
1.3.4.3.1. Factor de descuento γ 33
1.3.4.3.2. Velocidad de aprendizaje 34
1.3.5. PERFILES ANGULARES DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL 34
1.3.5.1. Parámetros recomendación UIT-R P.1816-4 ANEXO 3 35
1.3.5.2. Perfil angular de llegada a largo plazo en la estación móvil para entornos NLoS en zonas urbanas y suburbanas 36
1.3.5.2.1. Perfil angular de llegada en la estación móvil NLoS 36
1.3.5.3. Perfil angular de llegada a largo plazo en la estación móvil para un entorno LoS en zonas urbanas y suburbanas 36
1.3.5.3.1. Estación base en la cima de un edificio ubicado en el lado izquierdo de la calle 36
1.3.5.3.2. Estación base en la cima de un edificio ubicado en el lado derecho de la calle 37
1.3.5.3.3. Estación base ubicada en la cima de un edificio frente al extremo de la calle 38
2. METODOLOGÍA 39
2.1. PROGRAMA DE BALANCEO DE CARGAS MEDIANTE APRENDIZAJE POR REFUERZO 39
2.1.1. DESCRIPCIÓN DE LA INTERFAZ GRÁFICA Y OBTENCIÓN DE DATOS INICIALES 40
2.1.2. CREACIÓN DE LAS CELDAS HEXAGONALES Y REPRESENTACIÓN GRÁFICA 43
2.1.3. GENERACIÓN ALEATORIA DE USUARIOS Y REPRESENTACIÓN GRÁFICA 45
2.1.4. CONDICIONES DE ERROR 47
2.1.5. CÁLCULO DE PRESUPUESTO DE ENLACE 48
2.1.6. SCRIPT PRESUPUESTO DEL ENLACE 49

2.1.7. PROCESO DE BALANCEO DE CARGAS PARA LA CELDA SOBRECARGADA 51
2.1.8. CÁLCULO DE MARGEN DE *HANDOVER* Y CONDICIÓN DE VELOCIDAD DE APRENDIZAJE 54
2.1.8.1. Proceso de *machine learning* – aprendizaje por refuerzo 55
2.1.8.2. Cálculo de recompensas a partir del algoritmo *Q-learning*....57
2.1.9. CONTADOR DE OPERACIONES PARA EL BALANCEO DE CARGAS 60
2.1.10. CONDICIÓN DE SENSIBILIDAD Y REPRESENTACIÓN GRÁFICA 61
2.1.10.1. Contador de *handovers* exitosos 63
2.1.11. CÁLCULO DE PARÁMETROS DE RENDIMIENTO DEL ALGORITMO 63
2.1.12. CREACIÓN DEL ARCHIVO DE TEXTO CON RESULTADOS 64
2.1.13. PRESENTACIÓN DE RESULTADOS Y GRÁFICAS 65
2.1.14. PRESENTACIÓN DE RESULTADOS EN LA VENTANA DE COMANDOS 66
2.1.15. CONFIGURACIÓN DE VARIABLES GLOBALES Y *RESET* DE RESULTADOS 68
2.2. PROGRAMA PARA EL CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL 69
2.2.1. DESCRIPCIÓN DE LA INTERFAZ GRÁFICA Y OBTENCIÓN DE DATOS INICIALES Y/O GLOBALES 69
2.2.2. ALGORITMO DEL CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL PARA UN ENTORNO NLOS 71
2.2.2.1. Representación gráfica del perfil angular de potencia de llegada en la estación móvil para un entorno NLoS. 73
2.2.3. ALGORITMO DEL CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL PARA UN ENTORNO LOS....74
2.2.3.1. Condición de valor del coeficiente R....75
2.2.3.2. Simulación del perfil angular de llegada en un entorno LoS cuando la estación base está a la izquierda respecto a la estación móvil....75
2.2.3.3. Simulación del perfil angular de llegada en un entorno LoS cuando la estación base está a la derecha respecto a la estación móvil 76
2.2.3.4. Simulación del perfil angular de llegada en un entorno LoS cuando la estación base está de frente a la estación móvil. 78
2.2.3.5. Representación gráfica del perfil angular de potencia de llegada en la estación móvil para el entorno LoS seleccionado. 78
2.2.4. *RESET* DE GRÁFICAS Y APERTURA DE LA RECOMENDACIÓN UIT-R P.1816-4 79
3. RESULTADOS Y DISCUSIÓN 81
3.1. PROGRAMA DE BALANCEO DE CARGA ANTE EL PROBLEMA DE SOBRECARGA EN UNA CELDA LTE 81
3.1.1. ESCENARIO Nº1: FRECUENCIA 850 MHZ, RADIO 1850 M, POTENCIA DE TRANSMISIÓN 0 DBM. 84
3.1.2. ANÁLISIS DE RESULTADOS 92

3.1.3. ESCENARIO Nº2: FRECUENCIA 850 MHZ, RADIO 18500 M, POTENCIA DE TRANSMISIÓN 20 DBM. 96
3.1.4. ESCENARIO Nº3: FRECUENCIA 1900 MHZ, RADIO 1150 M, POTENCIA DE TRANSMISIÓN 0 DBM. 96
3.1.5. ESCENARIO Nº4: FRECUENCIA 1900 MHZ, RADIO 11500 M, POTENCIA DE TRANSMISIÓN 20 DBM. 97
3.1.6. ESCENARIO Nº5: FRECUENCIA 2100 MHZ, RADIO 1050 M, POTENCIA DE TRANSMISIÓN 0 DBM. 97
3.1.7. ESCENARIO Nº6: FRECUENCIA 2100 MHZ, RADIO 10500 M, POTENCIA DE TRANSMISIÓN 20 DBM. 98
3.1.8. COMPARACIÓN DE RESULTADOS RESPECTO A LA CANTIDAD DE SIMULACIONES....99
3.1.9. SIMULACIÓN DE UN ESCENARIO DE 200 USUARIOS DE SOBRECARGA EN LA CELDA CENTRAL Y TRES CELDAS VECINAS SOBRECARGADAS. 100
3.2. CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL SEGÚN LA RECOMENDACIÓN UIT-R P.1816-4, ANEXO 3. 102
3.2.1. CÁLCULO DEL DE PERFIL ANGULAR DE POTENCIA DE LLEGADA PARA UN ENTORNO NLOS 103
3.2.2. CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA PARA UN ENTORNO LOS....105
3.2.2.1. Simulación del perfil angular de potencia de llegada para un entorno LoS, estación base ubicada a la derecha de la calle. 106
3.2.2.2. Simulación del perfil angular de potencia de llegada para un entorno LoS cuando la estación base está ubicada a la izquierda de la calle.....107
3.2.2.3. Simulación del perfil angular de potencia de llegada para un entorno LoS, estación base de frente al extremo de la calle....108
4. CONCLUSIONES Y RECOMENDACIONES 110
4.1. CONCLUSIONES 110
4.2. RECOMENDACIONES....112
5. REFERENCIAS BIBLIOGRÁFICAS....114
6. ANEXOS....118
ANEXO A 119
ANEXO B 127
Anexo B.1. Código de la interfaz “presentación”. 128
Anexo B.2. Código de la interfaz “Balanceo_Cargas”. 130
Anexo B.3. Código de la función “presupuesto”. 154
Anexo B.4. Código de la interfaz “results”. 155
Anexo B.5. Código de la interfaz “Perfil_Angular”. 158
ANEXO C 166

RESUMEN

En el presente trabajo se muestra la implementación y simulación de un algoritmo fundamentado en *hard handover* que permitirá realizar el proceso de balanceo de cargas desde una celda sobrecargada con ancho de banda de 5 MHz hacia celdas vecinas no sobrecargadas, además, se obtiene el perfil angular de potencia de llegada en la estación móvil de un usuario que ha realizado el proceso de *handover* para entornos con y sin línea de vista (LoS y NLoS).

El primer capítulo detalla los conceptos básicos sobre *hard handover*, balanceo de cargas, presupuesto de enlace, *machine learning* y algoritmo *Q-Learning*. Además, se detalla el cálculo del perfil angular descrito en la recomendación UIT-R P.1816-4 ANEXO 3.

En el segundo capítulo se describen los algoritmos implementados en el *software* MATLAB para las simulaciones del programa para el balanceo de cargas y el programa del perfil angular de potencia de llegada en la estación móvil.

En el tercer capítulo se verifican los resultados de la simulación del algoritmo de *machine learning*, respecto al promedio de *handover*s exitosos por segundo y la variación de margen de *handover*. Además, se analizan los perfiles de potencia de llegada en la estación móvil respecto al ángulo de llegada y el nivel de potencia mínimo de la señal.

En el cuarto capítulo se presentan las conclusiones y recomendaciones en base a los resultados obtenidos.

Finalmente, se presentan los anexos con los diagramas de flujo de los procesos realizados, los resultados obtenidos de las simulaciones y los códigos de cada programa.

PALABRAS CLAVE: *hard handover*, balanceo de cargas, margen de *handover*, perfil angular, *Q-Learning*.

ABSTRACT

This work shows the implementation and simulation of an algorithm based on *hard-handover* that will allow the *load balancing* process to be carried out from an over*load*ed cell with a 5 MHz bandwidth to neighboring cells that are not over*load*ed, in addition, the calculation of the arrival power angular profile is performed in the mobile station of a user who has carried out the *handover* process for environments with and without line of sight (LoS and NLoS).

The first chapter details the basics of *hard handover*, *load balancing*, link budget, *machine learning*, and the *Q-Learning* algorithm. In addition, the calculation of the angular profile described in recommendation ITU-R P.1816-3 ANNEX 3 is detailed.

The second chapter describes the algorithms implemented in the MATLAB *software* for the simulations of the *load balancing* program and the arrival power angular profile program at the mobile station.

In the third chapter, the results of the simulation of the *machine learning* algorithm are verified, regarding the average of successful *handover*s per second and the variation of the *handover* margin. In addition, the arrival power profiles at the mobile station are analyzed regarding the arrival angle and the minimum signal power level.

The fourth chapter presents the conclusions and recommendations based on the results obtained.

Finally, the annexes are presented with the flow diagrams of the processes carried out, the results obtained from the simulations and the codes of each program.

KEYWORDS: *hard handover*, *load balancing*, *handover* margin, angular profile, *Q-Learning*.

1. INTRODUCCIÓN

El crecimiento exponencial de las telecomunicaciones móviles ha creado la demanda de tecnologías que ofrezcan el mejor servicio de velocidad de transmisión, cobertura y capacidad para los dispositivos móviles de la actualidad.

La tecnología *Long Term Evolution* (LTE) es parte de la cuarta generación de los sistemas móviles celulares, que se ha desarrollado y especificado según el 3GPP (*The 3rd Generation Partnership Project*) como la tecnología sucesora de UMTS (*Universal Mobile Telecommunications System*). LTE se ha desarrollado para alcanzar velocidades de transmisión de hasta 150 Mbps según el 3GPP *release* 8 y, además, para poder tener la capacidad de soportar la incesante y creciente demanda de los servicios móviles.

Dichas necesidades han generado varios problemas, uno de ellos es la sobrecarga de usuarios, que se ocasiona cuando la carga existente se presenta de una manera en que determinadas zonas de la red celular se sobrecargan, sufriendo una disminución en la calidad del servicio.

El balanceo de cargas y la movilidad son aspectos importantes para LTE que posibilitan solucionar el problema de la sobrecarga de usuarios. La tecnología LTE tiene inmerso el proceso de *handover* con una de sus alternativas vigentes como el proceso de *hard-handover*, este último puede ser aprovechado como solución para el problema de sobrecarga de usuarios incluso para los servicios en tiempo real [1].

Por su parte el *Machine Learning* consiste en un subconjunto de la inteligencia artificial, que es una forma viable para automatizar el proceso de *hard-handover* en LTE, ya que sus técnicas permiten mejorar progresivamente su desempeño a través de un proceso de entrenamiento. Su desarrollo y ejecución se realiza mediante métodos estadísticos para analizar datos previos sobre los mismos fenómenos y poder ajustarlos para obtener mejores resultados [2].

En este contexto, el objetivo de este proyecto de titulación es desarrollar un algoritmo basado en el aprendizaje por refuerzo, que es parte del *Machine Learning*, para solucionar el problema de sobrecarga de terminales activos en una celda de una red celular LTE, además, se analizarán sus resultados mediante diferentes parámetros que indiquen el desempeño del algoritmo desarrollado. Lo propuesto será simulado con la ayuda del *software* MATLAB.

Además, como complemento del programa de *machine learning* para el balanceo de cargas, se calculará el perfil angular de potencia de llegada para entornos con línea de vista (LoS) y sin línea de vista (NLoS) entre la estación móvil y la estación base de la celda seleccionada del proceso de *handover*. El objetivo del desarrollo de este programa es conocer la incidencia de las señales a lo largo del rango del ángulo de llegada de la señal recibida en la estación móvil que ha realizado el proceso de *handover*. Este cálculo no forma parte del proceso de aprendizaje por refuerzo y será simulado con la ayuda del *software* MATLAB.

1.1 OBJETIVOS

El objetivo general de este Proyecto Técnico es desarrollar un algoritmo basado en el aprendizaje por refuerzo para solucionar el problema de sobrecarga de terminales activos en una celda de una red celular.

Los objetivos específicos de este Proyecto Técnico son:

- Estudiar y describir los conceptos básicos del aprendizaje por refuerzo y el proceso de *handover* en una red celular.
- Desarrollar un algoritmo basado en *Q-Learning* y *hard handover* para el tratamiento del problema de sobrecarga de usuarios en una celda con un canal de 5 MHz.
- Simular con la ayuda del *software* MATLAB el algoritmo desarrollado para el balanceo de cargas.
- Analizar los resultados obtenidos mediante los parámetros: porcentaje de *handover* exitosos y velocidad de procesamiento.

1.2 ALCANCE

En este trabajo se simulará una red de comunicaciones móviles compuesta de 7 celdas omnidireccionales que disponen de canales de ancho de banda de 5 MHz, se considerará que la celda central puede alcanzar un nivel de sobrecarga. Para la evaluación de las condiciones de transmisión se empleará el modelo del espacio libre que permitirá calcular las pérdidas de propagación. El cálculo del presupuesto del enlace se ejecutará con la aplicación de la siguiente ecuación:

$$P_{RX} = P_{TX} + G_{TX} - L_{TX} - L_{FS} - L_M + G_{RX} - L_{RX} + G_{HO} \quad (1.1)$$

En donde:

P_{RX} = Potencia recibida en la estación móvil [dBm].

P_{TX} = Potencia de transmisión de la estación base [dBm].

G_{TX} = Ganancia de transmisión de la antena de la estación base [dBi].

L_{TX} = Pérdidas de transmisión (cables coaxiales, conectores, etc) [dB].

L_{FS} = Pérdidas de trayecto, pérdida de espacio libre [dB].

L_M = Margen de desvanecimiento [dB].

G_{RX} = Ganancia de recepción de la antena de la estación móvil [dBi].

L_{RX} = Pérdidas de recepción (cables coaxiales, conectores, etc) [dB].

G_{HO} = Ganancia de *handover* [dB].

El algoritmo propuesto para el balanceo de carga en la red se ejecutará considerando los valores de la capacidad máxima de usuarios activos en una celda LTE de 5 MHz. De esta manera se determinará si la celda bajo análisis se encuentra sobrecargada o no; en consecuencia, si la celda se encuentra en un estado de sobrecarga se ejecutará el algoritmo para realizar el balanceo de carga.

En el algoritmo se utilizarán los conceptos del *hard handover* en una red de comunicaciones móviles y sus parámetros asociados en el cálculo del presupuesto de enlace (margen de *handover* y ganancia de *handover*) [3]. En una parte del algoritmo se establecerá que terminales móviles deben realizar el *handover* y a qué celda objetivo deben realizar este traspaso.

El algoritmo consiste en un proceso de aprendizaje por refuerzo (*machine learning*) basado en el algoritmo de *Q-Learning* que será aplicado al balanceo de cargas de una celda sobrecargada hacia celdas vecinas no sobrecargadas que disponen de suficientes recursos para recibir conexiones [4].

Para la ejecución del algoritmo se tomará una instantánea del estado de la red cada cierto tiempo y se establecerá la posición de los terminales de usuario en la celda. Dentro del algoritmo, para la selección de la celda destino, se realizará el cálculo del presupuesto del enlace entre la ubicación actual del usuario y la estación base de cada una de las celdas vecinas, el mayor valor obtenido indicará la celda destino, y el éxito de dicho *handover* se

alcanzará si se cumple con la sensibilidad mínima del receptor. A continuación, se muestra la ecuación para determinar la condición de *handover* [4]:

$$P_{RX(CS)} + M_{HO(CS)} - \emptyset_{offset(CS)} \leq P_{RX(CO)} + \emptyset_{offset(CO)} \quad (1.2)$$

En donde:

$P_{RX(CS)}$ = Potencia recibida en la estación móvil respecto a la celda sobrecargada [dBm].

$P_{RX(CO)}$ = Potencia recibida en la estación móvil respecto a la celda objetivo o vecina [dBm].

M_{HO} = Margen de *handover* [dB].

$\emptyset_{offset}$ = Variación del margen de *handover* [dB].

El valor de $\emptyset_{offset}$ es una cantidad en dB que se sumará (positivamente) a la sensibilidad de la estación base de la celda objetivo y se restará (negativamente) en la estación base de la celda sobrecargada, permitiendo que la cobertura de las celdas objetivo aumente, mientras que se disminuye la cobertura de la celda central (sobrecargada) en la misma proporción. Lo mencionado consiste en un proceso de aprendizaje por refuerzo, ya que las estaciones base irán ajustando sus parámetros de *handover* hasta conseguir la redistribución de la carga, y, por tanto, la red alcanzará autónomamente y de forma automática un estado de balanceo de carga. Habiendo obtenido el resultado de dicha condición se dará una recompensa positiva o negativa, según sea el éxito del *handover* del usuario hacia la celda objetivo.

En base al nivel de sobrecarga y a la distribución aleatoria de los terminales de usuario en el área de cobertura, el algoritmo simulado en Matlab permitirá determinar la cantidad de terminales que deberán ejecutar el *handover* a las celdas vecinas y la variación del margen de *handover* que se debe establecer para el balanceo de carga.

Para medir el rendimiento del algoritmo se determinará mediante simulación la cantidad de *handover* exitosos y la velocidad con la que se logra conseguir el balanceo de carga. A partir de estos parámetros se calculará el promedio de *handovers* por estación móvil por segundo a partir de la siguiente fórmula:

$$HO_{AVG} = \frac{HO_{Total}}{J*T} \quad (1.3)$$

En donde:

HO_{AVG} = Promedio de *handovers* por unidad móvil por segundo.

HO_{Total} = Número total de *handover*s exitosos.

J = Número total de usuarios que realizan el *handover*.

T = Tiempo total de la simulación.

Del trabajo de titulación se obtendrá una simulación para probar la ejecución del algoritmo, por esto, no se tendrá un producto final demostrable.

1.3. MARCO TEÓRICO

1.3.1. *HANDOVER* EN LTE

El proceso de *Handover* se refiere a la transferencia de la conexión de un usuario desde un canal de radio a otro que puede ser desde la misma o diferente celda [5]. Además, *handover* es uno de los procesos claves para asegurar a los usuarios la libertad de movilidad mientras se encuentre bajo la cobertura de red y de esta forma obtener la continuidad del servicio.

Uno de los objetivos principales de LTE es proveer un rápido y eficiente *handover* desde una celda a otra mientras se mantiene el manejo de la red de la forma más simple posible. Por lo que, la optimización del proceso de *handover* es muy importante para alcanzar el rendimiento requerido y ese es uno de los problemas más importantes en redes LTE.

Uno de los tipos del proceso de *handover* en LTE es llamado *hard-handover*. El uso de *hard-handover* reduce la complejidad de la arquitectura de la red LTE. De todas formas, este proceso puede resultar ineficiente para el rendimiento de LTE, ya que puede incrementar el número de *handovers* y disminuir la velocidad de transmisión de la misma forma en que se incrementa el retardo de la red [1].

1.3.1.1. Técnicas de *handover*

El proceso de *handover* tiene varias técnicas, pero estas se pueden categorizar como *hard-handover* y *soft-handover*, que también son conocidas como *break-before-conect* BBC y *Conect (Entry)-Before-Break* (CBB) respectivamente [3]. Ambas técnicas de *handover* serán descritas en las siguientes subsecciones.

1.3.1.1.1. Soft Handover

Soft-handover es una técnica de *handover* donde el enlace de radio es añadida y abandonada en cierta forma en que la unidad móvil siempre mantiene al menos un enlace de radio con la UTRAN [5]. Para este tipo de técnica existe una controladora centralizada

llamada *Radio Network Controller* (RNC) que realizará el control de *handover* para cada unidad móvil que se encuentre en la red [3].

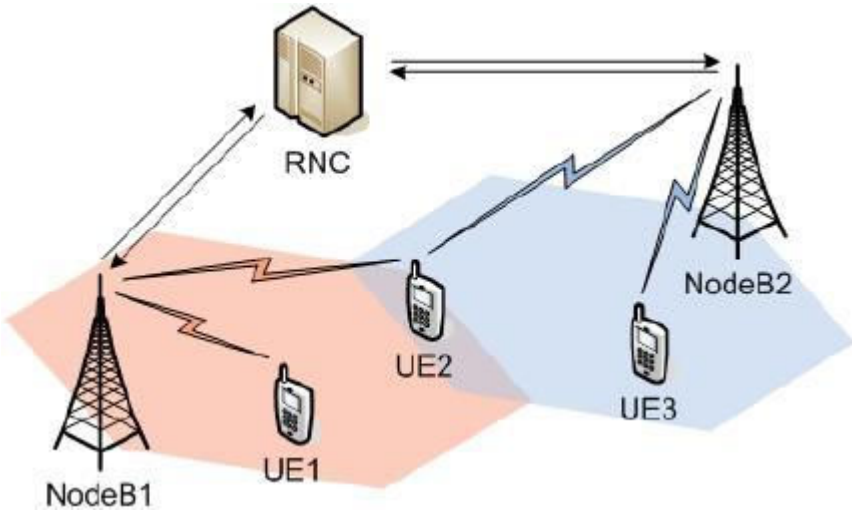


Figura 1.1 Esquema de soft-*handover* [6].

Tal como se puede observar en la figura 1.1, la característica distintiva de este tipo de *handover* es que es posible para una unidad móvil estar conectada simultáneamente a dos o más celdas durante un traspaso de la conexión.

1.3.1.1.2. Hard Handover

Hard handover es otra técnica de traspaso. Esta técnica requiere que un usuario rompa o abandone la conexión existente con la celda actual o la celda de servicio y realice una nueva conexión hacia la celda objetivo.

Este tipo de *handover* es controlado por la red y asistido por la unidad móvil. Se toma en consideración la información y las mediciones hechas por el móvil (por ejemplo: cálculo del presupuesto del enlace, nivel de la potencia de la señal recibida, etc.) para la ejecución del *handover*. [7]

Para redes celulares LTE existen algoritmos de traspaso de conexión basados tanto *en soft handover* como en *hard handover*. En el presente proyecto de titulación, el proceso de *handover* se fundamentará en el algoritmo de *hard handover*, lo que significa una corta interrupción en el servicio en el orden de milisegundos cuando el *handover* es ejecutado [3].

En esta técnica de *handover*, tal como indica la figura 1.2, se realiza una reserva de recursos en la celda objetivo antes de hacer el *handover*, dicho proceso será descrito en la sección 1.3.1.4.

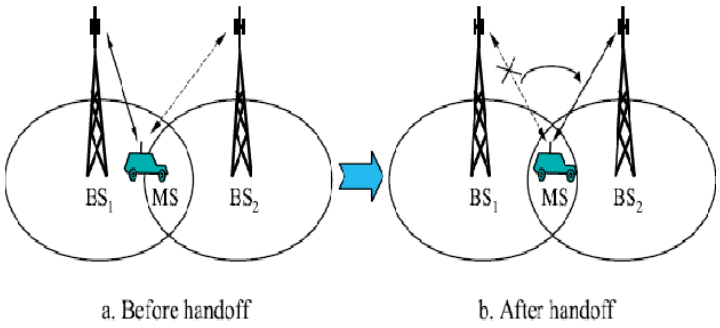


Figura 1.2. Esquema de *hard-handover* [8].

La interrupción en la conexión de esta técnica tiene varias ventajas, como la reducción de la complejidad del proceso de *handover* y minimiza la probabilidad de la ejecución de *handover* excesivos [7].

1.3.1.2. Parámetros de *handover*

En el proceso de *handover* se tienen diferentes parámetros que están involucrados y que son utilizados para mejorar el rendimiento de dicho proceso. En LTE, el inicio del *handover* usualmente se basa en medidas del estado del enlace y de otros parámetros que son utilizados para mejorar y evaluar el rendimiento del proceso. A continuación, se mencionan los aspectos principales del *handover*.

1.3.1.2.1. Margen de handover

El Margen de *handover* (HOM) es un parámetro que representa un valor límite de la diferencia de potencia de la señal recibida entre la celda de servicio y la celda objetivo, tal como indica la figura 1.3.

El valor de este parámetro es utilizado para asegurar que la celda objetivo sea la más apropiada para ser el destino del *handover* y recibir la conexión de la estación móvil.

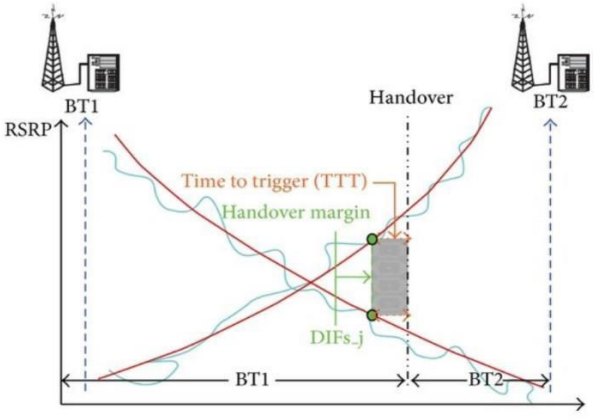


Figura 1.3. Margen de *handover* y Tiempo de disparo (TTT) [9].

Como se puede observar en la figura 1.3, mientras el móvil se acerca más a la celda objetivo, mayor será el nivel de la señal recibida respecto a esa estación base, por lo que la celda de servicio no es la estación base que tiene el nivel de señal más fuerte.

1.3.1.2.2. Time To Trigger (TTT) o Tiempo de disparo

El tiempo de disparo es el momento en que el proceso de *handover* se inicia, tal como se puede observar en la figura 1.3. En este instante de tiempo se hace énfasis en que los requerimientos iniciales se puedan cumplir en un intervalo de tiempo. Este parámetro puede decrecer el número de *handovers* innecesarios, pero también puede incrementar el *delay* o retardo del proceso de *handover*, el cual incrementa la probabilidad de *handovers* fallidos [10].

1.3.1.2.3. RSRP Potencia recibida de la señal de referencia

La potencia recibida de la señal de referencia (RSRP) se menciona en los documentos del 3GPP, y es definida como la potencia promedio recibida en vatios o en dBm de las señales que llevan señales de referencia (RS) de las celdas específicas dentro del ancho de banda analizado. Esta medida generalmente se expresa en dBm, y es utilizada para ordenar las diferentes celdas LTE de acuerdo a la intensidad de su señal, siendo una medida fundamental para el proceso de *handover*, esta medida se utiliza como un parámetro de entrada para la decisión de la selección de la celda objetivo [5].

Cabe recalcar que el número de recursos dentro del ancho de banda considerado por un tiempo determinado es utilizado por el usuario para determinar el RSRP.

1.3.1.2.4. RSRQ Calidad recibida de la señal de referencia

RSRQ o calidad recibida de la señal de referencia se define por el 3GPP, como la métrica que provee el valor de la calidad de la señal[1] recibida respecto a una celda específica, que al igual que el RSRP, es usada para ordenar las celdas LTE candidatas respecto a la calidad de sus señales.

Esta medida es una entrada para el proceso de *handover* y las decisiones de elección de la nueva celda de servicio. Por ejemplo, en los casos para los que la medida de RSRP no provea de la suficiente información para un rendimiento confiable en término de decisiones de movilidad [5].

[1] La calidad de la señal recibida es medida por el ruido o interferencia que existe en la transmisión. El término calidad describe cuanta interferencia existe entre la estación base y la estación móvil. Esto indica que tan buena u óptima será la señal recibida cuando existan condiciones de lluvia, nieve o tormentas eléctricas [42].

$$RSRQ = \frac{N*RSRP}{RSSI} \quad (1.4)$$

Donde:

N = Número de *resource block*s (RBs) de la portadora LTE en donde se mide el RSRP y RSSI en el ancho de banda del sistema.

$RSSI$ = La potencia total de las portadoras, incluyendo las interferencias adicionales que son recibidas o determinadas por el terminal.

Cabe recalcar que las medidas de RSSI y RSRP son ejecutadas sobre los mismos recursos que usa la estación móvil.

1.3.1.3. Algoritmo de *hard-handover* en LTE

El algoritmo de *hard-handover* en LTE es un proceso básico, pero muy efectivo en redes de cuarta generación para garantizar la continuidad de las comunicaciones. El funcionamiento de este algoritmo de *handover* considera tres parámetros fundamentales que son el margen de *handover* (HOM), el tiempo de disparo o *Time To Trigger* (TTT) y la potencia recibida de la señal de referencia respecto a una estación de base (RSRP) que para este proyecto su valor será tomado del resultado del presupuesto del enlace.

Tal como se describe en la figura 1.4, el funcionamiento de este algoritmo inicia cuando un móvil empieza a alejarse de la celda de servicio actual.

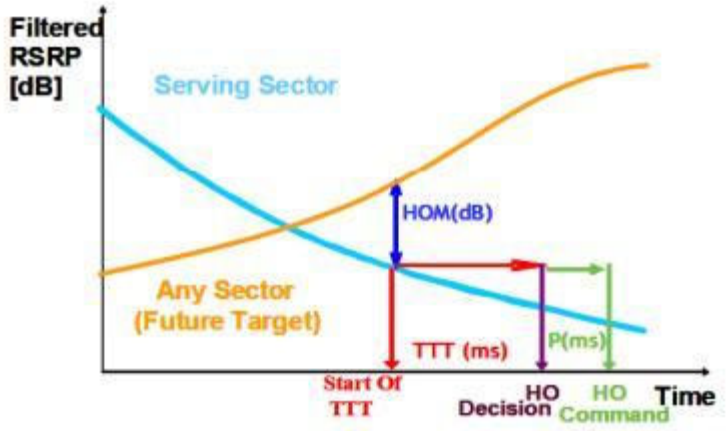


Figura 1.4. Diagrama del proceso de decisión de *hard-handover* [11].

La intensidad de la señal que se puede determinar mediante el presupuesto del enlace respecto a la celda de servicio empieza a disminuir con el tiempo, mientras el terminal se aleja de la estación base.

Con este problema presente, precisa tomar una decisión, la cual se ha representado con una ecuación que se satisface durante todo el intervalo TTT [12]:

$$RSRP_T > RSRP_S + HOM \tag{1.5}$$

Donde:

$RSRP_T$ = Potencia recibida de la señal de la celda vecina en función del cálculo del presupuesto del enlace.

$RSRP_S$ = Potencia recibida de la señal de la celda de servicio en función del cálculo del presupuesto del enlace.

HOM = Margen de *handover*.

Si dentro de un instante de tiempo TTT, la condición de la ecuación 1.5 se cumple, el proceso de *handover* se ejecuta.

Durante el intervalo de tiempo TTT, si el $RSRP_S$ es mayor que $RSRP_T$ o si la diferencia entre el $RSRP_T$ y $RSRP_S$ es menor que el margen de *handover*, el intervalo TTT se reiniciará.

El número de *handovers* o traspasos normalmente depende del valor del TTT y del HOM; mientras menor sean estos valores, mayor será el número de *handovers* [12].

1.3.1.4. Proceso de *handover* en LTE

Para realizar el proceso de *handover* la estación controladora E-UTRAN[2] configura en la estación móvil la realización de mediciones y recepción de reportes cuando se den las condiciones necesarias para realizar el proceso de *handover* [2].

Los parámetros que se configuran en los dispositivos móviles son:

- Los objetivos de medida, donde se definen las estaciones base respecto a las cuales se realizará la medida del nivel de la señal.
- Las configuraciones de informes, donde se configura el criterio sea eventual o periódico, en el que se activan los informes junto con lo que se espera del dispositivo móvil para informar, es decir, la señal de referencia elegida [2].

[2] E-UTRAN (*Evolved-UMTS Terrestrial Radio Access Network*) es la interfaz de aire en una red celular LTE y consiste de un conjunto de eNodeBs [44].

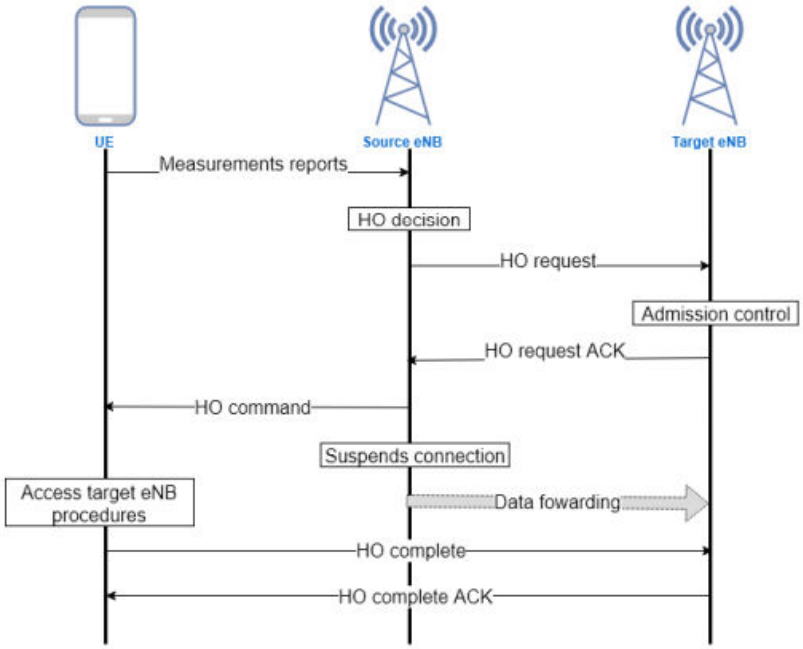


Figura 1.5. Esquema proceso de *Hard-Handover* [2]

Según la figura 1.5, los pasos para realizar el procedimiento de *handover* entre dos estaciones base sobre la interfaz[3] X2 son:

- La unidad móvil envía hacia la estación base de servicio los reportes de medida del nivel de la señal de referencia respecto a la estación base de servicio y también respecto a las estaciones base de las celdas vecina.
- Respecto a los reportes del nivel de la señal se toma la decisión de realizar el *handover* hacia la estación base de la celda vecina con la que se tiene mayor nivel de señal.
- La estación base de servicio envía una solicitud para realizar el proceso de *handover* hacia la estación base vecina (HO *Request*).
- La estación base de la celda vecina hace un control de admisión, verifica la disponibilidad de recursos y el nivel de la señal hacia la unidad móvil que realizaría el *handover*.
- Luego del control de admisión responde a la estación base de servicio con un ACK a la solicitud de *handover* (HO *request* ACK).
- La estación base de servicio envía el comando u orden para realizar el proceso de *handover* a la unidad móvil (HO *Command*).

[3] La interfaz X2 es la interfaz de interconexión entre dos eNodeB en la red LTE y soporta tanto el plano de control como el plano de usuario [43].

- Luego se produce la interrupción de la conexión (propiedad de *hard-handover*).
- La unidad móvil realiza el proceso para conectarse hacia la estación base objetivo, mientras que la estación base de servicio reenvía los datos de la conexión a la estación base objetivo.
- La unidad móvil confirma el traspaso con la estación base objetivo (HO *Complete*).
- La estación base objetivo envía un ACK de respuesta a la confirmación del traspaso de la conexión de la unidad móvil (HO *Complete* ACK).

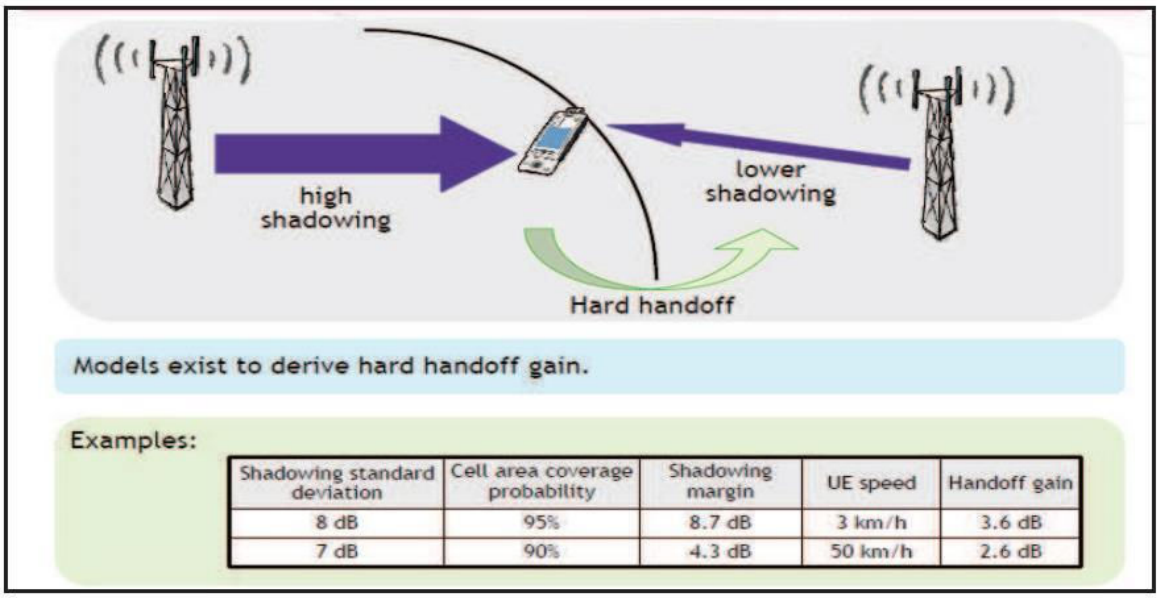
Como se puede ver en la figura 1.5, se sigue un procedimiento o una estrategia predefinida. Cada estrategia o algoritmo está compuesto por uno o más eventos basados en la señal de referencia de la estación base de servicio y sus vecinas.

Un algoritmo de *handover* también necesita definir una señal de referencia para ser utilizada por los eventos. La señal de referencia puede estimar la potencia del enlace entre la unidad móvil y la estación base o su calidad, para lo cual se denominan potencia recibida de la señal de referencia (RSRP) y calidad recibida de la señal de referencia (RSRQ), respectivamente [5].

1.3.1.5. Ganancia de *handover*

Se conoce que los enlaces de comunicación en una red celular inalámbrica están sujetos a un desvanecimiento aleatorio. Por lo tanto, se requiere un margen de enlace para garantizar un cierto nivel de disponibilidad de enlace. Es bien sabido que el margen de enlace requerido se reduce debido a la transferencia móvil en el borde de la celda [13], esta es la ganancia de *handover*. Tradicionalmente, la ganancia de *handover* considera solo la reducción del requisito de margen de enlace debido a la presencia de múltiples rutas aleatorias de desvanecimiento lento; el requisito de tasa de error no se tiene en cuenta de acuerdo a lo indicado en [13].

La ganancia resultante por la reducción del margen de enlace requerido, también puede ser representado como la ganancia de *handover*. En el caso de *hard-handover*, usa un enlace de radio en un momento dado, por lo tanto, el requisito de tasa de error está implícito en el requisito de desvanecimiento, pero para evitar la respuesta, *hard handover* requiere histéresis, es decir, el enlace objetivo debe ser de unos pocos dBs más alto que el enlace de host actual para que continúe una transferencia. La histéresis reduce la ganancia de traspaso [13].



Shadowing standard deviation	Cell area coverage probability	Shadowing margin	UE speed	Handoff gain
8 dB	95%	8.7 dB	3 km/h	3.6 dB
7 dB	90%	4.3 dB	50 km/h	2.6 dB

Figura 1.6. Esquema de la ganancia de *hard-handover* [14]

Como se puede observar en la figura 1.6, la velocidad con la que se mueve la estación móvil es el parámetro que en el proceso de *hard handover* marca la ganancia de *hard handover*. Se tiene que en el rango de 3 Km/h hasta 50 Km/h, la ganancia de *hard handover* estaría en el rango de 3,6 dB hasta 2,6 dB [14].

1.3.1.6. Medidas de *handover*

Los algoritmos aplicados respecto al *handover* en LTE pueden ser evaluados en base al promedio de *handovers* o traspasos por unidad móvil por segundo. Este parámetro permitirá evaluar el rendimiento del algoritmo desarrollado en este proyecto de titulación.

1.3.1.6.1. Promedio de handovers por unidad móvil por segundo

El promedio de *handovers* por unidad móvil por segundo representa el número de *handovers* realizados durante una simulación. Y está representado en la ecuación 1.6 [3]:

$$HO_{AVG} = \frac{HO_{Total}}{J*T} \tag{1.6}$$

En donde:

HO_{AVG} = Promedio de *handovers* por unidad móvil por segundo.

HO_{Total} = Número total de *handovers* exitosos.

J = Número total de usuarios que realizan el *handover*.

T = Tiempo total de la simulación.

HO_{Total} se incrementa si y solo si un *handover* es realizado exitosamente. Un *handover* se considera exitoso cuando un usuario ha sido traspasado desde una celda de servicio hacia una celda objetivo, mientras se mantiene el flujo de transmisión de datos sin interrupciones.

1.3.1.7. Sobrecarga de usuarios (Problema de *handover*)

La tecnología *Long Term Evolution* (LTE) tiene como objetivo proporcionar la mejor experiencia de conexión a todos los usuarios que la utilicen, aumentando la velocidad de transmisión de datos, mejorando la cobertura y brindando calidad de servicio (QoS) [15].

La carga de usuarios en las redes de comunicaciones móviles se presenta generalmente de manera no uniforme, pudiendo existir sobrecargas en determinadas zonas del área de cobertura. Este problema crea la necesidad de tener un balanceo de cargas, tal como se define en los estándares de las redes auto-organizadas LTE (LTE *self-organization network*, LTE SON) [4].

Una gran cantidad de equipos de usuario (UE) tratando de tener acceso a la red en un periodo de tiempo puede sobrecargar la Red de Acceso de Radio (RAN). Esta situación provoca que los equipos de usuario deban intentar acceder varias veces a la red hasta disponer de recursos habilitados para su ocupación, lo que conlleva a una disminución de la calidad del servicio [16]. Este problema trae consigo una situación donde una o varias celdas sobrecargadas sufren un alto índice de llamadas bloqueadas o llamadas caídas.

Una de las posibles soluciones consiste en aprovechar los recursos disponibles o no utilizados en celdas vecinas no sobrecargadas, proceso que se conoce como balanceo de carga (*Load Balancing*).

Si no se aplica el balanceo de cargas entre celdas provocaría el deterioro del desempeño de la red, por lo que el concepto de balanceo de cargas con la aplicación de técnicas adaptativas podría mejorar la calidad de servicio de las redes LTE [17].

Para conseguir lo planteado, en este trabajo de titulación se estudiarán los conceptos de *handover* en LTE, balanceo de cargas y las redes auto-organizadas, y se aplicarán los principios de *machine learning* en lo referente al aprendizaje por refuerzo. De esta manera, se planteará y probará mediante simulación un algoritmo que le permita a la red determinar por sí misma la distribución de cargas mediante un proceso iterativo de aprendizaje.

1.3.1.8. Capacidad máxima de usuarios activos en una celda LTE con ancho de banda de 5 MHz.

El cálculo de la capacidad máxima de una celda LTE será determinada a partir de las definiciones de *Physical Resource Block* (PRB) y número de subportadoras dentro del ancho de banda especificado, que, para el presente proyecto de titulación es de 5 MHz.

Este valor será tomado como referencia para el mínimo valor de usuarios generados dentro de la celda central que necesita el proceso de balanceo de cargas, el máximo valor de usuarios permitidos en el arreglo celular y en cada una de las celdas vecinas, tal como se describió anteriormente, tienen recursos disponibles para recibir nuevas conexiones.

El bloque de recurso físico o *Physical Resource Block* (PRB) es definido por el *release* 8 del 3GPP, como el mínimo elemento de información que puede ser asignado por el eNodeB a una estación móvil o usuario. El bloque de recurso físico tiene un ancho de banda de 180 kHz, que equivale a 12 subportadoras equiespaciadas en 15 kHz [18].

El número de *Physical Resource Block* (PRB) para cada ancho de banda o canalización permitida en LTE se describe en la tabla 1.

Tabla 1.1. Número de PRBs en función del ancho de banda del canal en LTE [18].

Canalización	**1,4 MHz**	**3 MHz**	**5 MHz**	**10 MHz**	**15 MHz**	**20 MHz**
Número de PRB	6	15	25	50	75	100

Considerando el número de PRBs para un canal LTE de ancho de banda de 5 MHz, el ancho de banda efectivo es de 4,5 MHz, tal como se describe en la ecuación 1.7. [18].

$$AB\ Efectivo\ celda\ de\ 5\ MHz = 25\ PRBs * 180\ kHz\ (AB\ de\ cada\ PRB) = 4{,}5\ MHz. \quad (1.7)$$

Por lo tanto, el número de subportadoras para una celda LTE de ancho de banda de 5 MHz es 300 subportadoras que pueden transmitir información, tal como se describe en la ecuación 1.8. [18].

$$\#\ subportadoras\ celda\ de\ 5\ MHz = 25\ PRBs * 12\ subportadoras\ por\ cada\ PRB = 300\ subportadoras \quad (1.8)$$

Cabe recalcar que en LTE se añade una subportadora más ya que se considera la subportadora central, que no se utiliza para transmitir información [19].

En el presente trabajo de titulación, se realiza una asignación de una portadora por usuario activo. Lo que establecería la capacidad máxima de una celda LTE con ancho de banda

de 5 MHz, a 300 usuarios por celda, lo que equivale a 2100 usuarios como la capacidad máxima total del arreglo celular compuesto por 7 celdas.

1.3.2. LTE SON (*LONG TERM EVOLUTION SELF-ORGANIZED NETWORKS*)

El manejo y administración de redes es una tarea difícil, especialmente para los sistemas de comunicación móvil celular debido a su complejidad inherente. Esta complejidad surge por el número de elementos de la red que deben ser desplegadas y gestionadas, pero también debido a interdependencias entre sus configuraciones. Estas tecnologías implementadas y sus paradigmas operativos son difíciles de manejar.

Las redes SON o redes auto-organizadas se pueden usar para reducir los costos operativos al reducir la carga de trabajo manual, protegiendo los ingresos y minimizando los errores humanos [20].

El objetivo de las redes SON es abordar los problemas de la gestión y administración de redes reduciendo en lo máximo posible la intervención humana al incrementar la automatización de los elementos de la red, la gestión de los dominios de la red o también llamada gestión de elementos de la red y la gestión de la red o gestión general.

3GPP ha abordado estos términos en la estandarización de un marco para las redes auto-organizadas en el estándar (3GPP TR32.500, 2011). La introducción de las características de las redes SON apuntan a reducir la carga de trabajo respecto a la operación y mantenimiento para el personal a través de la automatización para liberarlos de tareas que consumen mucho tiempo para que puedan de esta forma concentrarse en problemas cruciales.

