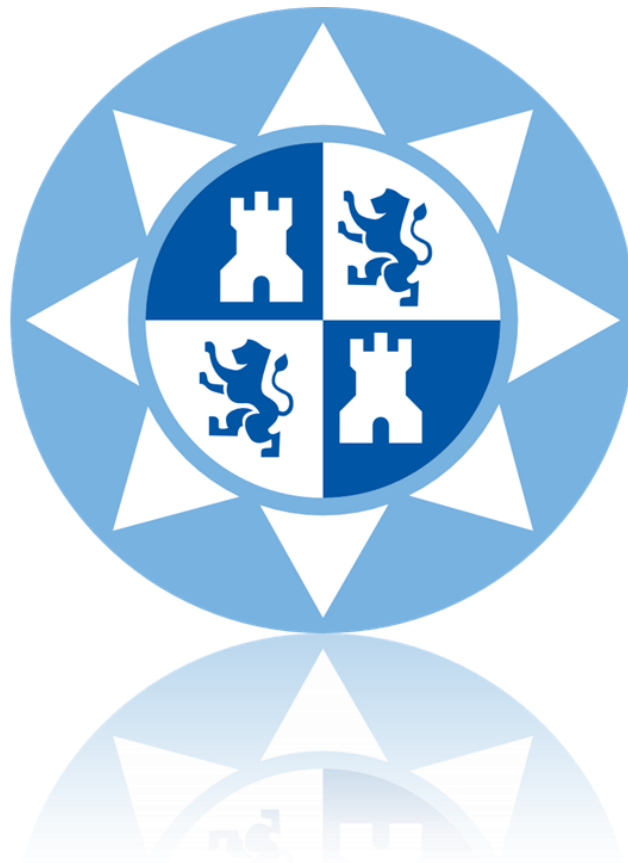


Proyecto Fin de Carrera

# **Escaneo Óptimo Del Espectro En Protocolos Mac De Radio Cognitiva Con Restricciones Hardware**

José Antonio Ayala Romero

Director: Juan José Alcaraz Espín



**Universidad Politécnica de Cartagena**

Escuela Técnica Superior de Ingeniería de Telecomunicación

Julio de 2012



# Contenido

---

<b>1. Introducción .....</b>	<b>6</b>
1.1 Motivación del estudio .....	6
1.2 Conceptos básicos sobre radio cognitiva .....	6
1.3 Escenario del estudio .....	7
1.4 Objetivos .....	7
1.5 Estructura del contenido .....	8
<b>2. HC-MAC .....</b>	<b>9</b>
2.1 Introducción .....	9
2.2 Motivaciones .....	9
2.2.1 Sistemas de Espectro Abierto .....	9
2.2.2 Limitaciones Hardware .....	9
2.2.3 Necesidad de escaneo inteligente .....	10
2.3 HC-MAC: Hardware-Constrained Multi-Channel Cognitive MAC .....	11
2.3.1 Resumen del protocolo .....	12
2.3.2 Diseño del protocolo .....	14
2.3.2.1 Sincronización o contienda .....	14
2.3.2.2 Escaneo .....	14
2.3.2.3 Transmisión .....	15
<b>3. Modelado del sistema .....</b>	<b>16</b>
3.1 Modelo teórico .....	16
3.1.1 Sistema primario .....	16

3.1.1.1 Definición de parámetros.....	16
3.1.1.2 Modelo del sistema primario .....	17
3.1.2 Sistema secundario .....	19
3.1.2.1 Definición de parámetros.....	19
3.1.2.2 Modelo del sistema secundario.....	20
3.2 Simulador.....	22
3.2.1 Elección del entorno de programación .....	22
3.2.2 Simulación por eventos discretos .....	22
3.2.2.1 Conceptos básicos sobre simulación .....	22
3.2.2.3 Control del avance del tiempo .....	23
3.2.2.4 Conceptos sobre Simulación por Eventos Discretos .....	24
3.2.2.5 Algoritmo de Simulación por Eventos Discretos .....	25
3.2.3 Programación del simulador .....	26
<b>4. Sistemas de Asignación .....</b>	<b>28</b>
4.1 Consideraciones .....	28
4.2 Tipos de asignación .....	28
4.2.1 Asignación Aleatoria .....	28
4.2.2 Asignación secuencial compacta .....	29
4.2.3 Asignación Secuencial.....	31
4.3 Planteamiento analítico.....	33
4.3.1 Cálculo de la probabilidad de estado .....	33
4.3.1.1 Probabilidad en Asignación Aleatoria .....	33
4.3.1.2 Probabilidad en Asignación Secuencial Compacta .....	33

4.3.1.3 Probabilidad en Asignación Secuencial.....	34
4.3.2 Cálculo del rendimiento.....	35
4.4 Resultados de simulación .....	36
4.4.1 Sistema vacío .....	37
4.4.2 Asignación Aleatoria .....	38
4.4.3 Asignación Secuencial Compacta.....	41
4.4.4 Asignación Secuencial.....	45
4.5 Convergencia de resultados .....	51
4.6 Conclusiones.....	53
<b>5. Escaneo Inteligente .....</b>	<b>55</b>
5.1 Objetivos.....	55
5.2 Políticas de escaneo .....	55
5.2.1 Escaneo hasta el final.....	55
5.2.2 Escaneo hasta ocupado .....	56
5.2.2 Escaneo hasta $m$ óptimo.....	56
5.2.3 Resultados de las políticas expuestas .....	57
5.3 Optimal Stopping.....	60
5.3.1 Introducción .....	60
5.3.2 Formulación .....	60
5.3.3 Optimal Stopping en el escaneo espectral .....	62
5.3.4 Interpretación mediante grafos .....	63
5.3.5 Implementación .....	65
5.3.6 Complejidad Computacional .....	68

5.4 $k$ -stage look-ahead .....	69
5.4 Resultados.....	70
<b>6. Conclusiones.....</b>	<b>75</b>
<b>Bibliografía.....</b>	<b>77</b>

# Capítulo 1

## Introducción

---

### 1.1 Motivación del estudio

A lo largo de la última década el uso de las tecnologías inalámbricas se ha incrementado de forma masiva, sobre todo en ámbitos de la telefonía móvil donde cada vez hay más operadores que quieren ofrecer sus servicios. Para permitir el máximo número de servicios y minimizar la interferencia entre ellos, el espectro radioeléctrico está cuidadosamente asignado y gestionado. La consecuencia de ello es que cada vez es más valioso y codiciado, sobre todo en bandas inferiores a los 3 GHz.

La radio cognitiva es una tecnología que se estudia actualmente para ser implantada próximamente. Su objetivo es usar las bandas espectrales que, aunque están asignadas de forma estática, son infrautilizadas por los propietarios de las mismas.

Por tanto, la motivación de este estudio es favorecer el desarrollo de este tipo de tecnologías que, sin duda, tendrán gran utilidad en futuros sistemas de comunicación inalámbricos.

### 1.2 Conceptos básicos sobre radio cognitiva

Las redes de radio cognitiva son redes inteligentes de acceso dinámico al espectro radioeléctrico y su principal objetivo es poder aprovechar las sub-bandas libres (también llamadas oportunidades espectrales) que no esté usando el usuario con licencia dentro de un determinado rango del espectro.

Se le llama usuario primario a aquel cuyo uso del espectro está regulado por una licencia otorgada por la administración competente, mientras que al usuario que accede al espectro de forma oportunista sin interferir los servicios primarios se le denomina usuario secundario.

La idea es en esencia que la tecnología de radio cognitiva se puedan adaptar al entorno de forma dinámica midiendo el estado del espectro electromagnético y usando la

heterogeneidad de tecnologías necesarias para permitir a los usuarios migrar a lo largo del tiempo entre las distintas bandas de frecuencia.

Las dos principales características que la radio cognitiva debe alcanzar son: la capacidad cognitiva, o capacidad de sensar la mayor cantidad de información útil posible dentro del entorno; y la capacidad de adaptación o reconfiguración, que le permite adaptarse al entorno de forma dinámica.

### **1.3 Escenario del estudio**

Se plantea un sistema en el que los usuarios primarios o con licencia usan el medio con un sistema de asignación centralizado, es decir, en el sistema hay un nodo central que se encarga de asignar los canales a los demás usuarios secundarios.

En cambio, el protocolo de asignación usado por los usuarios secundarios es distribuido, por lo que los usuarios sin licencia crean redes ad-hoc y acceden de forma oportunista al espectro que no está siendo usado por los usuarios primarios. Todo ello sin interferir las comunicaciones primarias.

Se toma como protocolo base el HC-MAC [1], que tiene como principal característica la consideración de las limitaciones hardware que tienen los terminales de radio cognitiva en el diseño del protocolo, y además plantea una política de escaneo de canales que optimiza el rendimiento del sistema frente a políticas de escaneo fijo usadas en trabajos anteriores.

### **1.4 Objetivos**

El protocolo HC-MAC resuelve el problema del escaneo óptimo de canales mediante la solución a un problema de Optimal Stopping [2], lo que permite a los usuarios secundarios estimar cuál es el número óptimo de canales que deben escanear en función de la situación externa.

El objetivo del proyecto es, por tanto, el estudio de otras políticas de escaneo además de la regla de parada óptima propuesta para HC-MAC y su evaluación en distintos tipos de asignación primaria no contemplados en [1].



## 1.5 Estructura del contenido

Los capítulos que a continuación se presentan dividen el trabajo realizado de la siguiente manera:

**Capítulo 2.** HC-MAC. Estudio del protocolo tratando desde las motivaciones que desembocan en su diseño hasta su funcionamiento detallado y las limitaciones hardware que lo caracterizan.

**Capítulo 3.** Modelado del sistema. Se definen y modelan los sistemas (tanto primario como secundario) para poder realizar el estudio. También se introduce de forma teórica la simulación orientada a eventos y se muestra el simulador empleado en capítulos posteriores.

**Capítulo 4.** Sistemas de asignación. Se exponen los tipos de asignación utilizados, para más tarde estudiar el comportamiento del sistema de forma analítica y mediante simulación observando la coincidencia de los resultados.

**Capítulo 5.** Escaneo inteligente. Se plantean y comparan distintas políticas de escaneo en los diferentes tipos de asignación utilizados.

**Capítulo 6.** Conclusiones. En este último capítulo se desarrollan las conclusiones más relevantes que se pueden extraer del trabajo.

# Capítulo 2

## HC-MAC

---

### 2.1 Introducción

En este capítulo se desarrolla el protocolo HC-MAC [1] (Hardware-Constrained Cognitive MAC o protocolo MAC cognitivo con limitaciones hardware). Este protocolo es la base del estudio que se realiza en los capítulos posteriores, por lo que es relevante desarrollar tanto su funcionamiento como las motivaciones que llevaron a su realización.

### 2.2 Motivaciones

#### 2.2.1 Sistemas de Espectro Abierto

La actual asignación fija del espectro radioeléctrico lleva a una infrautilización de éste. El concepto de Sistemas de Espectro Abierto tiene como objetivo flexibilizar el uso del espectro. Para un sistema de espectro abierto dado, algunas bandas frecuenciales son de interés común tanto para usuarios primarios como para secundarios. Los usuarios primarios poseen licencia de esas bandas espectrales otorgada por el Gobierno, ya que suelen tener sistemas desplegados en esa área. Sin embargo la utilización de ese espectro es muy baja y como sólo queda una pequeña parte de éste disponible no pueden ser desplegados más sistemas de comunicación que lo usen. Sin embargo con la ayuda del Espectro Abierto, esos sistemas pueden funcionar requiriendo un uso de las bandas libres a los usuarios primarios. Tanto los usuarios primarios como los secundarios se benefician de ello, ya que los primarios reciben unos ingresos extras por el alquiler del espectro y los secundarios pueden realizar comunicaciones que no podrían realizar de otro modo.

#### 2.2.2 Limitaciones Hardware

El hardware de radio cognitiva está dando sus primeros pasos en la actualidad; incluso en el futuro, cuando la radio cognitiva esté al orden del día y pueda cambiar sus parámetros de escaneo y transmisión, su costo podría ser muy alto. Dicho de otro modo,

la radio cognitiva tendrá limitaciones hardware. Cuando se escanea en banda ancha existen algunas limitaciones como, por ejemplo, el tiempo que se consume. Por lo tanto, comúnmente, se asume que se puede escanear un ancho de banda limitado durante un cierto tiempo (lo que se conoce como restricción de escaneo).

Después de escanear por un periodo de tiempo, cada usuario secundario tiene información sobre el estado del espectro; basándose en esa información, los usuarios secundarios pueden acceder de forma oportunista al espectro usando las bandas libres, sin embargo, deben estar seguros de que sus transmisiones no generan interferencia grave a los usuarios primarios. En algunos casos el espectro libre es discontinuo. OFDM es muy adecuado para la agregación de espectro discontinuo debido a la posibilidad de cancelar las subportadoras no deseadas y, por tanto, producir una señal con fragmentos frecuenciales no contiguos que pueden ser adaptados a los canales que se encuentren disponibles. Por lo tanto, el espectro que puede ser utilizado por un único nodo secundario está restringido por las limitaciones hardware (llamadas restricciones de transmisión).

### **2.2.3 Necesidad de escaneo inteligente**

Las operaciones de los terminales secundarios tienen dos etapas que se asumen secuenciales: escaneo y transmisión. De esta forma se detecta previamente la actividad de los usuarios primarios y se mantiene su QoS. La capa PHY adapta los parámetros y esquemas de modulación para detectar las señales de los usuarios primarios en los diferentes canales, mientras que la capa MAC determina qué canales debe escanear el usuario secundario y cuándo debe hacerlo. Después de obtener información sobre los canales en la fase de escaneo, los datos se transmiten por los canales que no estaban siendo usados por los usuarios primarios durante la fase de escaneo. *Este proyecto se centra en las decisiones de escaneo de la capa MAC.*

El problema de escaneo inteligente en Sistemas de Espectro Abierto no ha sido investigado completamente en los trabajos anteriores a [1]. En algunos de ellos se escanea un número fijo de canales y se usan los que se encuentran libres, como en IEEE 802.22 [3]. En el presente trabajo se amplía el estudio hecho en [1] sobre escaneo inteligente, considerando más políticas de escaneo y más tipos de asignaciones por parte de los usuarios primarios.

## 2.3 HC-MAC: Hardware-Constrained Multi-Channel Cognitive MAC

En esta sección se presenta el diseño del protocolo MAC multicanal de radio cognitiva con limitaciones hardware, HC-MAC. Es necesario asumir las siguientes hipótesis:

1. Hay un total de  $N$  canales adyacentes de interés. El término canal se refiere a un canal físico con un determinado ancho de banda. No se consideran canales lógicos los diferentes esquemas de codificación en Code Division Multiple Access (CDMA). Para simplificar, se asume que todos los canales tienen el mismo ancho de banda  $B$ .
2. Un canal común  $ch_0$  está disponible para los usuarios secundarios en cualquier momento. En la práctica puede ser una banda sin licencia. Ese canal común es usado como canal de control donde los usuarios secundarios compiten y colaboran como se describirá más adelante.
3. Los usuarios primarios usan  $N$  canales para sus transmisiones. El estado de los  $N$  canales en un instante de tiempo  $t$  y en cualquier localización viene dado por  $\{X_1(t), X_2(t), \dots, X_N(t)\}$ , donde  $X_i(t)$  vale  $\{0$  (libre),  $1$  (ocupado) $\}$ . Si el tráfico de los usuarios primarios sigue un modelo de Poisson, la probabilidad de los estados  $\{X_i(t)\}$  puede ser calculada.
4. Cada nodo secundario está equipado con un único equipo de radio cognitiva. El nodo puede transmitir o escuchar, pero no ambas a la vez. Basándose en el coste del hardware, habrá limitaciones en el número máximo de canales libres o en el número máximo de fragmentos espectrales que el equipo puede usar para la transmisión. El tiempo usado para la detección de señal primaria depende de los diferentes mecanismos de escaneo del espectro y del tipo de señal primaria. Se denota  $t$  el tiempo necesario para detectar la señal primaria y se asume que los resultados de esta detección son precisos.
5. Existe cierto grado de interferencia producido por los usuarios secundarios que los usuarios primarios pueden soportar. Se define  $T$  como el tiempo máximo de interferencia tolerable por los usuarios primarios. Se asume que cualquier actividad primaria en un canal es de duración relativamente larga comparada con  $T$ , por lo que sea cual sea la duración de la transmisión secundaria impuesta por

el protocolo MAC, si no excede el límite impuesto por  $T$ , se considera segura para los usuarios primarios.

Asumiendo estas hipótesis, se encuentran los siguientes retos a la hora de diseñar un protocolo MAC cognitivo que examina la oportunidad de transmisión en múltiples canales disponibles:

1. *Escaneo espectral en búsqueda de usuarios primarios antes de transmisión.* Como la situación del canal es desconocida a priori, para proteger a los usuarios primarios se debe escanear primero. Para ello, se debe buscar la decisión óptima de parada mediante una política de escaneo.
2. *Sincronización entre transmisor y receptor.* Los resultados del escaneo deben ser intercambiados entre el emisor y el receptor. El coste que causa ese intercambio de información debe ser incluido en el cálculo de la regla de parada óptima.
3. *Problema de terminal oculto en sistemas multicanal.* En sistemas multicanal, especialmente en aquellos que solo poseen un único equipo de radio, aparecen nuevos problemas de terminal oculto. Esos problemas surgen cuando un único terminal de radio debe escuchar en diferentes canales, lo cual dificulta el uso del escaneo de una portadora virtual para evitar el problema del terminal oculto.