El plan de las redes SON es avanzar gradualmente hacia una supervisión y dirección totalmente automatizadas para las redes auto-organizadas, lo que implica partir desde una planificación y configuración manual hacia procesos totalmente automatizados. Finalmente, solo se deben incluir ciertas pautas y actualizar el sistema en vez de supervisar los cambios actuales a través de la gestión de configuración [20].

En la práctica las redes auto-organizadas son una combinación de las funcionalidades estandarizadas por el 3GPP y las funcionalidades específicas del proveedor.

1.3.2.1. Características de las SON

Las redes SON intentan implantar una estrategia de auto-organización, teniendo como características la auto-configuración, la auto-optimización y la auto-corrección.

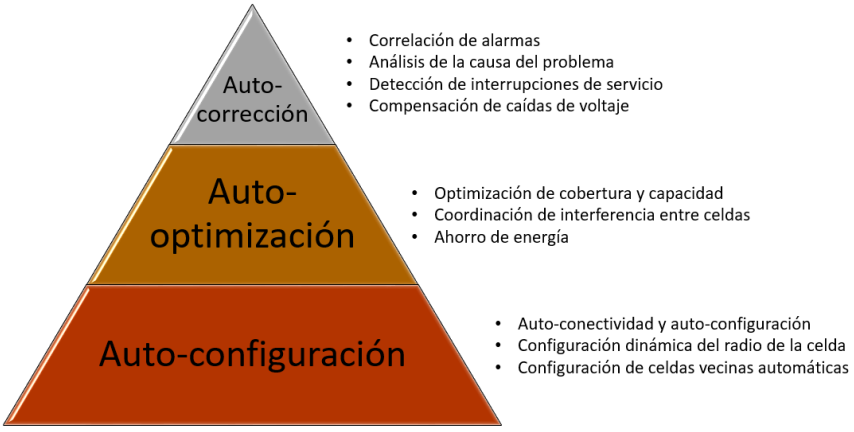


Figura 1.7. Características redes auto-organizadas SON [20].

1.3.2.1.1. Auto-configuración

La auto-configuración es el proceso mediante el cual se pone en servicio un nuevo elemento de red o partes de un elemento de red con la mínima intervención del operador humano [20].

La idea de la auto-configuración en las redes de comunicaciones móviles es que el sistema pueda realizar configuraciones iniciales para nuevas estaciones base automáticamente. La configuración inicial se realiza a través de una fase de planificación y toma de decisiones, que además de las mediciones de las señales se consideran aspectos como pueden ser la ubicación de la estación base y las políticas del operador[4].

Dentro de la configuración de las redes de telecomunicaciones, la configuración inicial se carga en el nodo, por ejemplo, se cargan algoritmos de auto-configuración SON. La eficiencia de la configuración inicial es evaluada por la nueva estación base y sus vecinas. Así, el rendimiento de las configuraciones en diferentes escenarios se puede aprender y tener en cuenta en la próxima implementación. Como resultado de este enfoque, el operador humano no tiene que crear una configuración inicial para cada nueva estación base manualmente, sino que define los objetivos operativos para los nuevos nodos.

1.3.2.1.2. Auto-optimización

La auto-optimización de las redes SON normalmente corrige una configuración subóptima cambiando ligeramente los valores de los parámetros y monitoreando la reacción de la red [20].

[4] Las políticas de operador son las técnicas que permiten abstraer detalles técnicos y cambiar el comportamiento de un sistema en ejecución por acción o configuración de un usuario [20].

En caso de que se produzca un problema de configuración específico dos veces, la optimización podría acelerarse aplicando directamente los mismos valores anteriores del parámetro. A partir de esta premisa, se necesitan expertos bien capacitados para configurar las funciones de optimización con el fin de garantizar un rendimiento óptimo.

Una representación inicial permite inferir el estado actual probable del sistema en función de un análisis previo y además intuir las razones posibles de los problemas de rendimiento, por ejemplo, se pueden detectar problemas de capacidad reducida o parámetros de traspaso subóptimos, a partir de mediciones del rendimiento de la red.

Con base en este análisis, el sistema puede crear varios planes de optimización y utilizar la toma de decisiones en condiciones de incertidumbre para llegar a una estrategia óptima que comprenda varios parámetros de configuración que logre los objetivos de un extremo a otro.

1.3.2.1.3. Auto-corrección

La auto-corrección de las redes SON se centra en la detección y el diagnóstico de problemas utilizando algunos conocimientos adquiridos y la respectiva compensación para lograr alcanzar un estado de mejores características.

El objetivo principal de un proceso de auto-corrección automatizado es planificar y decidir las contramedidas más efectivas y eficientes, es decir, las acciones, para una falla diagnosticada dados algunos objetivos de un extremo a otro [20].

El proceso de auto-corrección incluye, entre otros, datos de configuración, datos de rendimiento y datos operativos como fecha y hora. Este conjunto de datos se utiliza para monitorear continuamente los efectos de las acciones realizadas y aprender la efectividad y eficiencia de las mismas en diferentes situaciones de falla. Esta información se puede utilizar para adaptar la toma de decisiones. Como resultado, el rendimiento mejorará con el tiempo gracias a la corrección aplicada.

1.3.3. PRESUPUESTO DE ENLACE

Dentro del escenario en el que se desarrolla este proyecto de titulación, es necesario la inclusión de una medida de potencia que le permita al algoritmo tomar las acciones necesarias para realizar el balanceo de cargas. Para ello, se utilizará el presupuesto del enlace cuya función será el cálculo de la potencia recibida en la estación móvil respecto a una estación base, considerando los parámetros de simulación descritos en la ecuación 1.9.

El presupuesto de enlace se calcula para estimar la cantidad de pérdida por trayectoria máxima permitida para cumplir con las condiciones mínimas de calidad de servicio de la señal recibida, que constituye uno de los factores principales en la ejecución de los procesos de *handover*. Potencias de transmisión, ganancia de antenas, pérdidas del sistema, márgenes de desvanecimiento, etc. son los diferentes parámetros que se toman en cuenta en el presupuesto del enlace.

El presupuesto del enlace nos entrega un valor medio de potencia recibida en base a las pérdidas máximas por trayectoria usando un modelo de propagación adecuado para el escenario específico donde se encuentra la red celular [21].

El cálculo del presupuesto del enlace se ejecutará con la aplicación de la siguiente fórmula general:

$$P_{RX} = P_{TX} + G_{TX} - L_{TX} - L_{FS} - L_M + G_{RX} - L_{RX} + G_{HO} \quad (1.9)$$

En donde:

P_{RX} = Potencia recibida en la estación móvil [dBm].

P_{TX} = Potencia de transmisión de la estación base [dBm].

G_{TX} = Ganancia de transmisión de la antena de la estación base [dBi].

L_{TX} = Pérdidas de transmisión (cables coaxiales, conectores, etc) [dB].

L_{FS} = Pérdidas de trayecto, pérdida de espacio libre [dB].

L_M = Margen de desvanecimiento [dB].

G_{RX} = Ganancia de recepción de la antena de la estación móvil [dBi].

L_{RX} = Pérdidas de recepción (cables coaxiales, conectores, etc) [dB].

G_{HO} = Ganancia de *handover* [dB].

1.3.3.1. Modelo de espacio libre

La propagación en el espacio libre puede calcularse de diferentes formas, pero para este trabajo de titulación el cálculo de las pérdidas en el trayecto se ejecutan considerando enlaces punto a punto entre los diferentes extremos de conexión en la red inalámbrica, por lo que se seguirán las relaciones matemáticas expuestas en la recomendación UIT-R P.525-2 [22].

1.3.3.1.1. Pérdida en trayectoria de espacio libre

Para un enlace punto a punto es preferible calcular la atenuación en el espacio libre entre antenas isótropas, denominada también pérdida básica de transmisión en el espacio libre o ecuación de Friis, de la siguiente manera [22]:

$$L_{bf} = 20 \log\left(\frac{4\,\pi\,d}{\lambda}\right) \qquad dB \tag{1.10}$$

Donde:

L_{bf} = pérdida básica de transmisión en el espacio libre (dB)

d = distancia

λ = longitud de onda

Se debe mencionar que d y λ se expresan en las mismas unidades.

La ecuación 1.10 puede también escribirse en función de la frecuencia en vez de la longitud de onda [22]:

$$L_{bf} = 32{,}4 + 20 \log f + 20 \log d \qquad dB \tag{1.11}$$

Donde:

f = frecuencia (MHz)

d = distancia (Km).

1.3.3.2. Margen de desvanecimiento

El margen de desvanecimiento es el factor en el que se toman en cuenta fenómenos como pérdidas por múltiples trayectorias, sensibilidad a la superficie del terreno, condiciones climatológicas, etc. Estos fenómenos podrían afectar la calidad de la señal recibida por lo que el margen de desvanecimiento se considera en el diseño de los enlaces con el objetivo mantener la calidad del sistema [23].

Tal como se indica en la figura 1.8, el nivel de la señal de entrada en función del tiempo, se podría encontrar en ciertos instantes por debajo del umbral o el nivel de recepción mínima. En este periodo de tiempo va a existir una degradación de rendimiento del sistema con una gran posibilidad de que se pueda ver interrumpida la conexión.

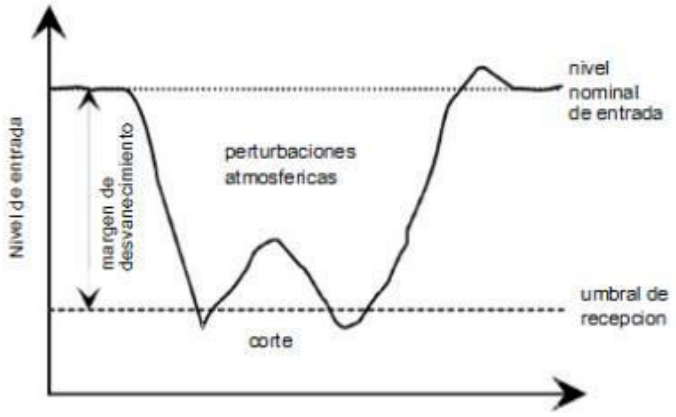


Figura 1.8. Esquema del margen de desvanecimiento.

Por lo que se puede concluir a partir de la figura 1.8, que, a mayor margen de desvanecimiento, menor será la probabilidad de que la señal caiga debajo del umbral. Lo que resulta en una mayor calidad de servicio, ya que la probabilidad de que la condición favorable de la conexión no cumpla se reduce.

1.3.3.2.1. Distribución probabilística de Rayleigh

La distribución de probabilidad de *Rayleigh* es una distribución de probabilidad continua de una variable aleatoria positiva. Dada una distribución de probabilidad normal bidimensional con dos variables aleatorias independientes y y z, de media cero y con la misma desviación típica σ, la variable aleatoria se describe en la siguiente ecuación [24]:

$$r = \sqrt{y^2 + z^2} \tag{1.12}$$

Dónde r se ajusta a una distribución de probabilidad de *Rayleigh*.

La función de distribución acumulativa de una distribución de probabilidad de *Rayleigh* viene dada por [24]:

$$F_R(r) = P[R \leq r] \begin{cases} 1 - e^{\left(-\frac{r^2}{2\sigma^2}\right)}, & r \geq 0 \\ 0, & r < 0 \end{cases} \tag{1.13}$$

Donde:

R = Es la variable aleatoria.

r = Representa el valor de umbral mínimo considerando el margen de desvanecimiento.

$P[R \leq r]$ = Es la probabilidad de que el nivel de la señal sea menor que el nivel del umbral más el margen de desvanecimiento.

El valor de σ^2 es la varianza calculada por [25]:

$$\sigma^2 = \left(\frac{4}{\pi} - 1\right) < r >^2 \quad (1.14)$$

Donde $< r >$ es el valor medio, o en este caso la potencia recibida calculada del presupuesto del enlace.

Siendo r el valor que se quiere despejar:

$$r = \sqrt{-2\sigma^2 \ln(1 - P[R \leq r])} \quad (1.15)$$

Cabe recalcar que el cálculo del valor de r será en vatios o mili vatios, por lo que el valor medio de la potencia recibida debe ser ingresada en las mismas medidas.

1.3.3.3. Sensibilidad del receptor en redes LTE

El concepto general de la sensibilidad del equipo receptor es un parámetro que determina el alcance del sistema. Este parámetro es de gran importancia pues determina el nivel de señal más débil que el receptor es capaz de recibir [26].

El nivel de potencia de sensibilidad de referencia en redes LTE es la potencia media mínima recibida en el conector de la antena del receptor, en la que el rendimiento debe ser el máximo posible para un canal de medición de referencia.

El ruido generado dentro del receptor limita la sensibilidad del receptor, siendo indispensables dos parámetros en la determinación de la calidad de la señal: La relación señal a ruido (SNR)[5] y la potencia de la señal en la salida [27].

Las medias de sensibilidad pueden definirse en términos de potencia (dBm) y/o tensión (dBμV); y como se puede observar en la tabla 2, el rango de valores de sensibilidad en la red LTE es la siguiente [28]:

Tabla 1.2. Rango de valores de sensibilidad en la red LTE.

Potencia de la señal	**Rango de nivel de potencia (dBm)**
Excelente	-75 a -88 dBm
Buena	-89 a -96 dBm
Regular	-97 a -105 dBm
Mala	-106 a -112 dBm
Pésima	-113 a -125 dBm

[5] SNR (Relación señal a ruido) El ruido de salida es un factor importante en cualquier medición de sensibilidad.

Para el desarrollo de este proyecto, durante la simulación se tomará el valor de referencia de -106 dBm como el nivel mínimo de sensibilidad que determinará el proceso de *handover* exitoso.

1.3.3.4. Variación del Margen de *Handover*

El valor de variación del Margen de *Handover*, cuya definición se detalla en la sección 1.3.1.2.1, es una cantidad en dB que se sumará (positivamente) a la sensibilidad de la estación base de la celda objetivo y se restará (negativamente) en la estación base de la celda sobrecargada, permitiendo que la cobertura de las celdas objetivo aumente, mientras que se disminuye la cobertura de la celda central (sobrecargada) en la misma proporción. Lo mencionado consiste en un proceso de aprendizaje por refuerzo que será explicado en la sección 1.3.4.1.3.

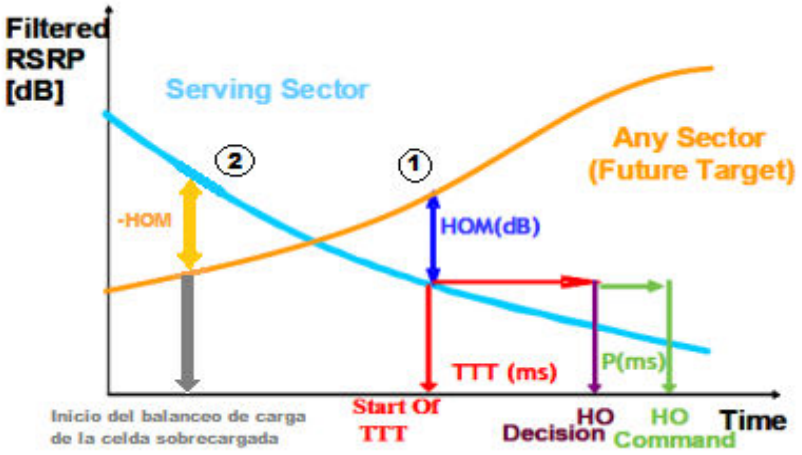


Figura 1.9. Esquema de la variación del Margen de *Handover* [11].

Tal como se puede observar en la figura 1.9. El primer escenario o instante (1) detalla el valor de margen de *handover* cuando la estación móvil se encuentra dentro de la zona de cobertura de otra celda que no es la de servicio, de esta forma la condición para que un *handover* sea exitoso en este escenario sería [4]:

$$P_{RX(CS)} + M_{HO(CS)} \leq P_{RX(CO)} \tag{1.16}$$

En donde:

$P_{RX(CS)}$ = Potencia recibida en la estación móvil respecto a la celda sobrecargada [dBm].

$P_{RX(CO)}$ = Potencia recibida en la estación móvil respecto a la celda objetivo o vecina [dBm].

M_{HO} = Margen de *handover* [dB].

El segundo escenario o instante (2) es el problema que se trata de resolver en este proyecto de titulación. En este escenario la estación móvil aún se encuentra dentro de la zona de cobertura de la celda de servicio. El problema de esta celda es que se encuentra sobrecargada y la estación móvil se encuentra dentro de las seleccionadas a realizar el *handover*, por esto, debe establecer la conexión hacia una estación base con la que ha obtenido menor valor de presupuesto de enlace calculado.

La variación de margen de *handover* permitirá cumplir con la condición de *handover* exitoso, por lo que, las estaciones base irán ajustando sus parámetros de *handover* hasta conseguir la redistribución de la carga, y, por tanto, la red alcanzará autónomamente y de forma automática un estado de balanceo de carga.

La condición de *handover* exitoso para este escenario o instante (2) es la siguiente:

$$P_{RX(CS)} + M_{HO(CS)} - \emptyset_{offset(CS)} \leq P_{RX(CO)} + \emptyset_{offset(CO)} \quad (1.17)$$

En donde:

$P_{RX(CS)}$ = Potencia recibida en la estación móvil respecto a la celda sobrecargada [dBm].

$P_{RX(CO)}$ = Potencia recibida en la estación móvil respecto a la celda objetivo o vecina [dBm].

M_{HO} = Margen de *handover* [dB].

$\emptyset_{offset}$ = Variación del margen de *handover* [dB].

Dicha variación de margen de handover se evaluará desde el valor del margen de handover calculado para la estación móvil que realizará el traspaso hasta que se cumpla condición descrita en la fórmula 1.17 [4].

1.3.4. *MACHINE LEARNING*

Una de las características de los seres humanos es que tienden a mejorar automáticamente su forma de abordar un problema. Los humanos aprenden de los errores anteriores y tratan de resolverlos corrigiéndolos o buscando nuevos enfoques para abordar el problema.

El campo del aprendizaje automático aborda este problema exacto e implica la creación de programas informáticos para que diferentes dispositivos tengan la posibilidad de poder aprender y, por lo tanto, mejorar su rendimiento mediante la recopilación de más datos y experiencia.

Las técnicas o clases de *machine learning* son algoritmos que tienen la habilidad de adquirir conocimiento extrayendo patrones de información o datos antiguos. Esta capacidad permite ejecutar tareas que no son complicadas para los seres humanos, pero que requiere mucho más conocimiento intuitivo y sujetos que lo puedan realizar, por lo tanto, el objetivo es enseñar a una computadora diferentes tareas usando un conjunto de reglas lógicas [20].

1.3.4.1. Clases de *Machine Learning*

Machine learning es utilizado para mejorar la efectividad y eficiencia de la etapa de decisión teniendo en cuenta nuevas experiencias aprendidas durante la operación. Por lo general, hay tres tipos de algoritmos de aprendizaje automático que se clasifican según el tipo de retroalimentación.

1.3.4.1.1. Aprendizaje supervisado

El aprendizaje supervisado se caracteriza por incluir todas las tareas en las que el algoritmo tiene acceso a la entrada y valores de salida.

Los métodos de aprendizaje supervisado suponen un conjunto de casos que contienen retroalimentación directa. Esto quiere decir que el conjunto de datos contiene premisas y además conclusiones. Esto significa que las estructuras de los datos ya se conocen y el objetivo de estos programas es asignar nuevos datos a las clases correctas [29].

Por ejemplo, el sistema presenta un conjunto de casos, cada uno con un conjunto de medidas (premisas) y un estado de falla de la red (conclusión), por lo que, el algoritmo aprende una función de mapeo entre mediciones y estados de falla [20].

1.3.4.1.2. Aprendizaje no supervisado

El aprendizaje no supervisado supone que el sistema no recibe ningún tipo de retroalimentación. Por lo tanto, el sistema no puede inferir ninguna conclusión de las premisas dadas. Sin embargo, aún puede aprender propiedades estadísticas, por ejemplo, distribuciones o grupos de los casos y, por lo tanto, se crea un modelo estadístico del entorno [20].

El Aprendizaje no supervisado incluye todas las tareas que no tiene acceso a los valores de salida y, por lo tanto, intenta encontrar estructuras dentro de los datos creando clases por su cuenta [29].

1.3.4.1.3. Aprendizaje por refuerzo

La estrategia general de aprendizaje por refuerzo utiliza un enfoque diferente al de las técnicas supervisadas. El aprendizaje por refuerzo es un área dentro del aprendizaje automático cuya meta es encontrar una política que mueva un agente de forma óptima por el entorno, del que generalmente se asume que es un Proceso de Decisión de *Markov* (MDP) [30].

Aprendizaje por refuerzo define el algoritmo de forma que se premie o castigue su comportamiento en función de las acciones que se vayan realizando. El agente que está conectado al entorno es capaz de interactuar con él, mediante la realización de determinadas acciones. Estas acciones y sus respectivas repercusiones llegarán en forma de refuerzo y harán transitar a otro estado diferente del que se encontraba [30].

- Procesos de decisión de Markov

Los procesos de decisión de *Markov* (MDPs) considera un tipo de problema de decisión secuencial donde se cumple la propiedad de *Markov*[6]. Estos procesos son utilizados para definir los problemas de aprendizaje por refuerzo.

- Elementos de un MDP

Los elementos de un proceso de decisión de *Markov* están definidos mediante el conjunto de variables (S, A, T, R) de forma que:

S: Conjunto de estados, donde st ∈ S es el estado en el que se encuentra el agente en el momento t.

A: Conjunto de acciones que el agente puede ejecutar para el estado st, donde at ∈ A(st) es la acción que el agente ejecuta en el instante t cuando transita el estado st.

T: S × A → P(S): Cada miembro de P(S) es una distribución de probabilidad sobre el conjunto de estados. Mientras que T(s, a, s0) es la probabilidad de que se transite del estado s a s0 ejecutando la acción a.

R: S × A → R: R(s, a): Es recompensa recibida tras ejecutar la acción a en el estado s.

[6] Propiedad de Markov es la regla de ciertos procesos estocásticos en que el estado que experimente el sistema en un tiempo t+1, solo depende del estado en el que se encontraba el agente en el tiempo t y la acción realizada en ese instante [30].

El agente tiene como objetivo encontrar la política, π : S × A → [0, 1], donde πt(s, a) es la probabilidad de que el agente seleccione la acción a en el estado s, maximizando de esta forma el valor de la recompensa recibida [31].

- Funcionamiento de un MDPs

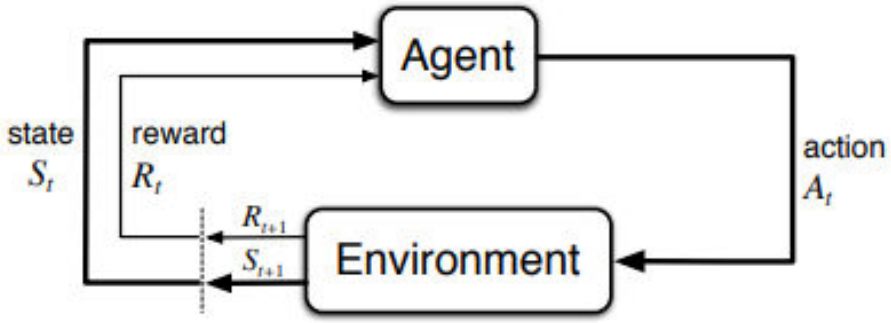


Figura 1.10. Diagrama de un proceso de decisión de Markov [30].

Como se puede observar en la figura 1.10, la descripción teórica del funcionamiento del aprendizaje por refuerzo se describe según los siguientes puntos:

- El agente aprende un comportamiento a través de interacciones de prueba y error en un ambiente dinámico e incierto [32].
- Una propiedad de este proceso es que el sistema no tiene alguna intuición preestablecida de que acción debe realizar, sino que el propio sistema determinará las acciones a tomar según el máximo beneficio [32].
- El agente tiene como entrada el estado actual (s ∈ S) y realiza una acción (a ∈ A). La acción cambia el estado del sistema y el agente recibe una recompensa (r ∈ R) [32].
- El agente sigue un comportamiento determinado decidiendo en cada momento las acciones que ha de ejecutar. Esta será determinada como una política denotada por π [31].
- La política del agente, π, elige acciones que incrementen la suma de todas las recompensas recibidas a lo largo del tiempo, siguiendo algún criterio de optimalidad [31].
- Los sistemas basados en procesos de decisión de *Markov* tienen las siguientes características:
- El agente seleccionará las acciones que pueden incrementar a largo plazo la suma de las recompensas totales.

- El objetivo del agente es encontrar una política (π, que mapea estados a acciones) que maximice a largo plazo el refuerzo acumulado.
- El ambiente del sistema es no-determinístico, es decir, la misma acción en el mismo estado puede dar resultados diferentes.

Para mejor entendimiento del funcionamiento del aprendizaje por refuerzo, a continuación, se presenta un ejemplo del proceso que realiza el algoritmo aplicando los conceptos descritos anteriormente sobre handover y balanceo de cargas.

Para este ejemplo se va a suponer que existen dos usuarios que están sobrecargando a la celda central, para ambos usuarios se han calculado el presupuesto del enlace respecto a la estación base de la celda central y la estación base vecina que recibirá la conexión.

Con el primer usuario, el presupuesto del enlace respecto a la celda central es de -102 dBm, y respecto a la celda vecina es -104 dBm. Mientras que, para el segundo usuario, el presupuesto del enlace respecto a la celda central es de -101 dBm, y respecto a la celda vecina es -105 dBm. La represntación gráfica del ejemplo propuesto se observa en la figura 1.11.

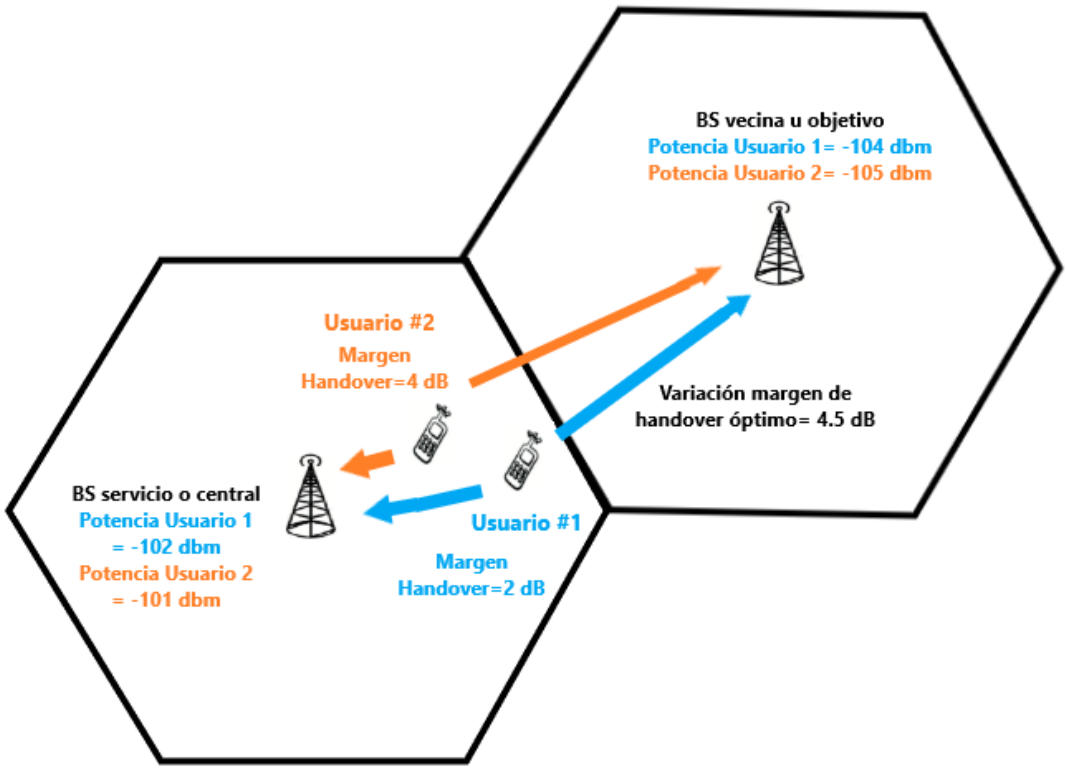


Figura 1.11. Diagrama y resultados del ejemplo de balanceo de cargas.

Considerando la condición de *handover* exitoso descrito en la ecuación 1.17 y las definiciones de margen de handover y variación de margen de handover, los resultados serían los siguientes:

- El margen de handover del primer usuario es de 2 dB, mientras que, para el segundo usuario es de 4 dB.
- El primer usuario no cumpliría la condición de handover exitoso por 4 dBm, mientras que, el segundo usuario no lo haría por 8 dBm.

A partir de este punto el algoritmo empezará a aumentar en pasos la variación de margen de handover una cierta cantidad de dB seleccionada por el usuario.

Para que el algoritmo no tenga fallos en la selección de la variación de margen de handover, con la ayuda del aprendizaje por refuerzo se aplica un sistema de recompensas que premia o castiga al agente (estación base) cada vez que se evalúe la condición de handover exitoso. El objetivo del algoritmo es alcanzar la meta que para este ejemplo sería el cumplir la condición de handover exitoso en ambos usuarios.

Si el valor del paso con el que se aumenta el valor del margen de handover es 0.5 dB, la recompensa positiva es +1 y la recompensa negativa es -1, el algoritmo necesitaría aumentar este valor 5 veces (sexta interacción) para que el primer usuario cumpla la condición de handover exitoso, mientras que, el segundo usuario cumpliría la condición en 10 interacciones.

La asignación de recompensas y el estado del sistema ante las acciones del agente serían las siguientes:

- Desde la primera interacción hasta la quinta se recompensa negativamente al agente (Estación base), por lo que, la recompensa total sería -2 (-1 por el primer usuario y -1 por el segundo usuario).
- En la sexta interacción el agente recibe su primera recompensa positiva, debido a que el primer usuario ha cumplido con la condición. La recompensa total sería 0 (+1 por el primer usuario y -1 por el segundo usuario).
- En la décima interacción el agente recibe una segunda recompensa positiva ya que el segundo usuario cumplió con la condición, la recompensa total sería +2, por lo que, el algoritmo ha logrado el objetivo de cumplir la condición para ambos usuarios.

Una vez que se ha alcanzado el objetivo, el agente guarda en una tabla de recompensas el valor de variación de margen de handover con el que se ha cumplido la condición para ambos usuarios, que para este ejemplo es 4.5 dB.

- Funciones de valor

El objetivo del aprendizaje por refuerzo es obtener la política de acción π con la aproximación de funciones de valor.

Las funciones de valor evalúan una determinada política, su evaluación consiste en determinar si es apropiado que el agente permanezca en un estado o si es mejor realizar una acción hacia otro estado. Existen dos tipos de funciones de valor, la función de valor-estado y la función de valor-acción [31].

Atender un problema con el uso de algoritmos basados en el aprendizaje por refuerzo consiste en buscar una política que alcance la mayor cantidad de recompensas positivas a largo plazo. Por lo que para un proceso de decisión de *Markov*, se debe precisar siempre las políticas óptimas.

- Función de valor-estado

La función de valor para un estado s y una política $\pi, V^{\pi}(s)$, es la recompensa que se espera obtener siguiendo una determinada política π a partir del estado s. Dicha función se puede observar en la ecuación 1.18. [31].

$$V^{\pi}(s) = E_{\pi}\{\sum_{k=0}^{\infty} \gamma^{k} r_{t+k+1} | s_t = s\} \quad (1.18)$$

Donde:

$V^{\pi}(s)$ = Función de valor estado.

π = Política de la función.

k= Número de repetición.

Eπ= Valor esperado cuando un agente sigue una determinada política π.

γ = Factor de descuento.

r= Recompensa del sistema.

s= Estado del sistema.

t= Instante de tiempo en el que se ejecuta la acción.

- Función de valor-acción

La función de valor-acción, Qπ (s, a), se define como la recompensa a recibir si el sistema aplica una determinada política de acción π tras ejecutar la acción a en el estado s [31]. Dicha función se puede observar en la ecuación 1.19:

$$Q^{\pi} = E_{\pi}\{\sum_{k=0}^{\infty} \gamma^{k} r_{t+k+1} | s_t = s,\ a_t = a\} \tag{1.19}$$

Por las propiedades del aprendizaje por refuerzo, el algoritmo siempre encontrará políticas mejores con el transcurso de las interacciones realizadas, estas políticas óptimas se denotan por π∗. La función se describe en la ecuación 1.20. [31]:

$$V^{*}(s) = \max_{\pi} V^{\pi}, \forall s\ \in S \tag{1.20}$$

Por su parte, la función de valor-acción óptima se describe en la ecuación 1.21. [31]:

$$Q^{*}(s,a) = \max_{\pi} Q^{\pi}(s,a), \forall s\ \in S, \forall a\ \in A \tag{1.21}$$

La importancia de dichas funciones es que a partir de estas definiciones matemáticas se obtiene la política óptima, como se puede observar en la ecuación 1.22. [31].

$$\pi^{*}(n) = \arg\max_{a} Q^{*}(s,a) \tag{1.22}$$

1.3.4.2. Programación dinámica (DP *Dynamic Programming*)

El término programación dinámica (DP) se refiere a un conjunto de algoritmos que se utilizan para calcular políticas óptimas como en los algoritmos basados en el aprendizaje por refuerzo que se basan en un modelo perfecto del entorno como un proceso de decisión de *Markov* (MDP). Los algoritmos de programación dinámica DP clásicos son de utilidad limitada en el aprendizaje por refuerzo por su suposición de un modelo perfecto. Este tipo de programación consume muchos recursos computacionales, pero por su utilidad siguen siendo importantes teóricamente [30].

La programación dinámica tiene un problema que es la necesidad de recursos. El número de recursos puede crecer exponencialmente con el número de variables de estado, pero sigue siendo mucho más eficiente y aplicable que cualquier otro método para problemas estocásticos.

Teniendo conocimiento completo del proceso de decisión de *Markov*, los algoritmos basados en programación dinámica actualizan constantemente los valores de las funciones de valor-estado $V^{*}(s)$ y de las funciones de valor-acción $Q^{*}(s,a)$, de tal forma que una vez

obtenidos los valores óptimos de cada función se obtiene la política de comportamiento óptima.

1.3.4.3. Algoritmo *Q-Learning*

Dentro de los aspectos con mayor trascendencia del aprendizaje por refuerzo es el desarrollo del algoritmo "fuera-de-política" conocido como *Q-learning*.

El algoritmo *Q-Learning* toma los conceptos más importantes del aprendizaje por refuerzo y se fundamenta en el aprendizaje a partir de la experiencia que se genera durante la exploración del entorno.

La expresión que define el funcionamiento del algoritmo *Q-learning* se describe en la ecuación 1.23. [4].

$$Q^{nuevo}(s_t, a_t) \leftarrow (1-\alpha) * Q(s_t, a_t) + \alpha * (r_t + \gamma * \overset{máx}{a} Q(s_{t+1}, a)) \quad (1.23)$$

Donde:

$Q^{nuevo}(s_t, a_t)$ = Nuevo valor de la función Q o valor Q actualizado según la nueva acción tomada.

α = Velocidad de aprendizaje.

r_t = Recompensa o refuerzo obtenido en el instante t.

$Q(s_t, a_t)$ = Valor antiguo de la función Q al momento de ejecutar la nueva acción.

γ = Factor de descuento.

$\overset{máx}{a} Q(s_{t+1}, a)$ = Valor futuro óptimo estimado.

Dependiendo del tipo y del número de acciones que el agente tome para interactuar con su entorno, llegará un instante en que el algortimo *Q-learning* garantizará la convergencia.

El proceso de convergencia del algoritmo de *Q-learning* se basa en los procesos de decisión de *Markov*, en donde la actualización de los valores *Q-learning* sigue los siguientes pasos:

- El agente inicializa su tabla Q[7] con un valor pequeño arbitrario para cada acción y estado de inicio.
- El agente comienza la interacción tomando un estado de su entorno.

[7] Los algoritmos *Q-learning* almacenan información cuantitativa en tablas. Estas tablas denominadas "tablas Q" guardan los valores de los estados del sistema y acciones que toma el agente en diferentes instantes de tiempo con la respectiva aproximación de la recompensa retornada por el sistema [46].

- Después el agente selecciona la acción que puede conducir al siguiente estado del valor Q más alto. Se debe tener en cuenta que en las primeras interacciones del agente con su entorno (etapa de aprendizaje temprano), se elige sin ningún conocimiento y de manera aleatoria la acción ejecutada por el agente, esto se debe al poco o nulo conocimiento que tiene el agente sobre su entorno.
- El entorno reacciona a la acción del agente y este le indica cual debería ser su próximo estado y la recompensa que obtiene de la acción tomada por el agente.
- El agente recibe la recompensa del entorno y actualiza los datos para calcular el valor Q de este par de estado-acción. Luego el agente actualiza el valor calculado en la tabla Q.
- El agente repetirá el proceso de elegir una acción y recibir una recompensa correspondiente hasta que la tabla Q converja[8].
- Una vez que la tabla Q converge, el agente siempre puede proporcionar la mejor acción [33].
- Con los valores de recompensa obtenidos se actualizará la fórmula del algoritmo *Q-learning* descrita en la ecuación 1.23.

1.3.4.3.1. Factor de descuento γ

El factor de descuento γ toma un valor dentro del rango entre 0 y 1, cuanto mayor sea el valor del factor de descuento mayor será la visión que tendrá el algoritmo, es decir, el algoritmo será capaz de tomar más en cuenta el valor de acciones para futuras recompensas, también denominadas recompensas de largo plazo o *long term reward*. Este concepto de recompensas a largo plazo indica la imposibilidad de obtener inmediatamente recompensas positivas después de aplicar una acción, el objetivo del factor de descuento con valores cercanos a 1 es de obtener recompensas positivas tras la ejecución de varias acciones cuando se conoce si una acción concreta es positiva [34].

También es posible usar procesos de decisión de *Markov* aplicados a algoritmos *Q-learning* sin factor de descuento, es decir, con el valor de factor de descuento igual a 0 cuando todas las secuencias pueden terminar o cuando solo se necesitará el valor de recompensas inmediatas. El valor calculado en la ecuación del algoritmo *Q-learning* descrito en la ecuación 1.21, solo se utilizará en la siguiente acción tomada por el agente [35], este será el caso del presente proyecto de titulación.

[8] La convergencia de valores de una tabla Q garantiza que, realizando un número adecuado de iteraciones, las recompensas obtenidas indican al agente la mejor acción a realizar para poder acercarse cada vez más al valor buscado u objetivo [45].

1.3.4.3.2. Velocidad de aprendizaje

La velocidad de aprendizaje α toma un valor mayor a 0 y menor a 1. Si el valor de la velocidad de aprendizaje es igual a 0 (α = 0) provocará que los valores calculados por el algoritmo *Q-learning* descrita en la ecuación 1.21, nunca sean actualizados, y, por lo tanto, provocará que el algoritmo nunca pueda aprender. En cambio, si el valor de la velocidad de aprendizaje es igual a 1 (α = 1), provocará la máxima actualización posible en cada iteración. Se debe tener en cuenta que en el proceso se puede también olvidar lo aprendido en acciones anteriores, si el valor de velocidad de aprendizaje es más cercano a cero. Encontrar e valor adecuado de velocidad de aprendizaje es fundamental para potenciar el rendimiento del *Q-Learning* [34], siendo esta la idea que el aprendizaje por refuerzo persigue.

1.3.5. PERFILES ANGULARES DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL

Dentro del desarrollo de este proyecto de titulación se analizará y calculará el perfil angular de potencia de llegada en la estación móvil. Este cálculo se desarrolla como una simulación adicional al programa principal de *machine learning* para el balanceo de cargas.

Esta simulación permite conocer la incidencia de las señales respecto al ángulo de llegada de la señal para entornos con línea de vista (LoS) y sin línea de vista (NLoS) entre la estación móvil y la estación base de la celda seleccionada del proceso de *handover*. Como parámetros de entrada para esta simulación, el programa obtiene la información de los usuarios que realizaron el proceso de *handover* anteriormente, la potencia de recepción calculada respecto a la celda objetivo que recibirá la conexión y la distancia entre estos dos elementos de red. Por lo que este cálculo no forma parte del proceso de aprendizaje por refuerzo.

El perfil angular de potencia de llegada instantáneo es la densidad de potencia de la respuesta al impulso respecto al ángulo de llegada en un instante y en una ubicación determinada [36]. Este parámetro se describe mediante la figura 1.12.

El perfil angular de llegada en la estación móvil está descrito en la Recomendación UIT-R P. 1816-4. en el Anexo 3.

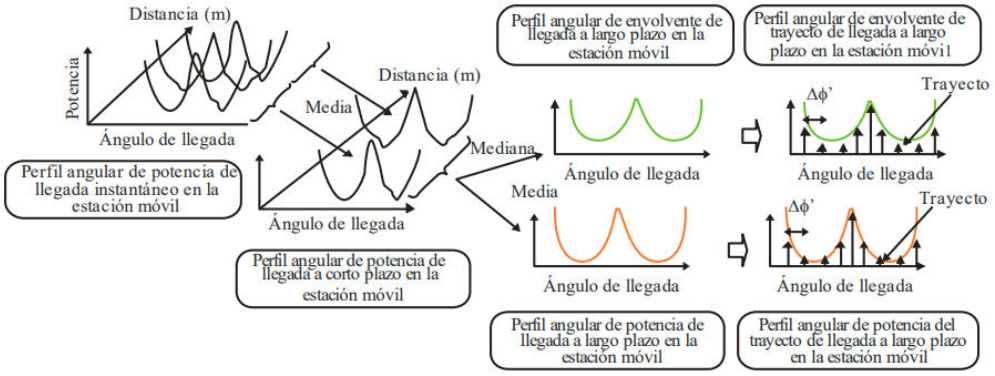


Figura 1.12. Perfiles angulares de llegada en la estación móvil [36].

1.3.5.1. Parámetros recomendación UIT-R P.1816-4 ANEXO 3

Los parámetros necesarios para el cálculo del perfil angular de llegada en la estación móvil y los valores recomendados para su cálculo según la UIT son los siguientes:

hb = Altura de antena de la estación de base, este parámetro se mide en metros y recomienda un valor de entre 5 a 150 metros; altura sobre el nivel del suelo de la estación móvil.

$< H >$ = Altura media del edificio, este parámetro se mide en metros y recomienda un valor de entre 5 a 50 metros; altura sobre el nivel del suelo de la estación móvil.

d = Distancia desde la estación de base, este parámetro se mide en kilómetros y recomienda un valor de entre 0,5 a 3 kilómetros.

W = Anchura de la calle, este parámetro se mide en metros y recomienda un valor de entre 5 a 50 metros.

f = Frecuencia portadora, este parámetro se mide en GHz y recomienda un valor de entre 0,7 a 9 GHz.

θ= Ángulo de la carretera, este es el ángulo agudo entre la dirección de la estación móvil y la dirección de la carretera, este parámetro se mide en grados y varía en el rango de 0 a 90 grados.

hs= Altura media de los edificios a lo largo de la carretera, este parámetro se mide en metros y recomienda un valor de entre 4 a 30 m.

φ'= Ángulo de llegada, este es el ángulo de llegada cuando se fija el ángulo de la carretera a un valor de 0 grados, este valor varía en el rango de -180 a 180 grados.

$< R >$ = Coeficiente de reflexión de potencia medio de los muros laterales del edificio, este párametro toma un valor constante comprendido entre 0,1 y 0,5 [36].

γ mW $= 10^{\gamma\, dBm/10}$ = Este es el valor del cálculo de presupuesto del enlace, que para este proyecto de titulación será calculado con el valor de nivel de señal recibido respecto a la celda objetivo.

1.3.5.2. Perfil angular de llegada a largo plazo en la estación móvil para entornos NLoS en zonas urbanas y suburbanas

1.3.5.2.1. Perfil angular de llegada en la estación móvil NLoS

El perfil angular de potencia de llegada en la estación móvil, $AOA_{NLoS,pow}(\varphi')$ viene dado por [36]:

$$AOA_{NLoS,pow}(\varphi') = \frac{1}{\sqrt{\cos\left(\varphi' * \frac{\pi}{180}\right) + \sin\left(\varphi' * \frac{\pi}{180}\right)/n^2}} \tag{1.24}$$

Donde [36]:

$$n = mín(1, [\frac{2.6}{h_s^{0,5}} * \{1 - e^{-0.03 * \theta}\} + 0.05]^{1.5}) \tag{1.25}$$

1.3.5.3. Perfil angular de llegada a largo plazo en la estación móvil para un entorno LoS en zonas urbanas y suburbanas

De acuerdo a la recomendación UIT-R P.1816-4 en el Anexo 3 se tienen varios entornos LoS considerados, que son descritos en las siguientes subsecciones.

1.3.5.3.1. Estación base en la cima de un edificio ubicado en el lado izquierdo de la calle

Como se puede observar en la figura 1.13, en este entorno LoS considerado, la estación base está situada en la cima de un edificio ubicado en el lado izquierdo de la calle y la estación móvil se encuentra en medio de la calle; en este caso la estación base tiene una visibilidad directa con la estación móvil.

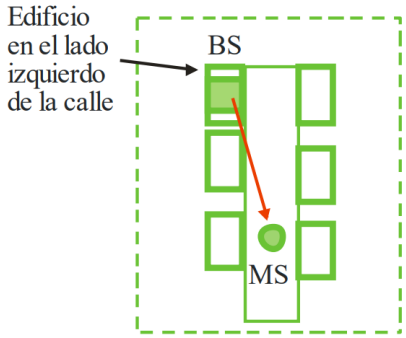


Figura 1.13. Entorno LoS cuando la BS está ubicada al lado izquierdo de la calle [36].

Para este caso el perfil angular de potencia de llegada en la estación móvil $AOA_{LoS,pow}(\varphi', d)$ se puede calcular mediante la ecuación 1.26. [36].

$$AOA_{LoS,pow}(\varphi', d) =$$

$$\begin{cases} AOA_{LoS,pow}(\varphi', d) =< R >^{1000*d*\left|\varphi'\right|*\frac{\pi}{180*W}-1} + \gamma * AOA_{NLoS,pow}(\varphi'), \quad (\varphi' \geq 0) \\ AOA_{LoS,pow}(\varphi', d) =< R >^{1000*d*\left|\varphi'\right|*\frac{\pi}{180*W}} + \gamma * AOA_{NLoS,pow}(\varphi'), \quad (\varphi' < 0) \end{cases} \tag{1.26}$$

1.3.5.3.2. Estación base en la cima de un edificio ubicado en el lado derecho de la calle

Como se puede observar en la figura 1.14, en este entorno LoS considerado, la estación base está situada en la cima de un edificio ubicado en el lado derecho de la calle y la estación móvil se encuentra en medio de la calle; en este caso la estación base tiene una visibilidad directa con la estación móvil [36].

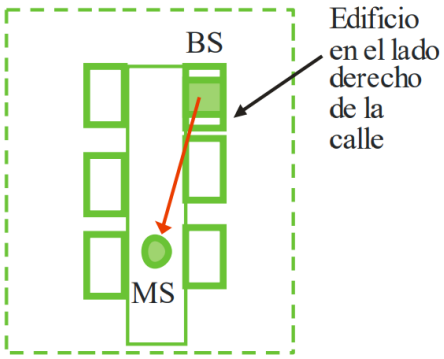


Figura 1.14. Entorno LoS en donde la BS está ubicada al lado derecho de la calle [36].