### 2.3.1 Resumen del protocolo

El tiempo total de un periodo temporal en HC-MAC se puede separar en 3 partes: contienda, escaneo y transmisión. Se introducen tres tipos de paquetes para facilitar estas operaciones. Todos estos paquetes se enviarán por el canal común  $ch_0$ .

- C-RTS/C-CTS: Contienda y reserva del espectro en la parte de contienda.
- S-RTS/S-CTS: Intercambio de la disponibilidad de los canales entre transmisor y receptor en cada slot de escaneo.
- T-RTS/T-CTS: Notificación a los nodos vecinos de que la transmisión ha finalizado.

*Figura 2.1* muestra el diagrama de estado de este HC-MAC. Si un nodo quiere transmitir, primero envía un C-RTS por  $ch_0$  después de un backoff aleatorio. El receptor responde con C-CTS por  $ch_0$ . Cualquier usuario secundario que escucha o el C-RTS o el

C-CTS deberá aplazar su operación y esperar un mensaje de notificación por  $ch_0$ . Después de reservar el periodo de escaneo, el transmisor y el receptor van escaneando cada canal e intercambiando S-RTS y S-CTS en  $ch_0$  si el canal está disponible en ambos extremos. Un fallo en S-RTS/S-CTS indica que el canal no está disponible. Se usará una política de parada para comunicar el estado en el que se para de escanear. Cuando se llega a un acuerdo entre emisor y receptor, se transmiten los datos en los canales seleccionados. Cuando la transmisión termina, vuelven al canal de control e intercambian T-RTS/T-CTS. El intercambio de estos últimos mensajes hace que termine el aplazamiento de los nodos vecinos y que participen en una nueva ronda de contienda.

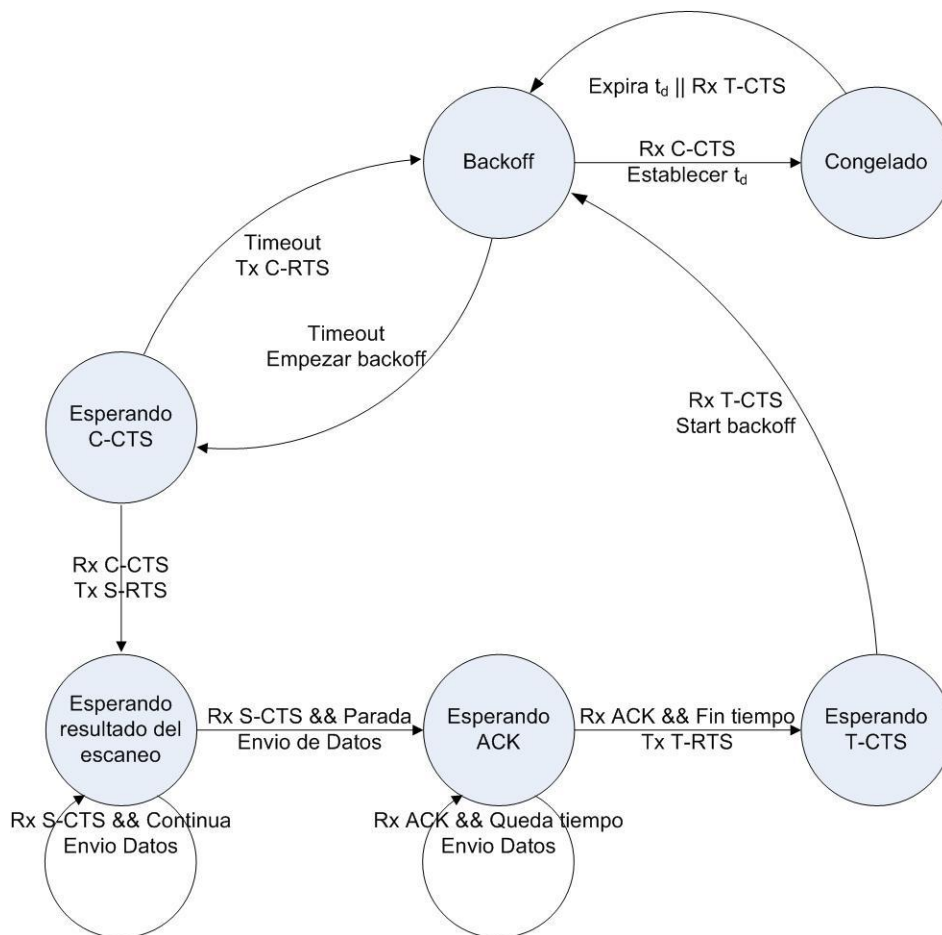


Figura: 2.1. Diagrama de estados de HC-MAC

## 2.3.2 Diseño del protocolo

### 2.3.2.1 Sincronización o contienda

HC-MAC no requiere sincronización centralizada. Cualquier nodo puede entrar en la red escuchando primero el canal de control  $ch_0$  durante un intervalo  $t_d = t \cdot N + T$ . Ello permite que el nuevo nodo observe las actividades que se están realizando sobre el espectro. Como ningún nodo vecino puede escanear más tiempo que  $t \cdot N$  ni transmitir más tiempo que  $T$ , el nuevo nodo no perderá ningún paquete de control de sus vecinos. Durante el periodo si se recibe un C-RTS/C-CTS, el nuevo nodo aplazará su actuación y esperará a un T-RTS/T-CTS. Si se recibe un T-RTS/T-CTS o se agota el tiempo  $t_d$  antes de recibirlo, el nuevo nodo participa en el proceso de contienda si desea transmitir.

Durante el periodo de contienda se usa un esquema de acceso al medio similar al modelo IEEE 802.11 DCF. Los nodos reservan el tiempo para el siguiente escaneo y transmisión con el conjunto de vecinos, intercambiando C-RTS/C-CTS con el nodo objetivo por el canal de control. Cuando un nodo quiere enviar un paquete a otro nodo, primero le envía un C-RTS por el canal de control; el receptor recibe el C-RTS y le contesta con un paquete C-CTS. Los nodos que escuchan esto aplazan sus operaciones y esperan una notificación del par que está transmitiendo o un timeout.

Cuando la transmisión termina, los nodos vuelven a competir en el canal de control con un backoff aleatorio. Cada uno de ellos elige un valor para la cuenta atrás dentro de una ventana de contención. Al comienzo, el tamaño de la ventana de contención es el mínimo. Cuando acaba el tiempo de backoff, el nodo intenta reservar el canal de control enviando un C-RTS al destino. Si colisionan varios C-RTS de vecino, el valor de la ventana de contención se dobla y se vuelve a la fase de backoff. Gana el nodo con menor tiempo de backoff, mientras que los demás detienen su cuenta atrás hasta el próximo periodo de contención.

### 2.3.2.2 Escaneo

El par de nodos que ganan el periodo de contención empiezan a escanear el espectro. La fase de escaneo consta de uno o más slots de escaneo, que a su vez constan de dos partes: el escaneo del canal actual y la negociación entre el emisor y el receptor. Como el emisor y el receptor están sincronizados, ambos escanean cada canal en el mismo intervalo de tiempo  $t_{scan}$ . Después de obtener los resultados, el emisor envía un S-RTS al receptor que incluye la disponibilidad del canal. El receptor recibe el S-RTS y le

contesta con un paquete S-CTS. Si el intercambio de paquetes ha sido exitoso, acaba el presente slot de escaneo. De esta forma, aparece otro costo derivado del intercambio de S-RTS y S-CTS que se denota como  $t_e$ . El coste total de obtener la información de un canal es  $t = t_{scan} + t_e$ .

La decisión de parar o continuar escaneando debe tomarse en el emisor y receptor de forma simultánea y precisa de futura negociación.

A veces, es necesario conocer la probabilidad de la disponibilidad de los canales. Si la probabilidad no es conocida a priori, puede estimarse con la información recogida en cada escaneo de canal. Si las condiciones de canal son similares para todos los canales, la información conjunta de todos los canales se usa para calcular la probabilidad común de disponibilidad. Por otra parte, la probabilidad individual es calculada para cada canal. Se crea una ventana con tamaño  $EW$  que puede ser usada para aproximar la probabilidad con la información recogida en los pasados slots de escaneo. La estimación previa entre transmisor y receptor debe ser la misma ya que, en caso contrario, cada uno de ellos tomará una decisión diferente. Esto se logra con los intercambios de paquetes RTS/CTS en las fases de contienda y escaneo. Cada RTS/CTS intercambia la estimación, mientras que la decisión final usa la media.

### 2.3.2.3 Transmisión

Después de que el emisor y el receptor tomen la decisión de parar de escanear, empiezan a usar el conjunto de canales disponibles para enviar paquetes. El tiempo máximo de transmisión es igual a  $T$ . Después de finalizar la transmisión, el emisor enviará un T-RTS para anunciar que la transmisión ha concluido, y acto seguido el receptor le responderá con un T-CTS. El intercambio de esta información finaliza la fase aplazamiento de los nodos vecinos que querían transmitir, por lo que empieza una nueva fase de contienda.



# Capítulo 3

## Modelado del sistema

---

### 3.1 Modelo teórico

Para poder estudiar el comportamiento del sistema, previamente se debe modelar. Hay que distinguir entre el sistema de los usuarios primarios y el de los usuarios secundarios, ya que aunque interactúan entre sí, funcionan de forma diferente.

#### 3.1.1 Sistema primario

A continuación se van a definir los parámetros del sistema de los usuarios primario y su modelo estadístico. Cabe destacar que el comportamiento de los usuarios secundarios no afecta al comportamiento de este sistema, ya que, al disponer los usuarios primarios de licencia, pueden hacer uso de los canales en su totalidad sin ninguna supeditación.

##### 3.1.1.1 Definición de parámetros

A continuación se describen los parámetros más relevantes del sistema:

*Número de canales ( $N$ ):* Es el número total de canales que tiene el sistema. Si los  $N$  canales están siendo usados y se intenta establecer una comunicación por otro canal, ésta será rechazada, ya que el sistema está lleno y no hay cola de espera.

*Tasa de ofrecida ( $\lambda$ ):* Número medio de peticiones que recibe el sistema en un periodo de tiempo. Se considera que la población es infinita, y se modelan como un proceso de Poisson. Por tanto, si  $X$  = “Número de peticiones que recibe el sistema en un periodo de tiempo aleatorio”:

$$X \sim P(\lambda)$$

Como consecuencia, el tiempo medio entre peticiones sigue un modelo exponencial, donde si  $Y = \text{“Tiempo entre peticiones”}$ :

$$Y \sim \text{Exp}(\lambda)$$

y el tiempo medio entre peticiones es:

$$E(Y) = \frac{1}{\lambda}.$$

*Tiempo medio de servicio* ( $\frac{1}{\mu}$ ): Es el tiempo medio que una petición está en el sistema.

Se modela de forma análoga a la tasa ofrecida. Si  $t_s = \text{“Tiempo de servicio de una petición escogida al azar”}$ ,

$$t_s \sim \text{Exp}(\mu).$$

*Tráfico ofrecido (TO)*: Se obtiene como relación entre la tasa ofrecida y el tiempo de servicio. Es el número medio de servidores ocupados, si el número de servidores en el sistema fuera infinitamente grande.

$$TO = \frac{\lambda}{\mu}$$

Este parámetro se va a usar para expresar la carga del sistema en las gráficas de las simulaciones.

### 3.1.1.2 Modelo del sistema primario

Una vez definidos los parámetros, se puede modelar el sistema como un proceso de nacimiento y muerte con  $N$  estados, tantos como canales tiene el sistema.

Si el sistema se encuentra en el estado  $n$  significa que hay  $n$  canales ocupados, no se especifica cuáles. Esto quiere decir que el modelo mediante cadenas de Markov es independiente del tipo de asignación de canales (aleatoria, secuencial...).

Según la nomenclatura de Kendall se trata de un sistema M/M/N/N, y su grafo asíncrono es el siguiente:

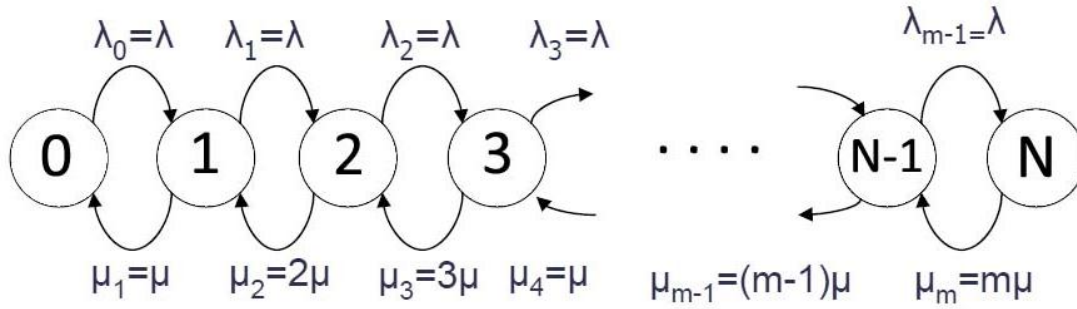


Figura: 3.1. Grafo asíncrono de la cola M/M/N/N

Se pueden observar en el grafo las transiciones entre estados; la tasa de llegada es siempre  $\lambda$  (población infinita), y la tasa de muertes depende del número de canales ocupados.

Algunos parámetros importantes de la cola M/M/N/N son los siguientes:

*Probabilidad de estado:* Es la probabilidad de que el sistema se encuentre en estado  $n$ , es decir, que haya  $n$  canales ocupados.

$$P(n) = \frac{\frac{\rho^n}{n!}}{\sum_{i=0}^m \frac{\rho^i}{i!}} \quad \text{con } \rho = \frac{\lambda}{\mu}.$$

*Probabilidad de pérdida:* Es la probabilidad de que se rechace una nueva petición que llega al sistema. Los valores de la siguiente expresión se encuentran tabulados en las tablas Erlang-B.

$$P_L = \frac{\frac{\rho^m}{m!}}{\sum_{i=0}^m \frac{\rho^i}{i!}} \quad \text{con } \rho = \frac{\lambda}{\mu}.$$

### 3.1.2 Sistema secundario

En este apartado se van a describir los parámetros y el comportamiento de los usuarios secundarios. Al contrario de lo que pasaba con el sistema primario, el comportamiento de los usuarios secundarios está condicionado por la ocupación y el tipo de asignación de usuarios primarios. Esto se detallará en los últimos capítulos, cuando se aborde el tema de las políticas de escaneo.

#### 3.1.2.1 Definición de parámetros

A continuación se describen los parámetros del sistema secundario:

$N$  (*Número de canales del sistema*): este parámetro es común al sistema primario y secundario, ya que ambos comparten el mismo medio.

$t_s$  (*Tiempo de sincronización*): es el tiempo que necesitan los terminales secundarios para sincronizarse antes de poder escanear y transmitir.

$t$  (*Tiempo de muestreo*): tiempo durante el que un terminal secundario escucha un canal para ver si está ocupado o no. Este tiempo viene dado por la tecnología física usada, por lo que hay muy poco margen de variación, además, en la práctica si se hace muy pequeño se corre el riesgo de detectar el canal como vacío cuando en realidad está siendo usado.

$T$  (*Tiempo de transmisión*): es el tiempo que un usuario secundario transmite ininterrumpidamente por un canal. Este parámetro es muy importante porque, por un lado, cuanto mayor sea mayor será el throughput de los usuarios secundarios; y, por tanto, mejor es el funcionamiento del sistema secundario. Por otro lado, un valor grande de  $T$  puede causar interferencias en los usuarios con licencia, con lo que no se estaría cumpliendo uno de los objetivos principales del protocolo. Por ello, para elegir el valor de  $T$  se va a coger el caso peor, en el que, nada más empezar a transmitir el usuario secundario, un usuario con licencia empieza también a transmitir por ese canal. En este

caso, el valor de  $T$  será el máximo tiempo que se puede interferir la transmisión del usuario primario sin que sufra pérdidas graves.

$m$ : es el número máximo de canales que un usuario secundario va a escanear antes de empezar a transmitir. Este es el parámetro que hay que optimizar para mejorar el rendimiento del sistema secundario.

$n$ : es el número máximo de canales por lo que puede transmitir de forma simultánea un usuario secundario. Este valor es una limitación de hardware del terminal secundario por lo que el protocolo deberá adaptarse a ello.

$R$  (*Tasa de transmisión de los usuarios secundarios en bits/s*): Este valor varía con la frecuencia pero para este estudio se va a considerar constante para cualquier canal.

$c$  (*Ratio de transmisión*): se define el ratio de transmisión como el cociente  $\frac{T}{t}$ . Se usará para ver cuán grande es el tiempo de transmisión respecto al de muestreo.

### 3.1.2.2 Modelo del sistema secundario

Este protocolo de comunicación consta de las siguientes fases para cada uno de los nodos:

1. Fase de sincronización

En la fase de sincronización o contienda todos los nodos se ponen de acuerdo sobre quién va a usar el medio para que no haya colisiones. Esta fase dura  $t_s$ , que se puede modelar como una variable aleatoria, o se le puede dar un valor fijo entendiéndose como la esperanza de esa variable.

2. Fase de escaneo

En esta fase, un nodo escanea canales de forma secuencial hasta que, o bien escanee  $m$  canales en total, o encuentre  $n$  canales libres (que son los máximos por los que puede transmitir de forma simultánea). Cuando se cumple una de estas dos condiciones se pasa a la fase de transmisión

### 3. Fase de transmisión

Esta fase dura un tiempo  $T$ , y en ella se transmite de forma simultánea por todos los canales libres encontrados. Por tanto, el número de canales por los que se transmite varía en función del estado del sistema, es decir, del tráfico primario que haya. Además, este valor debe ser menor o igual a  $n$ . Seguidamente se vuelve a la fase de sincronización o contienda.