Para este caso el perfil angular de potencia de llegada en la estación móvil $AOA_{LoS,pow}(\varphi', d)$ se puede calcular mediante la ecuación 1.27. [36].

$$AOA_{LoS,pow}(\varphi', d) =$$

$$\begin{cases} AOA_{LoS,pow}(\varphi', d) =< R >^{1000*d*|\varphi'|*\frac{\pi}{180*W}} + \gamma * AOA_{NLoS,pow}(\varphi'), \ \ (\varphi' \geq 0) \\ AOA_{LoS,pow}(\varphi', d) =< R >^{1000*d*|\varphi'|*\frac{\pi}{180*W}-1} + \gamma * AOA_{NLoS,pow}(\varphi'), \ \ (\varphi' < 0) \end{cases} \quad (1.27)$$

1.3.5.3.3. Estación base ubicada en la cima de un edificio frente al extremo de la calle

Como se puede observar en la figura 1.15. En este entorno LoS considerado, la estación base está situada aproximadamente en el centro del tejado de un edificio que se encuentra en el extremo de la calle y la estación móvil está en medio de la calle [36].

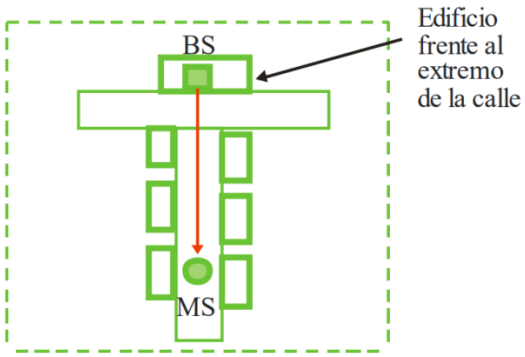


Figura 1.15. Entorno LoS en donde la BS está frente al extremo de la calle [36].

Para este caso el perfil angular de potencia de llegada en la estación móvil $AOA_{LoS,pow}(\varphi', d)$ se puede calcular mediante la ecuación 1.28.

$$AOA_{LoS,pow}(\varphi', d) =< R >^{1000*d*|\varphi'|*\pi/(180*W)} + \gamma * AOA_{NLoS,pow}(\varphi') \quad (1.28)$$

La recomendación UIT-R P.1816-4 en el Anexo 3, recomienda que los valores de γ y $< R >$ tomen los valores de −15 dB y 0,3 (−5 dB), respectivamente, cuando la altura media del edificio sea superior a 20 m.

2. METODOLOGÍA

En este capítulo se describe el proceso de balanceo de cargas en una celda LTE desarrollando un algoritmo con las herramientas del aprendizaje por refuerzo simulado en una interfaz *GUIDE* de MATLAB.

Además, se detalla el proceso para la simulación en una interfaz *GUIDE* de MATLAB el cálculo del perfil angular de potencia de llegada para entornos NLoS y LoS, tal como se describe en la recomendación UIT-R P.1816-4 en el anexo 3. Esta segunda simulación se ejecutará para uno de los usuarios que realizarán el proceso de *handover* según se desarrolle el proceso de balanceo de cargas.

Los procedimientos realizados en los scripts se detallarán mediante diagramas de flujo, en los cuales se indican parámetros, procedimientos y funciones.

Los scripts desarrollados realizarán las siguientes tareas:

- Creación de un sistema LTE que experimenta sobrecarga de usuarios en la celda central.
- Selección de los usuarios que realizarán el balanceo de cargas hacia celda vecinas con recursos disponibles.
- Cálculo de la variación de margen de *handover* óptimo mediante aprendizaje por refuerzo (Algoritmo *Q-learning*).
- Representación gráfica del perfil angular de potencia de llegada en la estación móvil para entornos NLoS y LoS.

Los diagramas de flujo que describen el funcionamiento de cada script están descritos en las figuras de Anexo A y el código de cada interfaz o función se encuentran en el Anexo B.

A continuación, se detalla el funcionamiento del programa de balanceo de cargas y el cálculo de la variación del margen de *handover* mediante aprendizaje por refuerzo.

2.1. PROGRAMA DE BALANCEO DE CARGAS MEDIANTE APRENDIZAJE POR REFUERZO

La primera simulación estima el caso de una red celular LTE que consta de un arreglo de 7 celdas de ancho de banda de 5 MHz cada una. La celda central tiene sobrecarga de usuarios, mientras que las celdas vecinas disponen de recursos para poder recibir las conexiones de los usuarios que sobrepasan la capacidad de la celda central.

El número de usuarios de cada celda y los parámetros de evaluación de la simulación serán ingresados por el usuario, y el programa simulará tanto gráfica como numéricamente cada uno de los procesos descritos en la figura 2.1 El diagrama de flujo que describe estos procesos se muestra en la Figura A.1 del Anexo A.

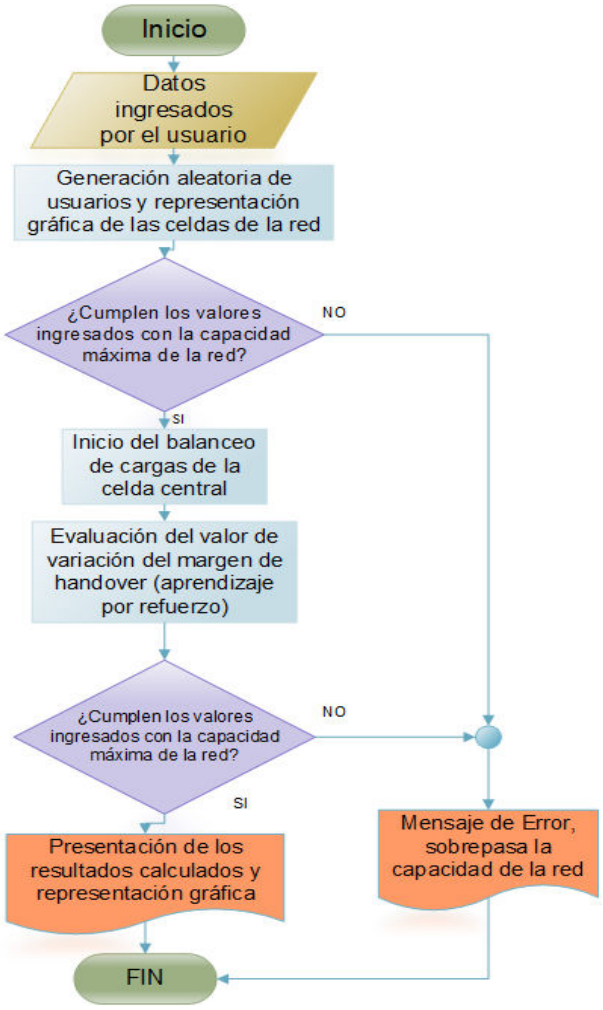


Figura 2.1. Diagrama de flujo del programa de balanceo de carga

2.1.1. DESCRIPCIÓN DE LA INTERFAZ GRÁFICA Y OBTENCIÓN DE DATOS INICIALES

La interfaz gráfica de la primera simulación está compuesta de varias entradas de datos ingresadas por el usuario y se tienen salidas como representaciones gráficas del sistema y resultados calculados. La interfaz está comentada en la figura 2.2.

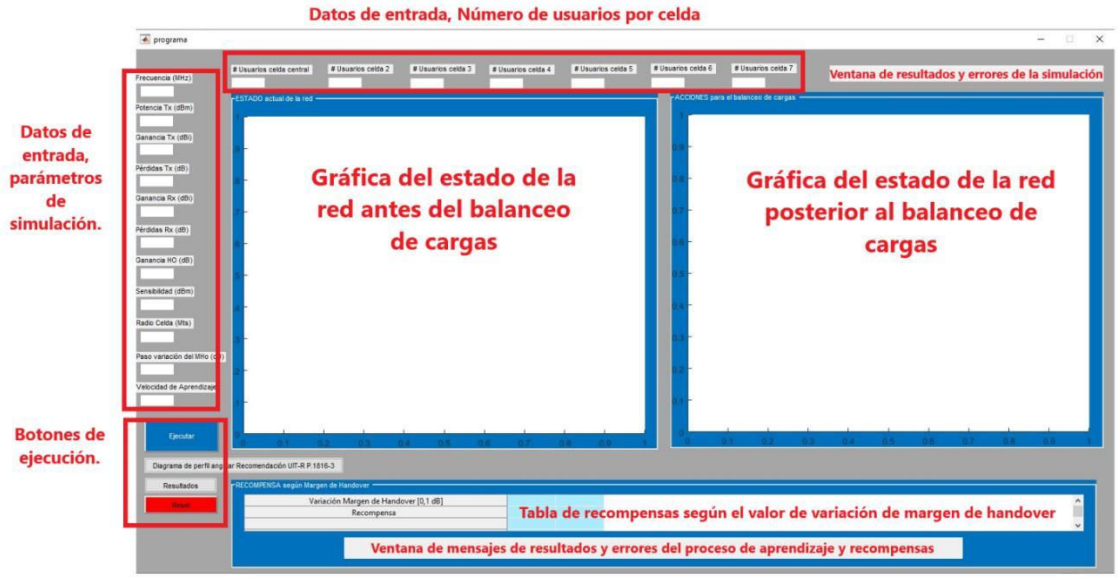


Figura 2.2. Interfaz del programa de balanceo de carga y aprendizaje por refuerzo

Para el inicio de la simulación el programa realiza la obtención de los valores ingresados por el usuario. El segmento de código 2.1, indica la lectura y conversión de los valores de los parámetros de la simulación.

```
%Obtención de los valores ingresados por el usuario
f=str2num(get(handles.f,'string'));            %Frecuencia de portadora
Ptx=str2num(get(handles.Ptx,'string'));        %Potencia de transmisión de la
estación base
Gtx=str2num(get(handles.Gtx,'string'));        %Ganancia de transmisión de la
antena de la estación base
Ltx=str2num(get(handles.Ltx,'string'));        %Pérdidas de transmisión
Grx=str2num(get(handles.Grx,'string'));        %Ganancia de recepción de la antena
de la estación móvil
Lrx=str2num(get(handles.Lrx,'string'));        %Pérdidas de recepción
Ghm=str2num(get(handles.Ghm,'string'));        %Ganancia de handover
Sen=str2num(get(handles.Sen,'string'));        %Sensibilidad de receptor
r=str2num(get(handles.r,'string'));            %Obtención valor de radio de la
celda celular
pasos=str2num(get(handles.pasos,'string'));    %Obtención valor de variación del
Margen de Handover
learning=str2num(get(handles.learning,'string'));     %Velocidad de Aprendizaje
```

Segmento de código 2.1. Comandos para la obtención de los valores ingresados por el usuario.

La sintaxis para la obtención de los valores ingresados por el usuario en un programa desarrollado en *GUIDE* de Matlab es la siguiente:

Variable = str2num (get(handles.tag,'string'))

El comando “*str2num*” realiza la conversión de una matriz de caracteres a una matriz numérica.
El comando “*get*” indica la obtención del *string* de datos ingresados por el usuario en el *textbox* o cuadro de texto del *tag* indicado.

En el segmento de código 2.1, se realiza la obtención y conversión numérica de las siguientes variables:

- Frecuencia de portadora LTE (MHz).
- Potencia de transmisión de la estación base (dBm).
- Ganancia de transmisión de la estación base (dB).
- Pérdidas en los terminales de recepción y transmisión (dB).
- Ganancia de *handover* (dB).
- Potencia de recepción de la estación base (dBm).
- Ganancia de recepción de la estación base (dB).
- Sensibilidad de recepción (dBm).
- Radio de la celda celular (m).
- Valor de variación del margen de *handover* (dB).

En el segmento de código 2.2, se realiza la obtención y conversión numérica de la cantidad de usuarios que serán generados y graficados de forma aleatoria en cada una de las celdas.

```
%Obtención Número de usuarios de cada celda
usuarioscentral=str2num(get(handles.usuarioscentral,'string'));     %Número     de
usuarios de la celda central #1
usuarios(2)=str2num(get(handles.usuarios2,'string'));          %Número de usuarios
de la celda #2
usuarios(3)=str2num(get(handles.usuarios3,'string'));          %Número de usuarios
de la celda #3
usuarios(4)=str2num(get(handles.usuarios4,'string'));          %Número de usuarios
de la celda #4
usuarios(5)=str2num(get(handles.usuarios5,'string'));          %Número de usuarios
de la celda #5
usuarios(6)=str2num(get(handles.usuarios6,'string'));          %Número de usuarios
de la celda #6
usuarios(7)=str2num(get(handles.usuarios7,'string'));          %Número de usuarios
de la celda #7
usuarios(1)=usuarioscentral;
```

Segmento de código 2.2. Comandos para la obtención del número de usuarios de cada celda.

2.1.2. CREACIÓN DE LAS CELDAS HEXAGONALES Y REPRESENTACIÓN GRÁFICA

Para la creación de la figura hexagonal, los comandos del segmento de código 2.3, realizan las operaciones matemáticas para determinar los valores en el eje X y Y de cada uno de los vértices de la celda. Para la celda central se toman los valores de centro en 0 para ambos ejes.

```
%CREACIÓN DE LA CELDA SOBRECARGADA

N=6; %NÚMERO DE LADOS, CELDA HEXAGONAL

cx(1)=0;     %Desfase del centro de la celda central en el eje X
cy(1)=0;     %Desfase del centro de la celda central en el eje Y

t = (1/(2*N):1/N:1)*2*pi;    % Traza un polígono de N lados
x = r*sin(t)+cx(1);          % Valores de los vértices de la celda en el eje X
y = r*cos(t)+cy(1);          % Valores de los vértices de la celda en el eje Y
x(7)=x(1);                   % Unión del primer y último vértice, eje X
y(7)=y(1);                   % Unión del primer y último vértice, eje Y
```

Segmento de código 2.3. Comandos para la creación de la celda central.

Cada vértice y segmento calculado son graficados para los diagramas de los dos estados de la red[9], además, en el centro de cada celda se desplegará el nombre de la celda con su número identificativo.

Los valores de los ejes X y Y son almacenados en la matriz de valores "*x*" y "*y*" respectivamente, para su posterior representación gráfica, tal como se describe en el segmento de código 2.4.

```
axes(handles.axes1)              %Primera Gráfica
hold on
grid on
plot(x,y,'b');                   %Gráfica de los segmentos de la celda
text(cx(1),cy(1),'Celda #1');    %# de la celda escrita en el centro de la celda
axis square;                     %Cuadrícula

axes(handles.axes2)              %Segunda Gráfica
hold on
grid on
plot(x,y,'b');                   %Gráfica de los segmentos de la celda
text(cx(1),cy(1),'Celda #1');    %# de la celda escrita en el centro de la celda
axis square;                     %Cuadrícula
```

Segmento de código 2.4. Representación gráfica de la celda central.

[9] La red celular antes y después del proceso de balanceo de cargas son los dos estados mencionados.

Para la creación de las demás celdas objetivo se realiza una modificación en el valor del centro de cada una de las celdas, para que de esta forma se pueda generar la red celular de 7 celdas adyacentes. El valor del desplazamiento en cada eje para cada una de las celdas, se describe en la tabla 2.1.

El valor de "*y(1)*" es el valor en el eje Y que tiene el segmento superior de la celda central, este valor se tomará de referencia para el desplazamiento de los centros de las celdas vecinas u objetivos.

Tabla 2.1. Generación de los centros de las celdas creadas en los ejes X y Y.

Número de celda	**Desplazamiento en el eje X**	**Desplazamiento en el eje Y**
Celda central / Celda N°1	0	0
Celda N°2	0	$y(1) * 2$
Celda N°3	0	$-y(1) * 2$
Celda N°4	$y(1) * 2 * \cos(\pi/6)$	$y(1)$
Celda N°5	$y(1) * 2 * \cos(\pi/6)$	$-y(1)$
Celda N°6	$-y(1) * 2 * \cos(\pi/6)$	$y(1)$
Celda N°7	$-y(1) * 2 * \cos(\pi/6)$	$-y(1)$

Los valores de cada uno de los centros son almacenados en una matriz. Estos serán las coordenadas de las estaciones base con las que se realizará el presupuesto del enlace para el posterior balanceo de cargas.

Los valores de los vértices calculados para las celdas vecinas se calculan respecto al desplazamiento de los centros descritos en la tabla 1.3. Estos valores son almacenados en las variables "*u*" y "*v*" respectivamente, tal como se detalla en el segmento de código 2.5.

```
%CREACIÓN DE LAS CELDAS VECINAS OBJETIVO Y SUS USUARIOS
cx(2)=0;         %Desfase del centro de la celda #2 en el eje X
cy(2)=y(1)*2;    %Desfase del centro de la celda #2 en el eje Y

t = (1/(2*N):1/N:1)*2*pi;    % Traza un polígono de N lados
u = r*sin(t)+cx(2);          % Valores de los vértices de la celda en el eje X
v = r*cos(t)+cy(2);          % Valores de los vértices de la celda en el eje Y
u(7)=u(1);                   % Unión del primer y último vértice, eje X
v(7)=v(1);                   % Unión del primer y último vértice, eje Y
```

Segmento de código 2.5. Comandos para la creación de la celda vecina #2.

2.1.3. GENERACIÓN ALEATORIA DE USUARIOS Y REPRESENTACIÓN GRÁFICA

Para la generación aleatoria de usuarios y sus coordenadas se ejecuta un lazo que con el comando "*rand*" calcula el valor en cada coordenada para cada usuario. El valor de salida del comando "*rand*" se encuentra en el rango de 0 a 1, por lo que se redimensiona el valor generado con la multiplicación de 2 veces el valor del radio de celda menos una vez el valor del radio, para que de esta forma también se puedan obtener valores negativos.

```
%GENERACION DE LOS PUNTOS(USUARIOS) ALEATORIAMENTE EN LA CELDA SOBRECARGADA
for i=1:usuarioscentral                 %Inicio del lazo, desde 1 hasta el # de
usuarios ingresado por el usuario
    puntos=0;                           %Configuración inicial parámetro puntos
    while ~puntos                       %Condición que realiza la acción mientas el
valor no sea 0
        usuariocenX(i)=rand(1)*2*r-r;   %Generación de valores aleatorios en el eje
X dentro del límite de la celda central
        usuariocenY(i)=rand(1)*2*r-r;   %Generación de valores aleatorios en el eje
Y dentro del límite de la celda central
        puntos=inpolygon(usuariocenX(i),usuariocenY(i),x,y);   %Retorno de los
valores en los ejes X y Y dentro de los límites de la celda
        numero(i)=i;                    %Almacenamiento del # de usuario creado
    end
end
```

Segmento de código 2.6. Generación aleatoria de los usuarios en la celda central.

La generación de los valores en los ejes X y Y para cada uno de los usuarios se almacena en las variables "*usuariocenX(i)*" y "*usuariocenY(i)*" respectivamente. Siendo "*i*" el número de usuario generado.

Los valores generados por el comando "*rand*" y su posterior redimensionamiento son generados para una figura cuadrangular, para obtener el retorno de valores dentro de los límites de una figura hexagonal se utiliza el comando "*inpolygon*", cuya sintaxis se describe a continuación:

Resultado de condición=inpolygon(limite en el eje X, límite en el eje Y, valor generado en el eje X, valor generado en el eje Y)

El comando "*inpolygon*" devuelve un valor de 1 si es que el valor generado en los ejes X y Y están dentro de los límites especificados; caso contrario, el comando devuelve un 0. La generación aleatoria de un valor dentro de los límites especificados se repite hasta que el comando "*inpolygon*" retorne un valor diferente de cero (variable puntos) como se especifica en el segmento de código 2.6.

Una vez obtenidos todos los valores de las coordenadas de cada uno de los usuarios almacenados en sus respectivas matrices (*usuarioscenX*, *usuarioscenY*), se grafican en cada uno de los diagramas de la interfaz como se detalla en el segmento de código 2.7.

```
%REPRESENTACIÓN GRÁFICA DE LOS USUARIOS EN LA CELDA CENTRAL
axes(handles.axes1)                         %Primera Gráfica
plot (usuariocenX,usuariocenY,'b+');        %Gráfica de los valores de la posición de
los usuarios en la celda central
axes(handles.axes2)                         %Segunda Gráfica
plot (usuariocenX,usuariocenY,'b+');        %Gráfica de los valores de la posición de
los usuarios en la celda central
```

Segmento de código 2.7. Representación gráfica de los usuarios en la celda central.

Las coordenadas de los usuarios de la celda central se almacenan en una matriz diferente, ya que estos serán utilizados posteriormente para el balanceo de cargas en caso de ser necesario. Las coordenadas de los usuarios de las celdas vecinas son almacenadas en diferentes variables. Los valores generados en el eje X y Y se almacenarán en las matrices "*d*" y "*s*" respectivamente.

Las matrices "*d*" y "*s*" serán reutilizadas en la generación aleatoria de valores de los usuarios de las celdas vecinas, ya que los usuarios generados en cada uno de ellas no realizarán *handover* o traspaso hacia otra celda por la condición inicial de que cada celda vecina dispone de recursos para el proceso de balanceo de cargas.

Para la generación de las coordenadas de los usuarios de las celdas vecinas se usará la misma redimensión del valor generado por el comando rand y, además, se adicionará el valor del centro de dicha celda, de esta forma se generarán valores dentro de los límites numéricos.

Para el retorno de un valor válido dentro de los límites de la celda se utilizará el comando "*inpolygon*", tal como se detalla en el segmento de código 2.8.

```
%GENERACIÓN Y REPRESENTACIÓN GRÁFICA DE LOS USUARIOS Y ESTACIÓN BASE
d=0;                                        %Variable que almacenará el valor de
la posición de los usarios eje X
s=0;                                        %Variable que almacenará el valor de
la posición de los usarios eje Y
for i=1:usuarios(2)                         %Inicio del lazo, desde 1 hasta el #
de usuarios ingresado por el usuario
puntos=0;                                   %Configuración inicial parámetro
puntos
while ~puntos                               %Condición que realiza la acción
mientas el valor no sea 0
d(i)=rand(1)*2*r-r+cx(2);                   %Generación de valores aleatorios en
el eje X dentro del límite de la celda #2
s(i)=rand(1)*2*r-r+cy(2);                   %Generación de valores aleatorios en
el eje Y dentro del límite de la celda #2
```

```
        puntos=inpolygon(d(i),s(i),u,v);      %Retorno de los valores en los ejes X
y Y dentro de los límites de la celda
    end
end

axes(handles.axes1)      %Primera Gráfica
plot (d,s,'k+');         %Gráfica de los valores de la posición de los usuarios en
la celda central
axes(handles.axes2)      %Segunda Gráfica
plot (d,s,'k+');         %Gráfica de los valores de la posición de los usuarios en
la celda central
```

Segmento de código 2.8. Generación aleatoria y representación gráfica de los usuarios de la celda vecina #2.

Para la representación gráfica de las coordenadas de los usuarios de las celdas vecinas se tomarán los valores de las matrices "*d*" y "*s*".

2.1.4. CONDICIONES DE ERROR

En el segmento de código 2.9, se detalla las condiciones de error, si el usuario ingresa valores mayores a los permitidos que puede soportar el sistema que está compuesta de 7 celdas con un ancho de banda de 5 MHz cada una. Esta información ha sido descrita en la sección 1.3.1.8.

Las condiciones iniciales de error serían las siguientes:

- El número de usuarios totales es mayor a 2100.
- El número de usuarios de cualquier celda vecina es mayor a 300.

En el caso de que alguna de estas condiciones no se cumpla, no se realizará el proceso de balanceo de cargas y se mostrará un mensaje de error tanto en la ventana de comandos como en el cuadro de resultados de la interfaz *GUIDE* de la simulación.

```
%CONDICIONES NECESARIAS PARA EJECUTAR EL BALANCEO DE CARGAS
%Condición que no ejecuta el balanceo de cargas si el número de usuarios totales
ingresados sobrepasa la capcidad del sistema
    if sum(usuarios)>2100
        fprintf("EL # de usuarios es mayor a la capacidad de la red \nDebe ser
menor a 700 \n"); %Mensaje de error en la ventana de comandos
        set(handles.error,'string','ERROR:  #  de  usuarios  mayor  a  2100  \n');
%Mensaje de error en la ventana de resultados del GUIDE
%Condición que no ejecuta el balanceo de cargas si el número de usuarios en la
celda #2 ingresados sobrepasa su capacidad
    elseif usuarios(2)>300
%Condición que no ejecuta el balanceo de cargas si el número de usuarios en la
celda #3 ingresados sobrepasa su capacidad
    elseif usuarios(3)>300
%Condición que no ejecuta el balanceo de cargas si el número de usuarios en la
celda #4 ingresados sobrepasa su capacidad
    elseif usuarios(4)>300
```

```
%Condición que no ejecuta el balanceo de cargas si el número de usuarios en la
celda #5 ingresados sobrepasa su capacidad
    elseif usuarios(5)>300
%Condición que no ejecuta el balanceo de cargas si el número de usuarios en la
celda #6 ingresados sobrepasa su capacidad
    elseif usuarios(6)>300
%Condición que no ejecuta el balanceo de cargas si el número de usuarios en la
celda #7 ingresados sobrepasa su capacidad
    elseif usuarios(7)>300
%Si no cumple ninguna de las anteriores condicioes se ejecuta el balanceo de cargas
    else
```

Segmento de código 2.9. Condiciones de capacidad máxima del sistema y por celda.

Para tener un escenario en el que se necesite realizar el proceso de balanceo de cargas es indispensable que el usuario ingrese un valor mayor a 300 para la celda central.

2.1.5. CÁLCULO DE PRESUPUESTO DE ENLACE

Para la determinación de los usuarios que realizarán el *handover*, se calcula el presupuesto del enlace respecto de la ubicación del usuario generado aleatoriamente con la estación base central que se encuentra en el centro de la celda central. Desde este punto se considera el inicio de proceso de balanceo de cargas. Para el posterior cálculo de rendimiento del algoritmo desarrollado se necesita un contador que mida el tiempo total del proceso, el comando "*tic*" inicia este contador y el comando "*toc*" lo detiene.

Para el cálculo del presupuesto del enlace se inicia un lazo que ejecuta esta iteración para cada uno de los usuarios de la celda central, tal como se detalla en el segmento de código 2.10.

```
%CÁLCULO DEL PRESUPUESTO DEL ENLACE DE LOS USUSARIOS DE LA CELDA
%SOBRECARGADA CON LA ESTACIÓN BASE CENTRAL
timerVal=tic;                           %Inicio de contador de tiempo, Inicio del
proceso de balanceo de cargas
i=1;
for i=1:usuarioscentral;                %Cálculo del presupuesto de enlace para
cada usuario de la celda central
    numero(i)=i;

[Prx,dis]=presupuesto(cx(1),cy(1),usuariocenX(i),usuariocenY(i),f,Ptx,Gtx,Ltx,Grx
,Lrx,Ghm);                              %Función que calcula el presupuesto de
enlace
    operaciones=operaciones+1;          %Aumento del número de operaciones para el
balanceo de carga
    Potencia(i)=Prx;                    %Almacenamiento del valor retornado del
calculo del presupuesto de enlace
    numero(i)=i;                        %Almacenamiento del # de usuario de la
celda central
end
```

Segmento de código 2.10. Comandos para llamar la función del cálculo de presupuesto de enlace.

La función "*presupuesto*" realiza el cálculo del presupuesto del enlace, en donde las variables de entrada de la función son:

- i= número de usuario del que se realiza el cálculo del presupuesto de enlace.
- cx (1): Valor de coordenada del centro de la celda central en el eje X.
- cy (1): Valor de coordenada del centro de la celda central en el eje Y.
- usuariocenx(i): Valor de coordenada de la posición del usuario i en el eje X.
- usuariocenY(i): Valor de coordenada de la posición del usuario i en el eje Y.
- f: Frecuencia de portadora.
- Ptx: Potencia de transmisión de la estación base.
- Gtx: Ganancia de transmisión de la antena de la estación base.
- Ltx: Pérdidas de transmisión en la estación base.
- Grx: Ganancia de recepción de la antena de la estación móvil (usuario).
- Lrx: Pérdidas de recepción en la estación móvil (usuario).
- Ghm: Ganancia de margen de *handover*.

Las variables de salida de la función son:

- Ptx: Potencia de recepción en la estación móvil (presupuesto del enlace).
- dis: Distancia calculada en metros desde la estación base de la celda hasta la ubicación del usuario i.

2.1.6. SCRIPT PRESUPUESTO DEL ENLACE

En el script de la función "*presupuesto*" las variables de entrada "*x*" y "*y*" corresponden a los valores de entrada del centro de la celda en los ejes X y Y respectivamente. En cambio, las variables "*a*" y "*b*" son los valores de coordenadas en el eje X y Y del usuario.

Dentro de la función se hace el cálculo de la distancia entre dos puntos con coordenadas rectangulares, estos puntos son la estación base que se encuentra dentro de la celda y la posición de la estación móvil o usuario, tal como se detalla en el segmento de código 2.11.

```
function [Prx,dis]=presupuesto(x,y,a,b,f,Ptx,Gtx,Ltx,Grx,Lrx,Ghm);
%--------------------------------------------------------------------
dis=sqrt((x-a)^2+(y-b)^2);  %Cálculo de la distancia vectorial
dis=dis*0.001;              %Conversión de metros a Kilómetros
```

Segmento de código 2.11. Comandos para el cálculo y conversión de la distancia.

Como se puede observar en el segmento de código 2.12, se realiza el cálculo de pérdidas por trayectoria en el espacio libre, está ecuación esta detallada en la ecuación 1.11.

```
%CÁLCULO DE PÉRDIDAS POR TRAYECTORIA EN EL ESPACIO LIBRE
Lfs=32.44+20*log10(f)+20*log10(dis);
```

Segmento de código 2.12. Cálculo de pérdidas por trayectoria en el espacio libre.

En el segmento de código 2.13, se calcula el presupuesto del enlace, el valor de la potencia recibida está detallada en la ecuación 1.9.

```
%POTENCIA RECIBIDA
Prx=Ptx+Gtx-Ltx-Lfs+Grx-Lrx+Ghm;
```

Segmento de código 2.13. Cálculo de potencia recibida en la estación móvil.

Además de la potencia recibida o cálculo del presupuesto del enlace se toma en cuenta el margen de desvanecimiento.

El margen de desvanecimiento toma un valor determinado por una distribución de *Rayleigh*. El cálculo del margen de desvanecimiento para la potencia recibida considerando una probabilidad de que la señal no se encuentre por debajo del umbral del receptor de 99% se describe en la sección 1.3.3.2.

```
%MARGEN DE DESVANECIMIENTO CON DISTRIBUCIÓN DE RAYLEIGH
Xmean=10^(Prx/10);              %Conversión de la Prx sin el margen de
desvanecimiento a mw
desv=Xmean^2*((4/pi)-1);        %Cálculo de la desviación estándar
prob=0.99;                      %Probabilidad de que la señal no esté por debajo
del margen de handover

x=sqrt(-log(0.99)*2*desv);      %Fórmula de la probabilidad de Rayleigh (Está
despejado x que es el margen)
Prx=10*log10(x);                %Prx ahora es el valor en dbm de la Prx + el Margen
de desvanecimiento
```

Segmento de código 2.14. Cálculo del margen de desvanecimiento y potencia recibida total.

En el segmento de código 2.14, se realiza el cálculo del margen de desvanecimiento siguiendo una distribución de *Rayleigh* para su posterior conversión de unidades. El valor de la potencia medida de dBm es retornada por la función al script principal.

2.1.7. PROCESO DE BALANCEO DE CARGAS PARA LA CELDA SOBRECARGADA

Luego de que se ha calculado el presupuesto del enlace de cada uno de los usuarios de la celda central, el algoritmo evalúa la necesidad de realizar el balanceo de cargas verificando si la celda central tiene un número de usuarios que sobrepasa la capacidad máxima de celda. Se debe considerar que hasta este punto del algoritmo se han cumplido las anteriores condiciones de capacidad del sistema.

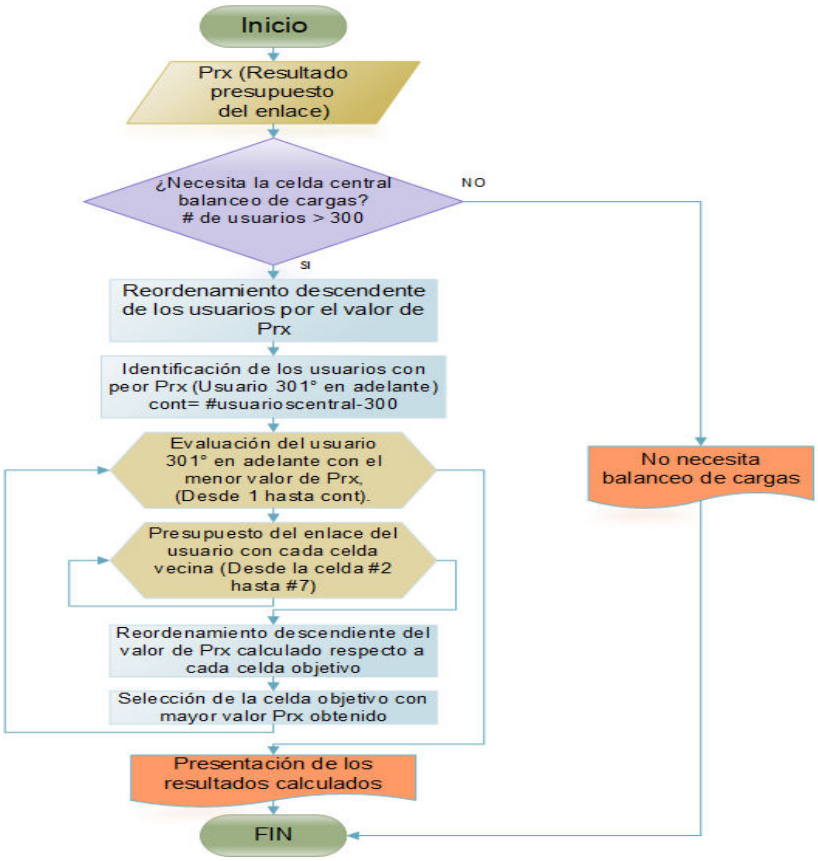


Figura 2.3. Diagrama de flujo del proceso de *handover* de la celda sobrecargada

El proceso de balanceo de cargas para la celda central sigue el procedimiento descrito en la figura 2.3, y el diagrama de flujo que describe este procedimiento se muestra en la Figura A.2 del Anexo A.

```
%ORDENAMIENTO DE LOS RESULTADOS DEL PRESUPUESTO DEL ENLACE
%Matriz que contiene los valores: # de usuario, Posiciones en los ejes X y
%Y y Cáclulo del presupuesto del enlace
datos=[numero' usuariocenX' usuariocenY' Potencia'];

[~, orden] = sort(datos(:, 4),'descend');     %Reordenamiento descendiente según
el valor del cálculo del presupuesto del enlace
format bank                                   %Formato de 2 decimales
```

```
fprintf("\n          orden        Pos X        Pos Y        Link budget \n");
resultado=datos(orden, :)                          %Matriz  que  almacena  los  datos
ordenados
operaciones=operaciones+1;                         %Aumento  del  número  de  operaciones
para el balanceo de carga
```

Segmento de código 2.15. Reorganización de los resultados obtenidos del presupuesto del enlace.

Si el número de usuarios ingresado por el usuario supera la capacidad máxima de la celda y además cumple con la capacidad máxima del sistema, el proceso de balanceo de cargas inicia con la obtención de los valores del presupuesto del enlace de los usuarios que realizarán el traspaso, además de sus coordenadas y orden según el reordenamiento ejecutado en el paso anterior. Este proceso de obtención de datos se detalla en el segmento de código 2.16.

```
%PROCESO DE HANDOVER PARA LA CELDA SOBRECARGADA
    if usuarioscentral > 300            %Condición de celda sobrecargada
        %Se obtienen los valores de # de usuario, Posiciones en los ejes X
        %y Y y Cáclulo del presupuesto del enlace de los usuarios con menor Prx
        usuariosOrden=resultado(301:end,[1]);
        usuariosHOx=resultado(301:end,[2]);
        usuariosHOy=resultado(301:end,[3]);
        POTcs=resultado(301:end,[4]);
        cont=usuarioscentral-300;       %Número de usuarios que harán el handover
```

Segmento de código 2.16. Obtención de los datos de los usuarios que realizarán el *handover*.

Si el número de usuarios en la celda central no supera la capacidad máxima de la celda, no habrá necesidad de realizar el balanceo de cargas. En el segmento de código 2.17, se indica los mensajes en la ventana de comandos y en la ventana de resultados si no se cumple con la mencionada condición.

```
 else

fprintf('\n La celda no está sobrecargada, no necesita un balanceo de cargas');
set(handles.error,'string','La celda no está sobrecargada, no necesita un balanceo
de cargas');

end
```

Segmento de código 2.17. Mensaje en el caso de que no exista sobrecarga en la celda central.

En el caso de que la condición anterior se cumpla, el algoritmo inicia el proceso de selección de la celda objetivo para cada usuario que realizará el *handover*.

El algoritmo inicia un lazo que realiza el cálculo de presupuesto del enlace del usuario con cada una de las celdas vecinas. Luego de llamar a la función "*presupuesto*" se almacena en una matriz el valor de la potencia recibida (presupuesto del enlace).

En el segmento de código 2.18, se almacena el número de la celda vecina con la que se hizo el presupuesto del enlace y aumenta el valor de un contador "*operaciones*" que se detalla en la sección 2.1.9.

```
for a=1:cont                        %Acción que se realizará para cada uno de los
usuarios que harán el handover
for b=2:7                           %Acción que se realizará el presupuesto de enlace
con cada centro de las celdas vecina #2-#7
[Prx,dis]=presupuesto(cx(b),cy(b),usuariosHOx(a),usuariosHOy(a),f,Ptx,Gtx,Ltx,Grx
,Lrx,Ghm);
c=b-1;                              %Reordenamiento del # de celda
HOpot(c)=Prx;                       %Cálculo del presupuesto del enlace recibida por
la función
celHO(c)=b;                         %# De la celda objetivo
operaciones=operaciones+1;          %Aumento del número de operaciones para el
balanceo de carga
end
```

Segmento de código 2.18. Cálculo del presupuesto del enlace respecto a las celdas vecinas.

Luego de que el algoritmo ha calculado el presupuesto del enlace del usuario que realizará el *handover* con todas las celdas vecinas, se realiza un proceso de reordenamiento de los resultados obtenidos, seleccionando la celda vecina con la mayor potencia de recepción como la celda de destino de la conexión. Este proceso se describe en el segmento de código 2.19.

```
HOpot=[celHO' HOpot'];              %Matriz que contiene el # de celda y el cálculo
de presupuesto de enlace de cada una
[~, orden] = sort(HOpot(:, 2),'descend'); %Reordenamiento descendiente según el
valor del cálculo del presupuesto del enlace
format bank                         %Formato de 2 decimales
fprintf("-------------------------------------------------------------------------
-------------");
HOresultado=HOpot(orden, :)         %Matriz que almacena los datos ordenados
%Se obtienen los valores de # de usuario, Posiciones en los ejes X y Y y Cálculo
del presupuesto del enlace del usuario con la celda objetivo
celdaObj=HOresultado(1,1);
POTco=HOresultado(1,2);
UO=usuariosOrden(a);
UX=usuariosHOx(a);
UY=usuariosHOy(a);
fprintf("El usuario # %d en la Pos en X: %.2f Pos en Y: %.2f realizara el Ho a la
celda # %d \n",UO,UX,UY,celdaObj);
usuarios(celdaObj)=usuarios(celdaObj)+1;      %Aumenta en uno el # de usuarios bajo
la cobertura de la celda objetivo luego del handover
operaciones=operaciones+1;                    %Aumento del número de operaciones
para el balanceo de carga
```

Segmento de código 2.19. Reordenamiento de los datos del presupuesto del enlace de los usuarios que realizarán el *handover*.

Como se puede observar en el segmento de código 2.20, los resultados del proceso de reordenamiento y selección de la nueva celda de servicio son presentados en la ventana de comandos y almacenados en una matriz que posteriormente será utilizada en el cálculo del perfil angular de potencia de llegada, descrita en la sección 2.2.

```
%Asignación de los valores obtenidos en una matriz para la presentación de los
datos
        usuarioshandover(a,1)=UO;
        usuarioshandover(a,2)=celdaObj;
        usuarioshandover(a,3)=dis;
        usuarioshandover(a,4)=POTco;
```

Segmento de código 2.20. Asignación de los valores obtenidos de los usuarios que realizarán el proceso de *handover*.

2.1.8. CÁLCULO DE MARGEN DE *HANDOVER* Y CONDICIÓN DE VELOCIDAD DE APRENDIZAJE

Tal como se describe en la sección 1.3.4.3.2 el valor ingresado por el usuario para la velocidad de aprendizaje "*learning*" debe ser entre 0 y 1, en el segmento de código 2.21, se detalla el mensaje de error si este valor no está dentro del rango permitido.

```
%CONDICION DE HANDOVER Y RECOMPENSA
if learning>1 || learning<=0;                  %Condición valor de velocidad de
aprendizaje
   fprintf('\n ERROR: EL valor de velocidad de aprendizaje tiene que estar entre 0
y 1');
   set(handles.ResultadoRecompensa,'string','ERROR: EL valor de velocidad de
aprendizaje tiene que estar entre 0 y 1');
```

Segmento de código 2.21. Condición del valor de la velocidad de aprendizaje.

Si se cumple la condición del valor de la velocidad de aprendizaje, el algoritmo procede a calcular el margen de *handover*. El valor de margen de *handover* es la diferencia de la potencia recibida en la estación móvil o presupuesto del enlace respecto a la celda de servicio y la celda objetivo que fue seleccionada anteriormente.

Las variables "*filapar*" y "*filaimpar*" se utilizan para la presentación de los resultados calculados en la tabla de recompensas. La variable "*filapar*" corresponde a la variación de margen de *handover,* mientas que la variable "*filaimprar*" corresponde al valor de recompensa del sistema obtenido. Ambas variables son declaradas en el segmento de código 2.22.

```
else
        filapar=2;                          %Fila que mostrará el valor de offset de HO
        filaimpar=1;
        %CÁLCULO MARGEN DE HANDOVER
        Mho=POTco-POTcs;                    %Margen de Handover sin offset
        if pasos>0 && pasos <=0.5           %Condición que valida pasos de handover
entre valores mayores 0 y 0.5
        z=1;                                %Número de paso de variación de handover
        offset=0;                           %Variable que llevará la suma de los pasos
(variaciones) ejecutadas
```

Segmento de código 2.22. Cálculo del margen de *handover* y condición del valor de paso de variación de margen de *handover*.

Luego de calcular el valor de margen de *handover*, el algoritmo valida el valor de paso de variación de margen de *handover*. Para que el algoritmo no tome valores de variación de *handover* muy altos, la variación de margen de *handover* debe estar en un rango mayor a 0 dB y menor o igual a 0,5 dB, dicha condición está descrita en el segmento de código 2.22.

Si la condición del valor de paso de variación de margen de *handover* no se cumple, se muestra un mensaje de error en la ventana de resultados del proceso de aprendizaje por refuerzo, tal como se describe en el segmento de código 2.23.

```
else
set(handles.ResultadoRecompensa,'string','ERROR: Variación del MHo no está dentro
del rango permitido (0 - 0.5 dB)');
end
```

Segmento de código 2.23. Mensaje de error del valor de paso de variación de margen de *handover* cuando esté fuera de rango.

2.1.8.1. Proceso de *machine learning* – aprendizaje por refuerzo

Una vez que el algoritmo ha validado las condiciones descritas en las secciones anteriores, inicia el proceso de aprendizaje por refuerzo. El procedimiento para el aprendizaje por refuerzo y la validación de las condiciones iniciales se describen en la figura 2.4, y el diagrama de flujo de la función que realiza estas tareas se muestra en la Figura A.3 del Anexo A.

Se recomienda un valor ideal de variación de margen de *handover* lo más pequeño posible, ya que de esta forma el sistema no necesitará mayor cambio de potencia para tener un proceso de balanceo de cargas exitoso.

Según el 3GPP, el margen de *handover* puede tomar valores que están entre 0 y 15 dB, modificándose con un paso de hasta 0,5 dB [37]. Para el presente trabajo de titulación el

valor de margen de *handover* está limitado a 12 dB[10], tal como de detalla en el segmento de código 2.24.

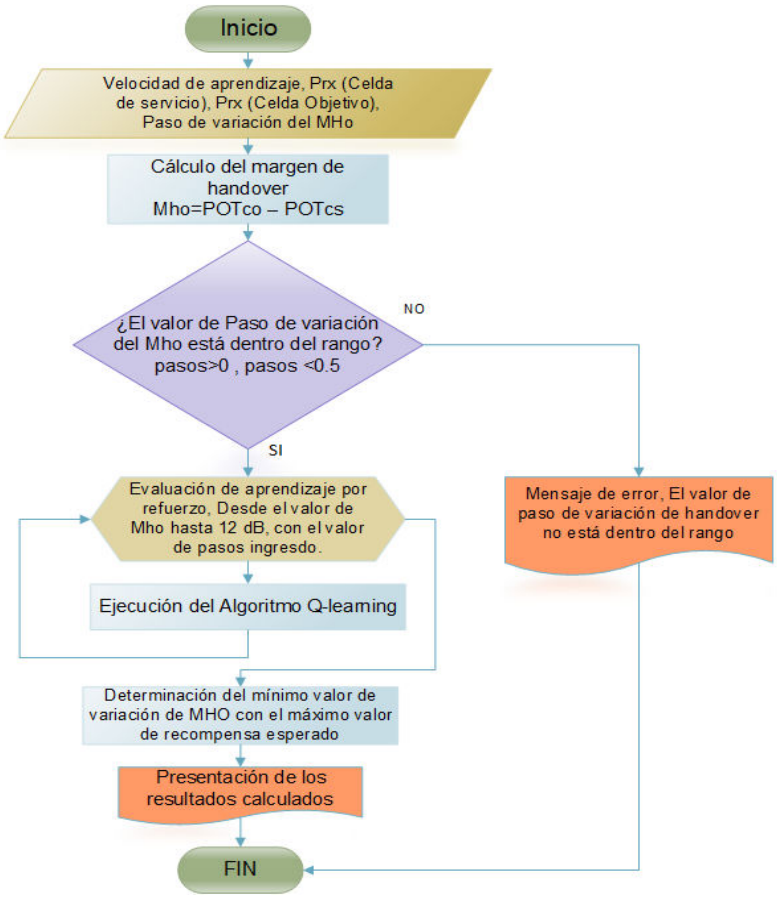


Figura 2.4. Diagrama de flujo del proceso basado en *Machine learning* – Aprendizaje por refuerzo.

La variable “cant” es el valor de margen de *handover* mas el valor de paso ingresado por el usuario. Tal como se puede observar en el segmento de código 2.24, la adición del valor del paso se realiza hasta alcanzar el valor máximo de 12 dB.

```
%PROCESO DE APRENDIZAJE
%Condición que realizará el proceso de aprendizaje hasta que llegue al valor máximo
de 12 dBm
for cant=Mho:pasos:12
    offset=offset+pasos;                %Suma del valor del offset actual más el
valor de paso
    recompensa(filaimpar,z)=offset;     %Asignación del valor de la variación en dB
```

Segmento de código 2.24. Comandos para la iteración del proceso de aprendizaje.