A continuación se muestra un diagrama en el que se ejemplifica gráficamente cómo funciona el protocolo. Para el ejemplo  $N = m = 3$  y  $n = 2$ .

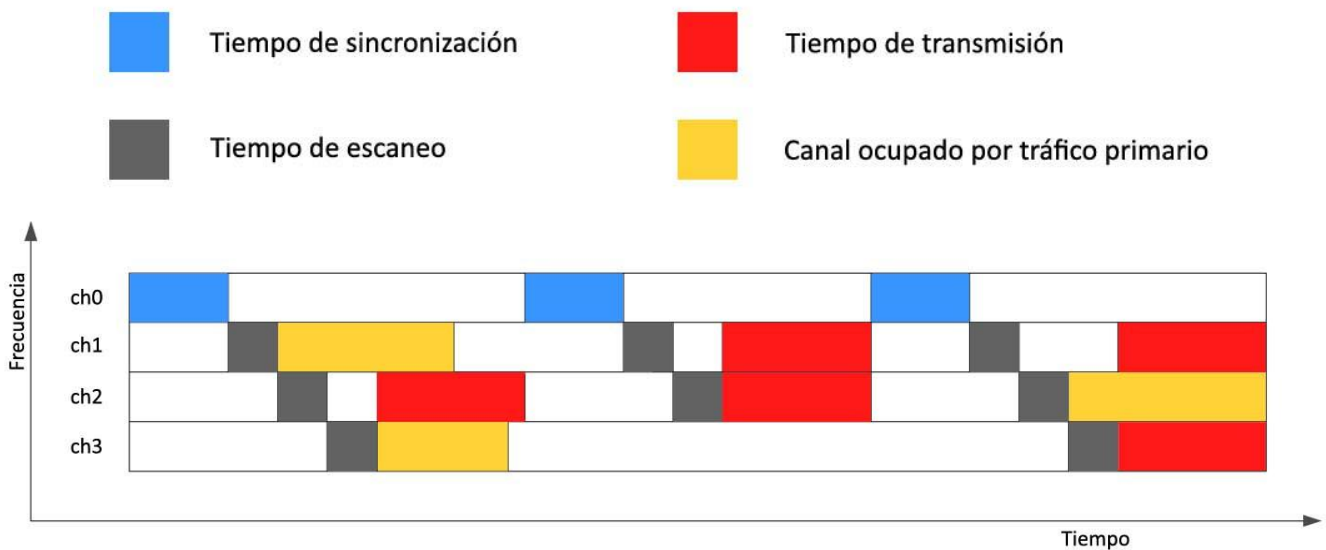


Figura: 3.2. Ejemplo de funcionamiento del protocolo

Se puede observar que, en principio, el sistema va a escanear todos los canales ( $m = N$ ), pero como sólo puede transmitir por dos de forma simultánea, cuando encuentra dos libres ya no escanea el tercero, ya que, aunque estuviera libre, no podría transmitir por él. También cabe destacar la aparición del *ch0* mencionado en el capítulo 2, que es un canal para control que siempre está disponible para los usuarios sin licencia.

## 3.2 Simulador

Una vez definidos matemáticamente los sistemas a programar, sólo queda elegir el entorno de programación y definir algunos conceptos básicos relacionados con la simulación para, posteriormente, realizar el simulador.

### 3.2.1 Elección del entorno de programación

Para programar el simulador se ha elegido, entre otras opciones, trabajar con el entorno MATLAB, por diversas razones:

- Lenguaje de alto nivel que proporciona mayor abstracción a la hora de codificar.
- Orientación a matrices que hace muy fácil el manejo de matrices y vectores que son muy recurrentes en el presente trabajo.
- Gran cantidad de librerías matemáticas.
- Framework amigable que facilita la tarea tanto de codificación como de depuración y ejecución del código.

### 3.2.2 Simulación por eventos discretos

En esta sección se van a abordar algunos conceptos sobre simulación, concretamente sobre simulación por eventos discretos.

#### 3.2.2.1 Conceptos básicos sobre simulación

La simulación es la imitación del comportamiento de un proceso o sistema real a lo largo del tiempo, mediante la generación de una historia artificial del sistema y la observación de dicha historia artificial, para extraer conclusiones sobre las características del funcionamiento de dicho sistema en la realidad.

La simulación de un sistema requiere como paso previo la realización de un modelo del sistema. El modelo es una representación abstracta del sistema real y estará basado en los parámetros y reglas de funcionamiento que caracterizan al sistema. Es especialmente importante obtener un modelo con un grado de detalle adecuado, que no contenga

detalles irrelevantes, que lo complicarían y lo harían más costoso, ni omite detalles esenciales, dando lugar a resultados erróneos.

Particularizando, un simulador será, por tanto, un programa de ordenador que representa la evolución temporal de un sistema real mediante un modelo del mismo.

Existen una serie de circunstancias en las cuales se considera que la simulación no es una herramienta adecuada para estudiar un sistema, por ejemplo, cuando existen modelos analíticos del sistema a simular. En este caso no se dispone de modelo analítico ya que al ser un protocolo distribuido se complica aún más la formulación del mismo.

Se define estado de un sistema como el grupo de variables necesarias para describir el sistema en cualquier momento dentro del ámbito estudiado.

### 3.2.2.3 Control del avance del tiempo

Se acaba de definir la simulación como la imitación del comportamiento de un sistema real a lo largo del tiempo. Como primer paso, por tanto, debemos decidir cómo se gestiona el paso del tiempo en el simulador. Hay dos enfoques básicos:

1. *Discretización temporal (time slicing)*: consiste en generar un modelo con un estado inicial, asociarlo a un reloj virtual que se incrementaría en intervalos suficientemente pequeños de tiempo y en cada uno de estos intervalos se examinaría el sistema para decidir si se ha producido algún evento o suceso que cambie el estado del sistema. Esta aproximación no resulta práctica en la mayoría de las ocasiones, ya que hay que simular todos los intervalos de tiempo, tanto aquellos en los que se producen eventos como en los que no, lo cual es ineficiente. Lo más razonable sería detenerse únicamente en los instantes de tiempo en los que realmente se produce un evento.
2. *Avance por eventos discretos*: en este caso, el reloj avanza directamente al instante temporal en el que realmente ocurre un evento en el sistema, es decir, el reloj salta de un instante de ocurrencia de un evento al siguiente.



La simulación por eventos discretos es una herramienta de análisis numérico de los sistemas discretos que se basa, precisamente, en simular únicamente los momentos en los que se producen eventos, adelantando el reloj de simulación directamente a estos momentos.

### 3.2.2.4 Conceptos sobre Simulación por Eventos Discretos

Además de los conceptos de Modelo y Estado del Sistema definidos en la *sección 3.2.2.1*, se definen los siguientes:

*Evento*: cualquier suceso instantáneo que cambia el estado del sistema. Pueden ser endógenos, si se generan dentro del sistema, y exógenos, si se generan fuera del sistema (en lo que se conoce por entorno del sistema). En el caso que nos ocupa, sólo se producirán dos tipos de eventos: llegadas de clientes al sistema (exógeno) y salidas de clientes del sistema (endógeno). El simulador deberá generar los eventos que se producirán en el sistema en el futuro, de los cuales se generará un registro (planificará) conteniendo, al menos, el tipo de evento y el momento en que ocurrirá.

*Entidad*: objeto o componente del sistema que requiere una representación explícita en el modelo. Por ejemplo, en un sistema de espera: el servidor o servidores, un cliente, la cola...

*Atributos*: propiedades de una entidad. Por ejemplo, la prioridad de un cliente, la longitud de la cola, etc...

*Lista de Eventos Futuros (F.E.L.)*: tabla o lista que contiene los registros con los eventos futuros planificados previamente. Estará ordenada por el tiempo de ocurrencia de los eventos.

*Actividad*: periodo de tiempo de cierta duración (puede ser aleatoria) del cual se sabe cuándo comienza. Puede estar definido por una distribución estadística. En nuestro caso, las actividades son los tiempos de servicio o los tiempos entre llegadas.

*Reloj*: variable que representa el tiempo real simulado o tiempo de simulación.

### 3.2.2.5 Algoritmo de Simulación por Eventos Discretos

El algoritmo empleado en la Simulación por Eventos Discretos es conocido como Event-Scheduling/Time-Advance (que se podría traducir como Programación de Eventos/Tiempo Anticipado).

Este algoritmo consiste en la generación de los momentos del tiempo de simulación en que se producirán los eventos, almacenar el registro del evento generado en la posición adecuada de tabla de eventos, y extraer los eventos cronológicamente, actualizando el reloj al momento de ocurrencia del evento extraído.

*La extracción de un evento de la tabla implica un cambio en el estado del sistema y puede implicar planificación del momento de ocurrencia de uno o más nuevos eventos.*

En un momento cualquiera  $t$  del tiempo de simulación, la Lista de Eventos Futuros (F.E.L.) contiene todos los eventos futuros previamente planificados, con sus tiempos de ocurrencia asociados ( $t_1, t_2, \dots$ ). La F.E.L. está ordenada cronológicamente, por lo que los tiempos de evento cumplen:

$$t \leq t_1 \leq t_2 \leq t_3 \leq \dots \leq t_n$$

$t$  es el valor del reloj, el valor actual del tiempo de simulación. El evento asociado a  $t_1$  es conocido como evento inminente. Una vez que el estado del sistema está actualizado al tiempo de simulación correspondiente a  $\text{reloj} = t$ , el reloj se hace avanzar al momento  $\text{reloj} = t_1$ , el registro del evento inminente es eliminado de la F.E.L. y el evento se ejecuta, lo que implica una variación en el estado del sistema y, en ciertos casos, la generación de al menos un evento futuro, que será insertado en su posición correspondiente de la F.E.L. Este proceso se repite hasta la finalización de la simulación.

### 3.2.3 Programación del simulador

De igual forma que se ha modelado el sistema en dos partes (primaria y secundaria), se va a programar el simulador. Para la primera parte, que es la parte correspondiente a los usuarios primarios, basta con programar una cola M/M/N/N. El código tiene esta forma:

```
Insertar_fel(primer llegada);

for i=1 : Neventos

    leer_fel(evento_inminente);
    borrar_fel(evento_inminente);

    switch evento_leido.tipo
        case llegada:
            insertar_fel(proxima llegada);
            if(recursos disponibles)
                insertar_fel(tiempo de servicio de la llegada);
                ocupar_recursos();
            endif
        case muerte:
            desocupar_recursos();
    endswitch
endfor
```

Después de verificar el funcionamiento de la cola M/M/N/N, sólo queda añadir la segunda parte al código en la que se simula el comportamiento de los usuarios secundarios. Es importante destacar que ambas partes son dos bloques de código separados que se encuentran dentro de un *bucle for* externo y que, aunque se ejecuten de forma independiente, tienen dos elementos comunes: la F.E.L. y los canales de comunicación. Es evidente que sólo es necesaria una F.E.L. para todo el simulador, donde se introducirán y sacarán eventos primarios y secundarios indistintamente, y también queda patente, por la propia definición del protocolo, que ambos usuarios deben compartir recursos.

Cuando el código está completo, hay que añadir las medidas oportunas que darán información y arrojarán resultados sobre el sistema simulado.

A continuación se muestra el pseudocódigo del simulador completo:

```

Insertar_fel(llegada,usuario_primario);
Insertar_fel(escaneo, usuario_secundario);

for i=1 : Neventos
    leer_fel(evento_inminente);
    borrar_fel(evento_inminente);

    if(evento_leido.usuario==usuario_primario)
        switch evento_leido.tipo
            case llegada:
                insertar_fel(llegada, usuario_primario);
                if(recursos disponibles)
                    insertar_fel(tiempo de servicio, usuario primario);
                    ocupar_recursos_primario();
                endif
            case muerte:
                desocupar_recursos_primario();
        endswitch
    else
        switch evento_leido.tipo

            case escaneo:
                if(canal_escaneado==libre)
                    vector_libre=concatenar(vector_libre,canal_escaneado);
                endif

                if(nº canales escaneados==m or longitud(vector_libre)==n)
                    Insertar_fel(transmisión, usuario_secundario);
                else
                    Insertar_fel(escaneo, usuario_secundario);
                endif

            case transmisión:
                for j=1:length(vector_libre)
                    ocupar_recursos_secundario(vector_libre(j));
                endfor
                Insertar_fel(espera_ts, usuario_secundario);

            case espera_ts:
                for j=1:length(vector_libre)
                    desocupar_recursos_secundario(vector_libre(j));
                endfor
                Insertar_fel(escaneo, usuario_secundario);

        endswitch
    endif
endfor

```

# Capítulo 4

## Sistemas de Asignación

---

### 4.1 Consideraciones

En este capítulo se va a estudiar el efecto que tiene el tipo de asignación de canales de los usuarios primarios sobre el desarrollo del protocolo y rendimiento de los usuarios secundarios.

En general, se va a evaluar el *throughput* y el *porcentaje de ocupación* del sistema en función del parámetro  $m$ , en un sistema con 10 canales ( $N = 10$ ), y para diferentes valores de ratio de transmisión ( $c$ ) y de carga del sistema ( $TO$ ). Para cada caso se obtendrá un valor de  $m$  que optimiza el *throughput* y con ello también el *porcentaje de ocupación* del sistema.

Otro aspecto a considerar es que, en principio, no se tendrán en cuenta las limitaciones impuestas por el parámetro  $n$ , que se valorará más tarde.

Para empezar, conviene desarrollar los posibles tipos de asignación que se pueden dar por parte de los usuarios primarios.

### 4.2 Tipos de asignación

A continuación se detallan los tres tipos de asignación de canales de los usuarios primarios.

#### 4.2.1 Asignación Aleatoria

En el instante en el que se procede a ocupar un canal, se escoge aleatoriamente uno de los que aún permanecen libres, es decir, la probabilidad de que un canal sea escogido se reparte de forma uniforme entre los canales libres.

Por ejemplo, si el sistema cuenta con 6 canales ( $N = 6$ ), y en un determinado instante es necesario utilizar un canal y dos de ellos se encuentran ya ocupados, cuando se proceda a ocupar un canal habrá una probabilidad  $p = 0.25$  de elegir cada uno de los restantes.

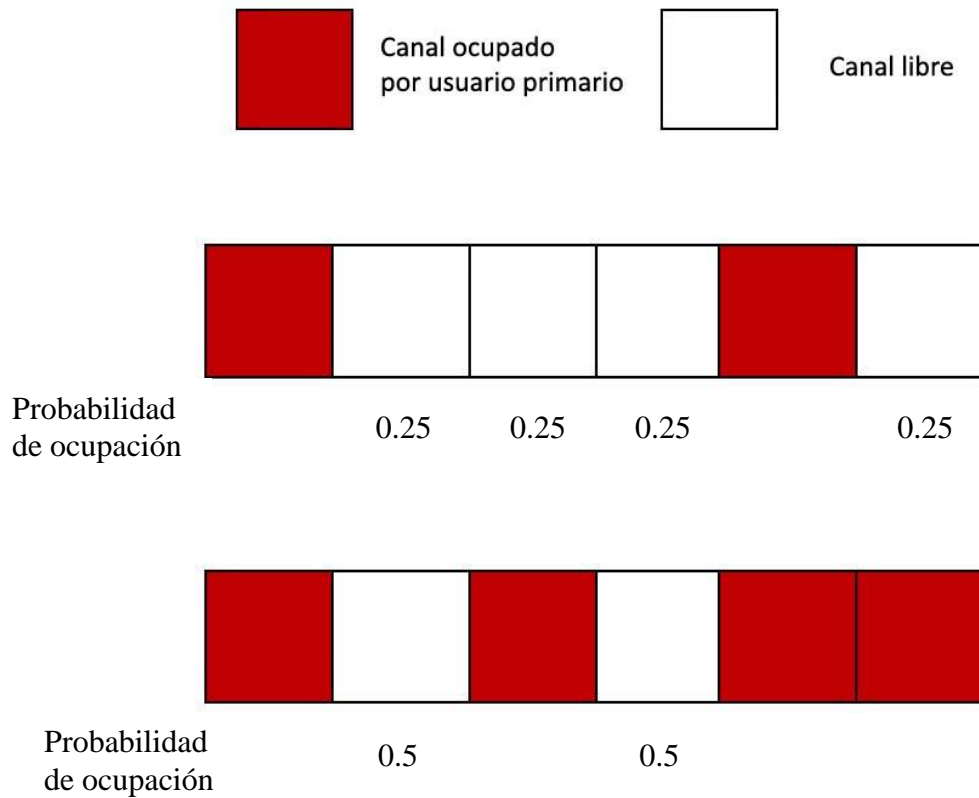


Figura: 4.1. Ejemplo Asignación Aleatoria

#### 4.2.2 Asignación secuencial compacta

Esta asignación se caracteriza por encontrarse todos los canales ocupados en un extremo del espectro; siempre se intentará ocupar los canales con el número más alto/bajo. Además, entre los canales ocupados no habrá huecos de canales libres. Para que esto sea así, en el instante en el que un canal se desocupa, los canales ocupados que quedan se desplazan para ocupar ese hueco de forma instantánea.

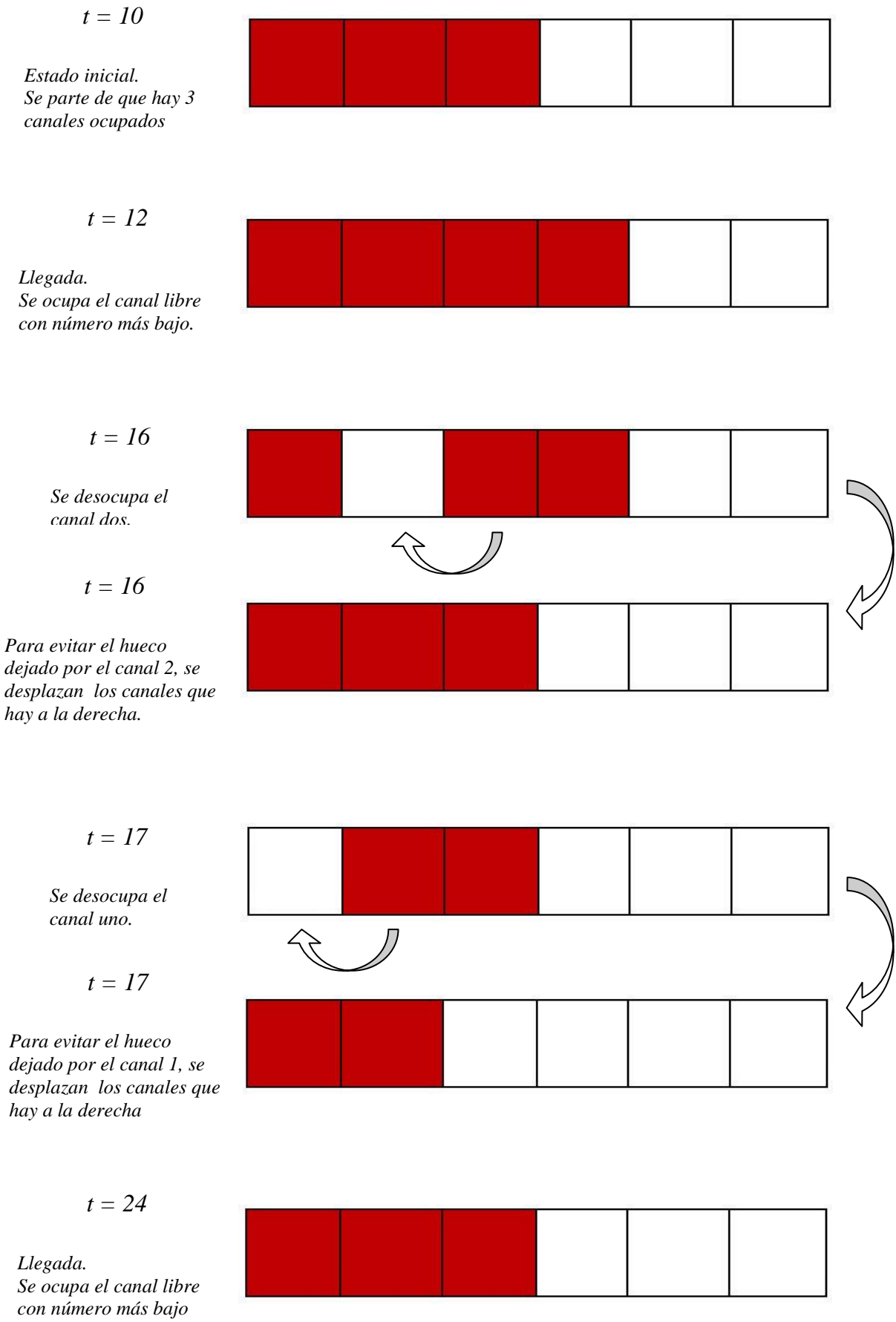


Figura: 4.2. Ejemplo Asignación Secuencial Compacta

Esto se puede interpretar de forma teórica de manera que el único canal que puede desocuparse es el último ocupado, y se desocupará en el instante en el que cualquiera de los canales ocupados acabe el tiempo de servicio.

Como se expuso en el capítulo anterior, el tiempo de servicio ( $ts$ ) de cualquier canal viene dado por una variable aleatoria exponencial con parámetro  $\mu$ .

$$ts \sim Exp(\mu)$$

En el instante en el que se ocupa (o se desocupa) el último canal el tiempo de servicio de todos los demás canales sigue siendo  $ts$ , ya que la variable aleatoria exponencial no tiene memoria.

$$P(ts > t_1 + t_2 | ts > t_1) = \frac{e^{-\lambda(t_1+t_2)}}{e^{-\lambda t_1}} = e^{-\lambda t_2} = P(ts > t_2)$$

Por tanto, se puede demostrar que el tiempo de servicio de ese último canal es

$$\min(ts_1, ts_2, \dots, ts_n) = ts_n \sim Exp(n\mu),$$

siendo  $n$  el estado del sistema, o lo que es lo mismo, el número de canales ocupados.

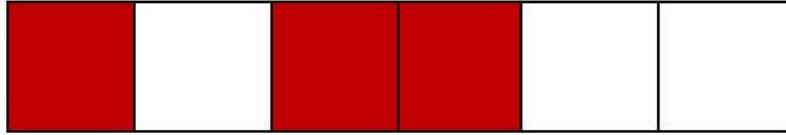
### 4.2.3 Asignación Secuencial

Este tipo de asignación también intenta ocupar un extremo del espectro pero, esta vez, no hay desplazamiento de canales ocupados, por lo que su funcionamiento se simplifica y se puede resumir en que cuando se necesita usar un canal, se busca el canal libre con un número más alto/bajo.



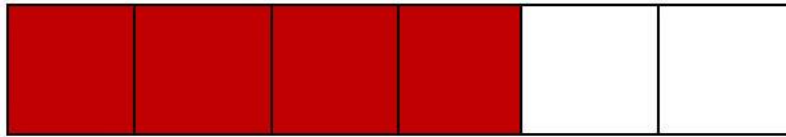
$t = 5$

*Estado Inicial.  
Se parte de estos tres  
canales ocupados.*



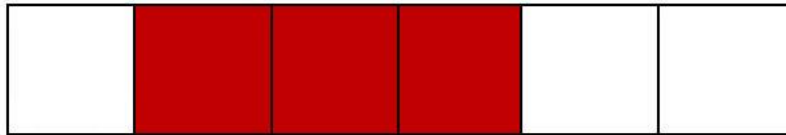
$t = 7$

*Llegada.  
Se ocupa el canal libre  
con número más bajo.*



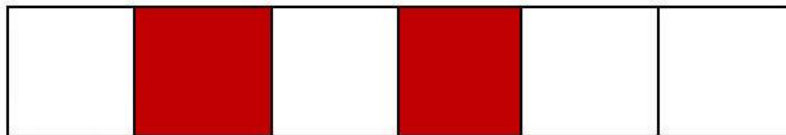
$t = 12$

*Se desocupa el  
canal uno.*



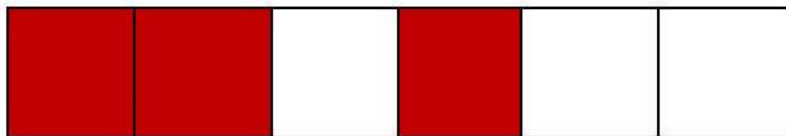
$t = 20$

*Se desocupa el  
canal tres.*



$t = 22$

*Llegada.  
Se ocupa el canal libre  
con número más bajo.*



$t = 23$

*Se desocupa el  
canal cuatro.*



Figura: 4.3. Ejemplo Asignación Secuencial

## 4.3 Planteamiento analítico

### 4.3.1 Cálculo de la probabilidad de estado

Para llevar a cabo el estudio analítico tal y como se ha planteado es necesario calcular la probabilidad de ocupación de los usuarios primarios, es decir, dado un canal cualquiera, es necesario saber la probabilidad de que esté ocupado por un usuario primario. Esta probabilidad no tiene por qué ser igual para todos los canales y, por supuesto, depende del tipo de asignación primaria. A continuación, se detalla este cálculo para cada tipo de asignación.

#### 4.3.1.1 Probabilidad en Asignación Aleatoria

Se va a usar el parámetro *Tráfico Cursado* ( $TC$ ) que en teoría de colas, por definición, es el número medio de peticiones en el sistema o el número medio de servidores ocupados. Se puede calcular a partir del *Tráfico Ofrecido* ( $TO$ ) y la *Probabilidad de Bloqueo* ( $P_L$ ):

$$TC = TO(1 - P_L)$$

Una forma de calcular la probabilidad de ocupación del sistema primario es dividiendo su  $TC$  entre el número de canales totales.

$$P_i = \frac{TC}{N} \quad \text{para } i = 1, 2, \dots, N.$$

Como se aprecia en la expresión anterior la probabilidad de ocupación de cualquier canal  $i$  es la misma en Asignación Aleatoria.

#### 4.3.1.2 Probabilidad en Asignación Secuencial Compacta

Para calcular la probabilidad de ocupación de los canales en este caso se plantea un proceso de nacimiento y muerte con una cadena idéntica a la de la *Figura 3.1* (Sistema Erlang-B con  $N$  servidores) y con asignación de canales compacta, tal y como se desarrolla en la *sección 4.2.2*.

De esta forma en el estado 1, sólo estará ocupado el primer canal; en el 2, el primero y el segundo; así sucesivamente hasta el estado  $N$ , en el que están todos los canales ocupados.

Así pues, la probabilidad de que el canal  $N$  esté ocupado será la probabilidad de que el sistema esté en estado  $N$ . La probabilidad de que el estado  $N-1$  esté ocupado es la probabilidad de que el sistema esté en el estado  $N$ , mas la probabilidad de que el sistema esté en el estado  $N-1$ , y así sucesivamente. Esto se puede generalizar mediante la siguiente expresión:

$$P_i = \sum_{j=i}^N P(j) \quad \text{para } i = 1, 2, \dots, N.$$

Siendo  $P_i$  la probabilidad de ocupación del canal  $i$ , y  $P(j)$  la probabilidad del estado  $j$  en el proceso de nacimiento y muerte descrito anteriormente.

#### 4.3.1.3 Probabilidad en Asignación Secuencial

En este caso, si se plantea el mismo proceso de nacimiento y muerte que en la sección 4.3.1.2 surge el problema de que, dado un estado, los canales que pueden estar ocupados no son únicos para ese estado. Por ejemplo, en el estado 1 podría estar ocupado el canal 1 o 2 o 3...o  $N$ , en el estado 2 podrían estar ocupados los canales 1 y 2, o 1 y 3 etc. De forma general cada estado se puede descomponer en  $\binom{N}{n}$  subestados, siendo  $N$  el número total de canales y  $n$  el estado del sistema de nacimiento y muerte.

Por tanto, el número total de subestados del sistema es:

$$N_{sub} = \sum_{n=0}^N \binom{N}{n} = 2^N$$

Una forma de proceder sería calcular la probabilidad de los subestados en la cadena de Markov propuesta y a partir de ello es sencillo calcular la probabilidad de ocupación de cada uno de los canales. El problema radica en que no hay expresiones cerradas para el cálculo de la probabilidad de los subestados para cualquier valor de  $N$ , por lo que se ha diseñado un algoritmo que para un cierto valor de  $N$ , calcula la probabilidad de ocupación de los canales. Dicho algoritmo sigue los siguientes pasos:

1. Se crea un vector de  $2^N$  posiciones, en cada una de la cuales se crea una estructura de datos donde se almacenará la información de cada subestado.
2. Se recorre el vector realizando las interconexiones de la cadena asíncrona de Markov entre los subestados.
3. Se plantea una ecuación de balance de flujo para cada uno de los subestados y la ecuación de conservación para el estado 0.
4. Se obtiene la probabilidad de cada subestado resolviendo las ecuaciones planteadas en el punto anterior de forma matricial.
5. Se calcula la probabilidad de ocupación de cada canal como la suma de las probabilidades de los subestados en los que ese canal está ocupado.

### 4.3.2 Cálculo del rendimiento

Una vez caracterizado el sistema por la probabilidad de que sus canales estén ocupados, se va a evaluar su rendimiento midiendo su throughput.

Se define  $Q_i$  como la probabilidad de que el canal  $i$  esté libre:

$$Q_i = 1 - P_i$$

Así pues, el número medio de canales libres que se espera encontrar al escanear los  $m$  primeros canales es:

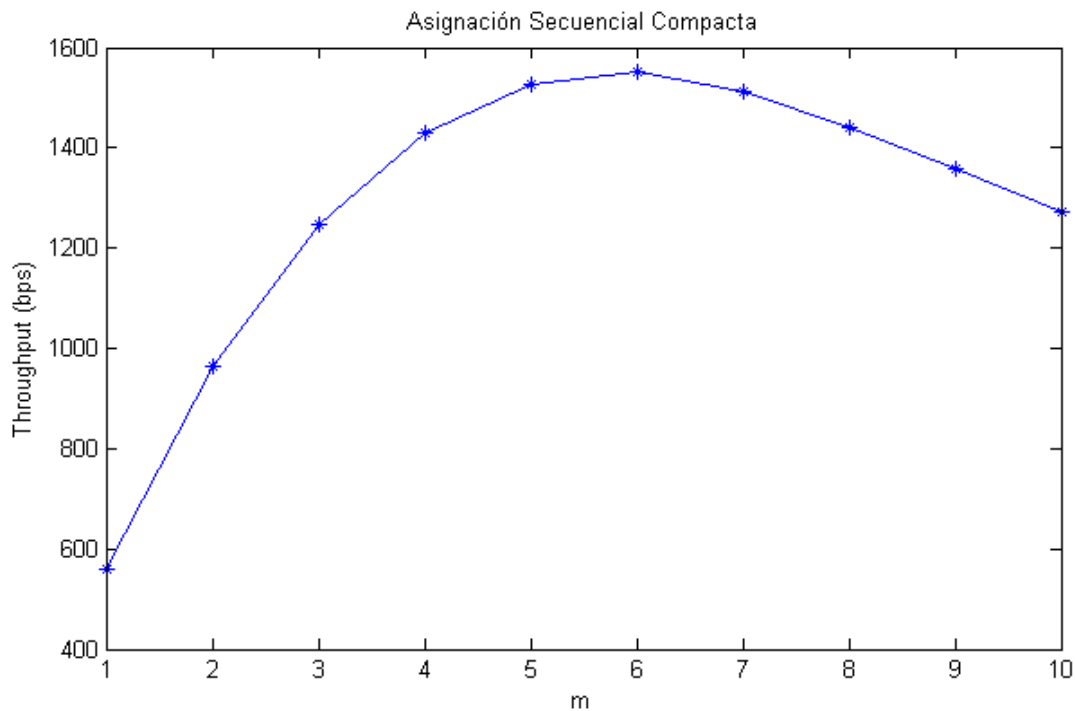
$$L = \sum_{i=1}^m Q_i$$

Por lo que, finalmente, el throughput del sistema al escanear  $m$  canales se calcula mediante la siguiente expresión:

$$\text{Throughput} = \frac{R \cdot T \cdot L}{T_s + t \cdot m + T}$$

Resulta interesante dibujar el valor del throughput barriendo todos los valores de  $m$  (número de canales que se escanean). De esta forma, se puede observar el valor de  $m$  que maximiza el throughput, si lo hubiera.

Como ejemplo se muestra en la *Figura 4.4* la curva que describe el throughput en función de  $m$  con una Asignación Secuencial Compacta y para un  $TO = 5$  Erlangs.



*Figura: 4.4. Cálculo analítico de  $m$  vs. throughput en Asignación Secuencial Compacta*

#### 4.4 Resultados de simulación

Una vez caracterizado el rendimiento del sistema de forma analítica, se va a simular y analizar su comportamiento de forma detallada, midiendo, además del throughput, el nivel de ocupación del sistema y caracterizándolo por el tipo de asignación. Posteriormente se comprobará que los resultados analíticos y los simulados coinciden dentro de los intervalos de confianza calculados.

### 4.4.1 Sistema vacío

Para una primera aproximación al problema se va a considerar el sistema vacío, es decir, no va a haber tráfico primario en él. Por tanto, siempre que un usuario secundario escanee un canal lo va a encontrar libre.

Como se puede ver de forma intuitiva, cuanto más canales se escaneen en una transmisión, más canales se usarán de forma simultánea y, por tanto, mayor será el throughput y la utilización del sistema. Y, por supuesto, a mayor valor de  $c$  mayor rendimiento.

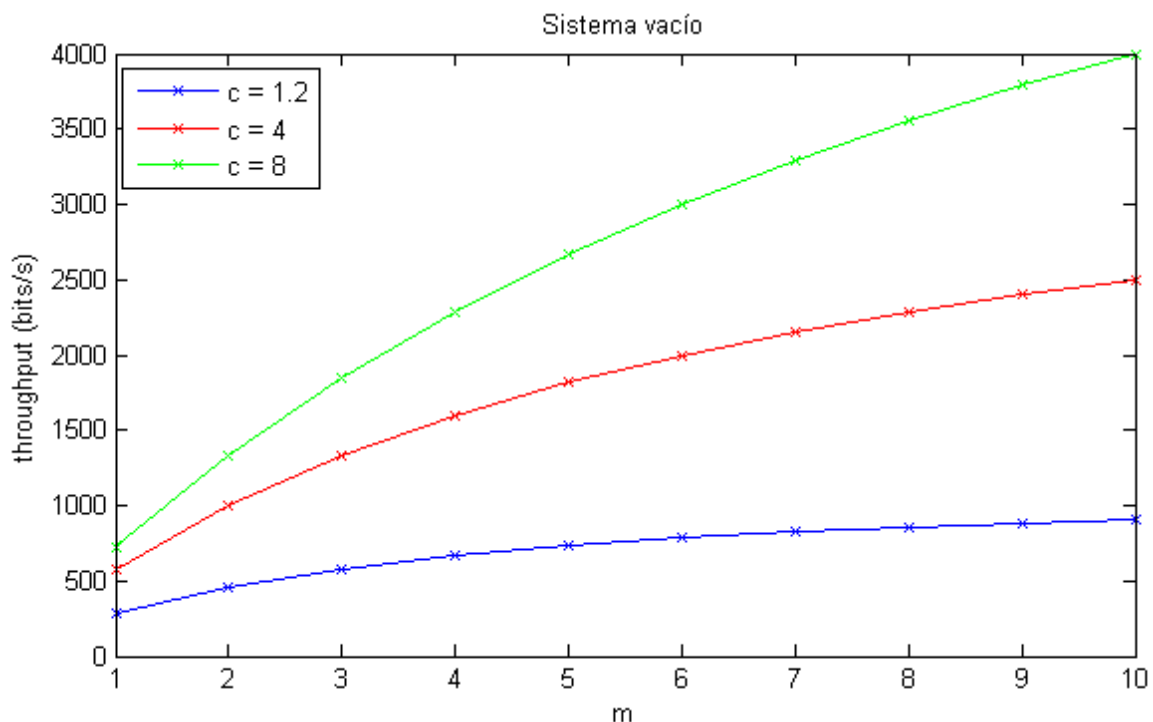


Figura: 4.5. Sistema vacío, throughput vs.  $m$

Se puede obtener el mismo resultado particularizando la expresión de la sección 4.3.2:

$$\text{Throughput} = \frac{R \cdot T \cdot m}{T_s + t \cdot m + T}$$

También se puede apreciar cómo la gráfica de la ocupación (Figura 4.6) del sistema evoluciona de forma similar ante los mismos parámetros.