[10] La variación del margen de *handover* (MHo) para el presente proyecto de titulación se ha limitado entre 0 y 12 dB; debido a que para valores mayores de variación de MHo se empieza a percibir una gran degradación de las prestaciones de la red celular LTE; tal como elevadas tasas de caída de llamadas y desconexiones entre los usuarios y las estaciones base [37].

La variable "*offset*" es la suma del valor de variación actual más el valor de paso ingresado por el usuario para la siguiente interacción. Esta adición se realizará hasta que el valor de margen de *handover* alcance el valor máximo de 12 dB. Se debe tomar en cuenta que el valor de la variable offset puede ser mayor al valor de margen de *handover* permitido, debido a que el valor de margen de *handover* en las primeras interacciones tiene un valor negativo, se necesita una mayor variación de margen de *handover* para llegar al valor máximo de 12 dB de margen de *handover*. El cálculo del margen de *handover* está descrito en la ecuación 2.1.

$$M_{HO(CS)} = P_{RX(CO)} - P_{RX(CS)} \quad (2.1)$$

En donde:

$P_{RX(CS)}$ = Potencia recibida en la estación móvil respecto a la celda sobrecargada [dBm].

$P_{RX(CO)}$ = Potencia recibida en la estación móvil respecto a la celda objetivo o vecina [dBm].

M_{HO} = Margen de *handover* [dB].

2.1.8.2. Cálculo de recompensas a partir del algoritmo *Q-learning*

El proceso de aprendizaje por refuerzo se realiza a partir del algoritmo *Q-learning*. Este algoritmo retorna una recompensa según la acción que el agente tome sobre el sistema. El procedimiento se detalla en la figura 2.5, y el diagrama de flujo que describe el funcionamiento del algoritmo se muestra en la Figura A.4 del Anexo A.

Para aplicar el algoritmo *Q-learning* se necesitan identificar los principales elementos del aprendizaje por refuerzo. El agente que interactuará con el sistema será la estación base de la celda central, puesto que será el elemento que indique el valor que deberán modificar las estaciones base vecinas (positivamente) como ella misma (negativamente), tal como se describe en la sección 1.3.3.4.

El estado actual de la red estará indicado por los valores de potencia calculados por la estación móvil respecto a la celda de servicio y la celda objetivo seleccionada. La acción que tomará el agente estará dada por el cumplimiento de la condición de *handover* exitoso, esta acción será el incremento de variación de margen de *handover* con el que se validará la condición de *handover* exitoso en cada interacción del agente con el sistema.

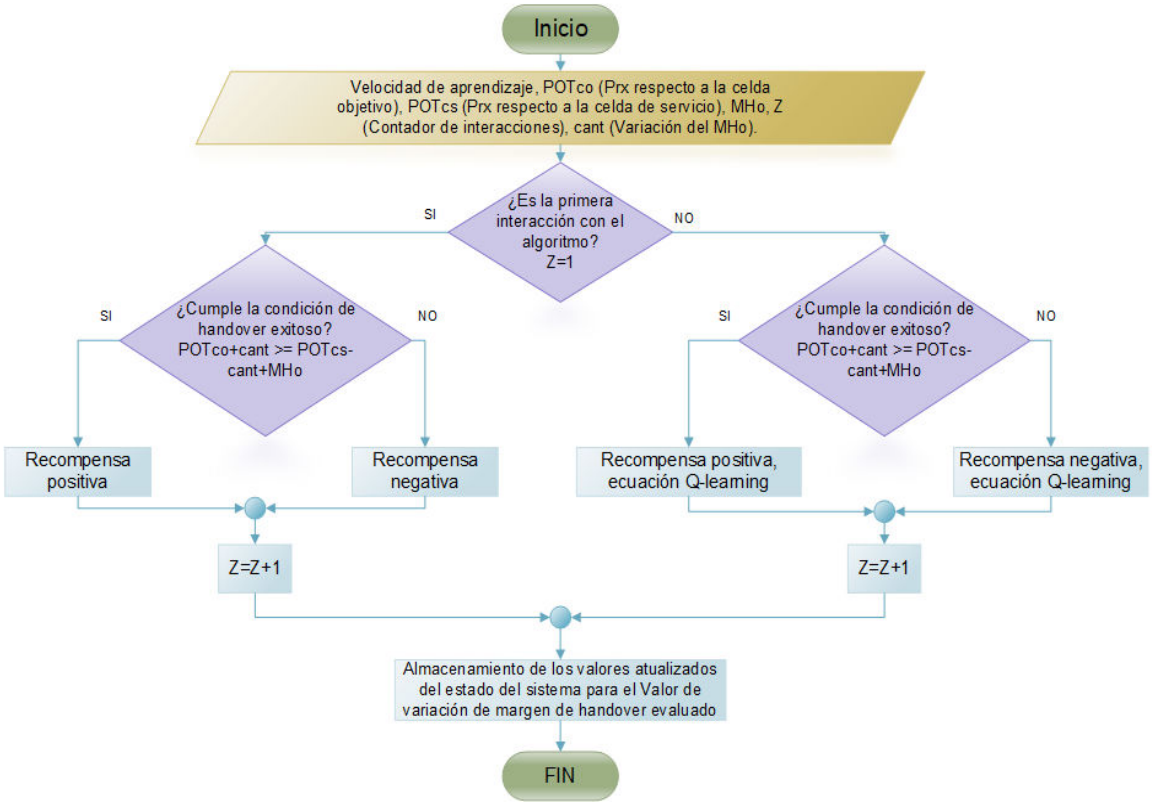


Figura 2.5. Diagrama de flujo del proceso de ejecución del algoritmo *Q-learning*

Los valores de recompensas para el aprendizaje por refuerzo tendrán los siguientes valores: Para una recompensa positiva se dará un valor de 10 y para una recompensa negativa el valor dado será -10. Estos valores serán asignados por el sistema según se cumpla o no la condición de *handover* exitoso descrita en la ecuación 2.2.

En la primera interacción del agente con el sistema aplicando un algoritmo basado en aprendizaje por refuerzo no existen estados ni acciones anteriores de la que pueda aprender el agente. En esta interacción se validará la condición de *handover* exitoso y se dará una recompensa positiva o negativa según sea el resultado que retorne la condición, tal como se describe en el segmento de código 2.25, en esta interacción no se toma en cuenta el valor de aprendizaje por refuerzo.

```
if z==1                          %Los siguientes pasos solo realizan cuando es el
primer paso de aprendizaje del agente
      if POTco+cant>= POTcs-cant+Mho;                    %Condición de handover
Exitoso
      recompensa(filapar,z)=recompensa(filapar,z)+10;      %Recompensa positiva
      else
      recompensa(filapar,z)=recompensa(filapar,z)-10; %    Recompensa negativa
      end
```

Segmento de código 2.25. Comandos para la primera interacción con el sistema.

Desde la segunda interacción, el agente calcula el estado del sistema a partir de las recompensas obtenidas aplicando el algoritmo de *Q-learning* descrito en la ecuación 1.21.

En la descripción de la ecuación del algoritmo de *Q-learning* existe la inclusión de dos parámetros claves, la velocidad de aprendizaje que es un valor de entrada ingresado por el usuario y el factor de descuento, que tal como se explica en la sección 1.3.4.3.1 tendrá un valor de 0.

Como se puede observar en el segmento de código 2.26, se realiza la evaluación de la condición de *handover* exitoso. Esta condición para este caso está descrita en la fórmula 1.17.

En el segmento de código 2.26, se valida la condición descrita en la ecuación 1.17 y además se ejecuta el algoritmo *Q-learning* tomando en cuenta el valor del estado de la red calculado en la interacción anterior y el retorno de su recompensa.

```
else              %los siguientes pasos se realizarán desde la segunda interacción
    if POTco+cant>= POTcs-cant+Mho;           %Condición de handover Exitoso
                    recompensa(filapar,z)=(1-learning)*recompensa(filapar,z-
1)+learning*10;     %Ecuación Q-learning (Recompensa Positiva)
                    recompensa(filapar,z)=round(recompensa(filapar,z),4);
%Redondeo de cuatro decimales del valor del estado de aprendizaje
    else
                    recompensa(filapar,z)=(1-learning)*recompensa(filapar,z-
1)+learning*(-10);  %Ecuación Q-learning (Recompensa Negativa)
                    recompensa(filapar,z)=round(recompensa(filapar,z),4);
%Redondeo de cuatro decimales del valor del estado de aprendizaje
    end
```

Segmento de código 2.26. Proceso de aprendizaje mediante el algoritmo *Q-learning*.

En el segmento de código 2.27, se hace el aumento del contador de interacciones "*z*"[11] y del contador de operaciones para el balanceo de carga, que está descrito en la sección 2.1.9.

```
z=z+1;                          %Aumento en uno del # de paso, para la siguiente
interacción
operaciones=operaciones+1;      %Aumento del número de operaciones para el
balanceo de carga
```

Segmento de código 2.27. Comandos para el aumento del contador de interacciones y operaciones.

[11] El contador de interacciones "z" indica el número de veces que se adiciona el valor de paso de margen de *handover* al valor de margen de *handover* hasta que este alcance el valor máximo de 12 dB. Por lo que, el número de interacciones estaría condicionado por el valor del paso de margen de *handover* y las potencias calculadas en el presupuesto del enlace.

Una vez recibidas todas las recompensas para cada interacción hasta llegar al valor de 12 dB de variación de margen de *handover*, se hace la presentación de la tabla de recompensas. En el segmento de código 2.28, se procede a buscar la mínima variación de margen de *handover* que ha llegado al valor de recompensa máximo, que se ha configurado inicialmente en 10.

```
operaciones=operaciones+1;              %Aumento del número de operaciones para el
balanceo de carga
valorRecompensa=recompensa(2,:);        %Extracción de los valores de recompensa
obtenidos
%Búsqueda del valor de variación de handover que alcanzó la recompensa máxima para
la presentación de resultados
posicionRecompensa=find(valorRecompensa==10,1);
varMHO=recompensa(1,posicionRecompensa);
```

Segmento de código 2.28. Búsqueda de variación de margen de *handover* más óptimo.

En el segmento de código 2.29, se presentan los resultados de variación de margen de *handover* y, además, se genera el mensaje de error cuando el algoritmo no ha podido llegar a la máxima recompensa, incluyendo adicionalmente la presentación de una recomendación al usuario de la herramienta de que aumente el valor de velocidad de aprendizaje.

```
if isempty(posicionRecompensa)
       set(handles.ResultadoRecompensa,'string','No se ha alcanzado la recompensa
máxima, aumente la velocidad de aprendizaje!');
       else
       recompensaResultados=sprintf('El valor de variacion de handover con el que
se tiene la maxima recompensa es %.2f dB \n',varMHO);
        set(handles.ResultadoRecompensa,'string',recompensaResultados);
        end
```

Segmento de código 2.29. Presentación de resultados de la tabla de recompensas.

La variación de margen de *handover* es el valor con el que todas las conexiones cumplirán con la condición de *handover* exitoso. Por lo que el algoritmo obtendría un 100% de rendimiento, si este valor se añade o disminuye en las estaciones base como se explicó anteriormente.

2.1.9. CONTADOR DE OPERACIONES PARA EL BALANCEO DE CARGAS

La variable "*operaciones*" es un contador que determina el número de operaciones claves[12] que posteriormente será utilizada para medir el rendimiento del algoritmo.

[12] Operación clave hace mención a los procesos que influyen directamente en el proceso de balanceo de cargas, sin contar acciones que realiza el algoritmo tales como: gráficas del estado de la red, creación de la red celular LTE, presentación de los resultados en la ventana de comandos, etc.

Los procesos que aumentan el valor del contador son las siguientes:

- Cada vez que el algoritmo realiza algún cálculo del presupuesto del enlace.
- Luego de reordenar los resultados obtenidos del presupuesto del enlace.
- Luego de cada interacción del agente usando el algoritmo de aprendizaje por refuerzo.
- Luego de cada aumento de variación de margen de *handover*.
- Después de cada validación de la condición de *handover* exitoso.
- Después de cada validación de la condición de sobrecarga de la capacidad de una celda.

2.1.10. CONDICIÓN DE SENSIBILIDAD Y REPRESENTACIÓN GRÁFICA

El valor de sensibilidad mínima es un dato de entrada ingresado por el usuario de la herramienta de simulación. Este parámetro y sus rangos de valores de potencia están descritos en la sección 1.3.3.3. En general, se recomienda la evaluación de la condición de sensibilidad en un caso de pésimo nivel de potencia de recepción.

La validación de la condición de sensibilidad y sus procesos subsecuentes están descritos en la figura 2.6; además, el diagrama de flujo que detalla este proceso se muestra en la Figura A.5 del Anexo A.

La validación de la condición de sensibilidad mínima se realizará para cada usuario que será traspasado. Si en el proceso de balanceo de carga existe alguna celda que deba recibir un usuario que supera la capacidad máxima de la celda, se considerará ese *handover* como fallido, por lo que no se validará la condición de sensibilidad mínima. Caso contrario, la condición de sensibilidad mínima será evaluada. Esta condición está descrita en la ecuación 2.4.

$$Sensibilidad \leq P_{RX(CO)} \tag{2.4}$$

En donde:

$Sensibilidad$ = Sensibilidad mínima del receptor [dBm].

$P_{RX(CO)}$ = Potencia recibida en la estación móvil respecto a la celda objetivo o vecina [dBm].

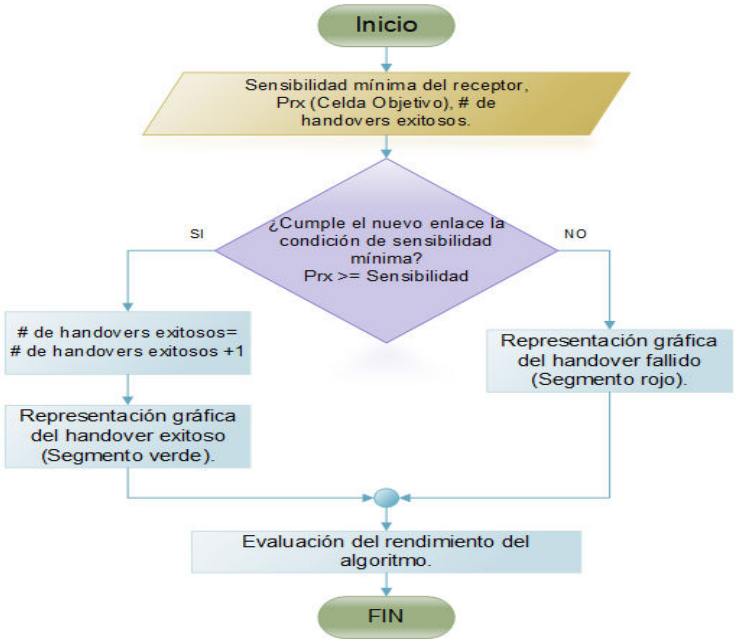


Figura 2.6. Diagrama de flujo de la condición para *handover* exitoso.

Si la condición de sensibilidad mínima se cumple, el *handover* se considerará como exitoso, y se representará el éxito de este traspaso con un segmento color de verde que unirá la posición del usuario que será traspasado con la celda que fue seleccionada anteriormente como destino. Caso contrario, el color del segmento será rojo, representando un *handover* fallido. Tal como se describe en el segmento de código 2.30.

```
%CONDICIÓN DE SOBRECARGA DE CELDAS VECINAS
        %Condición que se ejecuta cuando no se sobrepasó la capacidad máxima de la
        celda vecina luego del balanceo de cargas
        if usuarios(celdaObj)<=300
        %CONDICIÓN DE SENSIBILIDAD
           if POTco >= Sen
               %# de handovers exitosos aumenta en uno si cumple la condición de
sensibilidad
               exito=exito+1;
               fprintf('Handover Exitoso \n');
               operaciones=operaciones+1;       %Aumento del número de operaciones
para el balanceo de carga
               %Representación gráfica de un segmento VERDE que une la posición
del usuario
               %con centro de la celda objetivo (BS) indicando el éxito del
               %handover en la segunda gráfica
               axes(handles.axes2)
               hold on
               P1=[UX UY];P2=[cx(celdaObj) cy(celdaObj)];
               plot([P1(1) P2(1)],[P1(2) P2(2)],'g')
           else  %Pasos que se ejecutarán cuando no cumple la condición de
sensibilidad
               fprintf('Handover no Exitoso, no cumple con la sensibilidad minima
\n');
               operaciones=operaciones+1;       %Aumento del número de operaciones
para el balanceo de carga
               %Representación gráfica de un segmento ROJO que une la posición del
usuario
```

```
            %con centro de la celda objetivo (BS) indicando el fallido
            %proceso de handover en la segunda gráfica
            axes(handles.axes2)
            P1=[UX UY];P2=[cx(celdaObj) cy(celdaObj)];
            plot([P1(1) P2(1)],[P1(2) P2(2)],'r')
        end
```

Segmento de código 2.30. Comandos para la presentación de resultados y representación gráfica de los *handover*s exitosos y fallidos.

En el segmento de código 2.31, se describe el mensaje de error cuando el proceso de balanceo de carga sobrepasa la capacidad de la celda vecina.

```
%Pasos que se ejecutarán cuando luego del balanceo de cargas una celda sobrepaso
su capacidad
        else
            fprintf('Handover no Exitoso, sobrepasa la cantidad maxima de la celda
(300 usuarios) \n');
            fallos=fallos+1;
            operaciones=operaciones+1;  %Aumento del número de operaciones para el
balanceo de carga
        end
        HOpot=0;                        %Reset de los valores calculados
        end
```

Segmento de código 2.31. Mensaje de error cuando se sobrepasa la capacidad de la celda vecina luego del *handover*.

2.1.10.1. Contador de *handovers* exitosos

La variable "*exito*" es el contador que determina el número de *handovers* exitosos. Este valor aumenta cada vez que se cumple la condición de sensibilidad mínima del receptor con cada usuario que realizará el traspaso de su enlace.

2.1.11. CÁLCULO DE PARÁMETROS DE RENDIMIENTO DEL ALGORITMO

Una vez presentados y representados gráficamente todos los resultados obtenidos, el algoritmo evalúa el rendimiento del proceso ejecutado.

La variable "*tiempo*" tiene el valor de tiempo total en segundos del proceso de balanceo de cargas. Este contador que fue iniciado con el comando *tic*, se detiene con el comando *toc*.

El rendimiento del proceso de balanceo de cargas es medido mediante dos parámetros. El porcentaje de *handovers* exitosos y el promedio de *handovers* por unidad móvil por segundo.

El porcentaje de *handovers* exitosos es calculado con la ecuación 2.5.

$$porcentaje = \frac{exito}{cont*100} \quad (2.5)$$

En donde:

$porcentaje$ = Porcentaje de *handovers* exitosos.

$exito$ = Número total de *handovers* exitosos.

$cont$ = Número total de usuarios que realizan el *handover*.

El promedio de *handovers* por unidad móvil por segundo se calcula con la ecuación 1.6.

Ambas medidas de rendimiento del algoritmo se describen en el segmento de código 2.32.

```
tiempo=toc(timerVal);            %Finalización de contador de tiempo, tiempo total
del proceso de balanceo de cargas
porcentaje=exito/cont*100;       %Porcentaje de exitoso del balanceo de carga
HOprom=exito/(cont*tiempo);      %Promedio de handovers exitoso por segundo por
usuario
```

Segmento de código 2.32. Comandos para el cálculo del rendimiento del sistema

2.1.12. CREACIÓN DEL ARCHIVO DE TEXTO CON RESULTADOS

El programa principal permite la generación de un archivo de texto que almacena todos los resultados calculados que son presentados en la ventana de comandos.

El comando "*diary*" ejecuta un proceso que activa y desactiva el inicio de sesión. Cuando el inicio de sesión está activado "*diary on*" MATLAB captura los comandos ingresados, la entrada del teclado y la salida de texto desde la ventana de comandos [38]. Este comando guarda la salida de la ventana de comandos en la carpeta actual como un archivo de texto codificado en UTF-8 llamado "*resultados.txt*", tal como se describe en el segmento de código 2.33.

```
function pushbutton1_Callback(hObject, eventdata, handles)
clc;
diary resultados.txt %Creación de un archivo de texto que almacena los resultados
de la ventana de comandos
diary on
```

Segmento de código 2.33. Creación del archivo de texto con los resultados de la ventana de comandos.

El comando "*diary off*" detiene la captura de la salida de la ventana de comandos.

```
diary off
```

Segmento de código 2.34. Comando que detiene la captura de los resultados de la ventana de comandos.

2.1.13. PRESENTACIÓN DE RESULTADOS Y GRÁFICAS

El segmento de código 2.35, presenta tanto en la ventana de comandos como en la ventana de errores de la interfaz los resultados calculados del rendimiento del algoritmo. Estos parámetros son el porcentaje de *handovers* exitosos, el promedio de *handovers* por unidad móvil por segundo, el número de operaciones realizadas en el proceso de balanceo de carga y el valor mínimo de variación de margen de *handover* que ha alcanzado la máxima recompensa.

```
%RESULTADOS
show=sprintf('Los Handovers exitosos para los %d usuarios de sobrecarga son %d
(%.2f por ciento de exito)\n',cont,exito,porcentaje);
fprintf("\n------------------------------------------------------------------------
---------------");
fprintf("\n--------------------------------RESULTADOS-----------------------------
--------\n");
fprintf('\n Los Handovers exitosos para los %d usuarios de sobrecarga son %d (%.2f
por ciento de exito)\n',cont,exito,porcentaje);
set(handles.error,'string',show);
fprintf('\n El Promedio de handovers por unidad movil por segundo es %.2f
\n',HOprom);
fprintf('\n La velocidad de procesamiento para los %d usuarios fue de %.2f segundos
por %d operaciones \n',cont,tiempo,operaciones);
fprintf('\n El valor de variacion de handover con el que se tiene la maxima
recompensa es %.2f \n \n',varMHO);
```

Segmento de código 2.35. Presentación de resultados en la ventana de la interfaz y ventana de comandos.

Con el segmento de código 2.36, se presenta la tabla de recompensas en la ventana de resultados del aprendizaje por refuerzo, indicando el valor de recompensa obtenida para cada valor de variación de margen de *handover* con el que fue evaluado el sistema.

```
set(handles.uitable1,'data',recompensa);
set(handles.pushbutton1,'BackgroundColor','r');
set(handles.reset,'BackgroundColor',[0, 0.45, 0.74]);
```

Segmento de código 2.36. Presentación de la tabla de recompensas.

La condición de cumplimiento de capacidad de las celdas vecinas es validada por el algoritmo, el segmento de código 2.37, muestra el mensaje de error si no se cumple dicha

condición cuando el usuario ingresó un valor mayor al permitido o cuando luego del proceso de balanceo de cargas se sobrepasó la capacidad máxima de la celda.

```
%CONDICION DE CUMPLIMIENTO DE CAPACIDAD DE LAS CELDAS VECINAS
for i=2:7 %Iteración que se realiza solo para las celdas vecinas
    if usuarios(i)>300
        fprintf("El balanceo de cargas excede la capacidad de la celda #%d con %d
usuarios \n",i,usuarios(i))
        set(handles.error,'string','El balanceo de cargas excede la capacidad de
una o varias celdas, verifique los resultados')
```

Segmento de código 2.37. Mensajes de error cuando se sobrepasa la capacidad de una celda luego del balanceo de cargas.

En el caso de que se ingrese un número mayor de usuarios a lo permitido por el sistema, el algoritmo no realizará el proceso de balanceo de cargas; en cambio, si luego del proceso de balanceo de cargas, una o varias de las celdas se saturaron, el algoritmo no contará ese *handover* como exitoso, y no se adicionará al número de usuarios activos en la celda. El mensaje de error de este caso se encuentra descrito en la sección de código 2.31.

2.1.14. PRESENTACIÓN DE RESULTADOS EN LA VENTANA DE COMANDOS

Tal como se indica en la figura 2.2, la interfaz del programa de la primera simulación muestra el *push button* "*Resultados*".

Este botón abre una interfaz que presenta la salida de la ventana de comandos que en esta simulación está completamente conformada por resultados del proceso de balanceo de cargas realizado anteriormente.

Todos los resultados que son presentados están almacenados en el archivo de texto creado "*resultados.txt*", la creación de este archivo de texto está descrito en la sección 2.1.12.

Para poder observar los resultados obtenidos del proceso de balanceo de cargas se debe oprimir el *push button* "*Mostrar Resultados*", tal como se puede observar en la figura 2.7.

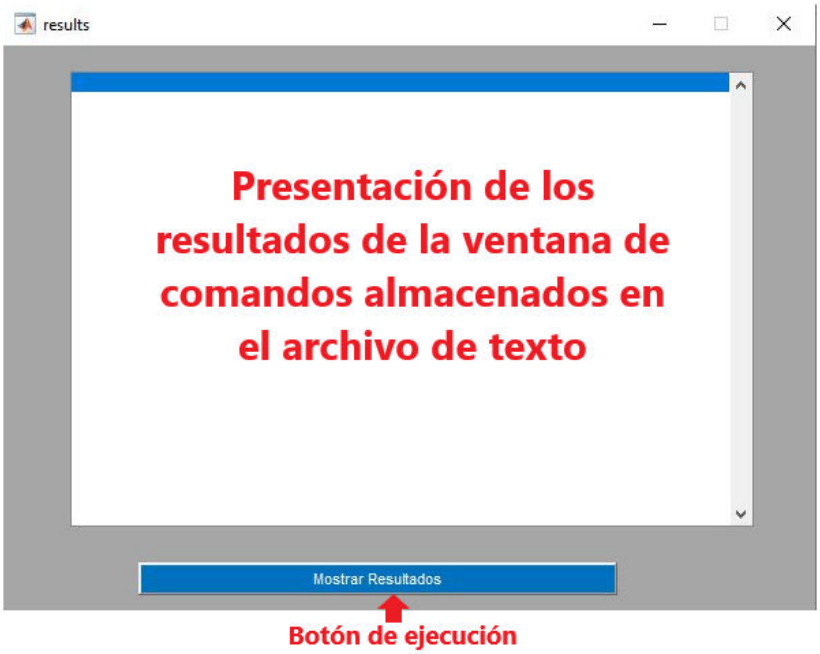


Figura 2.7. Interfaz del programa de presentación de resultados

En el segmento de código 2.38, se muestra el comando que abre el programa "*resultados*". Esta función se encuentra dentro de la sintaxis del *push button* "*resultados*" de la primera simulación.

```
% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
results();
```

Segmento de código 2.38. Comando que abre el programa de resultados.

En el código principal del programa "*resultados*" se realizan tres acciones: la apertura y lectura del archivo de texto, la conversión de caracteres del archivo de texto y la presentación de los resultados en la ventana de texto.

```
function pushbutton1_Callback(hObject, eventdata, handles)
%Apertura del archivo de texto con los resultados del balanceo de carga
    fid = fopen('resultados.txt', 'r');
    c = fread(fid, inf, 'uint8=>char')';     %Conversión de caracteres
    set(handles.texto,'String',c)            %Presentación de resultados
```

Segmento de código 2.39. Lectura, conversión y presentación del archivo de texto con los resultados.

2.1.15. CONFIGURACIÓN DE VARIABLES GLOBALES Y *RESET* DE RESULTADOS

Tal como se indica en la figura 2.2, la interfaz del programa de la primera simulación muestra el *push button* "*Reset*".

Este botón realiza el *reset* de las gráficas y los resultados presentados en la simulación; además, hace la limpieza de las variables del programa principal como de la variable global.

Se recomienda oprimir este botón antes de empezar la simulación debido a que las variables globales se guardan en la memoria de MATLAB incluso si el programa fue cerrado.

En el segmento de código 2.40, se puede observar la configuración de la variable global "*usuarioshandover*" que contiene la información del orden de los usuarios que realizan el *handover*, la distancia respecto a la celda objetivo seleccionada, la potencia recibida calculada en la función del presupuesto del enlace y el número de la celda objetivo.

```
%Configuración de las variables globales para la segunda simulación
setappdata(0,'usuarioshandover',usuarioshandover);  %Configuración de variable
global
```

Segmento de código 2.40. Configuración de la variable global utilizada para la segunda simulación.

El comando que realiza la configuración de la variable global es "*setappdata*" cuya sintaxis es la siguiente:

setappdata(obj,name,val)

El comando "*setappdata*" almacena el contenido de la variable "*val*". El número de variable global almacenada es "*obj*", y el identificador de la variable "name", identifica de forma única los datos para su posterior recuperación. Esta variable global se utilizará para el cálculo del perfil angular de potencia de llegada, descrito en la sección 2.2.1.

El segmento de código 2.41, muestra los comandos para el reinicio general de la simulación: *reset* de las gráficas del estado de la red antes y después del balanceo de cargas, el *reset* de la matriz "*recompensas*" que contiene el valor de variación de margen de *handover*, el *reset* de la tabla de recompensas, el *reset* de las ventanas de resultados y errores, el cambio de color de los botones de "*ejecución*" y "*reset*" y, además, el *reset* del archivo de texto creado con los resultados obtenidos en la ventana de comandos.

```
function reset_Callback(hObject, eventdata, handles)
%Reset de gráficas, varibales y parámetros generales
clc;
cla(handles.axes1,'reset');                    %reset primera gráfica
cla(handles.axes2,'reset');                    %reset segunda gráfica
recompensa=0;                                  %reset matriz de recompensa
set(handles.uitable1,'data',recompensa);       %reset Tabla de recompensa
set(handles.pushbutton1,'BackgroundColor',[0, 0.45, 0.74]);
set(handles.reset,'BackgroundColor','r');
set(handles.error, 'String', '');
set(handles.ResultadoRecompensa, 'String', '');
clear all;                                     %clear de variables
fopen('resultados.txt','w');                   %Reset archivo de texto
```

Segmento de código 2.41. Reset de las gráficas, tabla de recompensa y cambio de color de botones de inicio y reset.

2.2. PROGRAMA PARA EL CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL

La segunda parte de la simulación de este proyecto de titulación calcula el perfil angular de potencia de llegada en la estación móvil para entornos con línea de vista (LoS) y sin línea de vista (NLoS) entre la estación móvil y la estación base. En la Figura A.6 y A.7 del Anexo A se muestran los diagramas de flujo de las funciones que describen el cálculo del perfil angular para ambos entornos respectivamente.

La estación móvil será uno de los usuarios que realizaron el proceso de *handover* y la estación base corresponderá al de la celda seleccionada como objetivo luego del proceso de balanceo de cargas de la primera simulación para el usuario seleccionado.

Esta simulación se ejecutará luego del proceso de balanceo de cargas de la primera simulación; la variable global "*usuarioshandover*" contiene la información necesaria para ejecutar esta simulación, permitiendo al usuario seleccionar el número de orden de usuario con el que se calculará el perfil angular de potencia de llegada.

2.2.1. DESCRIPCIÓN DE LA INTERFAZ GRÁFICA Y OBTENCIÓN DE DATOS INICIALES Y/O GLOBALES

La interfaz gráfica de la segunda parte de la simulación está compuesta de varias entradas de datos ingresadas por el usuario que se utilizarán para la configuración de los parámetros iniciales. La interfaz permitirá al usuario seleccionar la posición de la estación base

respecto a la estación móvil para la simulación en un entorno NLoS tomando en cuenta las especificaciones establecidas en la recomendación de la UIT.

El programa además presentará una tabla con la información de los usuarios que realizaron el proceso de *handover* en la primera simulación y, como resultados se obtendrán representaciones gráficas del sistema para los diferentes entornos descritos en la recomendación UIT-R P.1816-4, Anexo 3.

La interfaz de esta simulación está descrita en la figura 2.8.

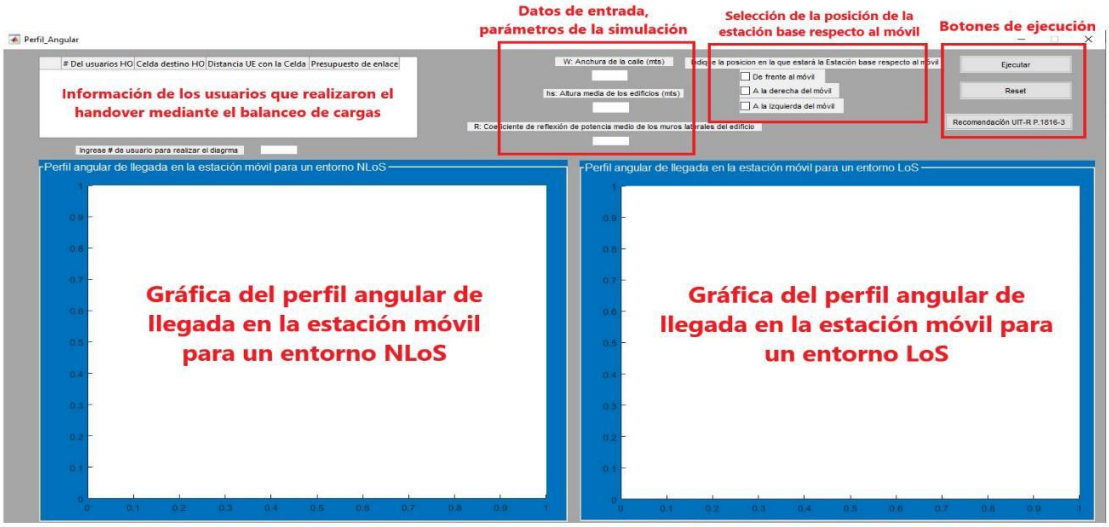


Figura 2.8. Interfaz del programa de simulación del perfil angular de llegada en la estación móvil.

En el segmento de código 2.42, se presenta el comando que abre la interfaz de la segunda simulación cuando se oprime el botón “Diagrama de perfil angular Recomendación UIT-R P.1816-4” de la primera simulación.

```
% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
Perfil_Angular();
```

Segmento de código 2.42. Comando que abre el programa de simulación del perfil angular de llegada en la estación móvil.

Dentro del código principal de la segunda simulación se realiza la obtención de la información de los usuarios que realizaron el proceso de *handover* anteriormente. Esta información está almacenada en la variable global “*usuarioshandover*”.

```
function ejecutar_Callback(hObject, eventdata, handles)
fprintf ('        Orden      Celda HO     Distancia (km)      Presupuesto de enlace
\n')
```

```
usuarioshandover=getappdata(0,'usuarioshandover')
```

Segmento de código 2.43. Lectura de los valores de la variable global.

Para realizar la obtención de datos de la variable global se utiliza el comando "*getappdata*" cuya sintaxis es la siguiente:

val = getappdata(obj,name)

getappdata devuelve el valor almacenado de la variable global etiquetada con el número "*obj*". El identificador de nombre, "*name"*, identifica de forma única el valor que se va a recuperar de los datos y val es el nombre de la variable que almacenará los datos obtenidos para su uso en el programa actual.

En el segmento de código 2.44, se realiza la obtención y conversión numérica para las variables de entrada:

- Altura media de los edificios (m).
- Coeficiente de reflexión medio de los muros laterales del edificio.
- Anchura de la calle (m).
- Número de usuario del que se obtendrá la información para la simulación.

```
%Obtención de los valores ingresados por el usuario
hs=str2num(get(handles.hs,'string'));               %Altura media de los
edificios (metros)
R=str2num(get(handles.R,'string'));                 %Coeficiente de reflexión de
potencia medio de los muros laterales del edificio
W=str2num(get(handles.W,'string'));                 %Anchura de la calle (metros)
usuario=str2num(get(handles.usuario,'string'));     %# de usuario con el que se
realizará la simulación
pocisionUsuario=find(usuarioshandover==usuario,1);  %Obtención de la posición del
usuario
```

Segmento de código 2.44. Comandos para la obtención de los valores ingresados por el usuario y lectura del usuario seleccionado.

2.2.2. ALGORITMO DEL CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL PARA UN ENTORNO NLOS

Una vez obtenidos los valores ingresados por el usuario se realiza el cálculo del perfil angular de potencia de llegada en la estación móvil para un entorno NLoS.

Este proceso está descrito en la sección 1.3.5.2. y el algoritmo sigue el procedimiento descrito en la figura 2.9.

El cálculo del perfil angular para el entorno NLoS necesita el valor de potencia recibida por el usuario respecto a la estación base de la celda seleccionada. El segmento de código 2.45, describe el proceso de obtención del valor de potencia y la conversión de unidades de potencia a mW.

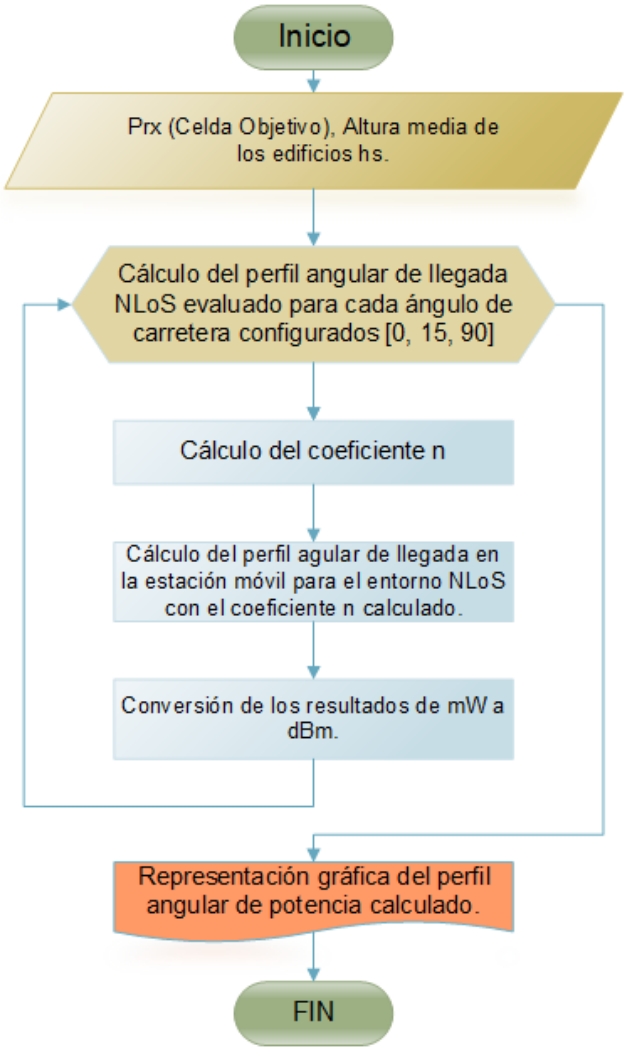


Figura 2.9. Diagrama de flujo del cálculo del perfil angular de potencia de llegada para una estación móvil en un entorno NLoS.

```
%Perfil angular de llegada en la estación móvil para un entorno NLoS
Prx=usuarioshandover(pocisionUsuario,4);      %Obtención del valor del presupuesto
del enlace
PrxdBm=Prx;
Prx=10^(Prx/10);                              %Conversión de dBm a mW
```

Segmento de código 2.45. Lectura de potencia del usuario seleccionado y conversión a mW.

En el segmento de código 2.46, se realiza la configuración del ángulo de inclinación de la carretera, que siguiendo la recomendación UIT-R P.1816-4, Anexo 3, toma valores de 0, 15 y 90 grados. Además, se establece el rango del ángulo de llegada que varía entre -180 y 180 grados. Una vez establecidos los ángulos de evaluación se realiza el cálculo del coeficiente n.

```
for carretera=[0 15 90]                 %Ángulo de evaluación

llegada=[-180:1:180];                   %Rango de ángulo de llegada

n=min(1,((2.6/(hs^0.5)*(1-exp(-0.03*carretera))+0.05)^1.5));    %Coeficiente n
```

Segmento de código 2.46. Cálculo del coeficiente n respecto al ángulo de la carretera y el ángulo de llegada.

A partir del cálculo del coeficiente n, se evalúa el perfil angular de potencia de llegada en la estación móvil para un entorno NLoS mediante la ecuación 1.24. Además, se realiza la multiplicación de la potencia con el perfil angular NLoS calculado y la conversión de unidades de potencia a dBm para su representación gráfica, tal como se puede observar en el segmento de código 2.47.

```
%Ecuación perfil angular de potencia de llegada en la estación móvil NLoS

AOAnlos=1./sqrt((cos(llegada.*(pi/180)).^2)+(sin(llegada.*(pi/180)).^2)./n^2);
AOAnlos=Prx*AOAnlos;        %Multiplicación del perfil con la PRx por la estación
móvil
AOAnlos=10*log10(AOAnlos);  %Conversión de mW a dBm
```

Segmento de código 2.47. Evaluación del perfil angular de potencia de llegada en la estación móvil NLoS.

2.2.2.1. Representación gráfica del perfil angular de potencia de llegada en la estación móvil para un entorno NLoS.

Para la representación gráfica del perfil angular de potencia NLoS se utiliza el valor del perfil angular de potencia en dBm y el valor del rango del ángulo de llegada, tal como se puede observar en el segmento de código 2.48.

```
%Representación gráfica del perfil angular de llegada entorno NLoS
    axes(handles.axes1)
    plot(llegada,AOAnlos)
    hold on
    grid on
    title ('AOAnlos')
    ylim([-inf -100])
```

```
    ylabel('potencia recibida (dBm)');
    xlabel('Angulo de llegada (grados)');
    if carretera == 0
        cero=['Ángulo de 0° entre la dirección de la BS y el UE, potencia mín=
',num2str(min(AOAnlos)),' dBm'];
    elseif carretera ==15
        quince=['Ángulo de 15° entre la dirección de la BS y el UE, potencia mín=
',num2str(min(AOAnlos)),' dBm'];
    else
        noventa=['Ángulo de 90° entre la dirección de la BS y el UE, potencia mín=
',num2str(min(AOAnlos)),' dBm'];
    legend(cero,quince,noventa)
    end
```

Segmento de código 2.48. Representación gráfica del perfil angular de llegada en el entorno NLoS.

2.2.3. ALGORITMO DEL CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL PARA UN ENTORNO LOS

Luego del cálculo del perfil angular de potencia de llegada para un entorno NLoS y con los valores ingresados por el usuario se realiza el cálculo del perfil angular de potencia de llegada en la estación móvil para un entorno LoS.

Este proceso está descrito en la sección 1.3.5.3. y el algoritmo sigue el procedimiento descrito en la figura 2.10.

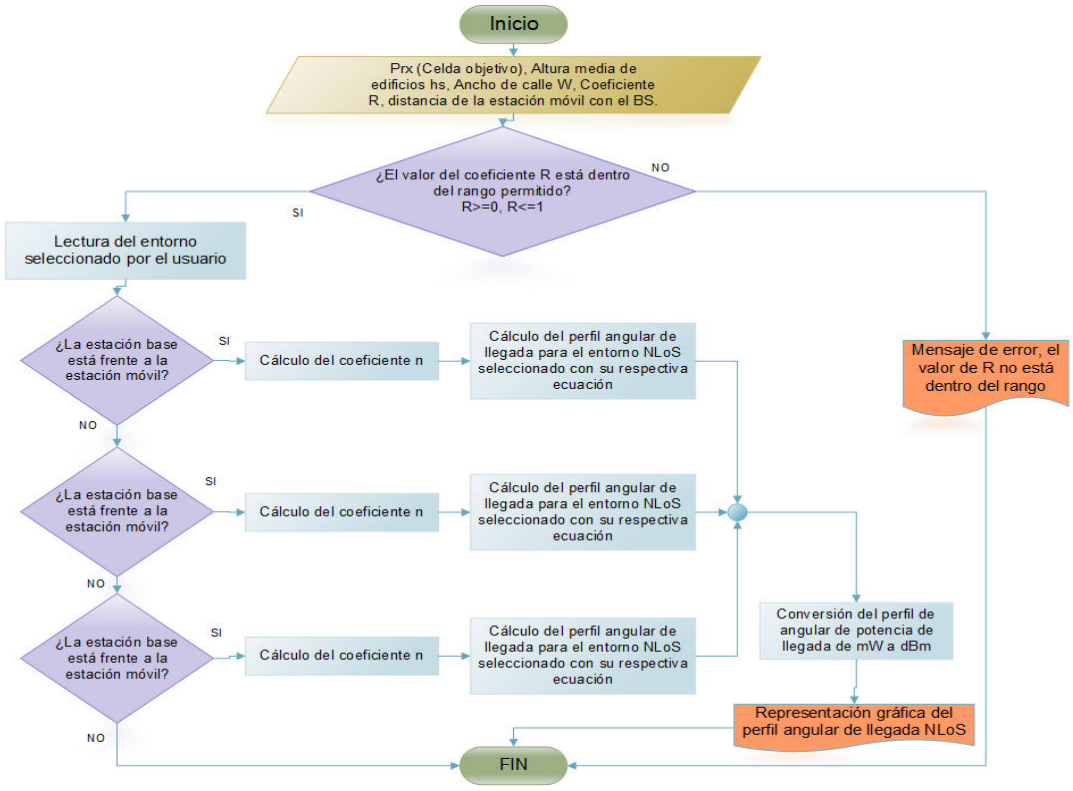


Figura 2.10. Diagrama de flujo del cálculo del perfil angular de potencia de llegada en la estación móvil para un entorno LoS.

La interfaz de la simulación permite al usuario escoger la posición de la estación base respecto a la estación móvil. La lectura del caso seleccionado se realiza mediante el comando "*get*", tal como se puede observar en el segmento de código 2.49.

```
% --- Executes on button press in frente.
function frente_Callback(hObject, eventdata, handles)
%Lectura de selección de opción
frente=get(hObject,'Value');

% --- Executes on button press in derecha.
function derecha_Callback(hObject, eventdata, handles)
%Lectura de selección de opción
derecha=get(hObject,'Value');

% --- Executes on button press in izquierda.
function izquierda_Callback(hObject, eventdata, handles)
%Lectura de selección de opción
izquierda=get(hObject,'Value');
```

Segmento de código 2.49. Lectura del entorno LoS seleccionado por el usuario.

2.2.3.1. Condición de valor del coeficiente R

Para el cálculo del perfil angular de potencia de llegada de un entorno LoS, el valor del coeficiente de reflexión medio "*R*" de los muros laterales del edificio que es ingresado por el usuario tiene que estar dentro del rango de 0 a 1. El segmento de código 2.50 valida el valor del coeficiente R y presenta el mensaje de error en la ventana de resultados en el caso de no cumplir con la condición.

```
%%Perfil angular de llegada en la estación móvil para un entorno LoS
%Condición del valor del coeficiente R
if R>=1 || R<=0
    fprintf('ERROR: EL valor del coeficiente R tiene que estar entre 0 y 1');
```

Segmento de código 2.50. Mensaje de error cuando el valor del coeficiente R está fuera de rango.

2.2.3.2. Simulación del perfil angular de llegada en un entorno LoS cuando la estación base está a la izquierda respecto a la estación móvil

En el segmento de código 2.51, se determina que la estación base se encuentra a la izquierda respecto de la estación móvil.

```
elseif handles.izquierda.Value==1; %Lectura del entorno seleccionado
       i=1;
```

Segmento de código 2.51. Comando de evaluación del entorno (izquierda) respecto al ángulo de llegada.

La evaluación del perfil angular LoS cuando la estación base se encuentra a la izquierda respecto a la estación móvil utiliza dos diferentes ecuaciones dependiendo del valor del ángulo de llegada, tal como se describe en la sección 1.3.5.3.1.

En el segmento de código 2.52, se realiza el cálculo del perfil angular de potencia de llegada para un entorno LoS cuando el valor del ángulo de llegada es mayor o igual a 0.

```
for llegada=-180:0.001:180; %Evaluación respecto al rango del ángulo de llegada
%Ecuación respecto al valor del ángulo de llegada
       if llegada>=0
       n=min(1,((2.6/(hs^0.5)*(1-exp(-0.03*carretera))+0.05)^1.5));
       %Coeficiente n
       %Ecuación perfil angular de potencia de llegada en la estación móvil NLoS
       AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
       %Ecuación perfil angular de potencia de llegada en la estación móvil LoS
       AOAlos(i)=(R^((1000*d*abs(llegada)*pi/(180*W))-1))+Prx*AOAnlos;
```

Segmento de código 2.52. Comandos para el cálculo del perfil angular de llegada LoS (izquierda) para ángulos mayores o iguales a 0.

En el segmento de código 2.53, se realiza el cálculo del perfil angular de potencia de llegada para un entorno LoS cuando el valor del ángulo de llegada es menor a 0.

```
else
       n=min(1,((2.6/(hs^0.5)*(1-exp(-0.03*carretera))+0.05)^1.5));
       %Coeficiente n
       %Ecuación perfil angular de potencia de llegada en la estación móvil NLoS
       AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
       %Ecuación perfil angular de potencia de llegada en la estación móvil LoS
       AOAlos(i)=(R^(1000*d*abs(llegada)*pi/(180*W)))+Prx*AOAnlos;
       end
i=i+1;
end
```

Segmento de código 2.53. Comandos para el cálculo del perfil angular de llegada LoS (izquierda) para ángulos menores a 0.

2.2.3.3. Simulación del perfil angular de llegada en un entorno LoS cuando la estación base está a la derecha respecto a la estación móvil

En el código principal de la simulación se realiza la lectura del entorno seleccionado por el usuario, es decir se identifica la posición existente de la estación base respecto a la estación móvil.

En el segmento de código 2.54, se determina que la estación base se encuentra a la derecha respecto de la estación móvil.

```
if handles.derecha.Value==1;     %Lectura del entorno seleccionado
        i=1;
```

Segmento de código 2.54. Comando de evaluación del entorno (derecha) para el cálculo del ángulo de llegada.

El cálculo del perfil angular de potencia de llegada LoS se determina a partir del valor del coeficiente n y del perfil angular de potencia de llegada NLoS.

En la evaluación del perfil angular LoS, cuando la estación base se encuentra a la derecha respecto a la estación móvil, se utilizan dos ecuaciones diferentes dependiendo del valor del ángulo de llegada, tal como se describe en la sección 1.3.5.3.2.

En el segmento de código 2.55, se calcula el perfil angular de potencia de llegada para un entorno LoS cuando el valor del ángulo de llegada es mayor o igual a 0.

```
for llegada=-180:0.001:180;     %Evaluación respecto al rango del ángulo de llegada
%Ecuación respecto al valor del ángulo de llegada
       if llegada>=0
       n=min(1,((2.6/(hs^0.5)*(1-exp(-0.03*carretera))+0.05)^1.5));
       %Coeficiente n
       %Ecuación perfil angular de potencia de llegada en la estación móvil NLoS
       AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
       %Ecuación perfil angular de potencia de llegada en la estación móvil LoS
       AOAlos(i)=(R^(1000*d*abs(llegada)*pi/(180*W)))+Prx*AOAnlos;
```

Segmento de código 2.55. Comandos para el cálculo del perfil angular de llegada LoS (derecha) para ángulos mayores o iguales a 0.

En el segmento de código 2.56, se realiza el cálculo del perfil angular de potencia de llegada para un entorno LoS cuando el valor del ángulo de llegada es menor a 0.

```
else
n=min(1,((2.6/(hs^0.5)*(1-exp(-0.03*carretera))+0.05)^1.5));     %Coeficiente n
%Ecuación perfil angular de potencia de llegada en la estación móvil NLoS
AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
%Ecuación perfil angular de potencia de llegada en la estación móvil LoS
AOAlos(i)=(R^((1000*d*abs(llegada)*pi/(180*W))-1))+Prx*AOAnlos;
        end
        i=i+1;
        end
```

Segmento de código 2.56. Comandos para el cálculo del perfil angular de llegada LoS (derecha) para ángulos menores a 0.

2.2.3.4. Simulación del perfil angular de llegada en un entorno LoS cuando la estación base está de frente a la estación móvil.

En el segmento de código 2.57 se reconoce si la estación base se encuentra de frente respecto de la estación móvil.

```
    elseif handles.frente.Value==1; %Lectura del entorno seleccionado
        i=1;
```

Segmento de código 2.57. Comando de evaluación del entorno (frente) respecto al ángulo de llegada.

La evaluación del perfil angular LoS cuando la estación base se encuentra de frente respecto a la estación móvil se describe en la sección 1.3.5.3.3.

En el segmento de código 2.58, se realiza el cálculo del perfil angular de potencia de llegada para un entorno LoS.

```
for llegada=-180:0.001:180; %Evaluación respecto al rango del ángulo de llegada
       n=min(1,((2.6/(hs^0.5)*(1-exp(-0.03*carretera))+0.05)^1.5));
       %Coeficiente n
       %Ecuación perfil angular de potencia de llegada en la estación móvil NLoS
       AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
       %Ecuación perfil angular de potencia de llegada en la estación móvil LoS
       AOAlos(i)=(R^(1000*d*abs(llegada)*pi/(180*W)))+Prx*AOAnlos;
i=i+1;
end
```

Segmento de código 2.58. Comandos para el cálculo del perfil angular de llegada LoS (de frente) para el valor de ángulo de llegada.

2.2.3.5. Representación gráfica del perfil angular de potencia de llegada en la estación móvil para el entorno LoS seleccionado.

Luego del cálculo del perfil angular de potencia de llegada en la estación móvil para el entorno LoS seleccionado se realiza la conversión de unidades de potencia a dBm y además, se determina el rango de ángulo de llegada en donde el perfil angular calculado supera el nivel de potencia recibida con la que el usuario realizó el proceso de *handover* en el programa de balanceo de cargas.

```
llegada=[-180:0.001:180];
AOAlos=10*log10(AOAlos);    %Conversión mW a dBm
posAOAlos=find(AOAlos>PrxdBm);
```

Segmento de código 2.59. Conversión de los valores del perfil calculado a dBm.

Para la representación gráfica del perfil angular de potencia LoS calculado se utiliza el valor del perfil angular de potencia en dBm y el valor del rango de ángulo de llegada, además, se muestra en la leyenda del programa el nivel de potencia mínima de la señal para el entorno seleccionado y el rango de ángulo de llegada en donde se superó el nivel de potencia del usuario que realizó el proceso de *handover*, tal como se puede observar en el segmento de código 2.60.

```
    %Representación gráfica del perfil angular de llegada entorno LoS
    axes(handles.axes3)
    hold on
    plot(llegada,AOAlos)
    hold on
    grid on
    title ('AOAlos')
    ylabel('potencia recibida (dBm)');
    xlabel('Angulo de llegada (grados)');
    legend('Estación base frente al móvil');
    frente=['Perfil angular LoS, potencia min= ',num2str(min(AOAlos)),' dBm,
rango de ángulos no válidos: ('...
        ,num2str((max(posAOAlos)-1)*0.001-180),'° hasta
',num2str((min(posAOAlos)-1)*0.001-180),'°)'];
    legend(frente)
    end
```

Segmento de código 2.60. Representación gráfica del perfil angular de llegada para el entorno LoS seleccionado.

2.2.4. *RESET* DE GRÁFICAS Y APERTURA DE LA RECOMENDACIÓN UIT-R P.1816-4

Tal como se indica en la figura 2.8, la interfaz del programa de la segunda simulación muestra el *push button* "*Reset*". Este botón realiza el *reset* de las gráficas de acuerdo al código 2.61.

```
function reset_Callback(hObject, eventdata, handles)
%Reset de gráficas
clc;
cla(handles.axes1,'reset');
cla(handles.axes3,'reset');
```

Segmento de código 2.61. Comandos para el reset de gráficas

Además, como se indica en la figura 2.8, la interfaz del programa de la segunda simulación muestra el *push button* "Recomendación UIT-R P.1816-4".

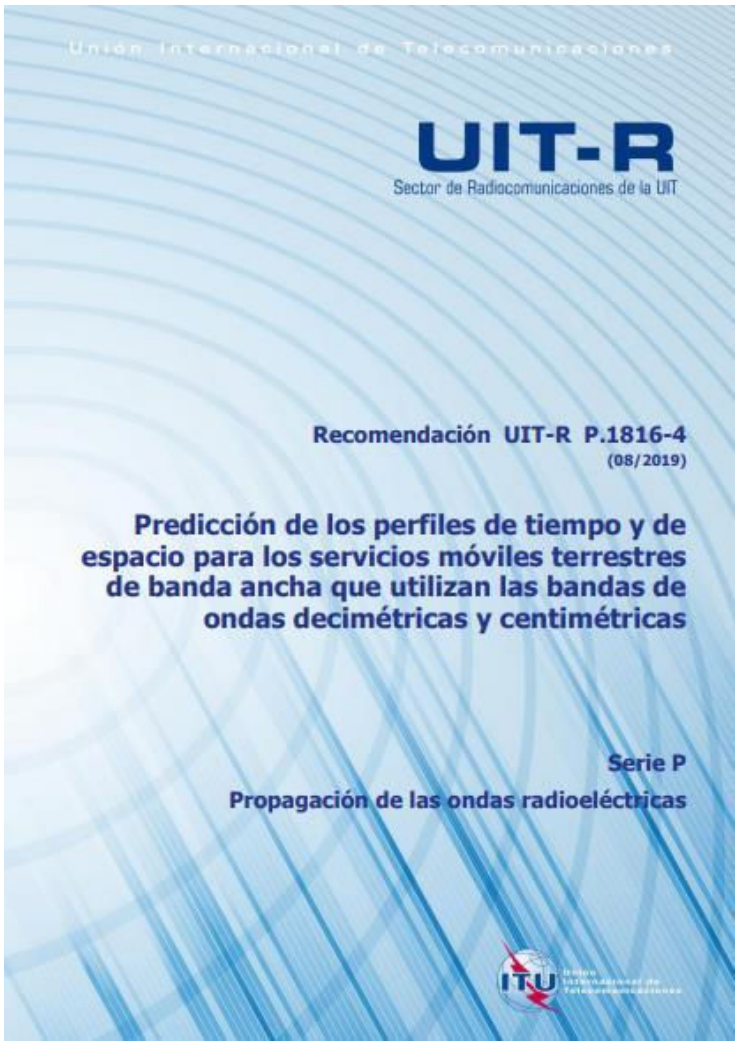


Figura 2.11. Recomendación UIT-R P.1816-4

Este botón realiza la apertura del documento que contiene el anexo 3 de la recomendación UIT-R P.1816-4 en base a lo que se muestra en el segmento de código 2.61.

```
function recomendacion_Callback(hObject, eventdata, handles)
%Este push button despliega la Recomendación UIT-R P.1816-3
winopen('R-REC-P.1816-4.pdf')
```

Segmento de código 2.61. Comandos para apertura de la recomendación UIT-R P.1816-4.

3. RESULTADOS Y DISCUSIÓN

En este capítulo se presentan los resultados obtenidos de las simulaciones del programa de balanceo de cargas mediante el uso de *machine learning* y el programa del cálculo del perfil angular de potencia de llegada en la estación móvil para entornos NLoS y LoS.

Los resultados de la primera simulación que corresponde al balanceo de cargas de una celda LTE de ancho de banda de 5 MHz, se muestran a partir de gráficas que detallan el estado de la red antes y después del balanceo de cargas, además, se presenta la tabla de recompensas para el valor de margen de variación de *handover* con el que fue evaluada la red según los conceptos descritos anteriormente del aprendizaje por refuerzo.

El rendimiento del algoritmo y los cálculos del presupuesto del enlace de cada usuario que realiza el traspaso se presentan tanto en la ventana de comandos como en la ventana de resultados en la interfaz del programa.

La segunda simulación presenta representaciones gráficas del perfil angular de potencia de llegada calculado para la estación móvil del usuario que ha realizado el proceso de *handover*.

3.1. PROGRAMA DE BALANCEO DE CARGA ANTE EL PROBLEMA DE SOBRECARGA EN UNA CELDA LTE

La simulación del programa de balanceo de cargas basado en un algoritmo de aprendizaje por refuerzo incluye en su interfaz varias variables de entrada; cada una de estas puede cambiar los resultados obtenidos y las representaciones gráficas del sistema.

Los parámetros clave del sistema son:

- La frecuencia de portadora LTE.
- La potencia de transmisión de la estación base.
- El valor de la sensibilidad mínima del receptor.
- El valor de paso de variación de *handover*.
- La velocidad de aprendizaje.
- El número de usuarios en la celda central.
- El radio de cobertura de las celdas.

En las secciones subsecuentes se presentan los resultados de la simulación con diferentes frecuencias de portadora que se encuentran en las bandas de frecuencia para LTE, los valores de frecuencia de portadora seleccionados son: 850 MHz, 1900 MHz, 2100 MHz.

Dentro de cada una de las secciones de las simulaciones a diferentes frecuencias, se establecen varios valores de potencia de transmisión de la estación base. En sistemas LTE se puede tener un valor máximo de 20 W o 43 dBm de potencia de transmisión para un canal de ancho de banda de 5 MHz, tal como se describe en el *release* 9 del 3GPP [18]. Los valores de potencia con los que se realizan las simulaciones son: 0 dBm y 20 dBm. Cabe considerar que, debido al modelo de espacio libre, las pérdidas son mucho menores a otros modelos de propagación; por lo que, para los valores de potencia de 20 dBm, se alcanzaron varios kilómetros de cobertura por la estación base.

El valor de sensibilidad mínima de receptor es una variable de entrada que es fundamental para el cálculo de *handovers* exitosos luego de realizar el proceso de balanceo de cargas. Los resultados de las simulaciones son evaluados para una sensibilidad de -106 dBm, que corresponde a un valor de potencia de señal recibida “mala”, tal como se describe en la tabla 1.1.

El radio de la celda LTE es uno de los parámetros que depende del valor de pérdidas en el espacio libre. Este parámetro de cobertura depende de las ganancias y pérdidas establecidas como valores de entrada por el usuario; para cada una de las simulaciones a diferentes valores de potencia de entrada y frecuencia de portadora LTE, se le sugerirá al usuario un valor de radio en metros, en el cual se empezará a experimentar un porcentaje de *handovers* exitosos y un porcentaje de *handovers* fallidos para cada escenario simulado.

El número de usuarios para cada una de las celdas es un valor de entrada ingresado por el usuario; en las siguientes simulaciones se establecieron 3 escenarios con diferente nivel de sobrecarga en la celda central, con el fin de poder observar los diferentes porcentajes de *handovers* exitosos, el tiempo de simulación y las demás métricas de evaluación de rendimiento del algoritmo descritas en la sección 1.3.1.6. Para el análisis los escenarios seleccionados son los siguientes, sin embargo, se debe indicar que el usuario de la herramienta de simulación puede ingresar valores diferentes:

- 200 usuarios de sobrecarga para la celda central (500 usuarios en total en la celda central).
- 20 usuarios de sobrecarga para la celda central (320 usuarios en total en la celda central).
- 2100 usuarios en total en la celda central (En este escenario la capacidad máxima de todo el sistema se encuentra en la celda central).

En el caso del escenario con la máxima capacidad del sistema, hay que tener en cuenta que, debido a la distribución aleatoria de los usuarios, el proceso de balanceo de cargas

puede sobrecargar una o varias de las celdas vecinas; cada uno de los traspasos o *handovers* que excedan la capacidad máxima de la celda será considerado como fallido. Por lo que, se añadirá una métrica que es el número de *handovers* fallidos provocados por la sobrecarga de las celdas vecinas.

Para los valores de velocidad de aprendizaje y paso de variación de margen de *handover* se utilizarán diferentes valores dentro de los rangos establecidos anteriormente[13] [14]. Cabe recalcar que para valores de velocidad de aprendizaje menores a 0,5, no se ha podido obtener el valor de recompensa máximo del sistema, las simulaciones y resultados con dichos valores no serán analizados o presentados en las secciones subsecuentes.

Los valores de los demás parámetros son establecidos respecto a las características técnicas de una antena omnidireccional para la estación base (transmisor) y la antena de un dispositivo móvil (receptor).

Cada uno de estos escenarios serán simulados 10 veces[15] y el promedio de estos resultados serán presentados y analizados en las secciones subsecuentes, los resultados tabulados de cada simulación se encuentran en el Anexo C.

Los resultados de las representaciones gráficas de los dos estados de la red[16] se visualizan en la interfaz del programa, tal como se puede observar en la figura 3.1.

Cabe recalcar que el algoritmo evalúa el sistema hasta que el margen de *handover* sea 12 dB[17], por lo que el tiempo transcurrido del proceso de balanceo de cargas descrito en las tablas de resultados, representará la simulación y evaluación de todo este proceso.

Antes del análisis de los resultados es necesario la descripción del concepto de “diferencia de variación de margen de *handover*” que es el resultado de la resta entre la variación de margen de *handover* con la que se alcanzó la máxima recompensa evaluado con el menor (0,5) y el mayor (0,99) valor de velocidad de aprendizaje. Este parámetro es importante ya que cuantifica la proximidad de resultados entre los diferentes pasos de variación de margen de *handover*, de esta forma el usuario puede determinar el paso adecuado para cada escenario.

---

[13] Rango de valores de la Velocidad de aprendizaje se ha establecido entre 0 y 1.
[14] Rango de valores de paso de variación de margen de *Handover* se ha establecido entre 0,1 y 0,5 dB.
[15] El número de simulaciones para cada escenario se ha establecido respecto a las características técnicas del computador con el que se ha realizado las simulaciones (Procesador Intel Core i7-7500U, sistema operativo de 64 bits, RAM 8 GB). Con un número mayor de simulaciones se empieza a experimentar caída de rendimiento del ordenador.
[16] La red celular antes y después del proceso de balanceo de cargas.
[17] El valor de margen *handover* máximo se establece según lo descrito en la sección 2.1.8.1.

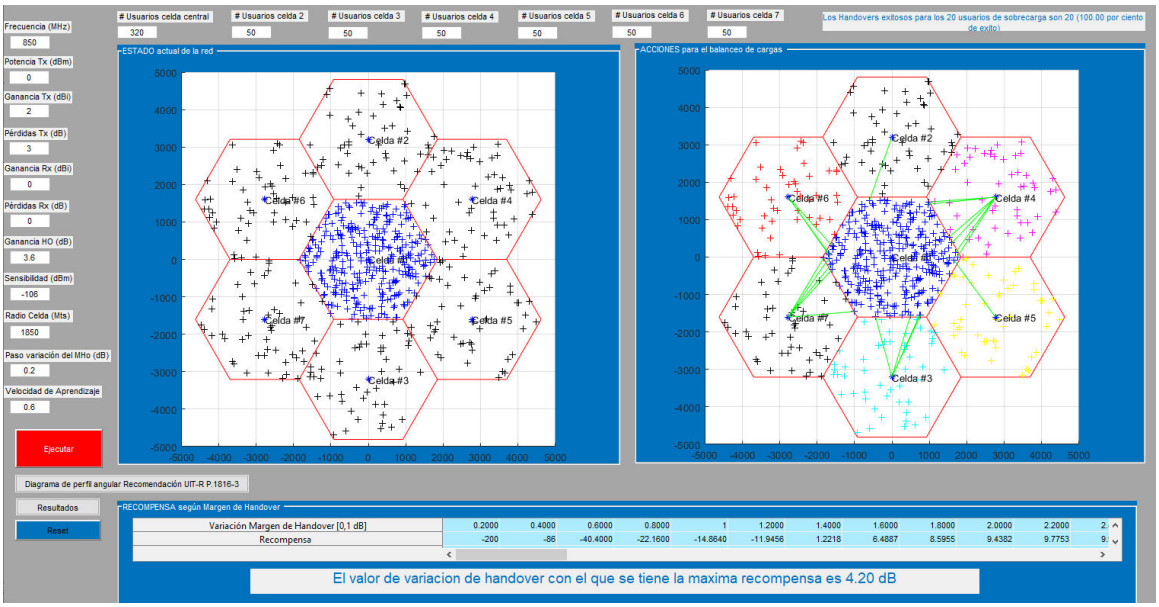


Figura 3.1. Resultados obtenidos en la interfaz gráfica del programa de balanceo de cargas.

La ejecución del proceso de balanceo de cargas que se realiza mediante *machine learning* permitirá a la red determinar por sí misma la variación de margen de *handover* óptimo, sin que el usuario ingrese cualquier valor que pueda generar errores en el proceso de balanceo de cargas o sobrecargas en las celdas vecinas, garantizando que la zona de cobertura de la celda central disminuirá lo necesario para que las demás celdas vecinas puedan brindar la zona de cobertura necesaria a los usuarios que realizarán el traspaso de la conexión.

3.1.1. ESCENARIO Nº1: FRECUENCIA 850 MHZ, RADIO 1850 M, POTENCIA DE TRANSMISIÓN 0 DBM.

Tal como se indicó en la sección anterior, la simulación del programa de balanceo de cargas se evaluará con diferente cantidad de usuarios de sobrecarga en la celda central; cada uno de los resultados de estos escenarios serán presentados y analizados para cada valor de velocidad de aprendizaje y para cada valor de paso de variación de margen de *handover*.

Los parámetros de simulación establecidos para una potencia de transmisión de la estación base de 0 dBm a una frecuencia de 850 MHz se describen en la tabla 3.1:

Tabla 3.1. Parámetros de simulación, Escenario N°1 [18] [39] [40].

Parámetros	**Valor**
Frecuencia de portadora	850 MHz
Potencia de Transmisión (Estación Base)	0 dBm
Ganancia de Transmisión (Estación Base)	2 dBi
Pérdidas de Transmisión (Estación Base)	3 dB
Ganancia de Recepción (Estación Móvil)	0 dBi
Pérdidas de Recepción (Estación Móvil)	0 dB
Ganancia de handover	3.6 dB

Cada uno de los valores establecidos para cada escenario son tomados según el *release* 10 del 3GPP para los parámetros de recepción; y para la antena transmisora omnidireccional se toman valores de antenas comerciales y del *release* 9 del 3GPP.

Para experimentar con valores de potencia de recepción que oscilen alrededor del nivel de sensibilidad mínima de la estación móvil con los valores descritos en la tabla 1.1, se recomienda ingresar un valor de radio mayor a 1850 metros.

Los resultados de las simulaciones del escenario 1 y sus promedios se encuentran en el Anexo digital C.1.

- Paso de variación de margen de *handover* de 0,1 dB.

En la tabla 3.2, se presentan los resultados obtenidos para el escenario 1, simulado con un valor de paso de variación de margen de *handover* de 0,1 dB, a diferentes valores de velocidad de aprendizaje y con el número de usuarios en la celda central para cada caso descrito en la sección 3.1.

Tabla 3.2. Resultados para el escenario 1 evaluado con un valor de paso de variación de *handover* de 0,1 dB.

ESCENARIO DE 500 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	4,89	95,20	14,65	0,07
0,6	4,41	95,20	14,24	0,07
0,7	4,05	95,75	14,88	0,06

0,8	3,94	95,00	14,81	0,07
0,9	3,60	95,80	15,15	0,06
0,99	3,33	94,80	15,12	0,06

ESCENARIO DE 320 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	2,79	100,00	1,04	0,99
0,6	2,49	100,00	1,04	0,96
0,7	2,14	100,00	1,07	0,94
0,8	2,00	100,00	1,08	0,93
0,9	1,69	100,00	1,12	0,90
0,99	1,40	100,00	1,00	1,03

ESCENARIO DE MÁXIMA CANTIDAD DE USUARIOS EN LA CELDA CENTRAL					
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**	***Handovers* fallidos por sobrecarga en las celdas vecinas**
0,5	11,13	50,38	82,56	0,01	38,20
0,6	10,66	50,75	84,10	0,01	35,50
0,7	10,40	50,28	82,29	0,01	42,80
0,8	10,33	50,35	82,37	0,01	34,40
0,9	9,87	50,93	83,89	0,01	41,50
0,99	9,61	50,46	83,08	0,01	34,80

Tal como se puede observar en la tabla 3.2 la diferencia de variación de margen de *handover* para el escenario de 500 usuarios en la celda central entre 4,89 dB y 3,33 dB es de 1,56 dB, para esta cantidad de usuarios, el tiempo transcurrido para el proceso de balanceo de cargas está en el rango de los 14,65 a 15,12 segundos y el promedio de *handover* exitosos por unidad móvil por segundo está en el rango de 0,06 a 0,07.

Para el escenario de 320 usuarios en la celda central la diferencia de variación de margen de *handover* entre 2,79 dB y 1,40 dB es de 1,39 dB. El tiempo transcurrido para el proceso de balanceo de cargas está en el rango de los 1,12 y 1 segundos y el promedio de *handover* exitosos por unidad móvil por segundo está en el rango de 1,03 a 0,90.

En el escenario de máxima cantidad de usuarios en la celda central la diferencia de variación de margen de *handover* entre 11,13 dB y 9,61 dB es de 1,52 dB. El tiempo

transcurrido para el proceso de balanceo de cargas está en el rango de los 82,56 y 84,10 segundos y el promedio de *handover* exitosos por unidad móvil por segundo es de 0,01.

En las siguientes tablas de resultados, se añadirá el análisis de cada parámetro luego de cada escenario.

- Paso de variación de margen de *handover* de 0,2 dB.

En la tabla 3.3, se presentan los resultados obtenidos para el escenario 1, simulado con un valor de paso de variación de margen de *handover* de 0,2 dB.

Tabla 3.3. Resultados para el escenario 1 evaluado con un valor de paso de variación de *handover* de 0,2 dB.

ESCENARIO DE 500 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	6,74	94,35	14,35	0,07
0,6	5,86	94,80	14,76	0,06
0,7	5,20	94,70	15,49	0,06
0,8	4,92	94,55	14,71	0,06
0,9	4,31	95,05	15,59	0,06
0,99	3,68	94,90	15,25	0,06
La diferencia de variación de margen de *handover* es de 3,16 dB. El tiempo transcurrido para el proceso está en el rango de los 15,59 y 14,35 segundos y el promedio de *handover* exitosos por unidad móvil por segundo está en el rango de 0,06 a 0,07.				
ESCENARIO DE 320 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	4,88	100,00	0,90	1,12
0,6	4,16	100,00	0,90	1,12
0,7	3,28	100,00	1,01	1,00
0,8	2,94	100,00	0,86	1,16
0,9	2,26	100,00	1,09	0,95
0,99	1,70	100,00	0,94	1,12

La diferencia de variación de margen de *handover* es de 3,18 dB El tiempo transcurrido para el proceso está en el rango de los 1,09 y 0,86 segundos y el promedio de *handover* exitosos por unidad móvil por segundo está en el rango de 1,16 a 0,95.

ESCENARIO DE MÁXIMA CANTIDAD DE USUARIOS EN LA CELDA CENTRAL					
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**	***Handovers* fallidos por sobrecarga en las celdas vecinas**
0,5	12,88	50,23	83,77	0,01	42,20
0,6	12,26	50,76	84,31	0,01	39,80
0,7	11,66	50,45	82,54	0,01	39,10
0,8	11,18	50,85	83,35	0,01	34,10
0,9	10,54	50,92	83,93	0,01	38,00
0,99	10,08	50,40	83,58	0,01	32,30

La diferencia de variación de margen de *handover* es de 2,80 dB. El tiempo transcurrido para el proceso está en el rango de los 84,31 y 82,54 segundos y el promedio de *handover* exitosos por unidad móvil por segundo es de 0,01.

- Paso de variación de margen de *handover* de 0,3 dB.

En la tabla 3.4, se presentan los resultados obtenidos para el escenario 1, simulado con un valor de paso de variación de margen de *handover* de 0,3 dB.

Tabla 3.4. Resultados para el escenario 1 evaluado con un valor de paso de variación de *handover* de 0,3 dB.

ESCENARIO DE 500 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	8,70	95,60	15,35	0,06
0,6	7,47	95,95	14,43	0,07
0,7	6,39	94,75	15,30	0,06
0,8	5,88	94,65	15,05	0,06
0,9	4,98	94,75	15,29	0,06
0,99	4,08	94,20	15,01	0,06

La diferencia de variación de margen de *handover* es de 4,62 dB. El tiempo transcurrido para el proceso está en el rango de los 15,35 y 14,43 segundos. El promedio de *handovers* exitosos está en el rango de 0,06 a 0,07.

ESCENARIO DE 320 USUARIOS EN LA CELDA CENTRAL

Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	6,90	100,00	1,05	0,96
0,6	5,67	100,00	1,09	0,93
0,7	4,53	100,00	1,07	0,94
0,8	3,78	100,00	1,06	0,95
0,9	2,88	100,00	1,02	0,98
0,99	2,01	100,00	0,84	1,20

La diferencia de variación de margen de *handover* es de 4,89 dB. El tiempo transcurrido para el proceso está en el rango de los 1,09 y 0,84 segundos. El promedio de *handover* exitosos por unidad móvil por segundo está en el rango de 1,20 a 0,93.

ESCENARIO DE MÁXIMA CANTIDAD DE USUARIOS EN LA CELDA CENTRAL

Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**	***Handovers* fallidos por sobrecarga en las celdas vecinas**
0,5	14,70	50,23	82,44	0,01	39,70
0,6	13,50	50,97	83,62	0,01	35,50
0,7	12,69	50,53	82,61	0,01	38,80
0,8	12,09	50,59	83,14	0,01	35,70
0,9	11,10	50,59	82,53	0,01	41,10
0,99	10,44	50,25	84,64	0,01	34,60

La diferencia de variación de margen de *handover* es de 4,26 dB. El tiempo transcurrido para el proceso está en el rango de los 83,62 y 82,44 segundos y el promedio de *handover* exitosos por unidad móvil por segundo es de 0,01.

- Paso de variación de margen de *handover* de 0,4 dB.

En la tabla 3.5, se presentan los resultados obtenidos para el escenario 1, simulado con un valor de paso de variación de margen de *handover* de 0,4 dB.

Tabla 3.5. Resultados para el escenario 1 evaluado con un valor de paso de variación de *handover* de 0,4 dB.

ESCENARIO DE 500 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	10,80	95,15	15,26	0,06
0,6	9,16	94,50	15,62	0,06
0,7	7,72	95,70	15,20	0,06
0,8	6,68	95,90	15,42	0,06
0,9	5,60	94,40	15,33	0,06
0,99	4,48	93,60	15,14	0,06
La diferencia de variación de margen de *handover* es de 6,32 dB. El tiempo transcurrido para el proceso está en el rango de los 15,62 y 15,14 segundos y el promedio de *handover* exitosos por unidad móvil por segundo está en el rango de 0,06 a 0,07.				
ESCENARIO DE 320 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	9,20	100,00	1,07	0,94
0,6	7,20	100,00	0,90	1,13
0,7	6,00	100,00	0,98	1,02
0,8	4,80	100,00	0,85	1,18
0,9	3,56	100,00	0,89	1,13
0,99	2,32	100,00	0,87	1,18
La diferencia de variación de margen de *handover* es de 6,88 dB. El tiempo transcurrido para el proceso está en el rango de los 1,07 y 0,87 segundos y el promedio de *handover* exitosos por unidad móvil por segundo está en el rango de 1,18 a 0,94.				
ESCENARIO DE MÁXIMA CANTIDAD DE USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**	**Handovers fallidos por sobrecarga en las celdas vecinas**
0,5	16,56	50,59	82,05	0,01	37,60
0,6	14,88	50,51	84,54	0,01	38,30
0,7	13,72	50,62	82,88	0,01	39,90

0,8	13,00	50,83	84,48	0,01	35,40
0,9	11,84	50,46	81,32	0,01	40,20
0,99	10,72	50,98	82,40	0,01	33,10
La diferencia de variación de margen de *handover* es de 5,84 dB. El tiempo transcurrido para el proceso está en el rango de los 84,54 y 81,32 segundos y el promedio de *handover* exitosos por unidad móvil por segundo es de 0,01.					

- Paso de variación de margen de *handover* de 0,5 dB.

En la tabla 3.6, se presentan los resultados obtenidos para el escenario 1, simulado con un valor de paso de variación de margen de *handover* de 0,5 dB.

Tabla 3.6. Resultados para el escenario 1 evaluado con un valor de paso de variación de *handover* de 0,5 dB, para 500 usuarios en la celda central.

ESCENARIO DE 500 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	13,35	95,00	15,13	0,06
0,6	10,70	94,85	14,71	0,07
0,7	8,80	95,55	14,78	0,07
0,8	7,70	94,95	14,57	0,07
0,9	6,25	94,25	15,32	0,06
0,99	4,75	94,90	14,79	0,06
La diferencia de variación de margen de *handover* es de 8,60 dB. El tiempo transcurrido para el proceso está en el rango de los 15,13 y 14,71 segundos y el promedio de *handover* exitosos por unidad móvil por segundo está en el rango de 0,06 a 0,07.				
ESCENARIO DE 320 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	11,50	100,00	1,00	1,00
0,6	9,00	100,00	0,90	1,12
0,7	7,25	100,00	0,86	1,17
0,8	5,80	100,00	0,82	1,22
0,9	4,15	100,00	0,84	1,20
0,99	2,85	100,00	0,85	1,18

La diferencia de variación de margen de *handover* es de 8,65 dB. El tiempo transcurrido para el proceso está en el rango de los 1 y 0,82 segundos y el promedio de *handover* exitosos por unidad móvil por segundo está en el rango de 1,22 a 1.

ESCENARIO DE MÁXIMA CANTIDAD DE USUARIOS EN LA CELDA CENTRAL

Velocidad de aprendizaje	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**	***Handovers* fallidos por sobrecarga en las celdas vecinas**
0,5	18,50	51,02	83,87	0,01	35,10
0,6	16,55	49,21	83,85	0,01	38,10
0,7	15,00	50,16	85,68	0,01	36,10
0,8	14,00	51,19	84,07	0,01	33,60
0,9	12,50	49,74	82,44	0,01	41,90
0,99	11,10	50,16	83,21	0,01	39,20

La diferencia de variación de margen de *handover* es de 7,40 dB. El tiempo transcurrido para el proceso está en el rango de los 85,68 y 83,21 segundos y el promedio de *handover* exitosos por unidad móvil por segundo es de 0,01.

3.1.2. ANÁLISIS DE RESULTADOS

Con los resultados obtenidos en la sección anterior se observa que para la obtención de la variación de margen de *handover* con el que el algoritmo alcanzó la máxima recompensa se puede destacar que mientras más cercano de la unidad es el valor de la velocidad de aprendizaje, menor será la variación de margen de *handover*, pero en términos de procesamiento de los resultados por parte del algoritmo, este utiliza una mayor cantidad de recursos.

En la figura 3.2 se puede observar la relación entre el valor de variación de margen de *handover* y la velocidad de aprendizaje tomando como ejemplo los resultados del escenario N°1 con 500 usuarios en la celda central y paso de variación de margen de *handover* de 0.5 dB.

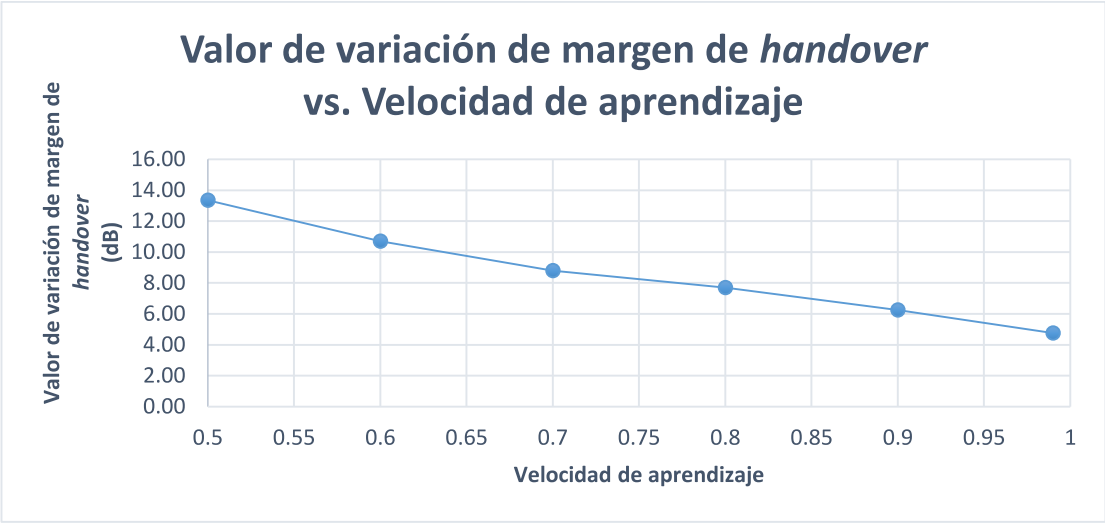


Figura 3.2. Relación entre el valor de variación de margen de *handover* y la velocidad de aprendizaje.

Un mayor valor de velocidad de aprendizaje corresponde a un mayor aprendizaje en base a los estados anteriores de la red.

La velocidad de aprendizaje influye directamente en el cálculo del valor de variación de margen de handover, pero no influye en el tiempo transcurrido para el proceso de balanceo cargas, ya que, el algoritmo realiza la misma cantidad de interacciones con el sistema para cada usuario que realiza el balanceo de cargas en el escenario establecido.

En la figura 3.3 se puede observar la relación entre la diferencia de variación de margen de *handover* y el paso de variación de margen de *handover* tomando como ejemplo los resultados del escenario N°1 con 2100 usuarios en la celda central.

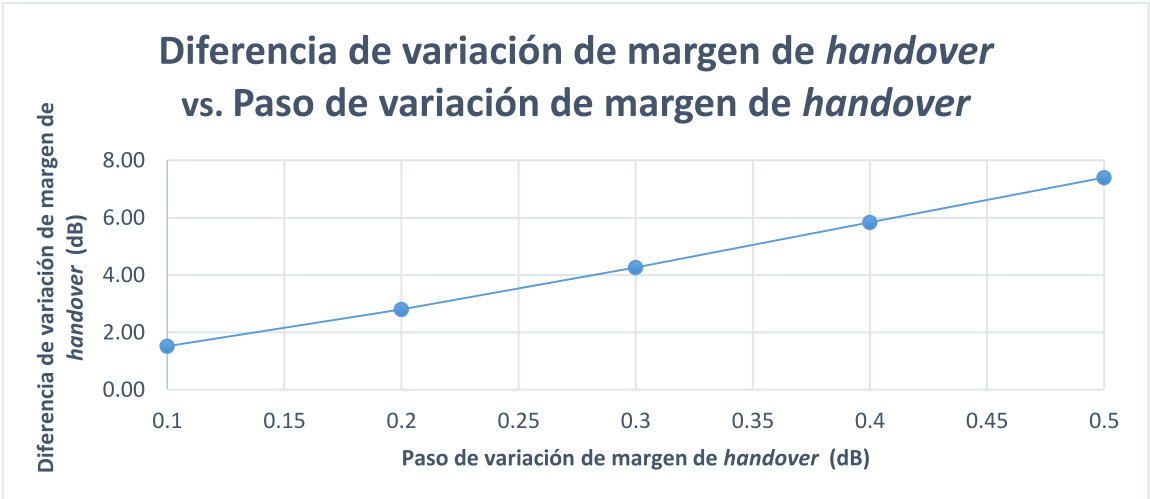


Figura 3.3. Relación entre la diferencia de variación de margen de *handover* y el paso de variación de margen de *handover*.

La diferencia de variación de margen de *handover* aumenta mientras mayor sea el paso de variación de margen de *handover*, debido a que un valor de paso mayor hace que la variación de margen de *handover* con el que el algoritmo interactúa en el sistema sea menos preciso.

En los escenarios con pequeña diferencia de variación de margen de *handover*, se puede configurar una velocidad de aprendizaje sin tener problemas de que el rendimiento del algoritmo varíe demasiado, en cambio en los escenarios con una diferencia de variación de margen de *handover* mayor (escenarios con paso de variación de margen de *handover* más grande) se debe seleccionar la velocidad de aprendizaje dependiendo del rendimiento que se quiera del sistema, si el usuario desea que los estados anteriores tengan una mayor ponderación en el aprendizaje o si el sistema puede realizar un ajuste de variación de margen de *handover* más grande.

En la figura 3.4 se puede observar la relación entre el valor de variación de margen de *handover* y el paso de variación de margen de *handover* tomando como ejemplo los resultados del escenario N°1 con 500 usuarios en la celda central.

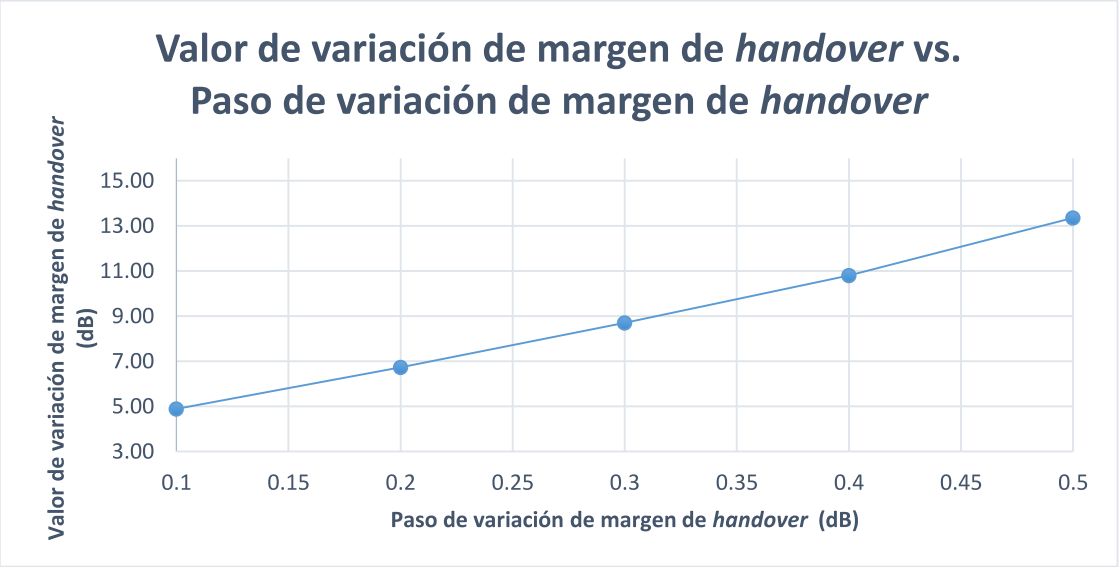


Figura 3.4. Relación entre el valor de variación de margen de *handover* y el paso de variación de margen de *handover.*

Disminuir el paso de variación de margen de *handover* es lo más óptimo para obtener un valor de variación de margen de *handover* menor, pero mayor será la cantidad de interacciones del algoritmo con el sistema para poder alcanzar la máxima recompensa establecida.

En la figura 3.5 se puede observar la relación entre el porcentaje de *handovers* exitosos y el número de usuarios en la celda central tomando como ejemplo los resultados del escenario N°1 con paso de variación de margen de *handover* de 0.3 dB y velocidad de aprendizaje igual a 0.9.

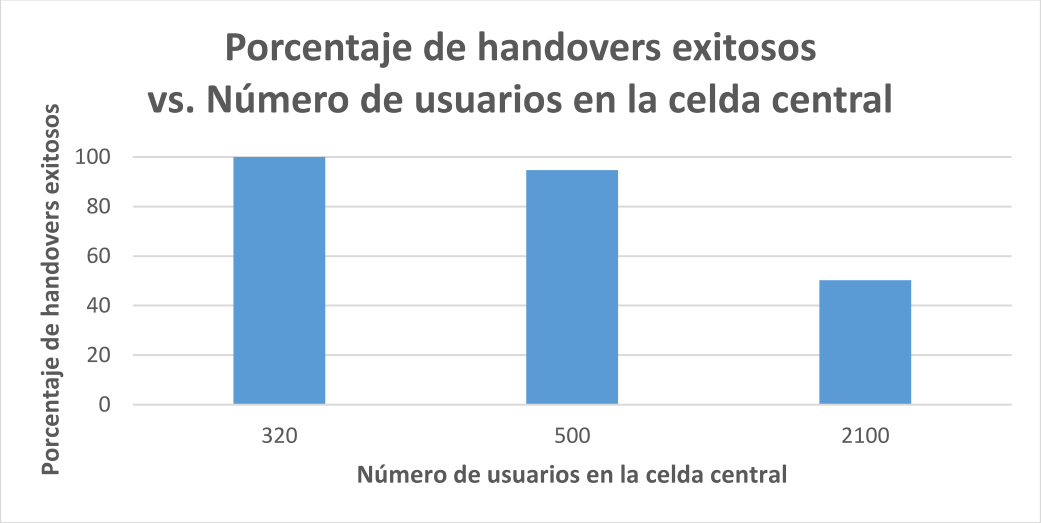


Figura 3.5. Relación entre el porcentaje de *handovers* exitosos y el número de usuarios en la celda central.

En los escenarios con la máxima capacidad de usuarios en la celda central, se puede observar un claro descenso en el porcentaje de *handovers* exitosos, esto se debe a la distribución aleatoria de los usuarios, que ha provocado que una o varias de las celdas vecinas se sobrecarguen y que los *handovers* de los usuarios que han sobrecargado a la celda sean fallidos. En estos escenarios la variación de margen de *handover* aumenta a comparación de los otros escenarios debido a que el algoritmo muestra el valor con el que se obtuvo la máxima recompensa, si existe un solo usuario que no cumple la condición descrita en la ecuación 2.3, el algoritmo aumenta la variación de margen de *handover* en un paso y se vuelve a evaluar hasta que cumpla la condición y se alcance la máxima recompensa, mientras mayor sea el número de usuarios de sobrecarga en la celda central mayor será la variación de margen de *handover* necesario.

La inclusión de *machine learning* en el proceso de balanceo de cargas permitiría a una red de telecomunicaciones activa auto-configurar a sus estaciones base un valor de variación de margen de *handover* que garantice el correcto traspaso de las conexiones que sobrecarguen a una celda, mediante los resultados obtenidos en una tabla de recompensas a partir de simulaciones previas, considerando el número de usuarios de la red, la velocidad de aprendizaje, paso de variación de margen de handover, frecuencia de portadora, ganancias y pérdidas del sistema.

Cabe destacar que en los siguientes escenarios establecidos se han podido observar resultados similares respecto al primer escenario. Por consiguiente, en las siguientes secciones se muestran los parámetros de simulación de cada escenario y para facilidad del lector sus resultados se encuentran en el anexo digital de cada uno de los escenarios descritos.

3.1.3. ESCENARIO Nº2: FRECUENCIA 850 MHZ, RADIO 18500 M, POTENCIA DE TRANSMISIÓN 20 DBM.

Los parámetros de simulación establecidos para una potencia de transmisión de la estación base de 20 dBm a una frecuencia de 850 MHz se describen en la tabla 3.7:

Tabla 3.7. Parámetros de simulación del escenario N°2 [18] [39] [40].

Parámetros	**Valor**
Potencia de Transmisión (Estación Base)	0 dBm
Ganancia de Transmisión (Estación Base)	2 dBi
Pérdidas de Transmisión (Estación Base)	3 dB
Ganancia de Recepción (Estación Móvil)	0 dBi
Pérdidas de Recepción (Estación Móvil)	0 dB
Ganancia de handover	3.6 dB

Para experimentar valores de potencia de recepción que oscilen el nivel de sensibilidad mínima de la estación móvil con los valores descritos en la tabla 1.1, se recomienda ingresar un valor de radio mayor a 18500 metros ya que se consideran pérdidas de propagación en el espacio libre.