Para la obtención de las gráficas de porcentaje de ocupación, para cada valor de  $m$ , se calcula el porcentaje de ocupación de cada canal y se promedia para obtener los valores que forman la curva.

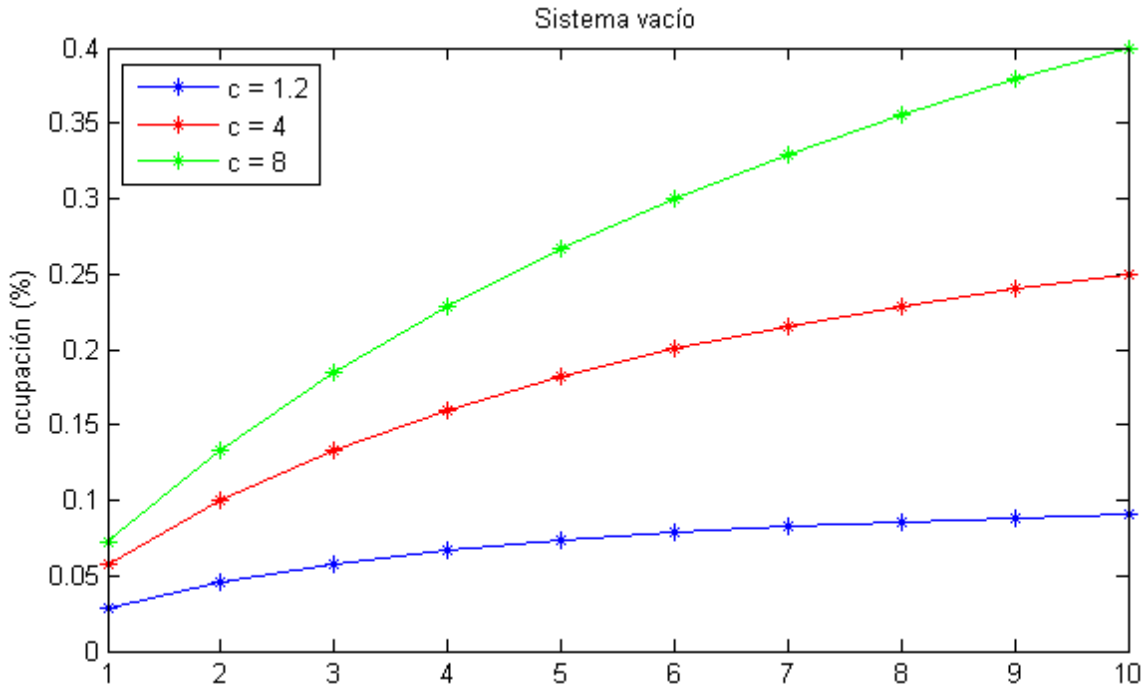


Figura: 4.6. Sistema vacío, ocupación vs. m

Debido al carácter determinista del protocolo con el sistema vacío, siempre se obtienen los mismos resultados ante los mismos parámetros, es decir, al eliminar el tráfico primario del sistema también se elimina la componente aleatoria de éste.

#### 4.4.2 Asignación Aleatoria

A continuación, se añade un tráfico primario con asignación aleatoria y se observan las diferencias y similitudes de los resultados respecto, a los del apartado anterior.

Si cuando el sistema está vacío hay una probabilidad de que el canal escaneado esté vacío igual a la unidad, ahora que se considera tráfico primario de Poisson no nulo, cuando se escanea un canal hay una probabilidad  $q$  de que el canal escaneado esté vacío, siendo  $q < 1$ .

Entendiendo así el sistema, se puede llegar a una conclusión similar a la anterior: ***Cuanto más canales se escaneen, más canales libres se encontrarán en media, por lo que mayor será el throughput y la utilización de canales.***

A continuación se muestran los resultados obtenidos en las simulaciones para diferentes cargas primarias.

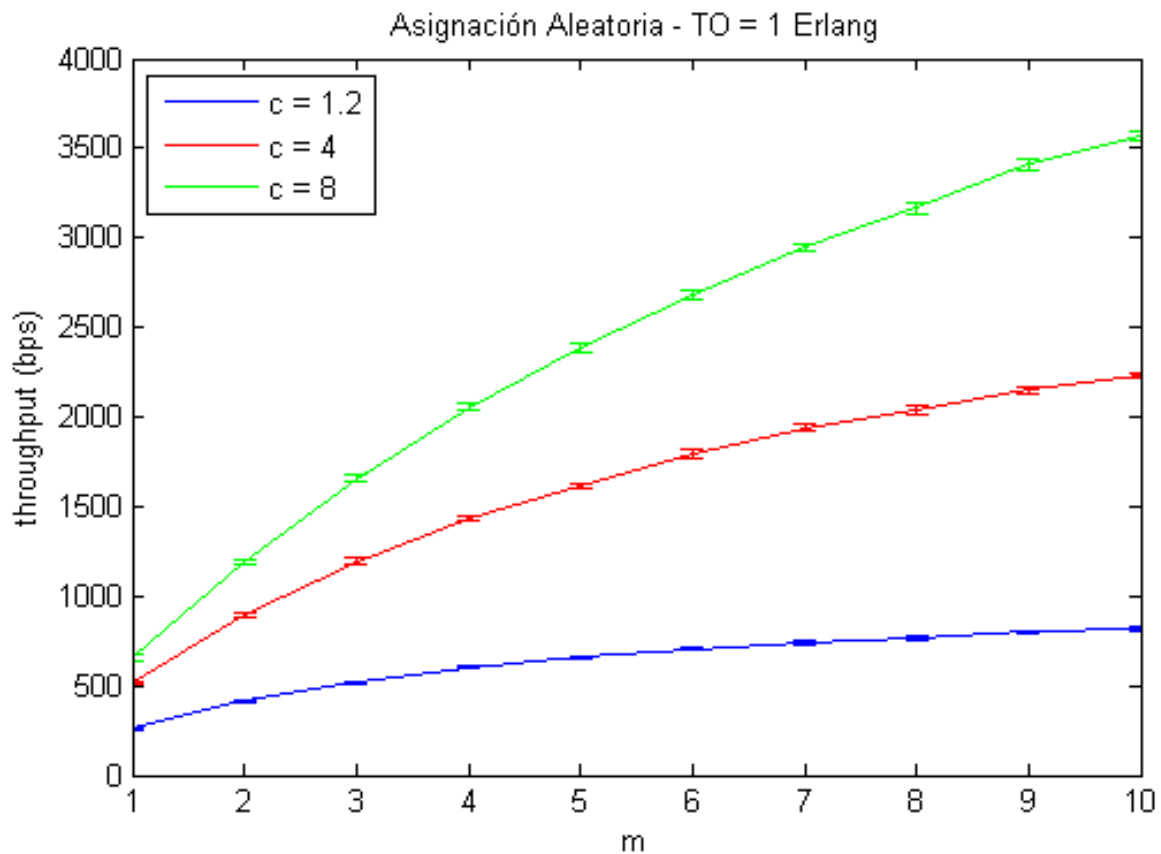


Figura: 4.7. Asignación Aleatoria, throughput vs. m, TO=1E

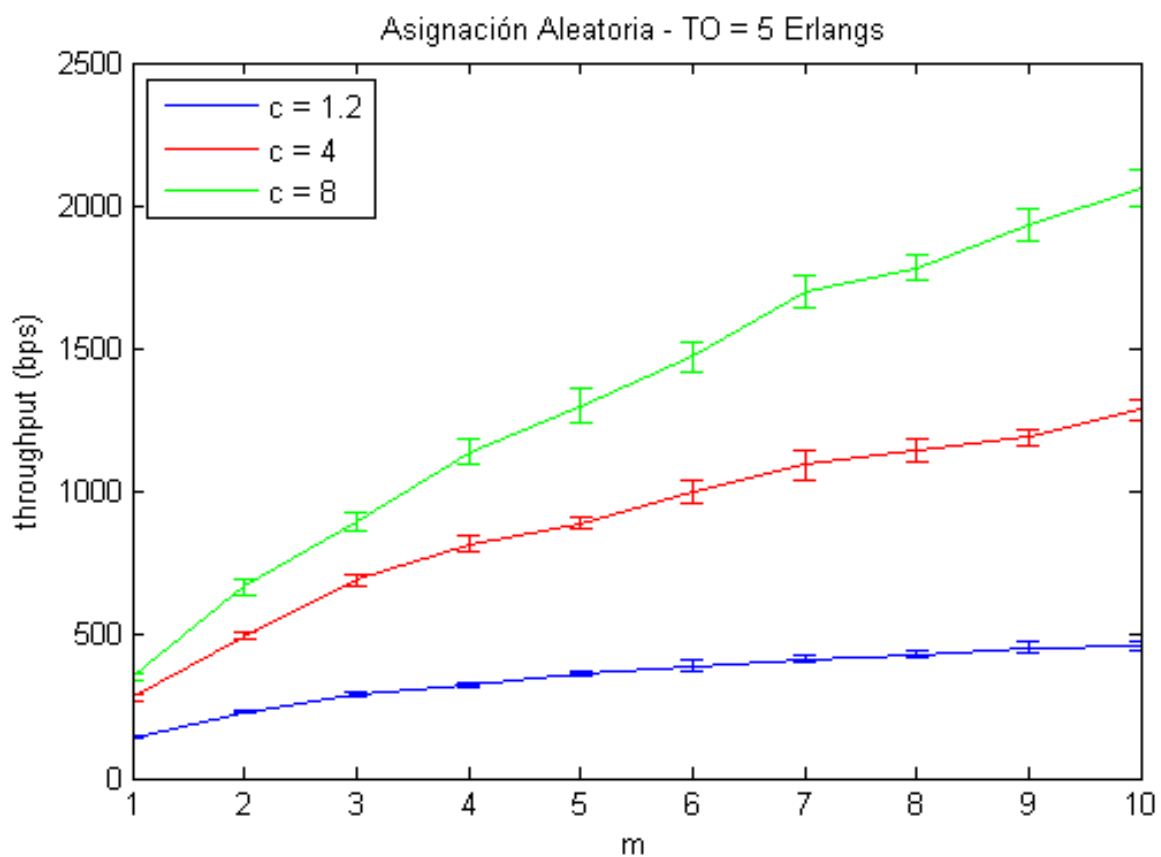


Figura: 4.8. Asignación Aleatoria, throughput vs. m, TO=5E



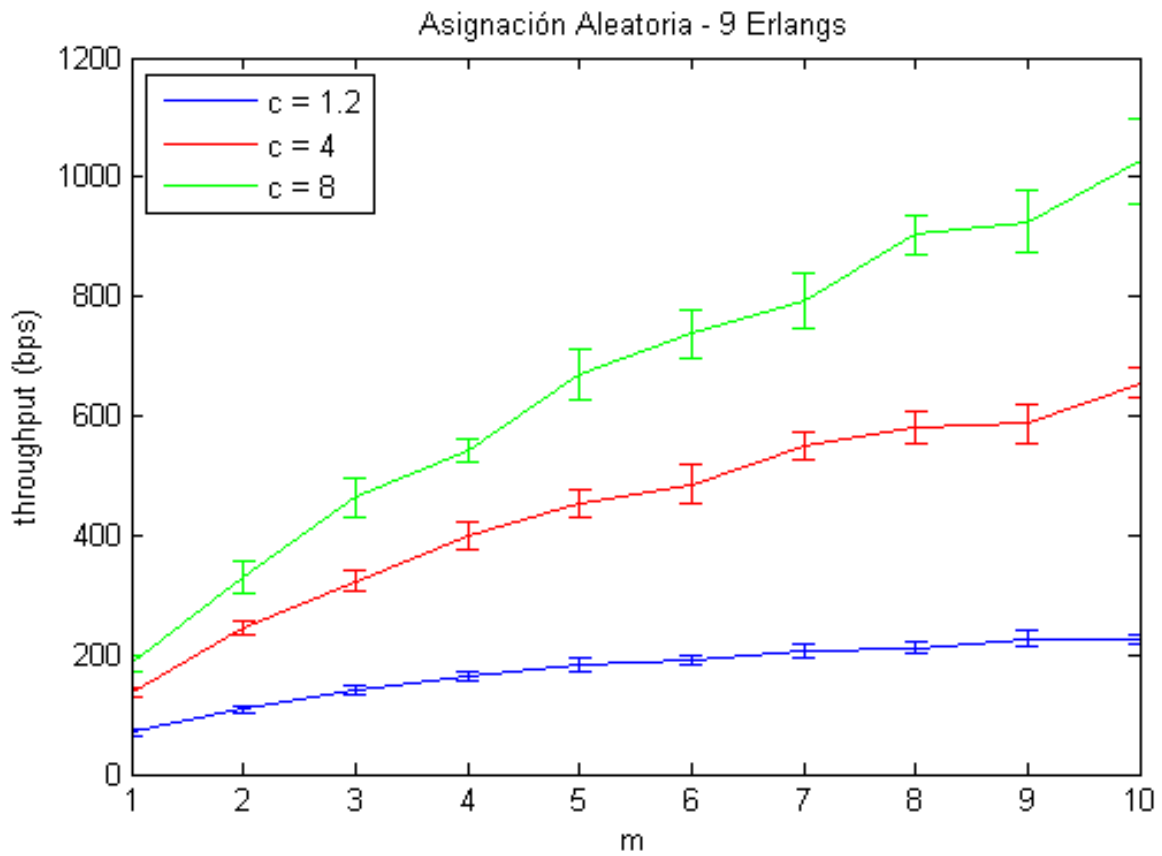


Figura: 4.9. Asignación Aleatoria, throughput vs. m,  $TO=9E$

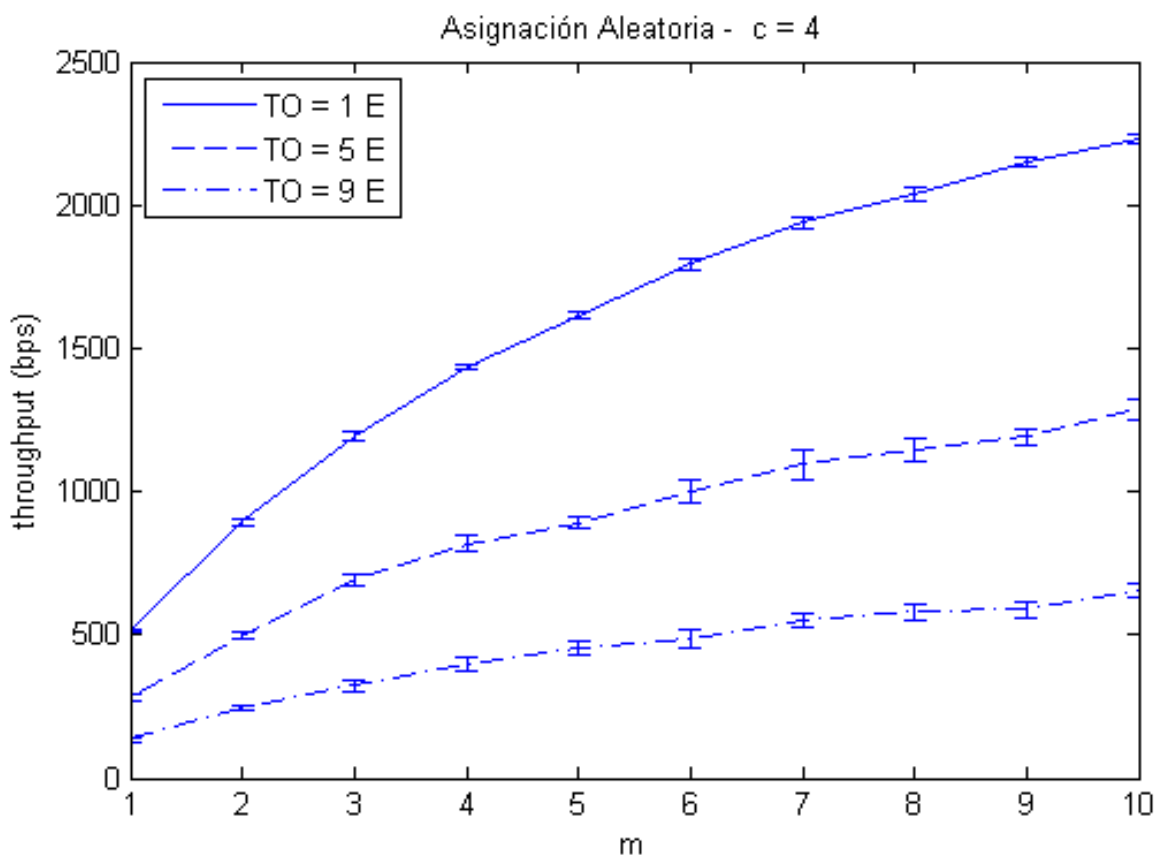


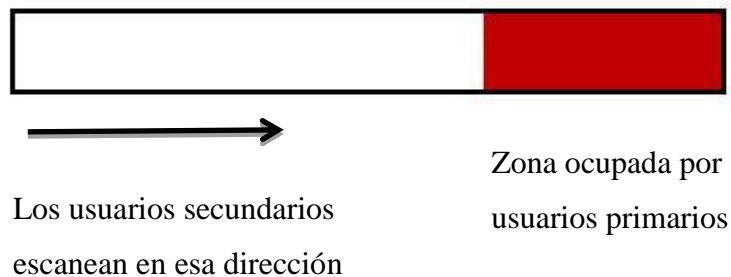
Figura: 4.10. Asignación Aleatoria, throughput vs. m,  $c = 4$

Se puede observar que, cuanto mayor es el tráfico ofrecido (y por tanto menos la probabilidad de que un canal esté libre), más se “achatan” las gráficas, pero su forma sigue siendo la misma. Esto se puede apreciar en la *Figura 4.10* donde para  $c = 4$ , se muestran las diferentes curvas del *throughput* en función de la carga del sistema.