Los resultados de las simulaciones del escenario 2 y sus promedios se encuentran en el Anexo digital C.2.

3.1.4. ESCENARIO Nº3: FRECUENCIA 1900 MHZ, RADIO 1150 M, POTENCIA DE TRANSMISIÓN 0 DBM.

Los parámetros de simulación establecidos para una potencia de transmisión de la estación base de 0 dBm a una frecuencia de 1900 MHz se describen en la tabla 3.8:

Tabla 3.8. Parámetros de simulación del escenario N°3 [18] [39] [40].

Parámetros	**Valor**
Potencia de Transmisión (Estación Base)	0 dBm
Ganancia de Transmisión (Estación Base)	5 dBi

Pérdidas de Transmisión (Estación Base)	3 dB
Ganancia de Recepción (Estación Móvil)	0 dBi
Pérdidas de Recepción (Estación Móvil)	0 dB
Ganancia de handover	3.6 dB

Para experimentar valores de potencia de recepción que oscilen el nivel de sensibilidad mínima de la estación móvil con los valores descritos en la tabla 1.1, se recomienda ingresar un valor de radio mayor a 1150 metros.

Los resultados de las simulaciones del escenario 3 y sus promedios se encuentran en el Anexo digital C.3.

3.1.5. ESCENARIO Nº4: FRECUENCIA 1900 MHZ, RADIO 11500 M, POTENCIA DE TRANSMISIÓN 20 DBM.

Los parámetros de simulación establecidos para una potencia de transmisión de la estación base de 20 dBm a una frecuencia de 1900 MHz se describen en la tabla 3.9:

Tabla 3.9. Parámetros de simulación del escenario N°4 [18] [39] [40].

Parámetros	**Valor**
Potencia de Transmisión (Estación Base)	20 dBm
Ganancia de Transmisión (Estación Base)	2 dBi
Pérdidas de Transmisión (Estación Base)	3 dB
Ganancia de Recepción (Estación Móvil)	0 dBi
Pérdidas de Recepción (Estación Móvil)	0 dB
Ganancia de handover	3.6 dB

Para experimentar valores de potencia de recepción que oscilen el nivel de sensibilidad mínima de la estación móvil con los valores descritos en la tabla 1.1, se recomienda ingresar un valor de radio mayor a 11500 metros.

Los resultados de las simulaciones del escenario 4 y sus promedios se encuentran en el Anexo digital C.4.

3.1.6. ESCENARIO Nº5: FRECUENCIA 2100 MHZ, RADIO 1050 M, POTENCIA DE TRANSMISIÓN 0 DBM.

Los parámetros de simulación establecidos para una potencia de transmisión de la estación base de 0 dBm a una frecuencia de 2100 MHz se describen en la tabla 3.10:

Tabla 3.10. Parámetros de simulación del escenario N°5 [18] [39] [40].

Parámetros	**Valor**
Potencia de Transmisión (Estación Base)	0 dBm
Ganancia de Transmisión (Estación Base)	5 dBi
Pérdidas de Transmisión (Estación Base)	3 dB
Ganancia de Recepción (Estación Móvil)	0 dBi
Pérdidas de Recepción (Estación Móvil)	0 dB
Ganancia de handover	3.6 dB

Para experimentar valores de potencia de recepción que oscilen el nivel de sensibilidad mínima de la estación móvil con los valores descritos en la tabla 1.1, se recomienda ingresar un valor de radio mayor a 1050 metros.

Los resultados de las simulaciones del escenario 5 y sus promedios se encuentran en el Anexo digital C.5.

3.1.7. ESCENARIO Nº6: FRECUENCIA 2100 MHZ, RADIO 10500 M, POTENCIA DE TRANSMISIÓN 20 DBM.

Los parámetros de simulación establecidos para una potencia de transmisión de la estación base de 20 dBm a una frecuencia de 850 MHz se describen en la tabla 3.11:

Tabla 3.11. Parámetros de simulación del escenario N°6 [18] [39] [40].

Parámetros	**Valor**
Potencia de Transmisión (Estación Base)	20 dBm
Ganancia de Transmisión (Estación Base)	5 dBi
Pérdidas de Transmisión (Estación Base)	3 dB
Ganancia de Recepción (Estación Móvil)	0 dBi
Pérdidas de Recepción (Estación Móvil)	0 dB
Ganancia de handover	3.6 dB

Para experimentar valores de potencia de recepción que oscilen el nivel de sensibilidad mínima de la estación móvil con los valores descritos en la tabla 1.1, se recomienda ingresar un valor de radio mayor a 10500 metros.

Los resultados de las simulaciones del escenario 6 y sus promedios se encuentran en el Anexo digital C.6.

3.1.8. COMPARACIÓN DE RESULTADOS RESPECTO A LA CANTIDAD DE SIMULACIONES.

En esta sección se realiza una comparación respecto a la cantidad de simulaciones con la finalidad de demostrar que la cantidad de simulaciones realizadas en los escenarios anteriores demuestran resultados fiables.

Para esto se ha realizado el cálculo de un escenario mediante 1000 simulaciones, estos resultados serán comparados respecto al mismo escenario simulado 10 veces.

Los resultados del escenario con 1000 simulaciones y su promedio se encuentran en el Anexo C.7.

El escenario se ha simulado con los siguientes parámetros:

- Escenario: 320 usuarios en la celda central (20 usuarios de sobrecarga).
- Velocidad de aprendizaje: 0,9.
- Paso de variación de margen de *handover*: 0,1 dB.
- Frecuencia de portadora: 850 MHz.
- Radio de cobertura de la celda: 1850 m.

Los demás parámetros establecidos se encuentran descritos en la tabla 3.1.

En las tablas 3.12 y 3.13 se pueden observar los resultados promediados del escenario descrito, simulado 1000 y 10 veces respectivamente.

Tabla 3.12. Resultados para el escenario simulado 1000 veces.

ESCENARIO DE 320 USUARIOS EN LA CELDA CENTRAL (1000 SIMULACIONES)				
Velocidad de aprendizaje (0,9)	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
PROMEDIO	1,63	100,00	1,13	0,89

Tabla 3.13. Resultados para el escenario simulado 10 veces.

ESCENARIO DE 320 USUARIOS EN LA CELDA CENTRAL (10 SIMULACIONES)				
Velocidad de aprendizaje (0,9)	**Variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
PROMEDIO	1,69	100,00	1,12	0,90

Tal como se puede observar en las tablas 3.12 y 3.13, se puede comprobar que el número de simulaciones realizadas no influye en gran medida en los resultados obtenidos.

La diferencia entre las variaciones de margen de *handover* es de 0,6 dB entre el escenario simulado 10 y 1000 veces, el porcentaje de *handovers* exitosos se ha mantenido en 100 %, el tiempo transcurrido del proceso de balanceo de cargas aumentó en 0,89 % y el promedio de *handovers* exitosos por unidad móvil por segundo disminuyó en 1,12%.

3.1.9. SIMULACIÓN DE UN ESCENARIO DE 200 USUARIOS DE SOBRECARGA EN LA CELDA CENTRAL Y TRES CELDAS VECINAS SOBRECARGADAS.

En esta sección se realiza una comparación de un escenario que tiene 200 usuarios de sobrecarga en la celda central y, además, hay tres celdas vecinas sobrecargadas (celdas N°3, N°4, N°6). De esta forma se puede observar el rendimiento del algoritmo respecto al mismo escenario sin celdas vecinas sobrecargadas.

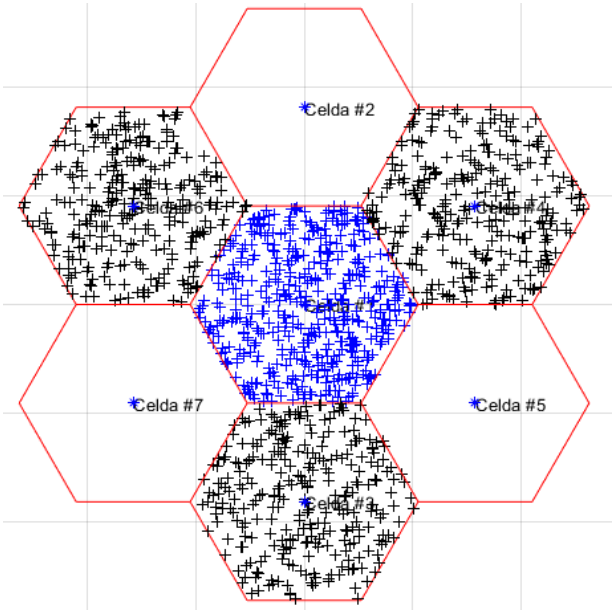


Figura 3.6. Representación gráfica del escenario simulado (celdas N°3, N°4, N°6 sobrecargadas).

El escenario se ha simulado con los siguientes parámetros:

- Escenario: 500 usuarios en la celda central (200 usuarios de sobrecarga).
- Paso de variación de margen de *handover*: 0,3 dB.
- Frecuencia de portadora: 2100 MHz.
- Radio de cobertura de la celda: 10500 m.

Los demás parámetros establecidos se encuentran descritos en la tabla 3.11.

Los resultados de las simulaciones del escenario con las celdas vecinas sobrecargadas y su promedio se encuentran en el Anexo C.8.

En la tabla 3.14 se pueden observar los resultados del escenario con las celdas vecinas sobrecargadas incluyendo un contador de *handovers* fallidos por la sobrecarga de las celdas vecinas.

Tabla 3.14. Escenario de 500 usuarios en la celda central con 3 celdas vecinas sobrecargadas.

ESCENARIO 500 USUARIOS EN LAUSUARIOS CELDA CENTRAL					
Velocidad de aprendizaje	**Valor de variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**	***Handovers* fallidos por sobrecarga en las celdas vecinas**
0.5	8.73	46.90	5.56	0.08	101.40
0.6	7.50	47.95	5.91	0.08	99.80
0.7	6.42	49.40	5.73	0.09	97.10
0.8	5.91	47.65	5.61	0.09	100.30
0.9	4.89	50.15	6.20	0.08	95.50
0.99	3.99	48.10	5.64	0.09	99.80

En la tabla 3.15, están los resultados del escenario sin celdas vecinas sobrecargadas.

Tabla 3.15. Escenario de 500 usuarios en la celda central sin celdas vecinas sobrecargadas.

ESCENARIO DE 500 USUARIOS EN LA CELDA CENTRAL				
Velocidad de aprendizaje	**Valor de variación de margen de *handover* (dB)**	**Porcentaje de *handovers* exitosos (%)**	**Tiempo transcurrido del proceso de balanceo de cargas (seg)**	**Promedio de *handover* exitosos por unidad móvil por segundo**
0,5	8,70	95,70	14,73	0,06
0,6	7,41	96,10	15,15	0,06
0,7	6,48	96,20	14,70	0,07
0,8	5,85	96,75	15,27	0,06
0,9	4,95	96,75	14,60	0,07
0,99	4,08	95,35	15,07	0,06

Tal como se puede observar en las tablas 3.14 y 3.15, se comprueba que, aumentando el número de usuarios en las celdas vecinas, en este caso sobrecargándolas, se produce un descenso en el rendimiento del algoritmo, incluyendo *handovers* fallidos por traspasos hacia las celdas ya sobrecargadas. Además, se observa un descenso en el tiempo transcurrido para el proceso de balanceo de cargas debido a que una vez que se sobrecarga una celda y se quiere añadir la conexión de un usuario, el algoritmo no realiza el cálculo de la variación de margen de *handover* para dicho usuario, y ese *handover* es considerado como fallido.

Realizando un promedio de los resultados de cada parámetro, se determinó que el porcentaje de *handovers* exitosos disminuyó en 49,80 %, el promedio de *handovers* exitosos por unidad móvil por segundo disminuyó en 52,63%, el tiempo transcurrido del proceso de balanceo de cargas disminuyó en 61,33%, y el promedio de *handovers* exitosos por segundo aumentó en 33.33%

3.2. CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA EN LA ESTACIÓN MÓVIL SEGÚN LA RECOMENDACIÓN UIT-R P.1816-4, ANEXO 3.

Los resultados obtenidos del programa que calcula el perfil angular de potencia de llegada en la estación móvil son representaciones gráficas que describen la incidencia de las señales respecto al ángulo de llegada de la señal para entornos con línea de vista (LoS) y sin línea de vista (NLoS).

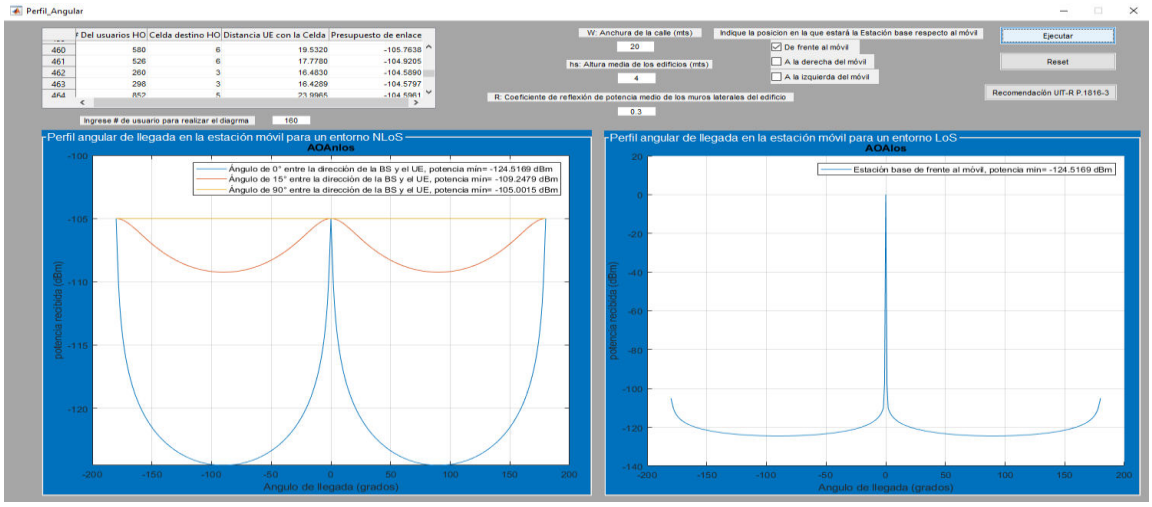


Figura 3.7. Representaciones gráficas del perfil angular de potencia de llegada de un usuario que ha realizado el proceso de balanceo de cargas.

Se debe recordar que los parámetros de entrada que ingresa el usuario de la herramienta de simulación son los siguientes:

- hs = Altura media de los edificios a lo largo de la carretera.
- W = Anchura de la calle.
- R = Coeficiente de reflexión de potencia medio de los muros laterales del edificio.

Las siguientes simulaciones se han ejecutado sobre un terminal móvil que ha realizado el *handover* en la simulación del programa de balanceo de cargas, el valor de potencia recibida respecto a la celda objetivo es de -105 dBm y el ancho de la calle es de 20 m.

3.2.1. CÁLCULO DEL DE PERFIL ANGULAR DE POTENCIA DE LLEGADA PARA UN ENTORNO NLOS

Para el cálculo del perfil angular de potencia de llegada en la estación móvil para un entorno sin línea de vista, el valor de la altura media de los edificios es el único parámetro del que depende esta simulación, por lo que se han seleccionado diferentes valores de la altura del edificio, que son 4, 10 y 30 metros.

Como se puede observar en las siguientes figuras, existen 3 perfiles calculados, cada uno de ellos corresponde a un ángulo agudo entre la dirección de la estación móvil y la dirección de la carretera para un entorno sin línea de vista. Los valores de ángulos seleccionados fueron 0, 15 y 90 grados, valores tomados según la recomendación UIT-R P.1816-4, Anexo 3 [36].

Para el entendimiento de las siguientes figuras hay que considerar que en el eje X de la gráfica se representa el nivel de potencia de la señal recibida en dBm a lo largo del rango del ángulo de llegada cuando se fija el ángulo de la carretera a un valor de 0 grados, representado en el eje Y.

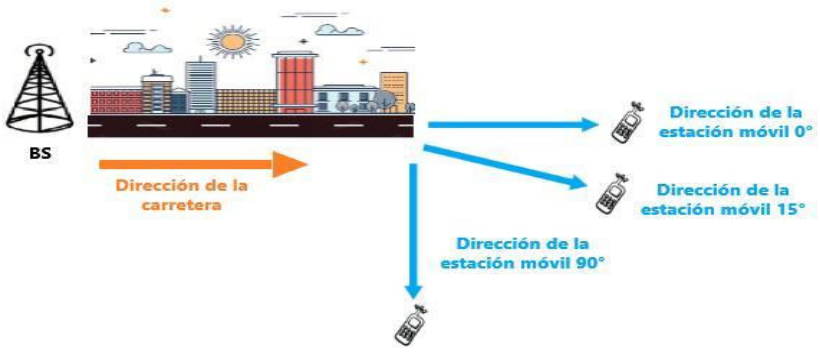


Figura 3.8. Diagrama del ángulo agudo entre la dirección de la estación móvil y la dirección de la carretera para un entorno sin línea de vista.

Tal como se puede observar en las siguientes figuras, se presentan las simulaciones del perfil angular de potencia de llegada en la estación móvil con una altura media de los edificios a lo largo de la carretera de 4, 10 y 30 metros.

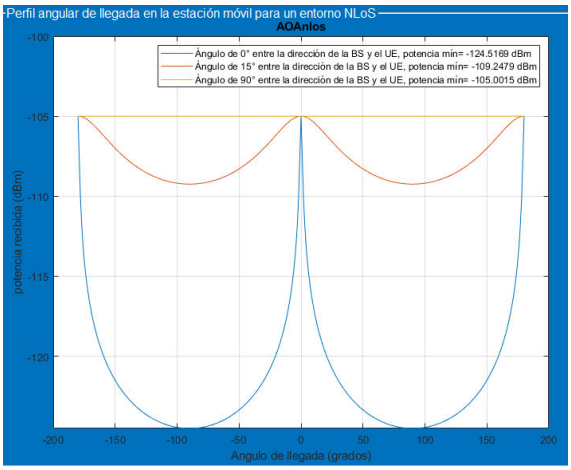


Fig 3.9 a

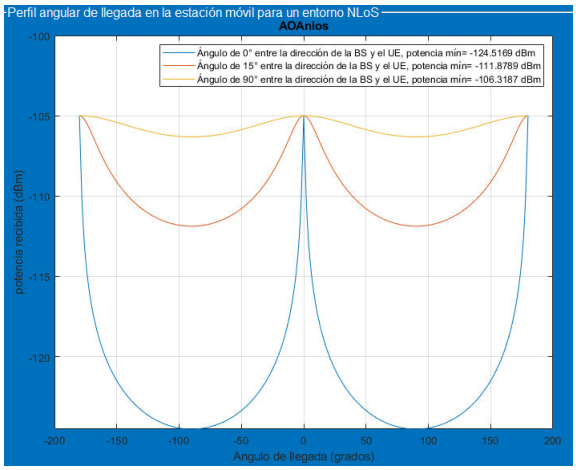


Fig 3.9 b

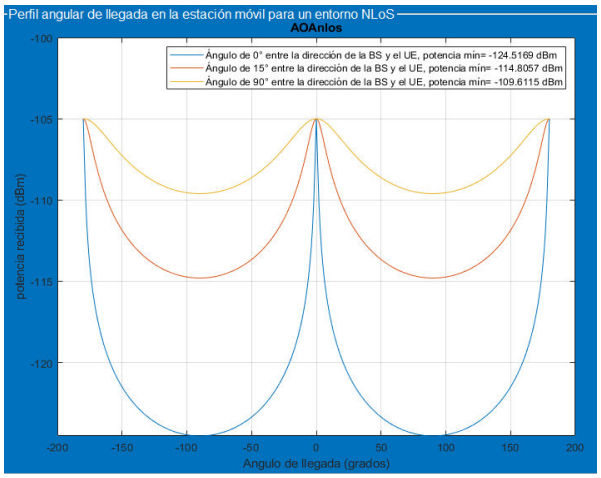


Fig 3.9 c

Figura 3.9. Perfil angular de potencia de llegada para un entorno NLoS, con una altura media de los edificios a lo largo de la carretera de a) 4 metros b) 10 metros c) 30 metros.

En los entornos simulados se puede observar que el menor nivel de la señal de llegada se obtiene cuando el ángulo de llegada se aproxima a valores de -90 y 90 grados.

Con los resultados obtenidos a una altura media de los edificios a lo largo de la carretera de 4 metros, se puede observar que, con un ángulo de 90 grados entre la dirección de la estación móvil y la dirección de la carretera, se tiene el mismo nivel de señal durante todo el rango del ángulo de llegada de la señal (-105 dBm); con un ángulo de 15 grados el nivel de la señal desciende hasta -111,88 dBm, y con un ángulo de 0 grados, el nivel de la señal de llegada desciende a -124.52 dBm.

Para una altura media de los edificios a lo largo de la carretera de 10 metros y con un ángulo de 90 grados entre la dirección de la estación móvil y la dirección de la carretera, el nivel de la señal desciende hasta -106,32 dBm; con un ángulo de 15 grados el nivel de la señal desciende hasta -109,24 dBm, y con un ángulo de 0 grados, el nivel de la señal de llegada desciende a -124.52 dBm.

Finalmente, para una altura media de los edificios a lo largo de la carretera de 30 metros y con un ángulo de 90 grados entre la dirección de la estación móvil y la dirección de la carretera, el nivel de la señal desciende hasta -109,61 dBm; con un ángulo de 15 grados el nivel de la señal desciende hasta -114,81 dBm, y con un ángulo de 0 grados, el nivel de la señal de llegada desciende a -124.52 dBm.

3.2.2. CÁLCULO DEL PERFIL ANGULAR DE POTENCIA DE LLEGADA PARA UN ENTORNO LOS

El cálculo del perfil angular de potencia de llegada en la estación móvil para un entorno con línea de vista toma en cuenta el cálculo del perfil angular de potencia de llegada para un entorno sin línea de vista para la misma estación móvil.

En la recomendación UIT-R P.1816-4 Anexo 3, para zonas urbanas se recomienda un valor de 0,3 para el coeficiente de reflexión de potencia de los muros laterales del edificio “R”, cuando la altura media de los edificios y la anchura de la calle son 10 y 20 metros respectivamente [36], estos dos últimos valores han sido seleccionados para las siguientes simulaciones.

Para la presentación de los resultados obtenidos, se han seleccionado diferentes valores del coeficiente de reflexión de potencia de los muros laterales del edificio “R”, estos valores son 0,1, 0,3 y 0,5. El valor del coeficiente “R” corresponde al material del que se encuentran hechos los muros de los edificios en los entornos simulados. Para muros de hormigón, la atenuación media de la señal es igual a -5 dB (0,3) en redes de telecomunicaciones en las bandas de frecuencia seleccionadas para este proyecto [41].

Los demás valores han sido seleccionados para observar de mejor manera la incidencia de la señal cuando el coeficiente de reflexión de potencia de los muros se acerca a los límites establecidos en la sección 1.3.5.1.

En el análisis de resultados se presenta el rango de ángulo de llegada de la señal en donde la potencia del perfil angular supera la potencia recibida del usuario seleccionado que ha realizado el *handover*, la representación gráfica del perfil angular en estos entornos es adecuada excluyendo dicho rango de ángulo de llegada, debido a que en estos casos no

se puede superar el nivel de potencia recibida inicial tal como se indica en la recomendación.

3.2.2.1. Simulación del perfil angular de potencia de llegada para un entorno LoS, estación base ubicada a la derecha de la calle.

En las siguientes figuras se pueden observar los perfiles angulares de potencia de llegada en la estación móvil calculado para un entorno con línea de vista, cuando la estación base se encuentra a la derecha de la calle, con valores del coeficiente de reflexión de potencia de los muros laterales del edificio “R” igual a 0,1, 0,3 y 0,5.

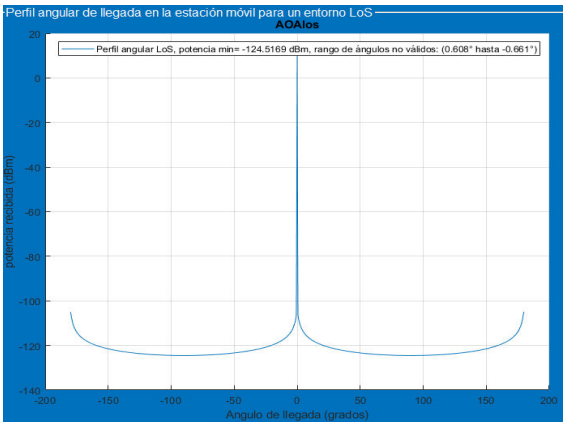


Fig 3.10 a

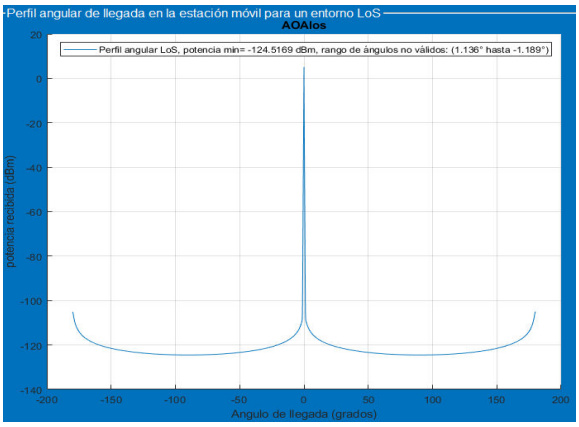


Fig 3.10 b

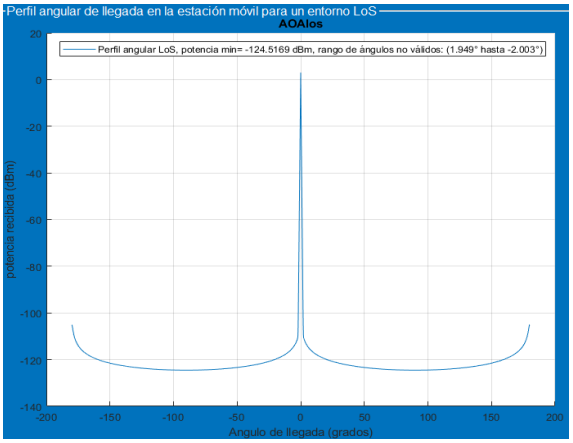


Fig 3.10 c

Figura 3.10. Perfil angular de potencia de llegada para un entorno LoS cuando la estación base está ubicada a la derecha de la calle con un valor de “R” igual a a) 0,1 b) 0,3 c) 0,5.

Tal como se puede observar en las figuras, el nivel de la potencia de la señal recibida desciende hasta -124,52 dBm cuando el valor del ángulo de llegada se distancia de cero. Para un valor del coeficiente “R” igual a 0,1 el rango de ángulo de llegada en donde se supera el nivel de la señal recibida (-105 dBm) es de 0,608° hasta -0,661° (1,269°); para

“R” igual a 0,3 el rango de ángulo de llegada es de 1,136° hasta -1,189° (2,325°) y finalmente, para “R” igual a 0,5 el rango de ángulo de llegada es de 1,949° hasta -2,003° (3,952°).

3.2.2.2. Simulación del perfil angular de potencia de llegada para un entorno LoS cuando la estación base está ubicada a la izquierda de la calle.

En las siguientes figuras se pueden observar los perfiles angulares de potencia de llegada en la estación móvil calculado para un entorno con línea de vista, cuando la estación base se encuentra a la izquierda de la calle, con valores del coeficiente de reflexión de potencia de los muros laterales del edificio “R” igual a 0,1, 0,3 y 0,5.

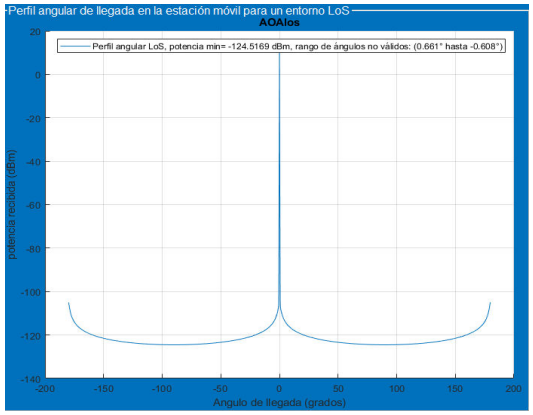


Fig 3.11 a

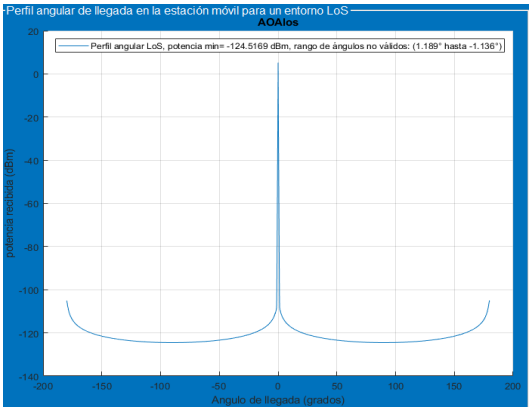


Fig 3.11 b

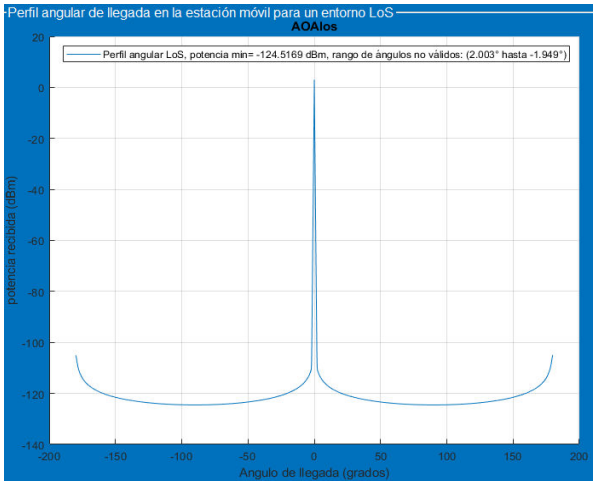


Fig 3.11 c

Figura 3.11. Perfil angular de potencia de llegada para un entorno LoS cuando la estación base está ubicada a la izquierda de la calle con un valor de “R” igual a a) 0,1 b) 0,3 c) 0,5.

Tal como se puede observar en las figuras, el nivel de la potencia de la señal recibida desciende hasta -124,52 dBm cuando el valor del ángulo de llegada se distancia de cero. Para un valor del coeficiente “R” igual a 0,1 el rango de ángulo de llegada en donde se supera el nivel de la señal recibida (-105 dBm) es de 0,661° hasta -0,608° (1,269°); para “R” igual a 0,3 el rango de ángulo de llegada es de 1,189° hasta -1,136° (2,325°) y finalmente, para “R” igual a 0,5 el rango de ángulo de llegada es de 2,003° hasta -1,949° (3,952°).

3.2.2.3. Simulación del perfil angular de potencia de llegada para un entorno LoS, estación base de frente al extremo de la calle.

En las siguientes figuras se pueden observar los perfiles angulares de potencia de llegada en la estación móvil calculado para un entorno con línea de vista, cuando la estación base se encuentra frente al extremo de la calle, con valores del coeficiente de reflexión de potencia de los muros laterales del edificio “R” igual a 0,1, 0,3 y 0,5.

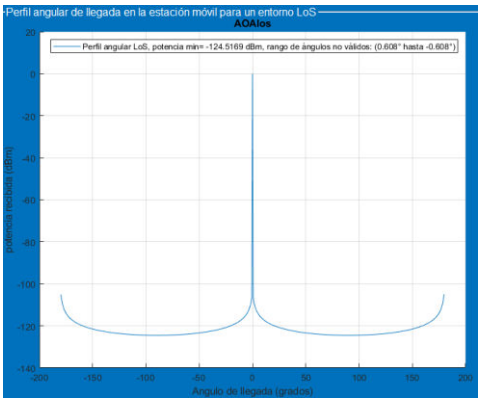


Fig 3.12 a

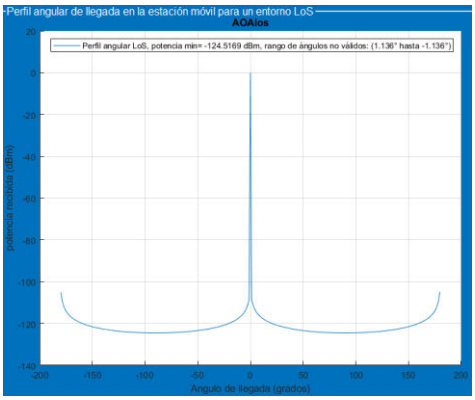


Fig 3.12 b

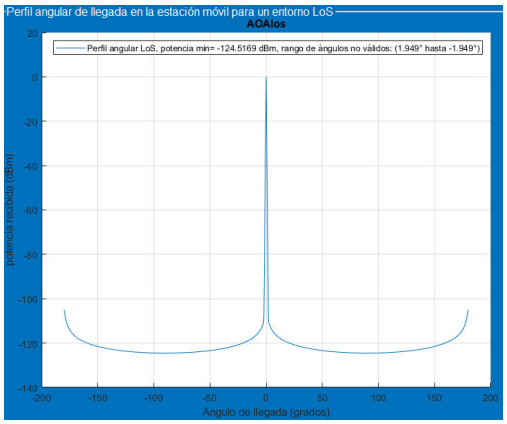


Fig 3.12 c

Figura 3.8. Perfil angular de potencia de llegada para un entorno LoS cuando la estación base está ubicada de frente al extremo de la calle con un valor de “R” igual a a) 0,1 b) 0,3 c) 0,5.

Tal como se puede observar en las figuras, el nivel de la potencia de la señal recibida desciende hasta -124,52 dBm cuando el valor del ángulo de llegada se distancia de cero. Para un valor del coeficiente “R” igual a 0,1 el rango de ángulo de llegada en donde se supera el nivel de la señal recibida (-105 dBm) es 0,608° hasta -0,608° (1,216°); para “R” igual a 0,3 el rango de ángulo de llegada es de 1,136° hasta -1,136° (2,272°) y finalmente, para “R” igual a 0,5 el rango de ángulo de llegada es de 1,949° hasta -1,949° (3,898°).

Como se puede observar en los resultados obtenidos el mínimo nivel de la señal (-124,52 dBm) es el mismo en los entornos simulados.

La diferencia entre las simulaciones respecto a los distintos valores del coeficiente “R” radica en que el rango de ángulo de llegada en que la potencia supera el nivel de la señal del usuario seleccionado es más grande cuando el valor del coeficiente “R” es mayor. Mientras que, respecto a la ubicación de la estación base, en el entorno en donde la estación base se encuentra de frente al extremo de la calle se puede observar simetría en el rango del ángulo de llegada considerando una misma atenuación en ambas direcciones; en cambio, cuando la estación base se encuentra a un lado de la calle el rango del ángulo de llegada es mayor del lado en donde se encuentra la estación base, indicando que la atenuación de la señal es menor en esa dirección de la señal de llegada.

4. CONCLUSIONES Y RECOMENDACIONES

4.1. CONCLUSIONES

- En las simulaciones del programa de balanceo de cargas, en los casos en los que se simuló con 20 usuarios de sobrecarga en la celda central, se obtuvieron resultados del 100% de éxito en los traspasos realizados, para los casos con 200 usuarios de sobrecarga se empezó a observar una caída en el rendimiento del algoritmo a porcentajes mayores al 90%, pero cuando se simuló con la cantidad máxima permitida por el sistema, es decir 2100 usuarios, se obtuvieron resultados en donde las celdas se sobrecargaron debido a la distribución aleatoria de los usuarios. En este último caso, el descenso del rendimiento del algoritmo alcanzó porcentajes de entre 60 a 50% de *handover* exitosos debido a la distancia de los usuarios hacia la estación base de la celda objetivo, la distribución aleatoria de los usuarios y la ocurrencia de *handovers* fallidos por no cumplir la sensibilidad mínima del receptor y por la sobrecarga de la capacidad de las celdas vecinas.
- El tiempo transcurrido para el proceso de balanceo de cargas en los diferentes casos simulados, depende solamente del número de usuarios con los que se realizó la simulación y se tiene una relación directamente proporcional. En los casos simulados con 320 usuarios en la celda central, se obtuvieron valores cercanos al segundo; para los casos simulados con 500 usuarios en la celda central, el tiempo de simulación estuvo en el rango de los 13 a 15 segundos de simulación, y finalmente, para los escenarios con la máxima cantidad de usuarios (2100 usuarios en la celda central), el tiempo de simulación tomó valores mucho más grandes que estaban en el rango de los 70 a 90 segundos.
- En las simulaciones del programa de balanceo de cargas cuando se tomó un valor de velocidad de aprendizaje menor a 0,5, no fue posible alcanzar el máximo valor de recompensa establecido por el sistema con el valor de margen de *handover* máximo de 12 dB, debido a que el algoritmo basado en el aprendizaje por refuerzo a menor valor de velocidad de aprendizaje aprende en menor medida de los estados anteriores para alcanzar la recompensa máxima. Con la inclusión del proceso basado en *machine learning*, mientras mayor sea la velocidad de aprendizaje el algoritmo alcanzará la recompensa máxima en menor cantidad de interacciones, pero mayor será la cantidad de recursos utilizados por el algoritmo.
- La variación de margen de *handover* con el que se llega al valor de recompensa máximo se reduce, mientras más cercano sea el valor de velocidad de aprendizaje

a 1, esto ocurre debido a que, a mayor valor de velocidad de aprendizaje, mayor será la actualización de las recompensas obtenidas por el sistema.

- Con los resultados obtenidos en el presente proyecto de titulación, se ha podido determinar que a mayor valor del paso de variación de margen de *handover*, mayor será la diferencia de variación de margen de *handover* con el que se obtuvo la máxima recompensa entre los diferentes valores de velocidad de aprendizaje. Mientras que, a menor valor del paso de variación de margen de *handover* con el que se evalúe el sistema, más preciso será para el algoritmo obtener el valor de variación de *handover* con la máxima recompensa, pero, mayor será la cantidad de interacciones que debe realizar el algoritmo para poder determinar la variación de margen de *handover*.
- Se puede concluir que con la implementación del algoritmo de *Q*-Learning apartir del *machine learning* y el aprovechamiento del proceso de *handover* es posible realizar efectivamente el proceso de balanceo de cargas de una celda central, que se limitaría por la capacidad disponible en las celdas circundantes.
- El algoritmo desarrollado en este trabajo de titulación se podría implementar en una red real agregando este proceso en las estaciones base que dan la conexión a los usuarios de la red, tomando como datos de entrada el nivel de potencia de la señal recibida de cada estación móvil para establecer la variación de margen de *handover*, información que deberá ser intercambiada mediante la interfaz X2 entre las estaciones base.
- Para el cálculo del perfil angular de potencia de llegada en la estación móvil para un entorno NLoS, mientras mayor sea la altura media de los edificios a lo largo de la carretera, se puede observar un menor nivel de la señal en los perfiles calculados con 90 y 15 grados entre la dirección de la estación móvil y la dirección de la carretera.
- Cuando el ángulo de llegada de la señal recibida se aproxima a valores de -90 y 90 grados y el perfil angular es calculado con un ángulo de 0 grados entre la dirección de la estación móvil y la dirección de la carretera, el nivel de la señal es la misma para cualquier valor de altura media de los edificios.
- El valor del coeficiente de reflexión de potencia de los muros laterales del edificio “R” determina el nivel de atenuación de la señal recibida cuando el ángulo de llegada de la señal se aleja de 0°, por lo que a mayor valor del coeficiente “R” se tiene un mayor rango de ángulo de llegada en donde se supera el nivel de potencia del usuario seleccionado.

- La diferencia entre los escenarios simulados para el cálculo del perfil angular de potencia de llegada en la estación móvil para un entorno LoS, radica en los escenarios en donde la estación base se encuentra a un lado de la calle, debido a que el rango del ángulo de llegada en donde se supera el nivel de potencia del usuario seleccionado, aumenta en la dirección en donde se encuentra la estación base, en cambio, para el escenario en donde la estación base está ubicada de frente al extremo de la calle, el rango de ángulo de llegada es simétrico en ambas direcciones.

4.2. RECOMENDACIONES

- Para las simulaciones del programa de balanceo de cargas, se recomienda el uso de computadores con buenas especificaciones, debido a que el tiempo de simulación del programa depende directamente de la velocidad de procesamiento del computador. Los resultados obtenidos se simularon en un computador con las siguientes características: Procesador Intel Core i7-7500U, sistema operativo de 64 bits, RAM 8 GB.
- Para una mayor exactitud en el proceso de obtención de la variación de margen de *handover*, se recomienda una mayor optimación en la presentación de resultados gráficos para poder realizar un mayor número de simulaciones por escenario y para que el algoritmo desarrollado no ocupe demasiados recursos del computador.
- Se recomienda que, en los escenarios de máxima capacidad de sobrecarga en la celda central, se omita o comente las líneas de código correspondientes a la representación gráfica de los resultados, debido a que el computador empezó a experimentar problemas de rendimiento y varios colapsos al momento de simular estos escenarios.
- Se recomienda incluir un contador que calcule el tiempo que el algoritmo tarda en determinar la variación de margen de *handover* con la que se alcanzó la máxima recompensa del sistema, ya que el algoritmo actual solo cuenta con un contador que indica el tiempo de procesamiento total hasta que el margen de *handover* llegue a 12 dB, de esta forma se podrá observar la diferencia de tiempo de procesamiento entre diferentes valores de paso de variación del margen de *handover*.
- Para el desarrollo de trabajos futuros, se recomienda incluir un script que calcule de forma automática el valor de radio de cobertura de las celdas, a partir de los valores ingresados para los parámetros de entrada y para un porcentaje de *handovers* exitosos dado.

- Se recomienda el uso de scripts para la automatización de varias simulaciones y el cálculo del promedio de sus resultados, lo cual optimizaría en gran manera el tiempo de simulación, debido a que se puede establecer una gran cantidad de escenarios.
- Para el desarrollo de trabajos futuros, se recomienda el desarrollo de scripts que incluyan el método de acceso múltiple aplicado en LTE, y además se tome en cuenta alguno de los métodos de asignación de recursos del canal de radio, por ejemplo, Round Robin, para la determinación de la capacidad de la red en la que se aplique este tipo de técnicas de balanceo de carga.
- Para la inclusión del algoritmo desarrollado en una red activa, se recomienda la simulación de un escernario en donde no solo la celda central se encuentre sobrecargada, sino cualquiera de las celdas de la red, brindando una herrameienta que apartir del proceso de machine learning permita el ajuste de la variación de margen de *handover* de forma dinámica en varias estaciones base del arreglo celular.
- Para trabajos futuros, se recomienda realizar una simulación en donde se pueda determinar la variación de margen de *handover* óptimo cuando las estaciones móviles se encuentren en movimiento, de esta forma el algoritmo se ejecutará mientras se represente de forma gráfica el movimiento de los usuarios durante todo el tiempo de simulación.
- Se recomienda la inclusión del contenido del anexo 2 de la recomendación UIT-R P.1816-4, para poder observar y calcular el perfil angular de potencia de llegada en la estación base y complementar el análisis de la incidencia de estas señales en el lado del transmisor.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] J. Chavarría, “LTE *Handover* Performance Evaluation Based on Power Budget *Handover* Algorithm”, Cataluña: Universitat Politécnica de Catalunya, 2014.

[2] T. Guerra, “*Machine Learning* Based *Handover* Management for LTE *Network*s with Coverage Holes”, Río Grande: Universidade Federal do Rio Grande do Norte, 2018.

[3] C. Lin, K. Sandrasegaran, H. Mohd Ramli, “Optimized Performance Evaluation of LTE *Hard Handover* Algorithm With Average RSRP Constraint”, Sydney: International Journal of Wireless & Mobile *Network*s (IJWMN) Vol. 3, No. 2, 2011.

[4] T, Stephen. S. Mwanje, “A *Q-Learning* Strategy for LTE Mobility *Load*”, 2013 IEEE 24th International Symposium in personal, indoor and Mobile Radio Communications, 2013.

[5] 3GPP, "3rd Generation Partnership Project”, “*Release* 8”, 2008. [En línea]. Disponible: https://www.etsi.org/deliver/etsi_ts/123500_123599/123509/08.00.00_60/ts_123509v080000p.pdf

[6] Z. Becvar, J. Zelenka, R. Bestak, “Comparison of *Handovers* In Umts And Wimax”, Praga: Czech Technical University in Prague, Department of Telecommunication Engineering, 2016.

[7] R. Agusti, F. Bernardo, F. Casadevall, R. Ferrús, “LTE: Nuevas Tendencias en Comunicaciones Móviles”, España: Fundación Vodafone, 2010.

[8] W. Benaatou, A. Latif, “Study of the Vertical Handoff in Heterogeneous *Network*s and Implement Based On Opnet”, Marrakech: University Cadi Ayyad, 2013.

[9] M. Thakkar, L. Agrawal, A. Rangisetti, B. Tamma, “Reducing ping-pong *handovers* in LTE by using A1-based measurements”, Chennai: Twenty-third National Conference on Communications (NCC), 2017.

[10] Y. Yang, “Optimization of *Handover* Algorithms within 3GPP LTE”, KTH, 2009.

[11] D. Aziz, R. Sigle, “Improvement of LTE *Handover* Performance through Interference Coordination”, Stuttgart: Alcatel-Lucent Bell Labs, 2009.

[12] C. Lin, “*Handover* Mechanisms in 3GPP *Long Term Evolution* (LTE) “, Sydney: University of Technology, 2013.

[13] H. Yang, “Differential Fade Margin and Handoff Gain in Wireless Cellular *Network*s”, Nueva Jersey: Nokia Bell Labs, 2012.

[14] H, Meza. L. Santín, “Estudio y diseño de una celda base de acceso inalámbrico con tecnología 4G LTE para el sector Iñaquito de la ciudad de Quito”, Quito: Escuela Politécnica Nacional del Ecuador, 2014. [En línea]. Disponible: http://bibdigital.epn.edu.ec/handle/15000/7387

[15] K. Dimou, M. Wang, Y. Yang, M. Kazmi, “*Handover* within 3GPP LTE: Design Principles and Performance”, Stockholm: Ericsson Research, 2009.

[16] C, Astudillo. T, Andrade. N. Fonseca, “Random Access Mechanism for RAN Over*load* Control in LTE/LTE-A *Network*s”, IEEEICC 2015 - Communications QoS, Reliability and Modeling Symposium, 2015.

[17] H. Jouini, “Radio Resource Management in LTE *Network*s: *Load Balancing* in Heterogeneous Cellular *Network*s”, Túnez: HAL archives-ouvertes.fr, 2018.

[18] 3GPP, "3rd Generation Partnership Project”, “*Release* 9”, 2010. [En línea]. Disponible: https://www.etsi.org/deliver/etsi_ts/136100_136199/136104/09.04.00_60/ts_136104v090400p.pdf

[19] H. Sauter, “From GSM to LTE-Advanced Pro and 5G, An Introduction to Mobile *Network*s and Mobile Broadband”, Colonia: Wiley, 2017.

[20] S. Hamalainen, H. Sanneck, C. Sartori, “Lte *Self*-Organising *Network*s (SON), *Network* Management Automation for Operational Efficiency”, Wiltshire: Nokia Siemens *Network*s, 2011.

[21] A. Basit, “Dimensioning of LTE *Network* Description of Models and Tool, Coverage and Capacity Estimation of 3GPP *Long Term Evolution* radio interface”, Helsinki: Helsinki University of Technology, 2009.

[22] RECOMENDACIÓN UIT-R P.525-2, “Cálculo de la atenuación en el espacio libre”, UIT-R Sector de Radiocomunicaciones de la Unión Internacional de Telecomunicaciones, 1994. [En línea]. Disponible: https://www.itu.int/dms_pubrec/itu-r/rec/p/R-REC-P.525-4-201908-I!!PDF-S.pdf

[23] D. Ulloa, “Modelo básico de un enlace de un Radioenlace para Costa Rica”, ISSUU, 9 Diciembre 2016. [En línea]. Disponible: https://issuu.com/denisulloa/docs/modelo_b__sico_de_propagaci__n_para. [Último acceso: 8 Julio 2020]

[24] RECOMENDACIÓN UIT-R P.1057-6, “Distribuciones de probabilidad para establecer modelos de propagación de las ondas radioeléctricas”, UIT-R Sector de Radiocomunicaciones de la Unión Internacional de Telecomunicaciones, 2019. [En línea]. Disponible: https://www.itu.int/dms_pubrec/itu-r/rec/p/R-REC-P.1057-6-201908-I!!PDF-S.pdf

[25] A. García, “Función de distribución de Rayleigh”, Universidad del País Vasco, 2016. [En línea]. Disponible: http://www.sc.ehu.es/sbweb/fisica3/datos/viento/estadistica_1.html. [Último acceso: 9 Agosto 2020]

[26] F. Ramos, “Radioenclaces Tecnologías inalámbricas y diseño de radioenlaces”, www.radioenlaces.es, 2019. [En línea]. Disponible: http://www.radioenlaces.es/articulos/*tag*/sensibilidad/. [Último acceso: 12 Julio 2020].

[27] “RECEPTORES DE RF”, Universidad Tecnológica Nacional de Argentina, [En línea]. Disponible: http://www1.frm.utn.edu.ar/aplicada3/apuntes/unidad8.pdf. [Último acceso: 27 Junio 2020].

[28] “Intensidad y nivel de la señal en redes móviles 2G, 3G y 4G”, norfipc.com, 2020. [En línea]. Disponible: https://norfipc.com/redes/intensidad-nivel-senal-redes-moviles-2g-3g-4g.php. [Último acceso: 22 Agosto 2020].

[29] M. Luckert, M. Schaefer-Kehnert, “Using *Machine Learning* Methods for Evaluating the Quality of Technical Documents”, 2015. [En línea]. Disponible: https://www.diva-portal.org/smash/get/diva2:920202/FULLTEXT01.pdf.

[30] R. Sutton, A. Barto, “Reinforcement *Learning*: An Introduction”, Londres: Bradford, 2015.

[31] F. Polo, “Aprendizaje por Refuerzo para la Toma de Decisiones Segura en Dominios con Espacios de Estados y Acciones Continuos”, Leganés: UNIVERSIDAD CARLOS III DE MADRID, 2012.

[32] E. Morales, “Aprendizaje por Refuerzo”, Puebla: Instituto Nacional de Astrofísica, Óptica y Electrónica.

[33] J. Li, “Reinforcement *Learning*: Introduction to Q *Learning*”, Medium.com, 30 Julio 2018. [En línea]. Disponible: https://medium.com/@fin.techology/reinforcement-*learning*-introduction-to-*q*-*learning*-444c951e292c. [Último acceso: 8 Julio 2020].

[34] G. Fernández, F. Gallego, “Ajustando *Q-Learning* para generar jugadores automáticos: un ejemplo basado en Atari Breakout”, Alicante: Universidad de Alicante.

[35] M. Silva, “Aprendizaje por Refuerzo: Procesos de Decisión de Markov — Parte 1”, Medium.com, 30 Mayo 2019. [En línea]. Disponible: https://medium.com/aprendizaje-por-refuerzo-introducci%C3%B3n-al-mundo-del/aprendizaje-por-refuerzo-procesos-de-decisi%C3%B3n-de-markov-parte-1-8a0aed1e6c59. [Último acceso: 02 Junio 2020].

[36] Recomendación UIT-R P.1816-4 ANEXO 3, “Predicción de los perfiles de tiempo y de espacio para los servicios móviles terrestres espacio para los servicios móviles terrestres ondas decimétricas y centimétricas”, Ginebra: UIT-R Sector de Radiocomunicaciones de la Unión Internacional de Telecomunicaciones, 2019. [En línea]. Disponible: https://www.itu.int/dms_pubrec/itu-r/rec/p/R-REC-P.1816-4-201908-I!!PDF-S.pdf

[37] I. de la Bandera, P. Munoz, R. Barco, M. Toril, “Auto-ajuste del margen de *handover* en redes LTE”, Malaga: Departamento de Ingeniería de Comunicaciones. Universidad de Málaga, 2014.

[38] “diary”, The MathWorks, Inc., 2020. [En línea]. Disponible: https://la.mathworks.com/help/matlab/ref/diary.html. [Último acceso: 9 Agosto 2020].

[39] Huawei Industrial Base, “In-Building Solution Products”, Shenzhen: HUAWEI TECHNOLOGIES CO., LTD., 2011.

[40] 3GPP, "3rd Generation Partnership Project”, “*Release* 10”, 2011. [En línea]. Disponible: https://www.etsi.org/deliver/etsi_ts/136100_136199/136101/10.03.00_60/ts_136101v100300p.pdf

[41] G. Antonini, A. Orlandi, S. D’elia, “Shielding Effects of Reinforced Concrete Structures to Electromagnetic Fields due to GSM and UMTS Systems”, Torino: IEEE TRANSACTIONS ON MAGNETICS, VOL. 39, NO. 3, Mayo 2003.

[42] “Difference Between Cell "Signal Strength" And "Signal Quality"“, signalbooster.com, 2020. [En línea]. Disponible: https://www.signalbooster.com/pages/difference-between-cell-signal-strength-and-signal-quality. [Último acceso: 25 Septiembre 2020].

[43] GL Communications Inc., “WIRELESS 4G CORE *NETWORK* IP *Network* Simulation”, 2020. [En línea]. Disponible: https://www.gl.com/lte-x2-application-protocol-

testing-maps.html#:~:text=The%20X2%20is%20the%20interconnecting,(Stream%20Control%20Transmission%20Protocol).. [Último acceso: 23 Julio 2020].

[44] 4G 5G World, “What is EUTRAN?”, [En línea]. Disponible: http://4g5gworld.com/ltefaq/what-eutran. [Último acceso: 19 Julio 2020].

[45] S. Chapra, “Métodos Numericos para Ingeniería”, Michigan: Mac Graw Hill., 2011.

[46] A. Printista, M. Errecalde, C. Montoya, “Una implementación paralela del algoritmo de *Q-Learning* basada en un esquema de comunicación con caché”, San Luis: Universidad Nacional de San Luis, 2020.

6. ANEXOS

ANEXO A. Diagramas de flujo de los procesos realizados.

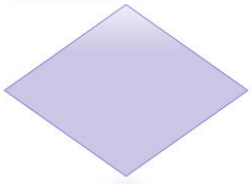
ANEXO B. Código de cada programa desarrollado en MATLAB.

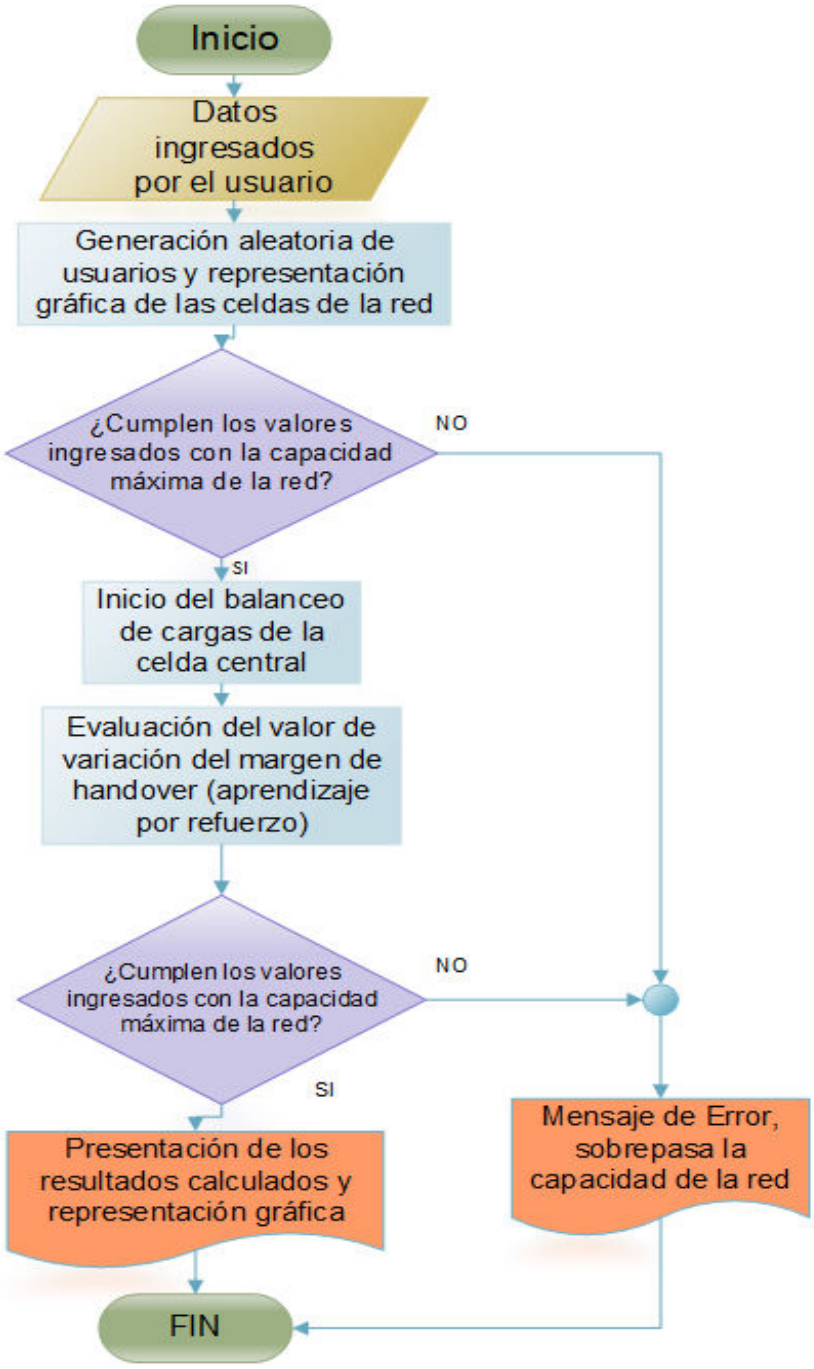
ANEXO C. Resultados de las simulaciones realizadas para cada escenario del programa de balanceo de cargas.

ANEXO A

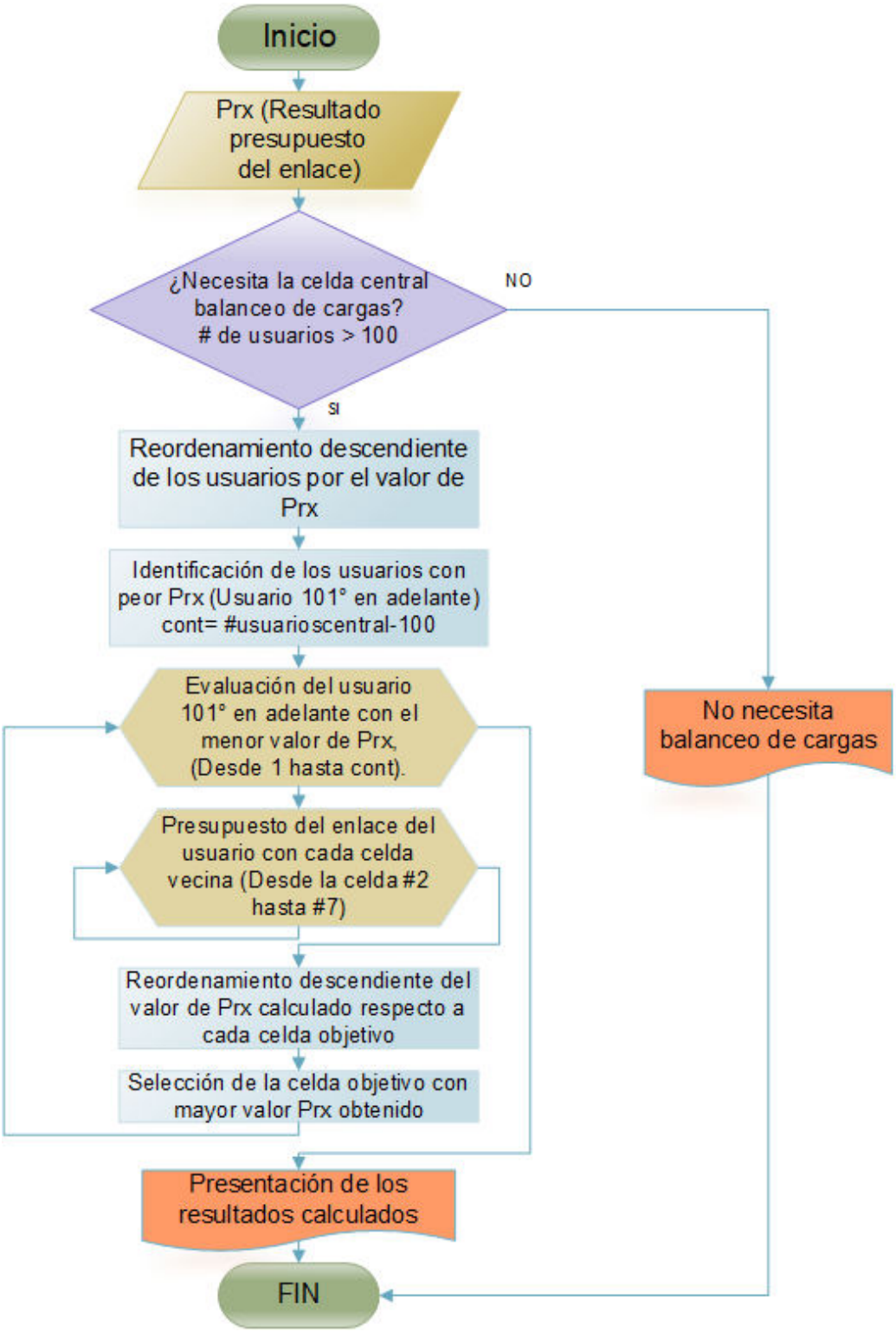
Este anexo contiene la simbología utilizada en los diagramas de flujo para las funciones utilizadas en el presente trabajo de titulación. La nomenclatura utilizada por los diagramas se detalla en la Tabla A.1.

Tabla A.1. Nomenclatura de los diagramas de flujo.

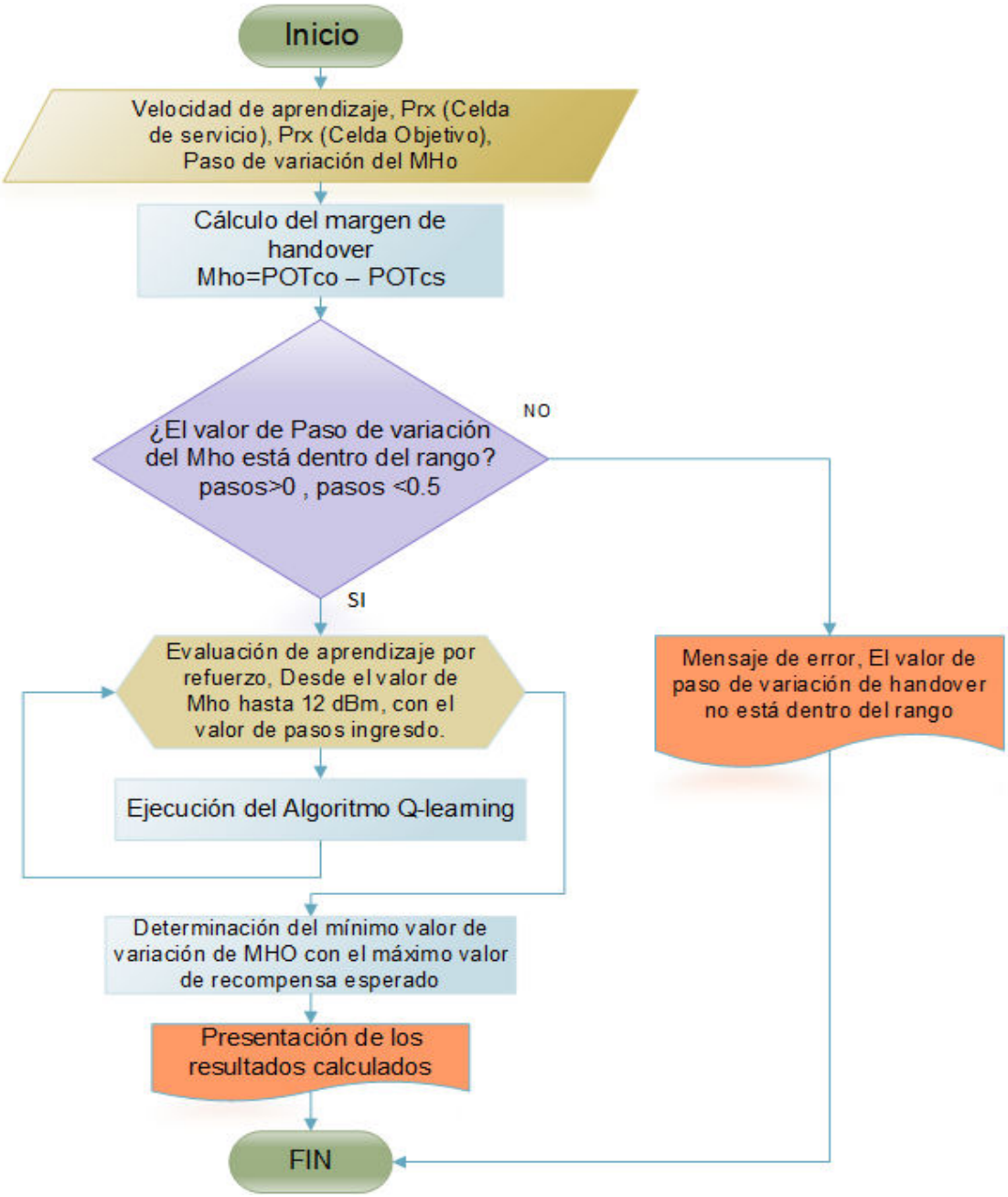
Diagrama	Nomenclatura
	Inicio o Fin
	Entrada de variables o funciones
	Representación gráfica, mensajes de error y salida de variables o funciones
	Operación o procesos
	Decisión
	Iteraciones o Lazos



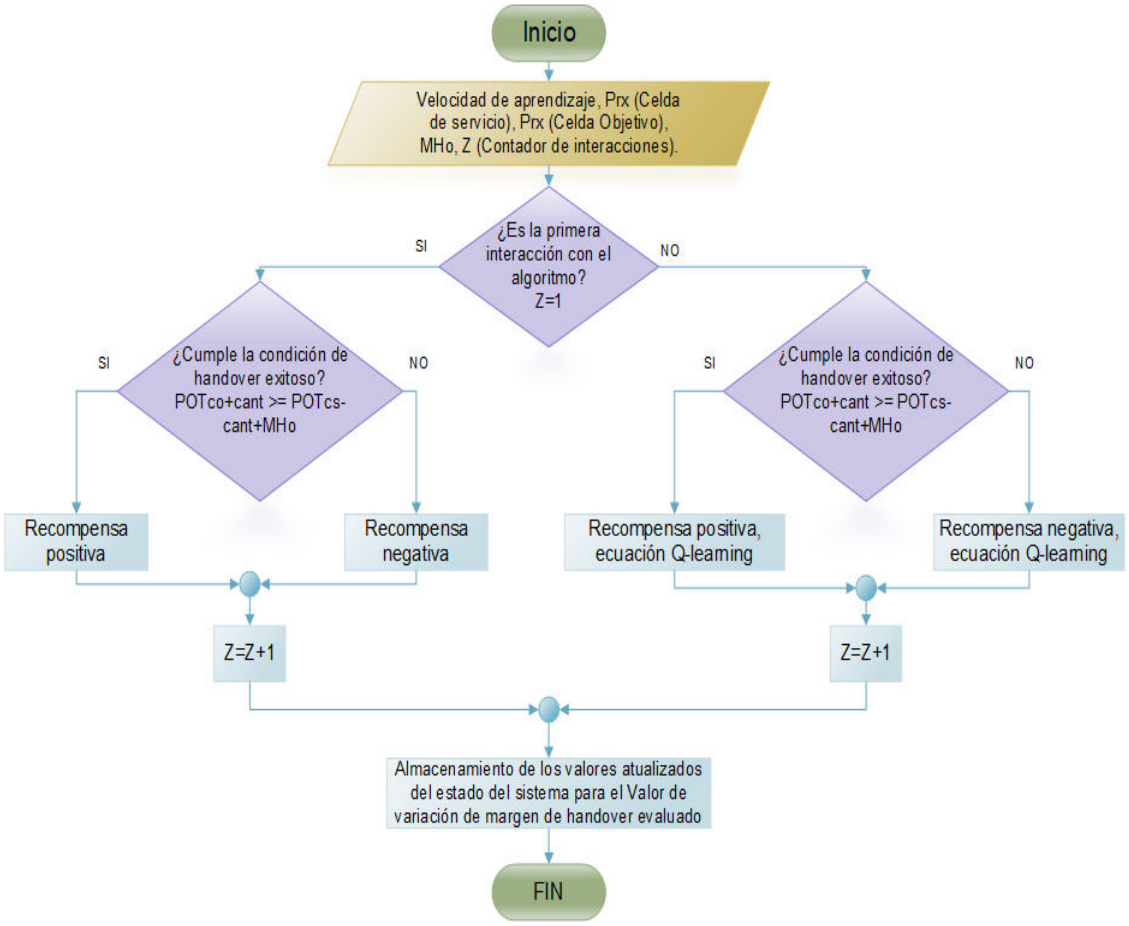
Anexo A.1. Diagrama de flujo del programa de balanceo de carga.



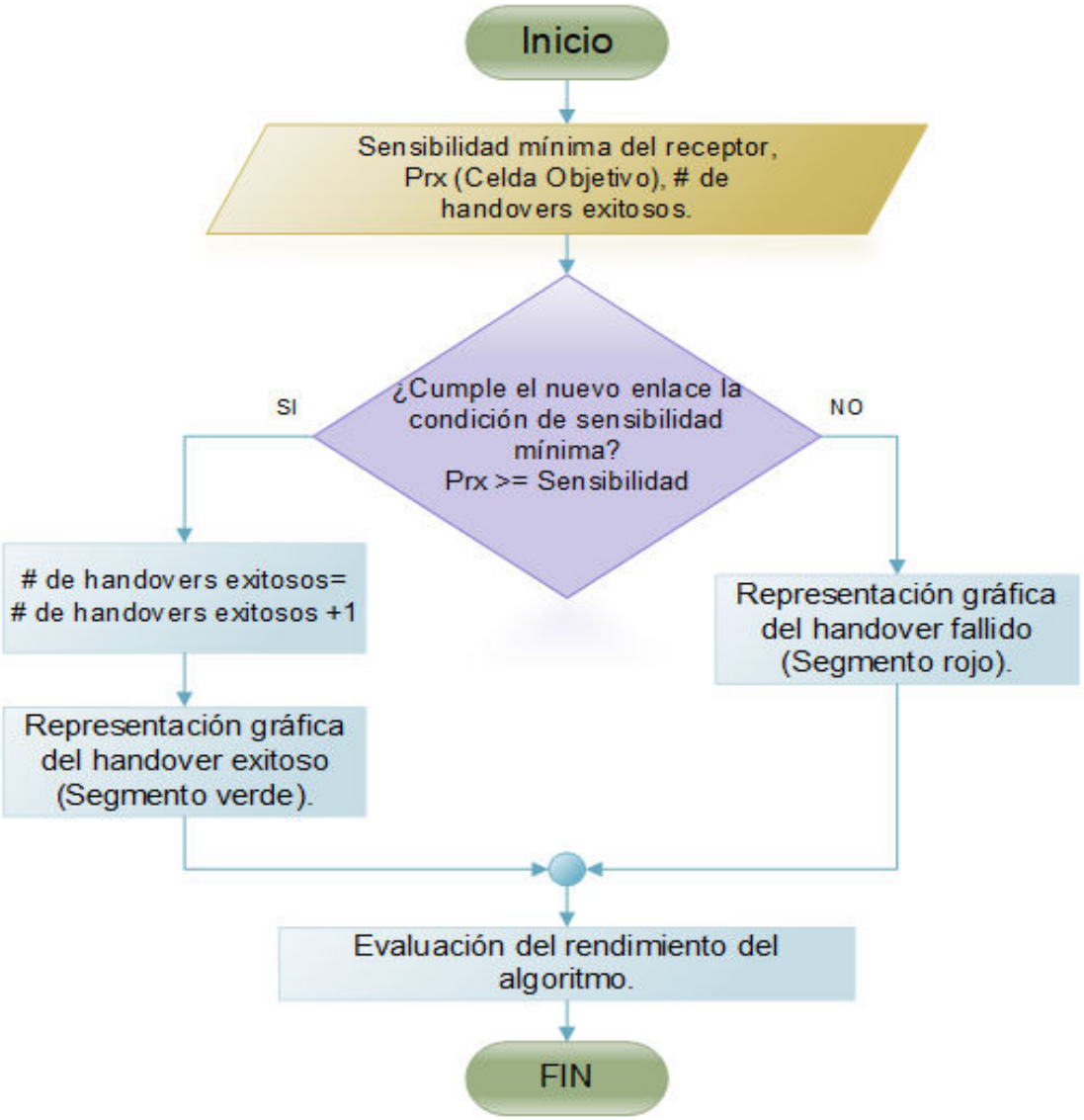
Anexo A.2. Diagrama de flujo del proceso de *handover* de la celda sobrecargada.



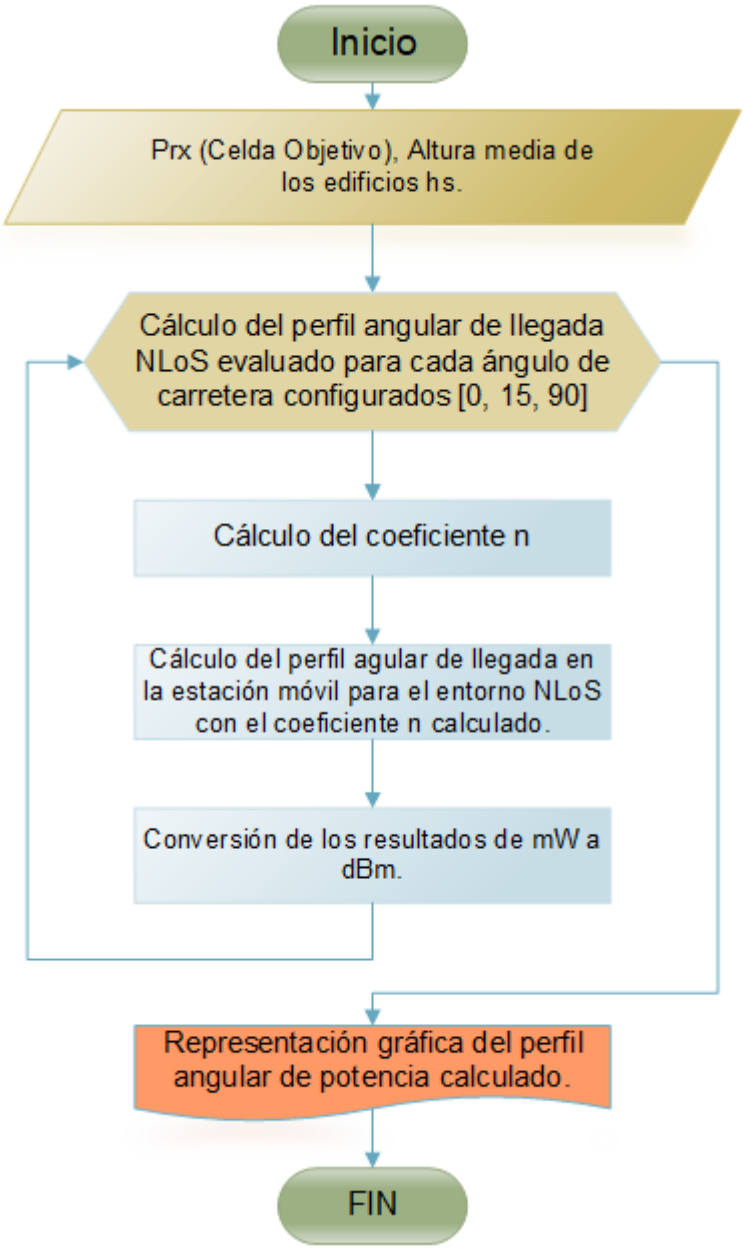
Anexo A.3 Diagrama de flujo del proceso basado en *machine learning*.



Anexo A.4. Diagrama de flujo del proceso de ejecución del algoritmo *Q-learning*



Anexo A.5. Diagrama de flujo de la condición para *handover* exitoso



Anexo A.6. Diagrama de flujo del cálculo del perfil angular de potencia de llegada en la estación móvil para un entorno NLoS.

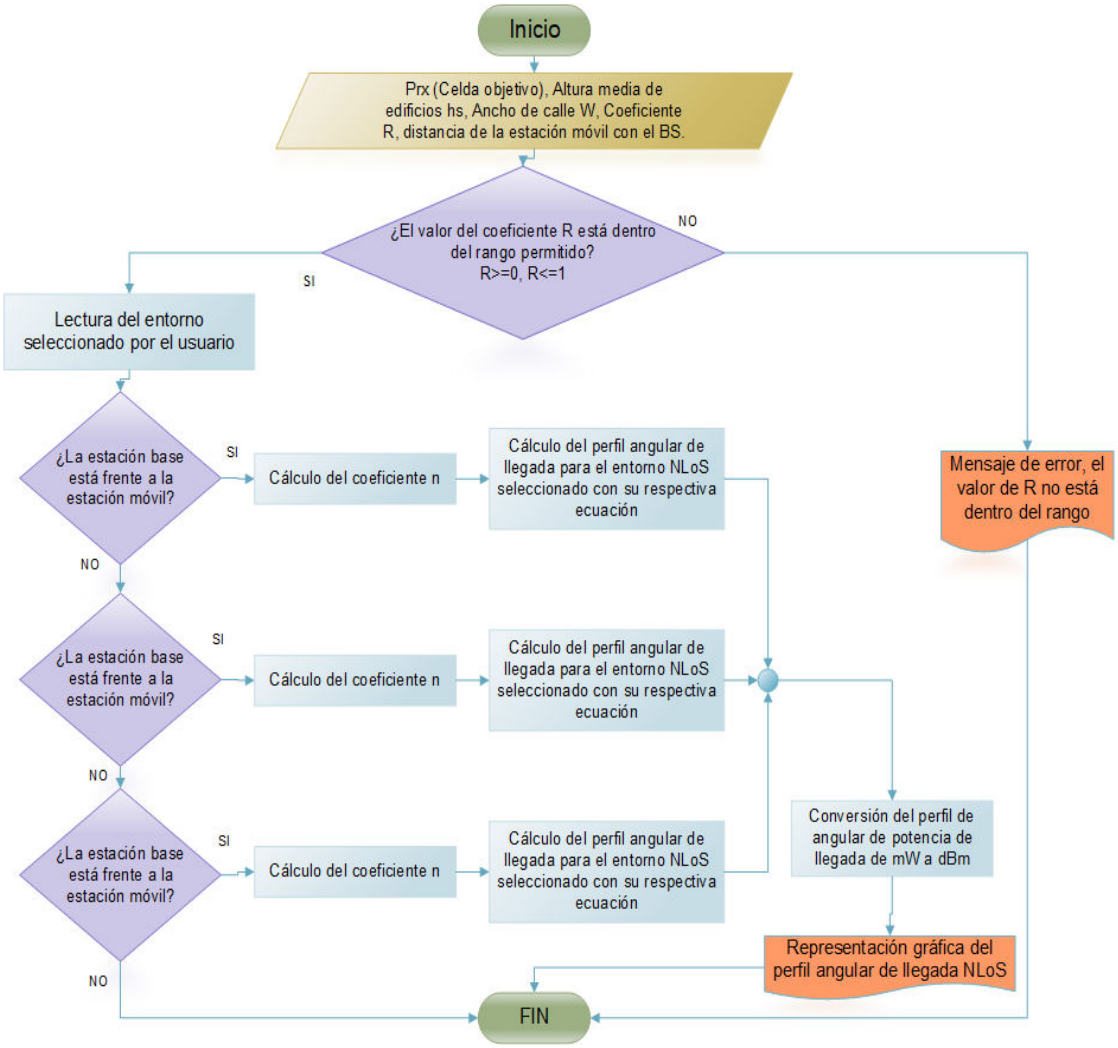


Figura A.7. Diagrama de flujo del cálculo del perfil angular de potencia de llegada en la estación móvil para un entorno LoS.

ANEXO B

Este anexo contiene el código de cada interfaz creado para este proyecto de titulación.

Anexo B.1. Código de la interfaz “presentación”.

Anexo B.2. Código de la interfaz “programa”.

Anexo B.3. Código de la función “presupuesto”.

Anexo B.4. Código de la interfaz “results”.

Anexo B.5. Código de la interfaz “Perfil_Angular”.

Anexo B.1. Código de la interfaz “presentación”.

```
function varargout = presentacion(varargin)
%-------------------------------------------------------------------------
% SIMULACIÓN EN MATLAB DE UN ALGORITMO BASADO EN MACHINE
% LEARNING Y HARD HANDOVER PARA EL BALANCEO DE CARGA EN UNA
% RED CELULAR LTE
% Trabajo de Titulación previo a la obtención del Título de Ingeniero en
% Electrónica y Telecomunicaciones.
% Realizado por: Carlos Alberto Parreño Muirragui
% Director: PhD. Pablo Lupera Morillo
% Descripción de la interfaz: Esta interfaz es la encargada de dirigir a
% la interfaz de balanceo de cargas.
%-------------------------------------------------------------------------
% PRESENTACION MATLAB code for presentacion.fig
%      PRESENTACION, by itself, creates a new PRESENTACION or raises the
existing
%      singleton*.
%
%      H = PRESENTACION returns the handle to a new PRESENTACION or the
handle to
%      the existing singleton*.
%
%      PRESENTACION('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in PRESENTACION.M with the given input
arguments.
%
%      PRESENTACION('Property','Value',...) creates a new PRESENTACION or
raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before presentacion_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to presentacion_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help presentacion

% Last Modified by GUIDE v2.5 06-Jul-2020 01:09:49

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @presentacion_OpeningFcn, ...
                   'gui_OutputFcn',  @presentacion_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before presentacion is made visible.
function presentacion_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to presentacion (see VARARGIN)

% Choose default command line output for presentacion
handles.output = hObject;
axes(handles.axes1);
imshow('EPN.jpg');
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes presentacion wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = presentacion_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in iniciar.
function iniciar_Callback(hObject, eventdata, handles)
Balanceo_Cargas();


% --- Executes during object creation, after setting all properties.
function axes1_CreateFcn(hObject, eventdata, handles)
```