En conclusión, si la asignación de los usuarios primarios es aleatoria siempre es ventajoso escanear todos los canales para maximizar el *throughput* y la *utilización del sistema*.

### 4.4.3 Asignación Secuencial Compacta

En este caso los usuarios primarios utilizan los canales de forma secuencial, de tal forma que no hay huecos entre los canales, como se ha explicado en la *sección 4.2.2*. Los usuarios secundarios empiezan a escanear por el lado contrario que los primarios, y van escaneando también de forma secuencial. Si escanean los suficientes canales llegará un punto en el que se encuentren con un canal ocupado, y a partir de ese límite los demás canales también estarán ocupados.



*Figura: 4.11. Diagrama ocupación Asignación Secuencial Compacta*

Se puede observar que en las figuras siguientes hay un máximo, que es un valor de  $m$  que nunca interesa sobrepasar, ya que a partir de ese valor el *throughput* empieza a disminuir. Ese máximo depende del tráfico primario, ya que cuantos más canales estén ocupados en media, menor será el número de canales que a un terminal secundario le interesa escanear para ser eficiente. El aumento del tráfico primario es equivalente a aumentar la zona de color rojo en la *Figura 4.11*.

Se vuelven a mostrar las gráficas para diferentes *ratios de transmisión* ( $c$ ) y es importante apreciar que el valor de  $m$  que maximiza el *throughput*, no varía de forma significativa con el valor de  $c$ , sólo en casos extremos.

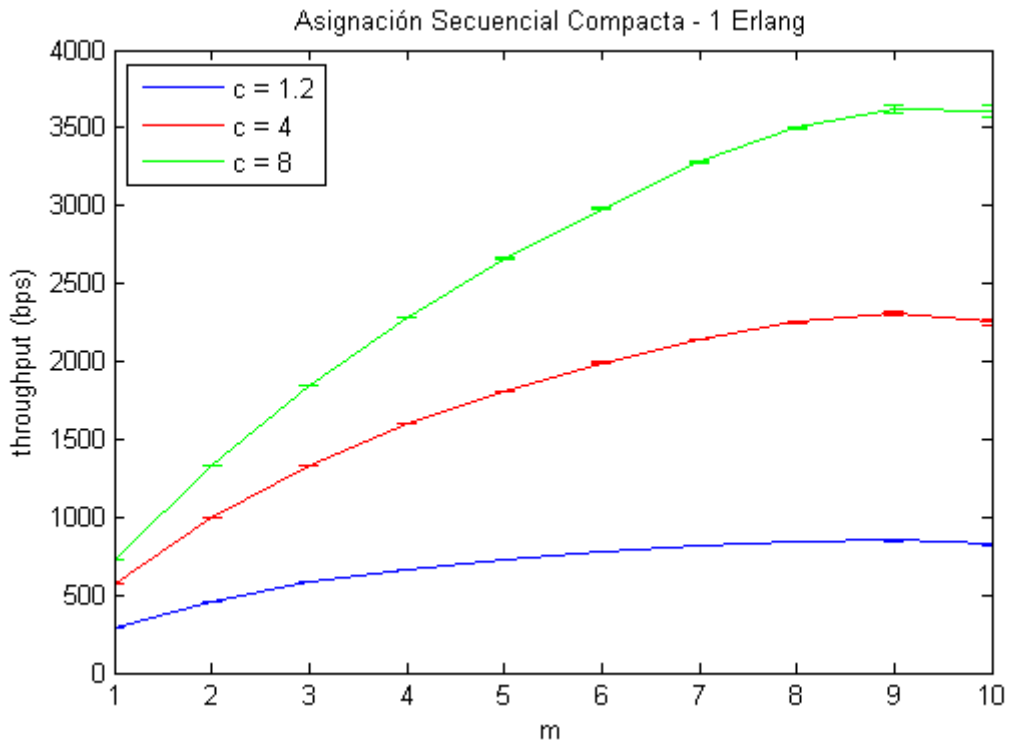


Figura: 4.12. Asignación Secuencial Compacta, throughput vs.  $m$ ,  $TO=1E$

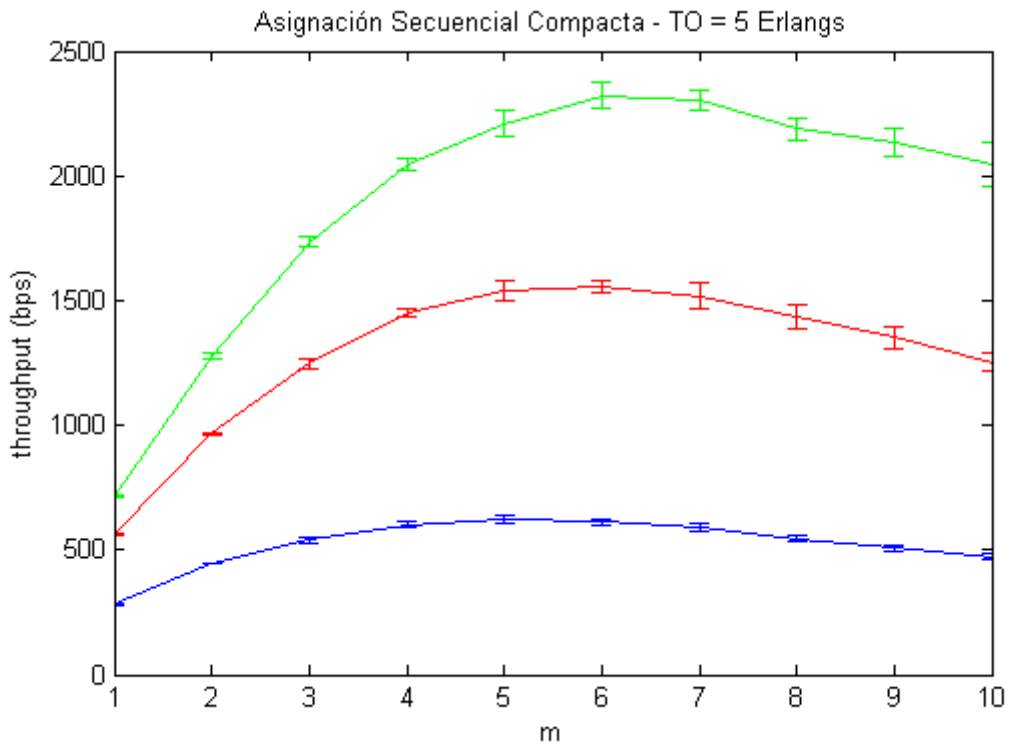


Figura: 4.13. Asignación Secuencial Compacta, throughput vs.  $m$ ,  $TO=5E$

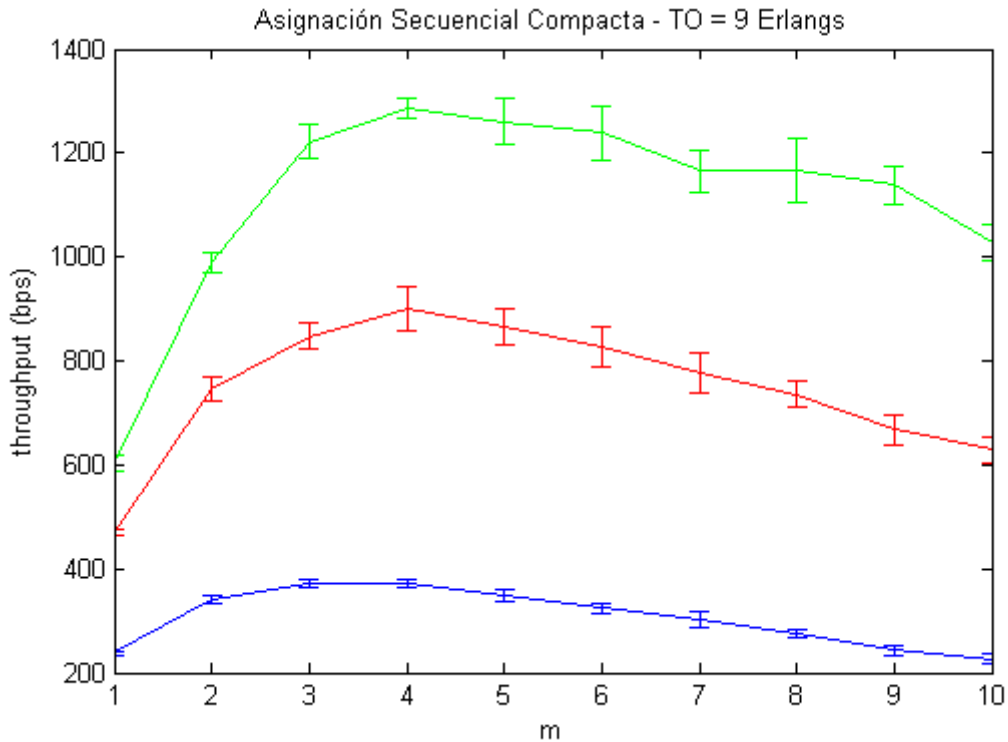


Figura: 4.14. Asignación Secuencial Compacta, throughput vs. m, TO=9E

Al igual que en el apartado anterior, las curvas del porcentaje de ocupación describen una forma similar a las del throughput. Esto es así debido a que el modelado del sistema primario (llegadas de Poisson, tiempos de servicio exponenciales, población infinita, etc.) implica una ocupación constante del sistema suponiendo un Tráfico ofrecido constante. Por eso, todo uso adicional de los canales supondrá una ocupación adicional del sistema, lo que quiere decir que cuanto mayor sea el throughput de los usuarios secundarios, mayor será la utilización del sistema. Entonces se puede afirmar que ***la optimización del throughput secundario implica una optimización del uso del sistema.***

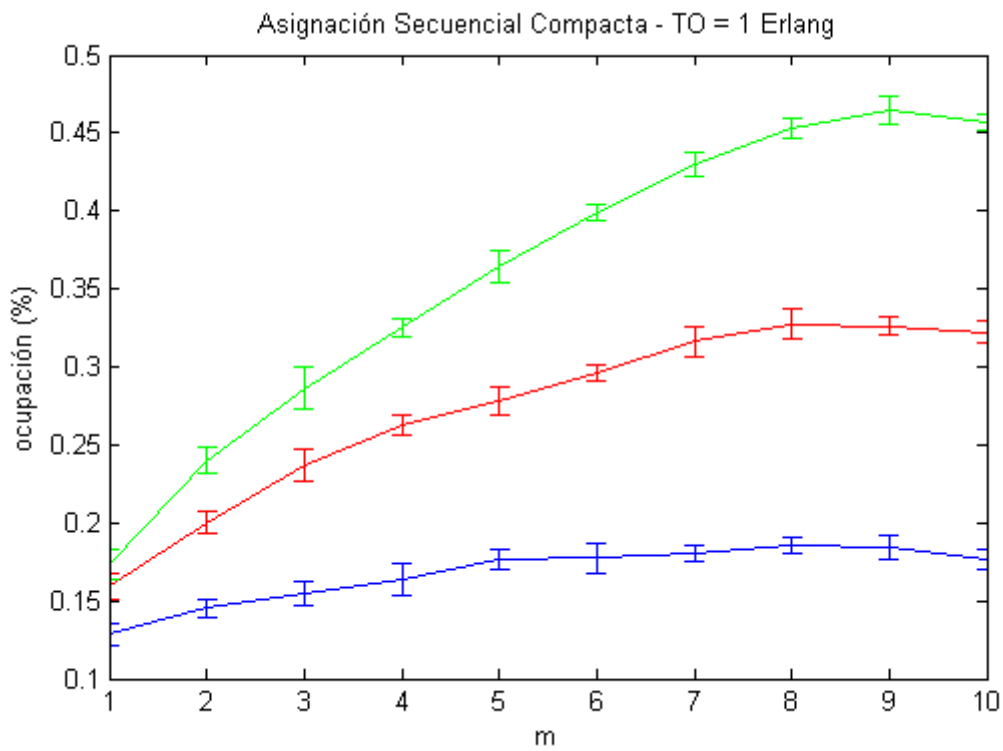


Figura: 4.15. Asignación Secuencial Compacta, ocupación vs.  $m$ ,  $TO=1E$

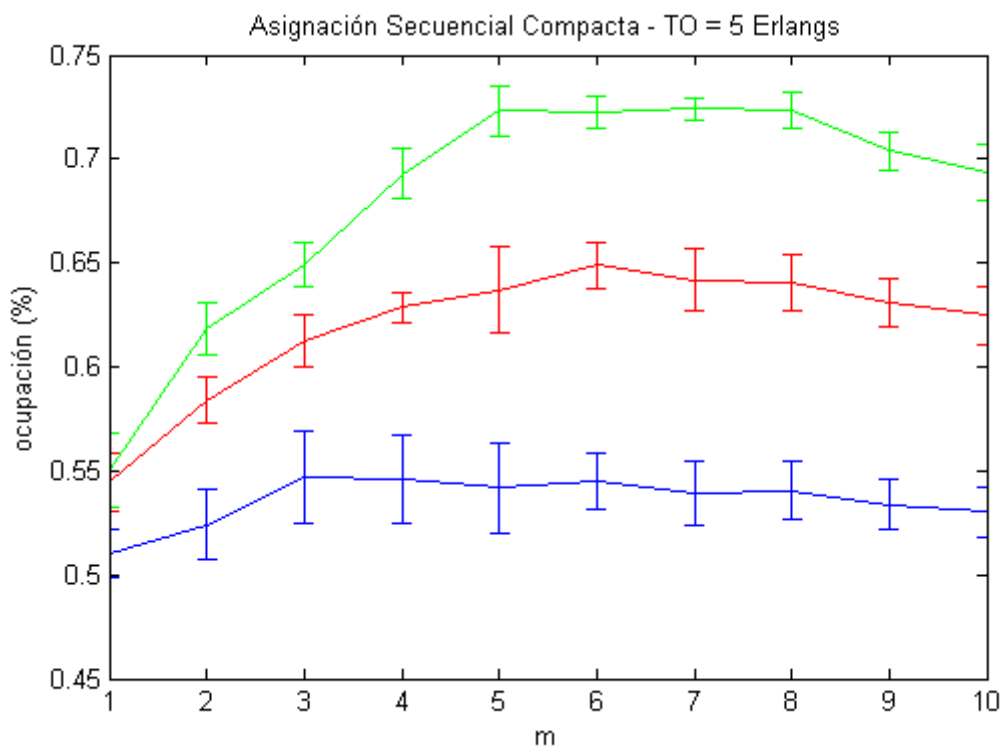


Figura: 4.16. Asignación Secuencial Compacta, ocupación vs.  $m$ ,  $TO=5E$

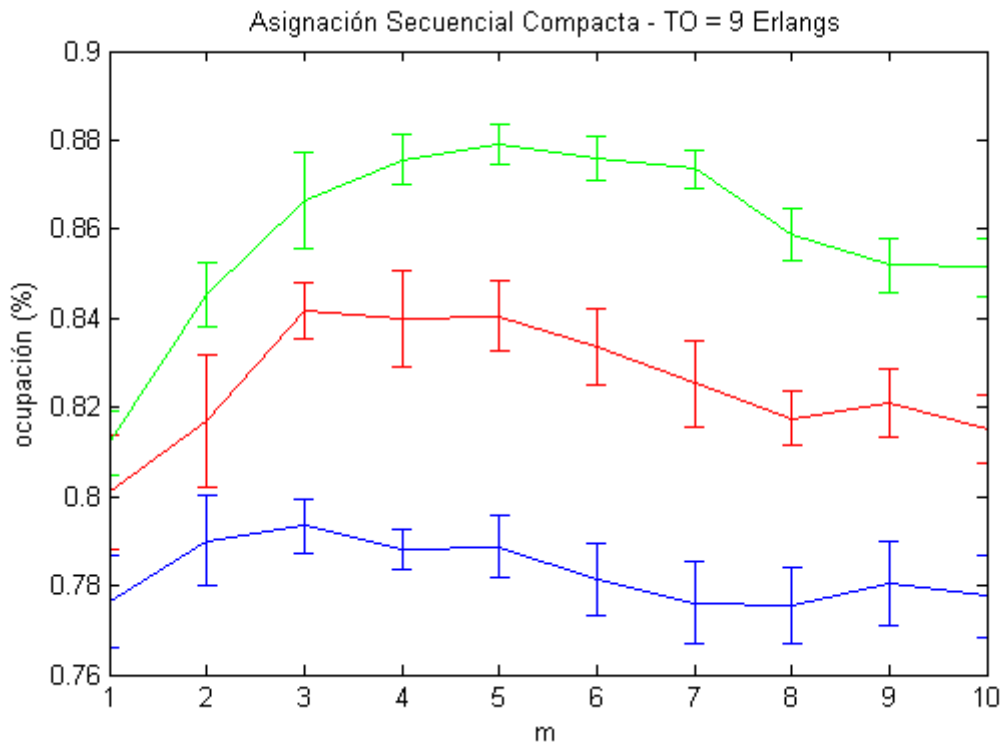


Figura: 4.17. Asignación Secuencial Compacta, ocupación vs. m, TO=9E

#### 4.4.4 Asignación Secuencial

En último lugar se va a estudiar el efecto de la Asignación Secuencial desarrollada en la *sección 4.2.3* sobre el desempeño del protocolo. No es casual que el estudio de esta asignación sea en último lugar. Los resultados de esta asignación son los más difíciles de predecir de forma intuitiva, por lo que el estudio previo de las otras asignaciones va a facilitar la interpretación de estos resultados.