Anexo B.2. Código de la interfaz “Balanceo_Cargas”.

```
function varargout = Balanceo_Cargas(varargin)
%-------------------------------------------------------------------------
% SIMULACIÓN EN MATLAB DE UN ALGORITMO BASADO EN MACHINE
% LEARNING Y HARD HANDOVER PARA EL BALANCEO DE CARGA EN UNA
% RED CELULAR LTE
% Trabajo de Titulación previo a la obtención del Título de Ingeniero en
% Electrónica y Telecomunicaciones.
% Realizado por: Carlos Alberto Parreño Muirragui
% Director: PhD. Pablo Lupera Morillo
% Descripción de la interfaz: Esta interfaz realiza la simulación del
% proceso de balanceo de cargas y el cálculo de la variación de margen de
% handover basado en el algoritmo Q-learning
%-------------------------------------------------------------------------
% BALANCEO_CARGAS MATLAB code for Balanceo_Cargas.fig
%      BALANCEO_CARGAS, by itself, creates a new BALANCEO_CARGAS or
raises the existing
%      singleton*.
%
%      H = BALANCEO_CARGAS returns the handle to a new BALANCEO_CARGAS or
the handle to
%      the existing singleton*.
%
%      BALANCEO_CARGAS('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in BALANCEO_CARGAS.M with the given input
arguments.
%
%      BALANCEO_CARGAS('Property','Value',...) creates a new
BALANCEO_CARGAS or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before Balanceo_Cargas_OpeningFcn gets called.
An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to Balanceo_Cargas_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Balanceo_Cargas

% Last Modified by GUIDE v2.5 22-Dec-2020 12:46:08

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Balanceo_Cargas_OpeningFcn, ...
                   'gui_OutputFcn',  @Balanceo_Cargas_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
```

```
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Balanceo_Cargas is made visible.
function Balanceo_Cargas_OpeningFcn(hObject, eventdata, handles,
varargin)
% Choose default command line output for Balanceo_Cargas
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% --- Outputs from this function are returned to the command line.
function varargout = Balanceo_Cargas_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
clc;
diary resultados.txt %Creación de un archivo de texto que almecena los
resultados de la ventana de comandos
diary on

%%
%Obtención de los valores ingresados por el usuario
f=str2num(get(handles.f,'string'));          %Frecuencia de portadora
Ptx=str2num(get(handles.Ptx,'string'));      %Potencia de transmisión de
la estación base
Gtx=str2num(get(handles.Gtx,'string'));      %Ganancia de transmisión de
la antena de la estación base
Ltx=str2num(get(handles.Ltx,'string'));      %Pérdidas de transmisión
Grx=str2num(get(handles.Grx,'string'));      %Ganancia de recepción de la
antena de la estación móvil
Lrx=str2num(get(handles.Lrx,'string'));      %Pérdidas de recepción
Ghm=str2num(get(handles.Ghm,'string'));      %Ganancia de handover
Sen=str2num(get(handles.Sen,'string'));      %Sensibilidad de receptor
r=str2num(get(handles.r,'string'));          %Obtención valor de radio de
la celda celular
pasos=str2num(get(handles.pasos,'string')); %Obtención valor de variación
del Margen de Handover
learning=str2num(get(handles.learning,'string'));     %Velocidad de
Aprendizaje
```

```
%%
%En caso de realizar múltiples simulaciones utilizar las siguientes
líneas de código

%repeticion=1;

%for learning=0.5:0.1:0.6
%for learning=0.7:0.1:0.8
%for learning=0.9:0.09:0.99

%for pasos=0.1:0.1:0.5
%for repeticiones=1:10
%fallos=0;

%%
%Obtención Número de usuarios de cada celda
usuarioscentral=str2num(get(handles.usuarioscentral,'string')); %Número
de usuarios de la celda central #1
usuarios(2)=str2num(get(handles.usuarios2,'string'));           %Número
de usuarios de la celda #2
usuarios(3)=str2num(get(handles.usuarios3,'string'));           %Número
de usuarios de la celda #3
usuarios(4)=str2num(get(handles.usuarios4,'string'));           %Número
de usuarios de la celda #4
usuarios(5)=str2num(get(handles.usuarios5,'string'));           %Número
de usuarios de la celda #5
usuarios(6)=str2num(get(handles.usuarios6,'string'));           %Número
de usuarios de la celda #6
usuarios(7)=str2num(get(handles.usuarios7,'string'));           %Número
de usuarios de la celda #7
usuarios(1)=usuarioscentral;



%%
%CREACIÓN DE LA CELDA SOBRECARGADA

N=6; %NÚMERO DE LADOS, CELDA HEXAGONAL

cx(1)=0;    %Desfase del centro de la celda central en el eje X
cy(1)=0;    %Desfase del centro de la celda central en el eje Y

t = (1/(2*N):1/N:1)*2*pi;   % Traza un polígono de N lados
x = r*sin(t)+cx(1);         % Valores de los vértices de la celda en el
eje X
y = r*cos(t)+cy(1);         % Valores de los vértices de la celda en el
eje Y
x(7)=x(1);                  % Unión del primer y último vértice, eje X
y(7)=y(1);                  % Unión del primer y último vértice, eje Y

axes(handles.axes1)             %Primera Gráfica
hold on
grid on
plot(x,y,'b');                  %Gráfica de los segmentos de la celda
text(cx(1),cy(1),'Celda #1');   %# de la celda escrita en el centro de la
celda
axis square;                    %Cuadrícula
```

```
axes(handles.axes2)             %Segunda Gráfica
hold on
grid on
plot(x,y,'b');                  %Gráfica de los segmentos de la celda
text(cx(1),cy(1),'Celda #1');   %# de la celda escrita en el centro de la
celda
axis square;                    %Cuadrícula

%%
%GENERACION DE LOS PUNTOS(USUARIOS) ALEATORIAMENTE EN LA CELDA
SOBRECARGADA
for i=1:usuarioscentral         %Inicio del lazo, desde 1 hasta el # de
usuarios ingresado por el usuario
    puntos=0;                   %Configuración inicial parámetro puntos
    while ~puntos               %Condición que realiza la acción mientas
el valor no sea 0
        usuariocenX(i)=rand(1)*2*r-r; %Generación de valores aleatorios
en el eje X dentro del límite de la celda central
        usuariocenY(i)=rand(1)*2*r-r; %Generación de valores aleatorios
en el eje Y dentro del límite de la celda central
        puntos=inpolygon(usuariocenX(i),usuariocenY(i),x,y); %Retorno de
los valores en los ejes X y Y dentro de los límites de la celda
        numero(i)=i;            %Almacenamiento del # de usuario creado
    end
end

%%
%REPRESENTACIÓN GRÁFICA DE LOS USUARIOS EN LA CELDA CENTRAL
axes(handles.axes1)                     %Primera Gráfica
plot (usuariocenX,usuariocenY,'b+');    %Gráfica de los valores de la
posición de los usuarios en la celda central
axes(handles.axes2)                     %Segunda Gráfica
plot (usuariocenX,usuariocenY,'b+');    %Gráfica de los valores de la
posición de los usuarios en la celda central

%%
%CREACIÓN DE LAS CELDAS VECINAS OBJETIVO Y SUS USUARIOS
cx(2)=0;        %Desfase del centro de la celda #2 en el eje X
cy(2)=y(1)*2;   %Desfase del centro de la celda #2 en el eje Y

t = (1/(2*N):1/N:1)*2*pi;   % Traza un polígono de N lados
u = r*sin(t)+cx(2);         % Valores de los vértices de la celda en el
eje X
v = r*cos(t)+cy(2);         % Valores de los vértices de la celda en el
eje Y
u(7)=u(1);                  % Unión del primer y último vértice, eje X
v(7)=v(1);                  % Unión del primer y último vértice, eje Y

axes(handles.axes1)             %Primera Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(2),cy(2),'Celda #2')    %# de la celda escrita en el centro de la
celda
axis square;                    %Cuadrícula

axes(handles.axes2)             %Segunda Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(2),cy(2),'Celda #2')    %# de la celda escrita en el centro de la
celda
```

```
axis square;                    %Cuadrícula

%GENERACIÓN Y REPRESENTACIÓN GRÁFICA DE LOS USUARIOS Y ESTACIÓN BASE
d=0;                                        %Variable que almacenará el
valor de la posición de los usarios eje X
s=0;                                        %Variable que almacenará el
valor de la posición de los usarios eje Y
for i=1:usuarios(2)                         %Inicio del lazo, desde 1
hasta el # de usuarios ingresado por el usuario
    puntos=0;                               %Configuración inicial
parámetro puntos
    while ~puntos                           %Condición que realiza la
acción mientas el valor no sea 0
       d(i)=rand(1)*2*r-r+cx(2);            %Generación de valores
aleatorios en el eje X dentro del límite de la celda #2
       s(i)=rand(1)*2*r-r+cy(2);            %Generación de valores
aleatorios en el eje Y dentro del límite de la celda #2
       puntos=inpolygon(d(i),s(i),u,v);     %Retorno de los valores en
los ejes X y Y dentro de los límites de la celda
    end
end

axes(handles.axes1)     %Primera Gráfica
plot (d,s,'k+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
axes(handles.axes2)     %Segunda Gráfica
plot (d,s,'k+');        %Gráfica de los valores de la posición de los
usuarios en la celda central

%%
%CREACIÓN DE LAS CELDAS VECINAS OBJETIVO Y SUS USUARIOS
cx(3)=0;            %Desfase del centro de la celda #3 en el eje X
cy(3)=y(1)*(-2);    %Desfase del centro de la celda #3 en el eje Y

t = (1/(2*N):1/N:1)*2*pi;   % Traza un polígono de N lados
u = r*sin(t)+cx(3);         % Valores de los vértices de la celda en el
eje X
v = r*cos(t)+cy(3);         % Valores de los vértices de la celda en el
eje Y
u(7)=u(1);                  % Unión del primer y último vértice, eje X
v(7)=v(1);                  % Unión del primer y último vértice, eje Y

axes(handles.axes1)             %Primera Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(3),cy(3),'Celda #3')    %# de la celda escrita en el centro de la
celda

axes(handles.axes2)             %Segunda Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(3),cy(3),'Celda #3')    %# de la celda escrita en el centro de la
celda

%GENERACIÓN Y REPRESENTACIÓN GRÁFICA DE LOS USUARIOS Y ESTACIÓN BASE
d=0;                                        %Variable que almacenará el
valor de la posición de los usarios eje X
s=0;                                        %Variable que almacenará el
valor de la posición de los usarios eje Y
for i=1:usuarios(3)                         %Inicio del lazo, desde 1
hasta el # de usuarios ingresado por el usuario
```

```
    puntos=0;                                   %Configuración inicial
parámetro puntos
    while ~puntos                               %Condición que realiza la
acción mientas el valor no sea 0
       d(i)=rand(1)*2*r-r+cx(3);                %Generación de valores
aleatorios en el eje X dentro del límite de la celda #3
       s(i)=rand(1)*2*r-r+cy(3);                %Generación de valores
aleatorios en el eje Y dentro del límite de la celda #3
        puntos=inpolygon(d(i),s(i),u,v);        %Retorno de los valores en
los ejes X y Y dentro de los límites de la celda
    end
end
axes(handles.axes1)     %Primera Gráfica
plot (d,s,'k+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
axes(handles.axes2)     %Segunda Gráfica
plot (d,s,'c+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
%%
%CREACIÓN DE LAS CELDAS VECINAS OBJETIVO Y SUS USUARIOS
ru=y(1)*2;          %Desfase de radio para las celdas 4,5,6,7
cx(4)=ru*cos(pi/6); %Desfase del centro de la celda #4 en el eje X
cy(4)=y(1);         %Desfase del centro de la celda #4 en el eje Y

t = (1/(2*N):1/N:1)*2*pi;   % Traza un polígono de N lados
u = r*sin(t)+cx(4);         % Valores de los vértices de la celda en el
eje X
v = r*cos(t)+cy(4);         % Valores de los vértices de la celda en el
eje Y
u(7)=u(1);                  % Unión del primer y último vértice, eje X
v(7)=v(1);                  % Unión del primer y último vértice, eje Y

axes(handles.axes1)             %Primera Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(4),cy(4),'Celda #4')    %# de la celda escrita en el centro de la
celda

axes(handles.axes2)             %Segunda Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(4),cy(4),'Celda #4')    %# de la celda escrita en el centro de la
celda

%GENERACIÓN Y REPRESENTACIÓN GRÁFICA DE LOS USUARIOS Y ESTACIÓN BASE
d=0;                                        %Variable que almacenará el
valor de la posición de los usuarios eje X
s=0;                                        %Variable que almacenará el
valor de la posición de los usuarios eje Y
for i=1:usuarios(4)                         %Inicio del lazo, desde 1
hasta el # de usuarios ingresado por el usuario
    puntos=0;                               %Configuración inicial
parámetro puntos
    while ~puntos                           %Condición que realiza la
acción mientas el valor no sea 0
       d(i)=rand(1)*2*r-r+cx(4);            %Generación de valores
aleatorios en el eje X dentro del límite de la celda #4
       s(i)=rand(1)*2*r-r+cy(4);            %Generación de valores
aleatorios en el eje Y dentro del límite de la celda #4
        puntos=inpolygon(d(i),s(i),u,v);    %Retorno de los valores en
los ejes X y Y dentro de los límites de la celda
```

```
    end
end
axes(handles.axes1)     %Primera Gráfica
plot (d,s,'k+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
axes(handles.axes2)     %Segunda Gráfica
plot (d,s,'m+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
%%
%CREACIÓN DE LAS CELDAS VECINAS OBJETIVO Y SUS USUARIOS
cx(5)=ru*cos(pi/6); %Desfase del centro de la celda #5 en el eje X
cy(5)=-y(1);        %Desfase del centro de la celda #5 en el eje Y

t = (1/(2*N):1/N:1)*2*pi;   % Traza un polígono de N lados
u = r*sin(t)+cx(5);         % Valores de los vértices de la celda en el
eje X
v = r*cos(t)+cy(5);         % Valores de los vértices de la celda en el
eje Y
u(7)=u(1);                  % Unión del primer y último vértice, eje X
v(7)=v(1);                  % Unión del primer y último vértice, eje Y

axes(handles.axes1)             %Primera Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(5),cy(5),'Celda #5')    %# de la celda escrita en el centro de la
celda

axes(handles.axes2)             %Segunda Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(5),cy(5),'Celda #5')    %# de la celda escrita en el centro de la
celda

%GENERACIÓN Y REPRESENTACIÓN GRÁFICA DE LOS USUARIOS Y ESTACIÓN BASE
d=0;                                    %Variable que almacenará el
valor de la posición de los usuarios eje X
s=0;                                    %Variable que almacenará el
valor de la posición de los usuarios eje Y
for i=1:usuarios(5)                     %Inicio del lazo, desde 1
hasta el # de usuarios ingresado por el usuario
    puntos=0;                           %Configuración inicial
parámetro puntos
    while ~puntos                       %Condición que realiza la
acción mientas el valor no sea 0
       d(i)=rand(1)*2*r-r+cx(5);        %Generación de valores
aleatorios en el eje X dentro del límite de la celda #5
       s(i)=rand(1)*2*r-r+cy(5);        %Generación de valores
aleatorios en el eje Y dentro del límite de la celda #5
       puntos=inpolygon(d(i),s(i),u,v); %Retorno de los valores en
los ejes X y Y dentro de los límites de la celda
    end
end
axes(handles.axes1)     %Primera Gráfica
plot (d,s,'k+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
axes(handles.axes2)     %Segunda Gráfica
plot (d,s,'y+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
%%
%CREACIÓN DE LAS CELDAS VECINAS OBJETIVO Y SUS USUARIOS
cx(6)=-ru*cos(pi/6);    %Desfase del centro de la celda #6 en el eje X
```

```
cy(6)=y(1);             %Desfase del centro de la celda #6 en el eje Y

t = (1/(2*N):1/N:1)*2*pi;   % Traza un polígono de N lados
u = r*sin(t)+cx(6);         % Valores de los vértices de la celda en el
eje X
v = r*cos(t)+cy(6);         % Valores de los vértices de la celda en el
eje Y
u(7)=u(1);                  % Unión del primer y último vértice, eje X
v(7)=v(1);                  % Unión del primer y último vértice, eje Y

axes(handles.axes1)             %Primera Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(6),cy(6),'Celda #6')    %# de la celda escrita en el centro de la
celda

axes(handles.axes2)             %Segunda Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(6),cy(6),'Celda #6')    %# de la celda escrita en el centro de la
celda

%GENERACIÓN Y REPRESENTACIÓN GRÁFICA DE LOS USUARIOS Y ESTACIÓN BASE
d=0;                                    %Variable que almacenará el
valor de la posición de los usuarios eje X
s=0;                                    %Variable que almacenará el
valor de la posición de los usuarios eje Y
for i=1:usuarios(6)                     %Inicio del lazo, desde 1
hasta el # de usuarios ingresado por el usuario
    puntos=0;                           %Configuración inicial
parámetro puntos
    while ~puntos                       %Condición que realiza la
acción mientas el valor no sea 0
       d(i)=rand(1)*2*r-r+cx(6);        %Generación de valores
aleatorios en el eje X dentro del límite de la celda #6
       s(i)=rand(1)*2*r-r+cy(6);        %Generación de valores
aleatorios en el eje Y dentro del límite de la celda #6
       puntos=inpolygon(d(i),s(i),u,v); %Retorno de los valores en
los ejes X y Y dentro de los límites de la celda
    end
end
axes(handles.axes1)     %Primera Gráfica
plot (d,s,'k+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
axes(handles.axes2)     %Segunda Gráfica
plot (d,s,'r+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
%%
%CREACIÓN DE LAS CELDAS VECINAS OBJETIVO Y SUS USUARIOS
cx(7)=-ru*cos(pi/6);    %Desfase del centro de la celda #7 en el eje X
cy(7)=-y(1);            %Desfase del centro de la celda #7 en el eje Y

t = (1/(2*N):1/N:1)*2*pi;   % Traza un polígono de N lados
u = r*sin(t)+cx(7);         % Valores de los vértices de la celda en el
eje X
v = r*cos(t)+cy(7);         % Valores de los vértices de la celda en el
eje Y
u(7)=u(1);                  % Unión del primer y último vértice, eje X
v(7)=v(1);                  % Unión del primer y último vértice, eje Y

axes(handles.axes1)             %Primera Gráfica
```

```
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(7),cy(7),'Celda #7')    %# de la celda escrita en el centro de la
celda

axes(handles.axes2)             %Segunda Gráfica
plot(u,v,'r');                  %Gráfica de los segmentos de la celda
text(cx(7),cy(7),'Celda #7')    %# de la celda escrita en el centro de la
celda

%GENERACIÓN Y REPRESENTACIÓN GRÁFICA DE LOS USUARIOS Y ESTACIÓN BASE
d=0;                                        %Variable que almacenará el
valor de la posición de los usuarios eje X
s=0;                                        %Variable que almacenará el
valor de la posición de los usuarios eje Y
for i=1:usuarios(7)                         %Inicio del lazo, desde 1
hasta el # de usuarios ingresado por el usuario
    puntos=0;                               %Configuración inicial
parámetro puntos
    while ~puntos                           %Condición que realiza la
acción mientas el valor no sea 0
       d(i)=rand(1)*2*r-r+cx(7);            %Generación de valores
aleatorios en el eje X dentro del límite de la celda #7
       s(i)=rand(1)*2*r-r+cy(7);            %Generación de valores
aleatorios en el eje Y dentro del límite de la celda #7
       puntos=inpolygon(d(i),s(i),u,v);     %Retorno de los valores en
los ejes X y Y dentro de los límites de la celda
    end
end
axes(handles.axes1)     %Primera Gráfica
plot (d,s,'k+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
plot (cx,cy,'b*')       %Gráfica de los centros de todas la celdas
hold off

axes(handles.axes2)     %Segunda Gráfica
plot (d,s,'k+');        %Gráfica de los valores de la posición de los
usuarios en la celda central
plot (cx,cy,'b*')       %Gráfica de los centros de todas la celdas


%%
%CONFIGURACIÓN DE PARÁMETROS INICIALES
exito=0;                %Creación variable que mide el número de
handovers exitosos
recompensa=zeros(2,50); %Creación de matriz de recompensas
operaciones=0;          %Creación del contador de operaciones

%%
%CÁLCULO DEL PRESUPUESTO DEL ENLACE DE LOS USUSARIOS DE LA CELDA
%SOBRECARGADA CON LA ESTACIÓN BASE CENTRAL
timerVal=tic;                           %Inicio de contador de tiempo,
Inicio del proceso de balanceo de cargas
i=1;
for i=1:usuarioscentral;    %Cálculo del presupuesto de enlace para cada
usuario de la celda central
    numero(i)=i;

[Prx,dis]=presupuesto(cx(1),cy(1),usuariocenX(i),usuariocenY(i),f,Ptx,Gtx
,Ltx,Grx,Lrx,Ghm); %Función que calcula el presupuesto de enlace
```

```
    operaciones=operaciones+1;  %Aumento del número de operaciones para
el balanceo de carga
    Potencia(i)=Prx;            %Almacenamiento del valor retornado del
calculo del presupuesto de enlace
    numero(i)=i;                %Almacenamiento del # de usuario de la
celda central
end

%%
%CONDICIONES NECESARIAS PARA EJECUTAR EL BALANCEO DE CARGAS
%Condición que no ejecuta el balanceo de cargas si el número de usuarios
totales ingresados sobrepasa la capacidad del sistema
    if sum(usuarios)>2100
        fprintf("EL # de usuarios es mayor a la capacidad de la red
\nDebe ser menor a 700 \n"); %Mensaje de error en la ventana de comandos
        set(handles.error,'string','ERROR: # de usuarios mayor a 2100
\n');     %Mensaje de error en la ventana de resultados del GUIDE
%Condición que no ejecuta el balanceo de cargas si el número de usuarios
en la celda #2 ingresados sobrepasa su capacidad
    elseif usuarios(2)>300
%Condición que no ejecuta el balanceo de cargas si el número de usuarios
en la celda #3 ingresados sobrepasa su capacidad
    elseif usuarios(3)>300
%Condición que no ejecuta el balanceo de cargas si el número de usuarios
en la celda #4 ingresados sobrepasa su capacidad
    elseif usuarios(4)>300
%Condición que no ejecuta el balanceo de cargas si el número de usuarios
en la celda #5 ingresados sobrepasa su capacidad
    elseif usuarios(5)>300
%Condición que no ejecuta el balanceo de cargas si el número de usuarios
en la celda #6 ingresados sobrepasa su capacidad
    elseif usuarios(6)>300
%Condición que no ejecuta el balanceo de cargas si el número de usuarios
en la celda #7 ingresados sobrepasa su capacidad
    elseif usuarios(7)>300
%Si no cumple ninguna de las anteriores condiciones se ejecuta el
balanceo de cargas
    else
%%
%ORDENAMIENTO DE LOS RESULTADOS DEL PRESUPUESTO DEL ENLACE
%Matriz que contiene los valores: # de usuario, Posiciones en los ejes X
y
%Y y Cáclulo del presupuesto del enlace
datos=[numero' usuariocenX' usuariocenY' Potencia'];

[~, orden] = sort(datos(:, 4),'descend');   %Reordenamiento descendiente
según el valor del cálculo del presupuesto del enlace
format bank                                 %Formato de 2 decimales
fprintf("\n         orden       Pos X       Pos Y       Link budget \n");
resultado=datos(orden, :)                   %Matriz que almacena los
datos ordenados
operaciones=operaciones+1;                  %Aumento del número de
opeaciones para el balanceo de carga
    %%
    %PROCESO DE HANDOVER PARA LA CELDA SOBRECARGADA
    if usuarioscentral > 300            %Condición de celda sobrecargada
        %Se obtienen los valores de # de usuario, Posiciones en los ejes
X
        %y Y y Cálculo del presupuesto del enlace de los usuarios con
menor Prx
```

```
        usuariosOrden=resultado(301:end,[1]);
        usuariosHOx=resultado(301:end,[2]);
        usuariosHOy=resultado(301:end,[3]);
        POTcs=resultado(301:end,[4]);
        cont=usuarioscentral-300;       %Número de usuarios que harán el
handover
        for a=1:cont                    %Acción que se realizará para
cada uno de los usuarios que harán el handover
            for b=2:7                   %Acción que se realizará el
presupuesto de enlace con cada centro de las celdas vecina #2-#7

[Prx,dis]=presupuesto(cx(b),cy(b),usuariosHOx(a),usuariosHOy(a),f,Ptx,Gtx
,Ltx,Grx,Lrx,Ghm);
            c=b-1;                      %Reordenamiento del # de celda
            HOpot(c)=Prx;               %Cálculo del presupuesto del
enlace recibida por la función
            celHO(c)=b;                 %# De la celda objetivo
            operaciones=operaciones+1;  %Aumento del número de
operaciones para el balanceo de carga
            end
        HOpot=[celHO' HOpot'];          %Matriz que contiene el # de
celda y el calculo de presupuesto de enlace de cada una
        [~, orden] = sort(HOpot(:, 2),'descend'); %Reordenamiento
descendiente según el valor del cálculo del presupuesto del enlace
        format bank                     %Formato de 2 decimales
        fprintf("----------------------------------------------------------
------------------------------");
        HOresultado=HOpot(orden, :)     %Matriz que almacena los datos
ordenados
        %Se obtienen los valores de # de usuario, Posiciones en los ejes
X
        %y Y y Cáclulo del presupuesto del enlace del usuario con la
celda objetivo
        celdaObj=HOresultado(1,1);
        POTco=HOresultado(1,2);
        UO=usuariosOrden(a);
        UX=usuariosHOx(a);
        UY=usuariosHOy(a);
        fprintf("El usuario # %d en la Pos en X: %.2f Pos en Y: %.2f
realizara el Ho a la celda # %d \n",UO,UX,UY,celdaObj);
        usuarios(celdaObj)=usuarios(celdaObj)+1;    %Aumenta en uno el #
de usuarios bajo la cobertura de la celda objetivo luego del handover
        operaciones=operaciones+1;                   %Aumento del número
de opeaciones para el balanceo de carga

        %Asignación de los valores obtenidos en una matriz para la
        %presentación de los datos
        usuarioshandover(a,1)=UO;
        usuarioshandover(a,2)=celdaObj;
        usuarioshandover(a,3)=dis;
        usuarioshandover(a,4)=POTco;

        %%
        %CONDICION DE HANDOVER Y RECOMPENSA
        if learning>1 || learning<=0;               %Condición valor de
velocidad de aprendizaje
            fprintf('\n ERROR: EL valor de velocidad de aprendizaje tiene
que estar entre 0 y 1');
            set(handles.ResultadoRecompensa,'string','ERROR: EL valor de
velocidad de aprendizaje tiene que estar entre 0 y 1');
```

```
        else
        filapar=2;                  %Fila que mostrará el valor de offset
de HO
        filaimpar=1;
        %CÁLCULO MARGEN DE HANDOVER
        Mho=POTco-POTcs;            %Margen de Handover sin offset
        if pasos>0 && pasos <=0.5   %Condición que valida pasos de
handover entre valores mayores 0 y 0.5
        z=1;                        %Número de paso de variación de
handover
        offset=0;                   %Variable que llevará la suma de los
pasos (variaciones) ejecutadas
        %%
        %PROCESO DE APRENDIZAJE
            %Condición que realizará el proceso de aprendizaje hasta que
            %llegue al valor máximo de 12 dBm
            for cant=Mho:pasos:12
                offset=offset+pasos;            %Suma del valor del
offset actual mas el valor de paso
                recompensa(filaimpar,z)=offset; %Asignación del valor de
la variación en dB
                if z==1                         %Los siguientes pasos
solo realizan cuando es el primer paso de aprendizaje del agente
                    if POTco+cant>= POTcs-cant+Mho;
%Condición de handover Exitoso
                    recompensa(filapar,z)=recompensa(filapar,z)+10;
%Recompensa postiva
                    else
                    recompensa(filapar,z)=recompensa(filapar,z)-10;
%Recompensa negativa
                    end
                else               %los siguientes pasos se realizarán
desde la segunda interacción
                    if POTco+cant>= POTcs-cant+Mho;
%Condición de handover Exitoso
                    recompensa(filapar,z)=(1-
learning)*recompensa(filapar,z-1)+learning*10;     %Ecuación Q-learning
(Recompensa Positiva)
                    recompensa(filapar,z)=round(recompensa(filapar,z),4);
%Redondeo de cuatro decimales del valor del estado de aprendizaje
                    else
                    recompensa(filapar,z)=(1-
learning)*recompensa(filapar,z-1)+learning*(-10);  %Ecuación Q-learning
(Recompensa Negativa)
                    recompensa(filapar,z)=round(recompensa(filapar,z),4);
%Redondeo de cuatro decimales del valor del estado de aprendizaje
                    end
                end
                z=z+1;                      %Aumento en uno del # de
paso, para la siguiente interacción
                operaciones=operaciones+1;  %Aumento del número de
opeaciones para el balanceo de carga
            end

            operaciones=operaciones+1;          %Aumento del número de
opeaciones para el balanceo de carga
            valorRecompensa=recompensa(2,:);    %Extracción de los
valores de recompensa obtenidos
            %Búsqueda del valor de variación de handover que alcanzó la
            %recompensa máxima para la presentación de resultados
```

```
            posicionRecompensa=find(valorRecompensa==10,1);
            varMHO=recompensa(1,posicionRecompensa);

            if isempty(posicionRecompensa)
                set(handles.ResultadoRecompensa,'string','No se ha
alcanzado la recompensa máxima, aumente la velocidad de aprendizaje!');
            else
                recompensaResultados=sprintf('El valor de variacion de
handover con el que se tiene la maxima recompensa es %.2f dB \n',varMHO);

set(handles.ResultadoRecompensa,'string',recompensaResultados);
            end
        else
            set(handles.ResultadoRecompensa,'string','ERROR: Variación
del MHo no está dentro del rango permitido (0 - 0.5 dB)');
        end
        end
        %%
        %CONDICIÓN DE SOBRECARGA DE CELDAS VECINAS
        %Condición que se ejecuta cuando no se sobrepasó la capacidad
        %máxima de la celda vecina luego del balanceo de cargas
        if usuarios(celdaObj)<=300
        %CONDICIÓN DE SENSIBILIDAD
           if POTco >= Sen
               %# de handovers exitosos aumenta en uno si cumple la
condición de sensibilidad
               exito=exito+1;
               fprintf('Handover Exitoso \n');
               operaciones=operaciones+1;       %Aumento del número de
opeaciones para el balanceo de carga
               %Representación gráfica de un segmento VERDE que une la
posición del usuario
               %con centro de la celda objetivo (BS) indicando el éxito
del
               %handover en la segunda gráfica
               axes(handles.axes2)
               hold on
               P1=[UX UY];P2=[cx(celdaObj) cy(celdaObj)];
               plot([P1(1) P2(1)],[P1(2) P2(2)],'g')
           else %Pasos que se ejecutarán cuando no cumple la condición de
sensibilidad
               fprintf('Handover no Exitoso, no cumple con la
sensibilidad minima \n');
               operaciones=operaciones+1;       %Aumento del número de
opeaciones para el balanceo de carga
               %Representación gráfica de un segmento ROJO que une la
posición del usuario
               %con centro de la celda objetivo (BS) indicando el fallido
               %proceso de handover en la segunda gráfica
               axes(handles.axes2)
               P1=[UX UY];P2=[cx(celdaObj) cy(celdaObj)];
               plot([P1(1) P2(1)],[P1(2) P2(2)],'r')
           end

        %Pasos que se ejecutarán cuando luego del balanceo de cargas una
celda sobrepaso su capacidad
        else
            fprintf('Handover no Exitoso, sobrepasa la cantidad maxima de
la celda (300 usuarios) \n');
            fallos=fallos+1;
```

```
            operaciones=operaciones+1;  %Aumento del número de
operaciones para el balanceo de carga
        end
        HOpot=0;                        %Reset de los valores calculados
        end

        %%%%%%%%%%%%%%%%%%%%%
        %%
        tiempo=toc(timerVal); %Finalización de contador de tiempo, tiempo
total del proceso de balanceo de cargas
        porcentaje=exito/cont*100; %Porcentaje de exitoso del balanceo de
carga
        HOprom=exito/(cont*tiempo); %Promerdio de handovers exitoso por
segundo por usuario

%%
%RESULTADOS

        show=sprintf('Los Handovers exitosos para los %d usuarios de
sobrecarga son %d (%.2f por ciento de exito)\n',cont,exito,porcentaje);
        fprintf("\n-----------------------------------------------------------
------------------------------");
        fprintf("\n--------------------------------RESULTADOS-----------
------------------------\n");
        fprintf('\n Los Handovers exitosos para los %d usuarios de
sobrecarga son %d (%.2f por ciento de exito)\n',cont,exito,porcentaje);
        set(handles.error,'string',show);
        fprintf('\n El Promedio de handovers por unidad movil por segundo
es %.2f \n',HOprom);
        fprintf('\n La velocidad de procesamiento para los %d usuarios
fue de %.2f segundos por %d operaciones \n',cont,tiempo,operaciones);
        fprintf('\n El valor de variacion de handover con el que se tiene
la maxima recompensa es %.2f \n \n',varMHO);
    else
        fprintf('\n La celda no está sobrecargada, no necesita un
balanceo de cargas');
        set(handles.error,'string','La celda no está sobrecargada, no
necesita un balanceo de cargas');
    end

    set(handles.uitable1,'data',recompensa);
    set(handles.pushbutton1,'BackgroundColor','r');
    set(handles.reset,'BackgroundColor',[0, 0.45, 0.74]);
end


%%
%CONDICION DE CUMPLIMIENTO DE CAPACIDAD DE LAS CELDAS VECINAS
for i=2:7 %Iteración que se realiza solo para las celdas vecinas
    if usuarios(i)>300
        fprintf("El balanceo de cargas excede la capacidad de la celda
#%d con %d usuarios \n",i,usuarios(i))
        set(handles.error,'string','El balanceo de cargas excede la
capacidad de una o varias celdas, verifique los resultados')
    end
end

diary off
```

```
%%
%Configuración de las varibales globales para la segunda simulación
setappdata(0,'usuarioshandover',usuarioshandover); %Configuración de
varibale global

%% Código para resultados de múltiples simulaciones
%totales(repeticiones,1)=varMHO;
%totales(repeticiones,2)=porcentaje;
%totales(repeticiones,3)=tiempo;
%totales(repeticiones,4)=HOprom;
%totales(repeticiones,5)=fallos;

%cla(handles.axes1,'reset');              %reset primera gráfica
%cla(handles.axes2,'reset');              %reset segunda gráfica
%recompensa=0;

%end
%learning
%totales

%end
function f_Callback(hObject, eventdata, handles)
% hObject    handle to f (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of f as text
%        str2double(get(hObject,'String')) returns contents of f as a
double


% --- Executes during object creation, after setting all properties.
function f_CreateFcn(hObject, eventdata, handles)
% hObject    handle to f (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Ptx_Callback(hObject, eventdata, handles)
% hObject    handle to Ptx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Ptx as text
%        str2double(get(hObject,'String')) returns contents of Ptx as a
double
```

```
% --- Executes during object creation, after setting all properties.
function Ptx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Ptx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Gtx_Callback(hObject, eventdata, handles)
% hObject    handle to Gtx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Gtx as text
%        str2double(get(hObject,'String')) returns contents of Gtx as a
double


% --- Executes during object creation, after setting all properties.
function Gtx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Gtx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Ltx_Callback(hObject, eventdata, handles)
% hObject    handle to Ltx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Ltx as text
%        str2double(get(hObject,'String')) returns contents of Ltx as a
double


% --- Executes during object creation, after setting all properties.
function Ltx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Ltx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Lm_Callback(hObject, eventdata, handles)
% hObject    handle to Lm (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Lm as text
%        str2double(get(hObject,'String')) returns contents of Lm as a
double


% --- Executes during object creation, after setting all properties.
function Lm_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Lm (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Grx_Callback(hObject, eventdata, handles)
% hObject    handle to Grx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Grx as text
%        str2double(get(hObject,'String')) returns contents of Grx as a
double


% --- Executes during object creation, after setting all properties.
function Grx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Grx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end



function Lrx_Callback(hObject, eventdata, handles)
% hObject    handle to Lrx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Lrx as text
%        str2double(get(hObject,'String')) returns contents of Lrx as a
double


% --- Executes during object creation, after setting all properties.
function Lrx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Lrx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Ghm_Callback(hObject, eventdata, handles)
% hObject    handle to Ghm (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Ghm as text
%        str2double(get(hObject,'String')) returns contents of Ghm as a
double


% --- Executes during object creation, after setting all properties.
function Ghm_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Ghm (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function Sen_Callback(hObject, eventdata, handles)
% hObject    handle to Sen (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Sen as text
%        str2double(get(hObject,'String')) returns contents of Sen as a
double


% --- Executes during object creation, after setting all properties.
function Sen_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Sen (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function r_Callback(hObject, eventdata, handles)
% hObject    handle to r (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of r as text
%        str2double(get(hObject,'String')) returns contents of r as a
double


% --- Executes during object creation, after setting all properties.
function r_CreateFcn(hObject, eventdata, handles)
% hObject    handle to r (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function usuarioscentral_Callback(hObject, eventdata, handles)
% hObject    handle to usuarioscentral (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of usuarioscentral as
text
%        str2double(get(hObject,'String')) returns contents of
usuarioscentral as a double


% --- Executes during object creation, after setting all properties.
function usuarioscentral_CreateFcn(hObject, eventdata, handles)
% hObject    handle to usuarioscentral (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in reset.
function reset_Callback(hObject, eventdata, handles)
%Reset de gráficas, varibales y parámetros generales
clc;
cla(handles.axes1,'reset');             %reset primera gráfica
cla(handles.axes2,'reset');             %reset segunda gráfica
recompensa=0;                           %reset matriz de recompensa
set(handles.uitable1,'data',recompensa); %reset Tabla de recompensa
set(handles.pushbutton1,'BackgroundColor',[0, 0.45, 0.74]);
set(handles.reset,'BackgroundColor','r');
set(handles.error, 'String', '');
set(handles.ResultadoRecompensa, 'String', '');
clear all;                              %clear de variables
fopen('resultados.txt','w');            %Reset archivo de texto



function usuarios3_Callback(hObject, eventdata, handles)
% hObject    handle to usuarios3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of usuarios3 as text
%        str2double(get(hObject,'String')) returns contents of usuarios3
as a double


% --- Executes during object creation, after setting all properties.
function usuarios3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to usuarios3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function usuarios2_Callback(hObject, eventdata, handles)
% hObject    handle to usuarios2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of usuarios2 as text
%        str2double(get(hObject,'String')) returns contents of usuarios2
as a double


% --- Executes during object creation, after setting all properties.
function usuarios2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to usuarios2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function usuarios7_Callback(hObject, eventdata, handles)
% hObject    handle to usuarios7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of usuarios7 as text
%        str2double(get(hObject,'String')) returns contents of usuarios7
as a double


% --- Executes during object creation, after setting all properties.
function usuarios7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to usuarios7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function usuarios4_Callback(hObject, eventdata, handles)
% hObject    handle to usuarios4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of usuarios4 as text
%        str2double(get(hObject,'String')) returns contents of usuarios4
as a double


% --- Executes during object creation, after setting all properties.
function usuarios4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to usuarios4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function usuarios5_Callback(hObject, eventdata, handles)
% hObject    handle to usuarios5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of usuarios5 as text
%        str2double(get(hObject,'String')) returns contents of usuarios5
as a double


% --- Executes during object creation, after setting all properties.
function usuarios5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to usuarios5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function usuarios6_Callback(hObject, eventdata, handles)
% hObject    handle to usuarios6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of usuarios6 as text
%        str2double(get(hObject,'String')) returns contents of usuarios6
as a double


% --- Executes during object creation, after setting all properties.
function usuarios6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to usuarios6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function Mho_Callback(hObject, eventdata, handles)
% hObject    handle to Mho (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Mho as text
%        str2double(get(hObject,'String')) returns contents of Mho as a
double


% --- Executes during object creation, after setting all properties.
function Mho_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Mho (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
results();


% --- Executes during object creation, after setting all properties.
function error_CreateFcn(hObject, eventdata, handles)
% hObject    handle to error (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
function pasos_Callback(hObject, eventdata, handles)
% hObject    handle to pasos (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of pasos as text
%        str2double(get(hObject,'String')) returns contents of pasos as a
double


% --- Executes during object creation, after setting all properties.
function pasos_CreateFcn(hObject, eventdata, handles)
% hObject    handle to pasos (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
Perfil_Angular();



function learning_Callback(hObject, eventdata, handles)
% hObject    handle to learning (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of learning as text
%        str2double(get(hObject,'String')) returns contents of learning
as a double


% --- Executes during object creation, after setting all properties.
function learning_CreateFcn(hObject, eventdata, handles)
% hObject    handle to learning (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Anexo B.3. Código de la función “presupuesto”.

```
function [Prx,dis]=presupuesto(x,y,a,b,f,Ptx,Gtx,Ltx,Grx,Lrx,Ghm);
%Función de presupuesto del enlace
%-------------------------------------------------------------------------
% SIMULACIÓN EN MATLAB DE UN ALGORITMO BASADO EN MACHINE
% LEARNING Y HARD HANDOVER PARA EL BALANCEO DE CARGA EN UNA
% RED CELULAR LTE
% Trabajo de Titulación previo a la obtención del Título de Ingeniero en
% Electrónica y Telecomunicaciones.
% Realizado por: Carlos Alberto Parreño Muirragui
% Director: PhD. Pablo Lupera Morillo
% Descripción de la función: Esta función realiza el cálculo de perdidas
en
% el espacio libre, la potencia recibida en la estación móvil respecto a
% una BS y el margen de desvanecimiento con distribución de Rayleigh
%-------------------------------------------------------------------------
dis=sqrt((x-a)^2+(y-b)^2);  %Cálculo de la distancia vecotrial
dis=dis*0.001;              %Conversión de metros a Kilómetros

%%
%CÁLCULO DE PÉRDIDAS POR TRAYECTORIA EN EL ESPACIO LIBRE
Lfs=32.44+20*log10(f)+20*log10(dis);

%%
%POTENCIA RECIBIDA
Prx=Ptx+Gtx-Ltx-Lfs+Grx-Lrx+Ghm;

%%
%MARGEN DE DESVANECIMIENTO CON DISTRIBUCIÓN DE RAYLEIGH
Xmean=10^(Prx/10);          %Conversion de la Prx sin el margen de
desvanecimiento a mw
desv=Xmean^2*((4/pi)-1);    %Cálculo de la desviación estándar
prob=0.99;                  %Probabildad de que la señal no esté por
debajo del margen de handover

x=sqrt(-log(0.99)*2*desv);  %Fórmula de la probabilidad de Rayleigh (Está
despejado x que es el margen)
Prx=10*log10(x);            % Prx ahora es el valor en dbm de la Prx + el
Margen de desvanecimiento
```

Anexo B.4. Código de la interfaz “results”.

```
function varargout = results(varargin)
%-------------------------------------------------------------------------
% SIMULACIÓN EN MATLAB DE UN ALGORITMO BASADO EN MACHINE
% LEARNING Y HARD HANDOVER PARA EL BALANCEO DE CARGA EN UNA
% RED CELULAR LTE
% Trabajo de Titulación previo a la obtención del Título de Ingeniero en
% Electrónica y Telecomunicaciones.
% Realizado por: Carlos Alberto Parreño Muirragui
% Director: PhD. Pablo Lupera Morillo
% Descripción de la interfaz: Esta interfaz muestra los resultados
% obtenidos del programa de balanceo de cargas.
%-------------------------------------------------------------------------
% RESULTS MATLAB code for results.fig
%      RESULTS, by itself, creates a new RESULTS or raises the existing
%      singleton*.
%
%      H = RESULTS returns the handle to a new RESULTS or the handle to
%      the existing singleton*.
%
%      RESULTS('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in RESULTS.M with the given input
arguments.
%
%      RESULTS('Property','Value',...) creates a new RESULTS or raises
the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before results_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to results_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help results

% Last Modified by GUIDE v2.5 08-Jul-2020 22:01:56

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @results_OpeningFcn, ...
                   'gui_OutputFcn',  @results_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before results is made visible.
function results_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to results (see VARARGIN)

% Choose default command line output for results
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes results wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = results_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
%Apertura del archivo de texto con los resultados del balanceo de carga
    fid = fopen('resultados.txt', 'r');
    c = fread(fid, inf, 'uint8=>char')';    %Conversión de caracteres
    set(handles.texto,'String',c)           %Presentación de resultados

% --- Executes on selection change in texto.
function texto_Callback(hObject, eventdata, handles)
% hObject    handle to texto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns texto contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from texto


% --- Executes during object creation, after setting all properties.
function texto_CreateFcn(hObject, eventdata, handles)
% hObject    handle to texto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Anexo B.5. Código de la interfaz “Perfil_Angular”.

```
function varargout = Perfil_Angular(varargin)
%-------------------------------------------------------------------------
% SIMULACIÓN EN MATLAB DE UN ALGORITMO BASADO EN MACHINE
% LEARNING Y HARD HANDOVER PARA EL BALANCEO DE CARGA EN UNA
% RED CELULAR LTE
% Trabajo de Titulación previo a la obtención del Título de Ingeniero en
% Electrónica y Telecomunicaciones.
% Realizado por: Carlos Alberto Parreño Muirragui
% Director: PhD. Pablo Lupera Morillo
% Descripción de la interfaz: Esta interfaz realiza la simulación del
% perfil angular de potencia de llegada en la estación movil para
entornos
% NLoS y LoS según la recomendación UIT-R P.1816-3, Anexo 3.
%-------------------------------------------------------------------------
% PERFIL_ANGULAR MATLAB code for Perfil_Angular.fig
%      PERFIL_ANGULAR, by itself, creates a new PERFIL_ANGULAR or raises
the existing
%      singleton*.
%
%      H = PERFIL_ANGULAR returns the handle to a new PERFIL_ANGULAR or
the handle to
%      the existing singleton*.
%
%      PERFIL_ANGULAR('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in PERFIL_ANGULAR.M with the given input
arguments.
%
%      PERFIL_ANGULAR('Property','Value',...) creates a new
PERFIL_ANGULAR or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before Perfil_Angular_OpeningFcn gets called.
An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to Perfil_Angular_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Perfil_Angular

% Last Modified by GUIDE v2.5 24-Aug-2020 23:12:00

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Perfil_Angular_OpeningFcn, ...
                   'gui_OutputFcn',  @Perfil_Angular_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
```

```
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Perfil_Angular is made visible.
function Perfil_Angular_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Perfil_Angular (see VARARGIN)

% Choose default command line output for Perfil_Angular
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Perfil_Angular wait for user response (see UIRESUME)
% uiwait(handles.figure1);

%%
%Obtención de varibles globales y valores ingresadas por el usuario
usuarios_handover=getappdata(0,'usuarios_handover')       %Lectura de la
varibale global
set(handles.TablaUsuariosHo,'data',usuarios_handover);   %Presentación de
la información para su selección inicial
% --- Outputs from this function are returned to the command line.

function varargout = Perfil_Angular_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

function hs_Callback(hObject, eventdata, handles)
% hObject    handle to hs (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of hs as text
%        str2double(get(hObject,'String')) returns contents of hs as a
double
```

```
% --- Executes during object creation, after setting all properties.
function hs_CreateFcn(hObject, eventdata, handles)
% hObject    handle to hs (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function R_Callback(hObject, eventdata, handles)
% hObject    handle to R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of R as text
%        str2double(get(hObject,'String')) returns contents of R as a
double


% --- Executes during object creation, after setting all properties.
function R_CreateFcn(hObject, eventdata, handles)
% hObject    handle to R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function W_Callback(hObject, eventdata, handles)
% hObject    handle to W (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of W as text
%        str2double(get(hObject,'String')) returns contents of W as a
double


% --- Executes during object creation, after setting all properties.
function W_CreateFcn(hObject, eventdata, handles)
% hObject    handle to W (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in ejecutar.
function ejecutar_Callback(hObject, eventdata, handles)
fprintf ('        Orden     Celda HO    Distancia (km)     Presupuesto de
enlace \n')
usuarios_handover=getappdata(0,'usuarios_handover')

%%
%Obtención de los valores ingresados por el usuario
hs=str2num(get(handles.hs,'string'));           %Altura media de los
edificios (metros)
R=str2num(get(handles.R,'string'));             %Coeficiente de reflexión
de potencia medio de los muros laterales del edificio
W=str2num(get(handles.W,'string'));             %Anchura de la calle
(metros)
usuario=str2num(get(handles.usuario,'string')); %# de usuario con el que
se realizará la simulación

pocisionUsuario=find(usuarios_handover==usuario,1); %Obtención de la
posición del usuario

%%
%Perfil angular de llegada en la estación móvil para un entorno NLoS
Prx=usuarios_handover(pocisionUsuario,4);
%Obtención del valor del presupuesto del enlace
PrxdBm=Prx;
Prx=10^(Prx/10);
%Conversión de dBm a mW
    for carretera=[0 15 90]
%Ángulo de evaluación
    llegada=[-180:1:180];
%Rango de ángulo de llegada
    n=min(1,((2.6/(hs^0.5)*(1-exp(-0.03*carretera))+0.05)^1.5));
%Coeficiente n

    %Ecuación perfil angular de potencia de llegada en la estación móvil
NLoS

AOAnlos=1./sqrt((cos(llegada.*(pi/180)).^2)+(sin(llegada.*(pi/180)).^2)./
n^2);
    AOAnlos=Prx*AOAnlos;        %Multiplicación del perfil con la PRx por
la estación móvil
    AOAnlos=10*log10(AOAnlos);  %Conversión de mW a dBm

    %Representación gráfica del perfil angular de llegada entorno NLoS
    axes(handles.axes1)
    plot(llegada,AOAnlos)
    hold on
```

```
    grid on
    title ('AOAnlos')
    ylim([-inf -100])
    ylabel('potencia recibida (dBm)');
    xlabel('Angulo de llegada (grados)');
    if carretera == 0
        cero=['Ángulo de 0° entre la dirección de la BS y el UE, potencia
mín= ',num2str(min(AOAnlos)),' dBm'];
    elseif carretera ==15
        quince=['Ángulo de 15° entre la dirección de la BS y el UE,
potencia mín= ',num2str(min(AOAnlos)),' dBm'];
    else
        noventa=['Ángulo de 90° entre la dirección de la BS y el UE,
potencia mín= ',num2str(min(AOAnlos)),' dBm'];
    legend(cero,quince,noventa)
    end
    end

%%
%%Perfil angular de llegada en la estación móvil para un entorno LoS
%Condición del valor del coeficente R
if R>=1 || R<=0
    fprintf('ERROR: EL valor del coeficiente R tiene que estar entre 0 y
1');
else
    carretera=0;                                %Angulo del suelo de la
carretera respecto al edificio de la BS
    d=usuarios_handover_(pocisionUsuario,3);  %Obtención de la distancia de
la Estación móvil respecto a la BS

    if handles.derecha.Value==1;    %Lecutra del entorno seleccionado
        i=1;
        for llegada=-180:0.001:180;     %Evaluación respecto al rango del
ángulo de llegada
            %Ecuación respecto al valor del ángulo de llegada
            if llegada>=0
                n=min(1,((2.6/(hs^0.5)*(1-exp(-
0.03*carretera))+0.05)^1.5));    %Coeficiente n
                %Ecuación perfil angular de potencia de llegada en la
estación móvil NLoS

AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
                %Ecuación perfil angular de potencia de llegada en la
estación móvil LoS

AOAlos(i)=(R^(1000*d*abs(llegada)*pi/(180*W)))+Prx*AOAnlos;
            else
                n=min(1,((2.6/(hs^0.5)*(1-exp(-
0.03*carretera))+0.05)^1.5));    %Coeficiente n
                %Ecuación perfil angular de potencia de llegada en la
estación móvil NLoS

AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
                %Ecuación perfil angular de potencia de llegada en la
estación móvil LoS
                AOAlos(i)=(R^((1000*d*abs(llegada)*pi/(180*W))-
1))+Prx*AOAnlos;
            end
        i=i+1;
        end
```

```
%%
    llegada=[-180:0.001:180];
    AOAlos=10*log10(AOAlos);    %Conversión mW a dBm
    posAOAlos=find(AOAlos>PrxdBm);
    %Representación gráfica del perfil angular de llegada entorno LoS
    axes(handles.axes3)
    hold on
    plot(llegada,AOAlos)
    hold on
    grid on
    title ('AOAlos')
    ylabel('potencia recibida (dBm)');
    xlabel('Angulo de llegada (grados)');
    der=['Perfil angular LoS, potencia min= ',num2str(min(AOAlos)),' dBm,
rango de ángulos no válidos: ('...
        ,num2str((max(posAOAlos)-1)*0.001-180),'° hasta
',num2str((min(posAOAlos)-1)*0.001-180),'°)'];
    legend(der)

    elseif handles.izquierda.Value==1; %Lecutra del entorno seleccionado
        i=1;
        for llegada=-180:0.001:180; %Evaluación respecto al rango del
ángulo de llegada
            %Ecuación respecto al valor del ángulo de llegada
            if llegada>=0
                n=min(1,((2.6/(hs^0.5)*(1-exp(-
0.03*carretera))+0.05)^1.5));    %Coeficiente n
                %Ecuación perfil angular de potencia de llegada en la
estación móvil NLoS

AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
                %Ecuación perfil angular de potencia de llegada en la
estación móvil LoS
                AOAlos(i)=(R^((1000*d*abs(llegada)*pi/(180*W))-
1))+Prx*AOAnlos;
            else
                n=min(1,((2.6/(hs^0.5)*(1-exp(-
0.03*carretera))+0.05)^1.5));    %Coeficiente n
                %Ecuación perfil angular de potencia de llegada en la
estación móvil NLoS

AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
                %Ecuación perfil angular de potencia de llegada en la
estación móvil LoS

AOAlos(i)=(R^(1000*d*abs(llegada)*pi/(180*W)))+Prx*AOAnlos;
            end
        i=i+1;
        end
    llegada=[-180:0.001:180];
    AOAlos=10*log10(AOAlos);    %Conversión mW a dBm
    posAOAlos=find(AOAlos>PrxdBm);
    %Representación gráfica del perfil angular de llegada entorno LoS
    %ylim(handles.axes3, [-20, 10]);
    axes(handles.axes3)
    hold on
    plot(llegada,AOAlos)
    hold on
    grid on
    title ('AOAlos')
```

```
    ylabel('potencia recibida (dBm)');
    xlabel('Angulo de llegada (grados)');
    legend('Estación base a la izquierda del móvil');
    izq=['Perfil angular LoS, potencia min= ',num2str(min(AOAlos)),' dBm,
rango de ángulos no válidos: ('...
        ,num2str((max(posAOAlos)-1)*0.001-180),'° hasta
',num2str((min(posAOAlos)-1)*0.001-180),'°)'];
    legend(izq)

    elseif handles.frente.Value==1; %Lecutra del entorno seleccionado
        i=1;
        for llegada=-180:0.001:180; %Evaluación respecto al rango del
ángulo de llegada
            n=min(1,((2.6/(hs^0.5)*(1-exp(-0.03*carretera))+0.05)^1.5));
%Coeficiente n
            %Ecuación perfil angular de potencia de llegada en la
estación móvil NLoS

AOAnlos=1/sqrt((cos(llegada*(pi/180))^2)+(sin(llegada*(pi/180))^2)/n^2);
            %Ecuación perfil angular de potencia de llegada en la
estación móvil LoS
            AOAlos(i)=(R^(1000*d*abs(llegada)*pi/(180*W)))+Prx*AOAnlos;
            i=i+1;
        end
    llegada=[-180:0.001:180];
    AOAlos=10*log10(AOAlos);    %Conversión mW a dBm
    posAOAlos=find(AOAlos>PrxdBm);
    %Representación gráfica del perfil angular de llegada entorno LoS
    axes(handles.axes3)
    hold on
    plot(llegada,AOAlos)
    hold on
    grid on
    title ('AOAlos')
    ylabel('potencia recibida (dBm)');
    xlabel('Angulo de llegada (grados)');
    legend('Estación base frente al móvil');
    frente=['Perfil angular LoS, potencia min= ',num2str(min(AOAlos)),'
dBm, rango de ángulos no válidos: ('...
        ,num2str((max(posAOAlos)-1)*0.001-180),'° hasta
',num2str((min(posAOAlos)-1)*0.001-180),'°)'];
    legend(frente)
    end
end

%%

% --- Executes on button press in reset.
function reset_Callback(hObject, eventdata, handles)
%Reset de gráficas
clc;
cla(handles.axes1,'reset');
cla(handles.axes3,'reset');



% --- Executes on button press in frente.
function frente_Callback(hObject, eventdata, handles)
%Lectura de selección de opción
frente=get(hObject,'Value');
```

```
% --- Executes on button press in derecha.
function derecha_Callback(hObject, eventdata, handles)
%Lectura de selección de opción
derecha=get(hObject,'Value');

% --- Executes on button press in izquierda.
function izquierda_Callback(hObject, eventdata, handles)
%Lectura de selección de opción
izquierda=get(hObject,'Value');



function usuario_Callback(hObject, eventdata, handles)
% hObject    handle to usuario (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of usuario as text
%        str2double(get(hObject,'String')) returns contents of usuario as
a double
%%

% --- Executes during object creation, after setting all properties.
function usuario_CreateFcn(hObject, eventdata, handles)
% hObject    handle to usuario (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

%%
% --- Executes on button press in recomendacion.
function recomendacion_Callback(hObject, eventdata, handles)
%Este push button despliega la Recomendación UIT-R P.1816-3
winopen('R-REC-P.1816-4.pdf')
```

ANEXO C

Este anexo contiene los resultados de cada una de las simulaciones realizadas para cada escenario descrito en el capítulo 3.

El anexo digital C.1. muestra los resultados de las simulaciones para el escenario 1 (frecuencia 850 MHz, radio 1850 m, potencia de transmisión 0 dBm).

El anexo digital C.2. muestra los resultados de las simulaciones para el escenario 2 (frecuencia 850 MHz, radio 18500 m, potencia de transmisión 20 dBm).

El anexo digital C.3. muestra los resultados de las simulaciones para el escenario 3 (frecuencia 1900 MHz, radio 1150 m, potencia de transmisión 0 dBm).

El anexo digital C.4. muestra los resultados de las simulaciones para el escenario 4 (frecuencia 1900 MHz, radio 11500 m, potencia de transmisión 20 dBm).

El anexo digital C.5. muestra los resultados de las simulaciones para el escenario 5 (frecuencia 2100 MHz, radio 1050 m, potencia de transmisión 0 dBm).

El anexo digital C.6. muestra los resultados de las simulaciones para el escenario 6 (frecuencia 2100 MHz, radio 10500 m, potencia de transmisión 20 dBm).

El anexo digital C.7. muestra los resultados del escenario simulado 1000 veces (frecuencia 850 MHz, radio 1850 m, potencia de transmisión 0 dBm, velocidad de aprendizaje 0,9, paso de variación de margen de *handover* 0,1 dB).

El anexo digital C.8. muestra los resultados del escenario simulado con las celdas vecinas sobrecargadas (frecuencia 2100 MHz, radio 10500 m, potencia de transmisión 20 dBm, paso de variación de margen de *handover* 0,3 dB).

ORDEN DE EMPASTADO