Se indica en la *sección 4.5* que en la Asignación Secuencial Compacta hay un límite definido del cual nunca interesa pasar, ya que, a partir de ahí, los canales siempre van a estar ocupados. En este caso, al no haber compactación, aunque la tendencia sea ocupar los canales de un extremo del espectro, cuando los canales se van desocupando van quedando huecos libres. Por tanto, este límite que antes era definido, ahora queda “difuminado”, por lo que cerca de ese límite se obtendrá un comportamiento muy parecido al aleatorio (*sección 4.1.2*).

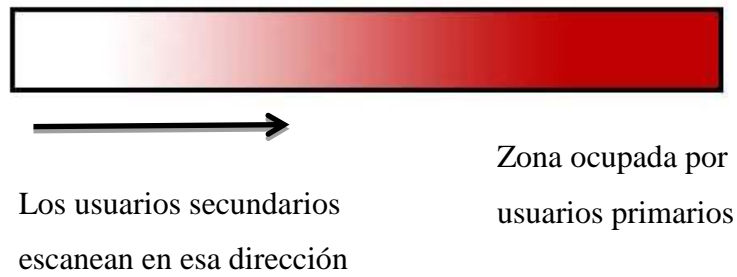


Figura: 4.18. Diagrama ocupación Asignación Secuencial

Mediante las figuras que se muestran seguidamente se puede comprobar que *cuando el tráfico primario es reducido, el sistema se comportará de forma muy similar a cuando se tiene Asignación Secuencial Compacta, y cuanto más tráfico primario hay más se parece el comportamiento del sistema al de uno con Asignación Aleatoria.*

Se observa que la Figura 4.19 en la que el tráfico primario es reducido ( $TO = 1$  Erlang), se asemeja mucho a la que se daba con Asignación Secuencial Compacta.

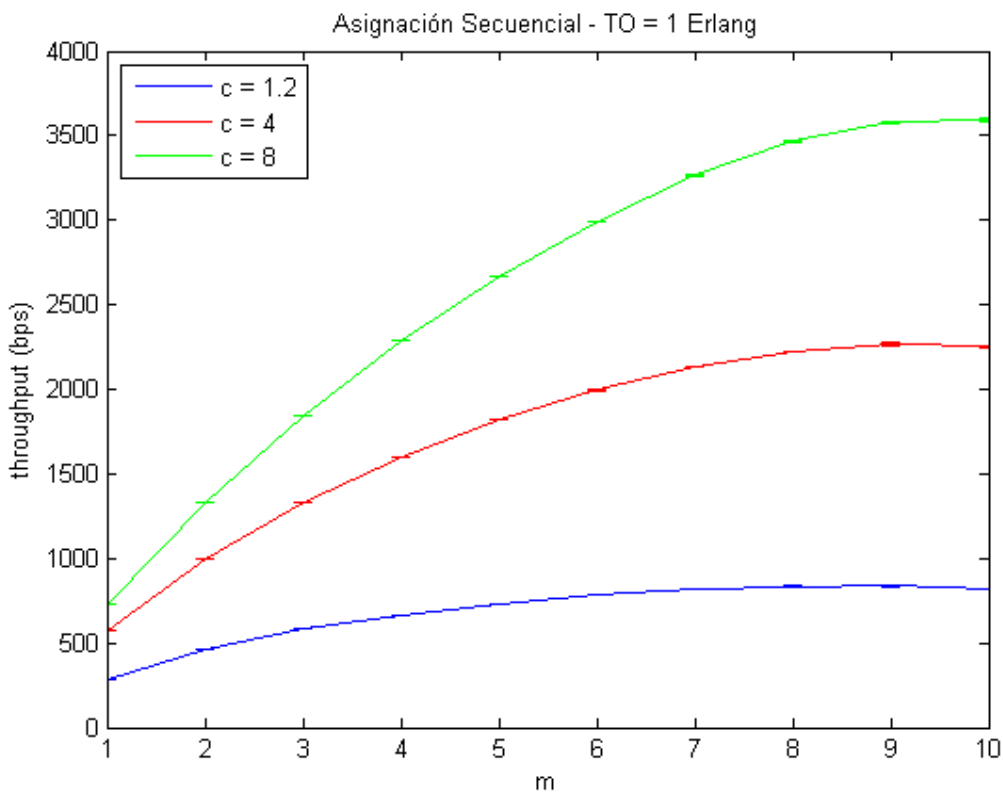
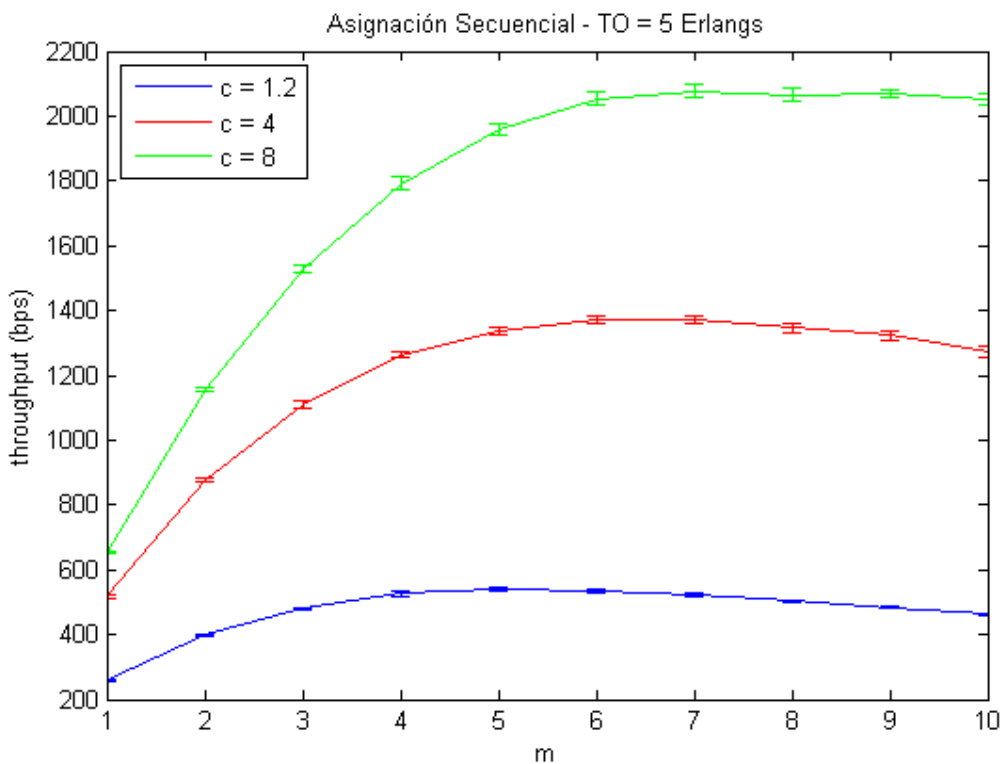


Figura: 4.19. Asignación Secuencial, throughput vs. m,  $TO=1E$

En la *Figura 4.20* el Tráfico Ofrecido es de 5 Erlangs (aproximadamente el 50% de los canales están siendo usados). Se puede apreciar que, al igual que en la Asignación Secuencial Compacta, hay un valor de  $m$  que maximiza el throughput que depende del TO y que no varía significativamente con el valor de  $c$ . Incluso los máximos se encuentran en torno a los mismos valores. Pero es importante apreciar, que ahora los máximos no son tan acusados como en el caso anterior, es decir, la diferencia entre el throughput en el máximo y el throughput en valores de  $m$  cercanos al máximo es pequeña. Esto es debido al comportamiento aleatorio que se da en la región donde los canales escaneados empiezan a ser usados por los usuarios primarios.



*Figura: 4.20. Asignación Secuencial, throughput vs. m, TO=9E*

Por último, en la *Figura 4.21* se muestra que el tráfico primario es elevado. En este caso casi desaparece el máximo para dejar paso a una zona casi plana, combinación entre Asignación Aleatoria (throughput siempre crece con  $m$ ), y Secuencial Compacta (throughput decrece de forma acusada para valores de  $m$  mayores que el máximo).



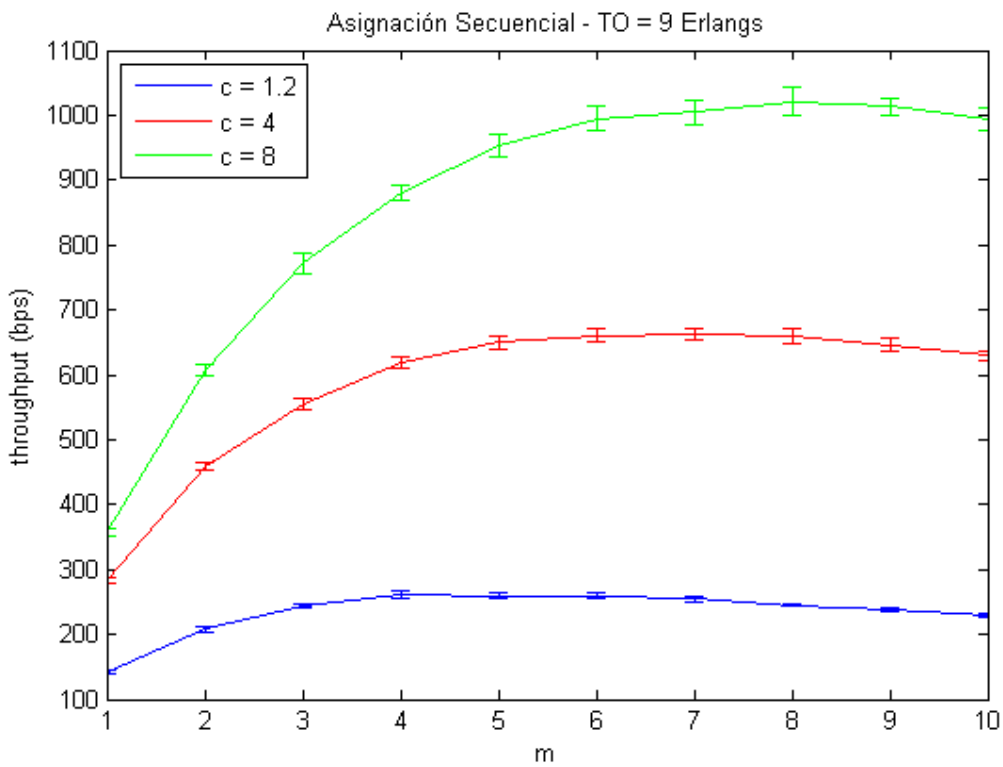


Figura: 4.21. Asignación Secuencial, throughput vs.  $m$ ,  $TO=9E$

En la *Figura 4.22* se muestra el impacto del tráfico primario en el throughput para un valor fijo de  $c$ .

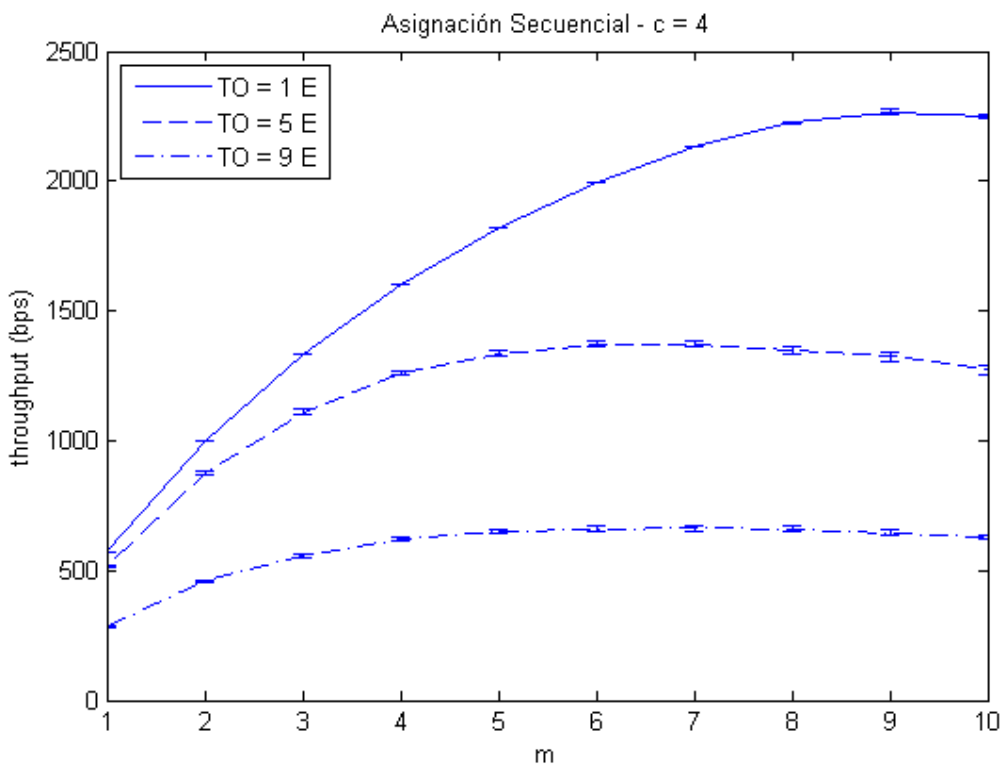


Figura: 4.22. Asignación Secuencial, throughput vs.  $m$ ,  $c = 4$

Y como se confirma en el apartado anterior, las curvas de la ocupación tienen el mismo comportamiento que las del throughput.

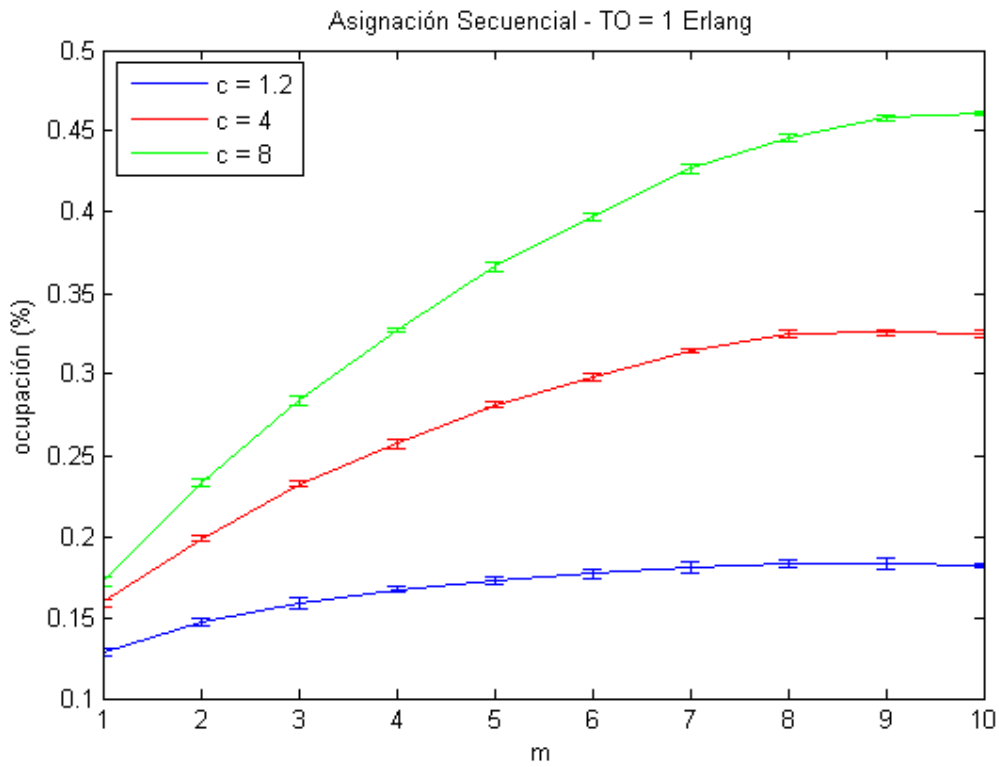


Figura: 4.23. Asignación Secuencial, ocupación vs. m, TO=1E

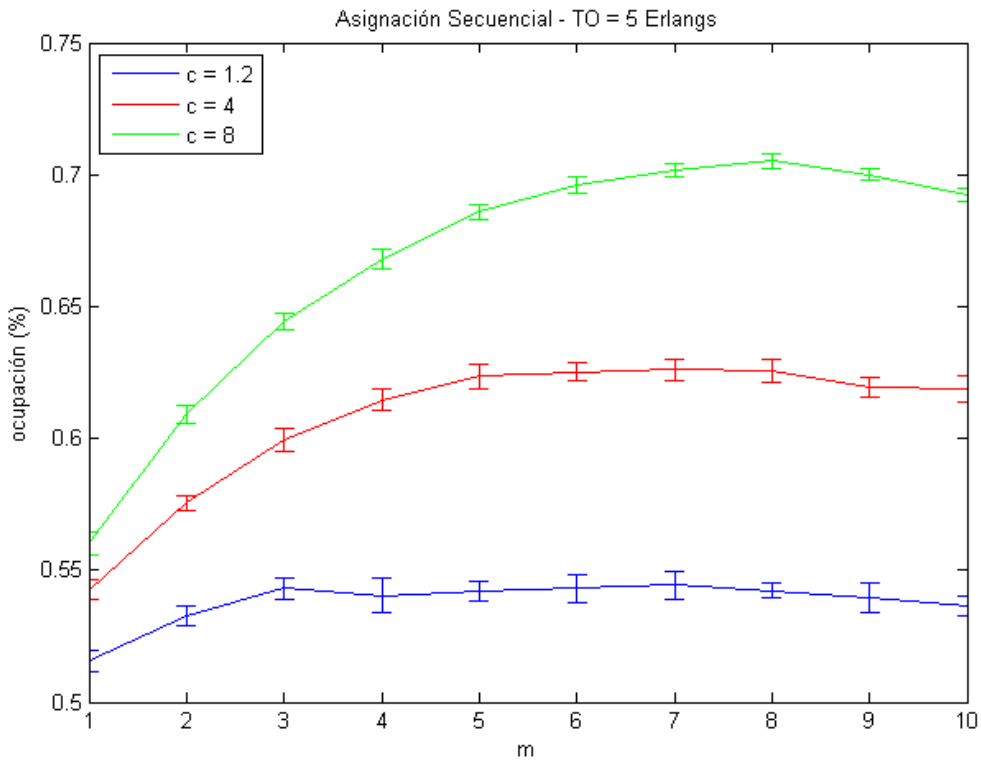
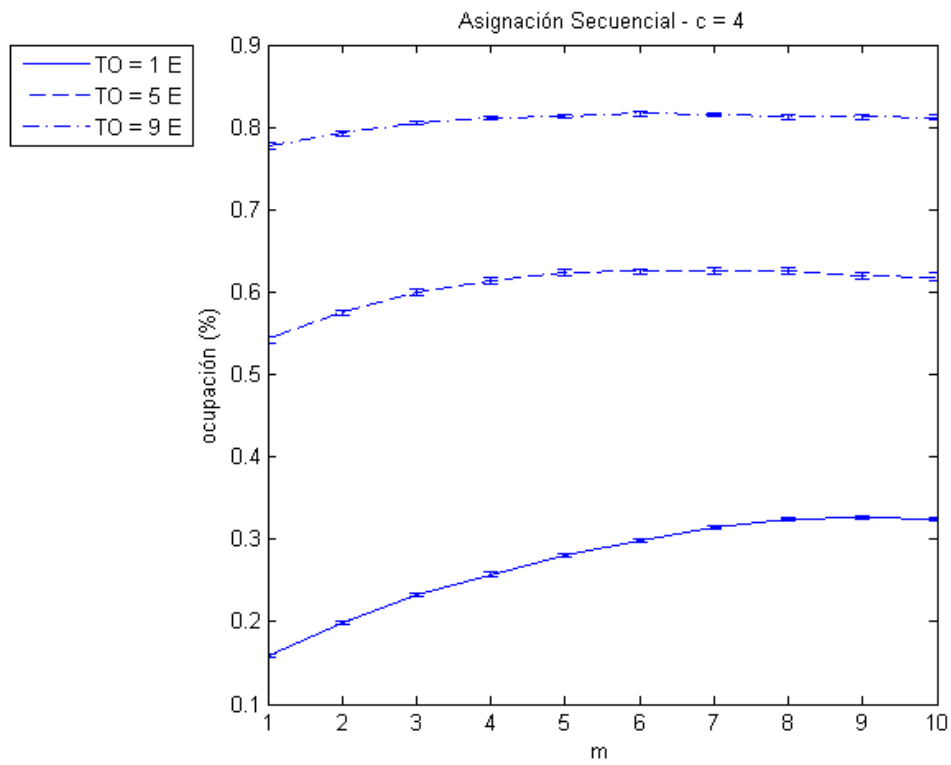


Figura: 4.24. Asignación Secuencial, ocupación vs. m, TO=5E

En la *Figura 4.25* se muestra la ocupación del sistema para  $c = 4$  y para diferentes valores del tráfico primario.



*Figura: 4.25. Asignación Secuencial, ocupación vs. m, c = 4*

## 4.5 Convergencia de resultados

A continuación, se va a comprobar la coincidencia entre los resultados de las simulaciones y los obtenidos con las funciones analíticas desarrolladas en la *sección 4.3.2*. Se dibujan las *Figuras 4.8, 4.13 y 4.20* y a continuación se superponen en color negro las curvas obtenidas con la función analítica. Se puede observar cómo coinciden dentro de los intervalos de confianza, que es de un 95%.

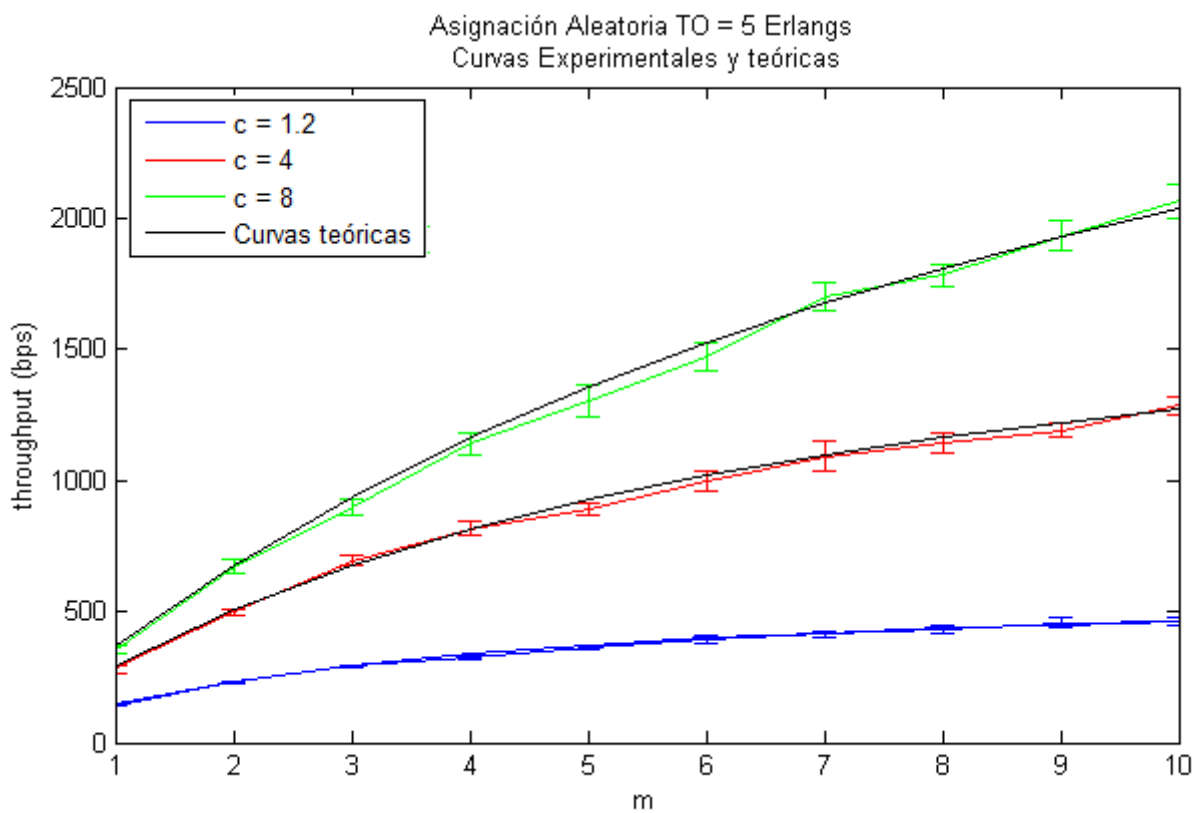


Figura: 4.26 Asignación Aleatoria, throughput vs.  $m$ ,  $TO = 5$  Erlangs  
Curvas Experimentales y teóricas

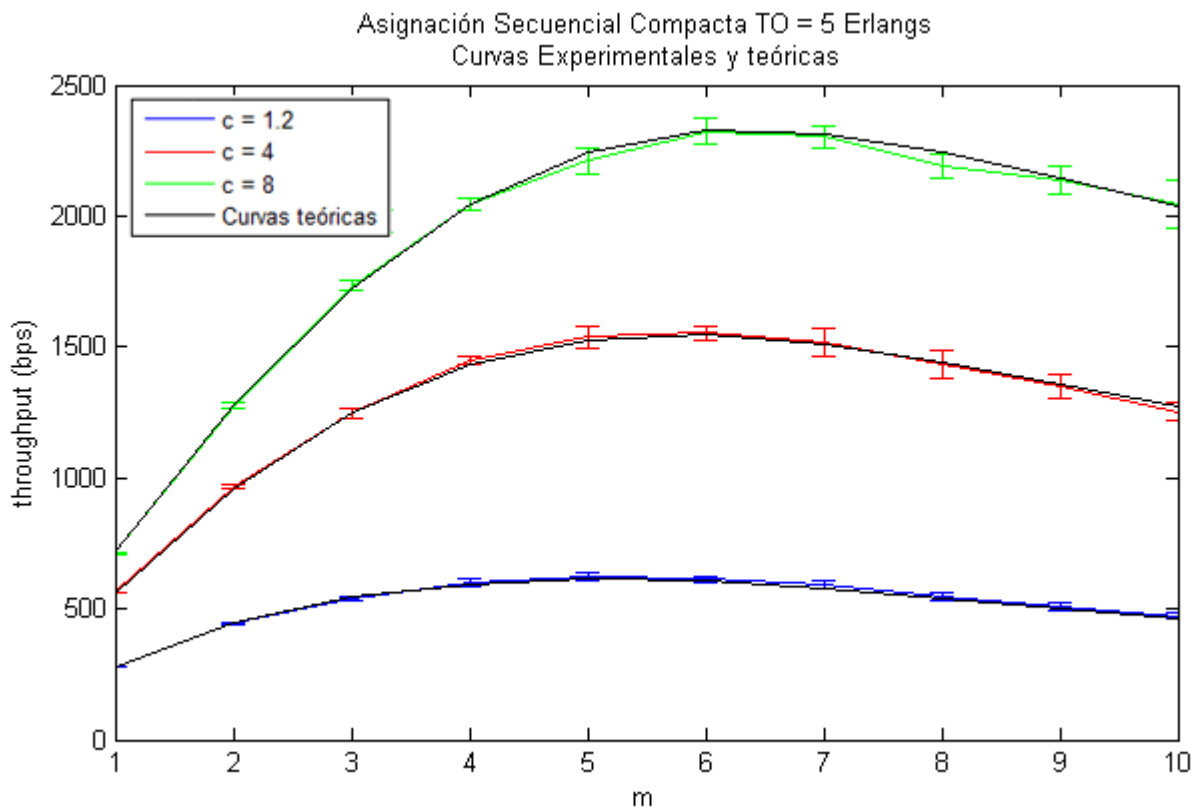


Figura: 4.27. Asignación Secuencial Compacta, throughput vs.  $m$ ,  $TO = 5$  Erlangs  
Curvas Experimentales y teóricas

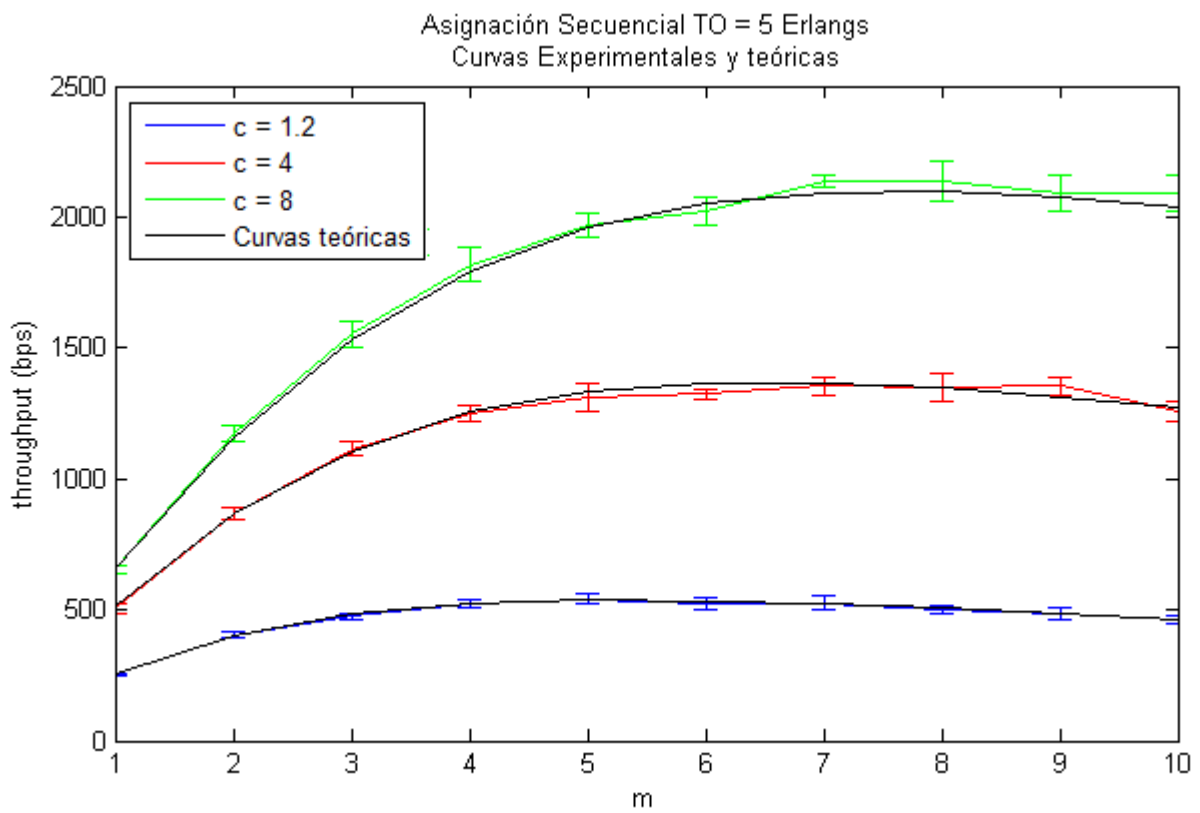


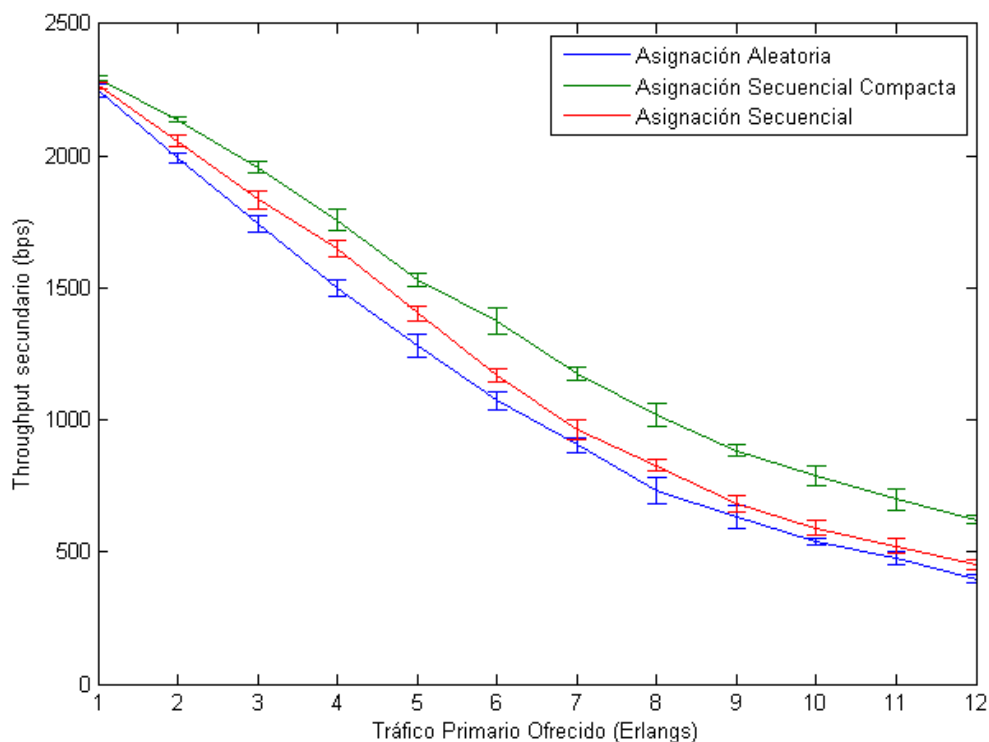
Figura: 4.28. Asignación Secuencial, throughput vs.  $m$ ,  $TO = 5$  Erlangs  
Curvas Experimentales y teóricas

## 4.6 Conclusiones

Como último punto del capítulo se resumen los resultados más relevantes que derivan de éste:

- Maximizar el throughput implica buscar una solución de compromiso entre el número de canales por los que se transmite de forma simultánea y el número de canales que se escanean.
- Se maximiza el throughput si y solo si se maximiza la función de utilización del sistema.
- Si los usuarios primarios usan Asignación Aleatoria siempre es óptimo escanear todos los canales.
- Si los usuarios primarios usan Asignación Secuencial Compacta, se encuentra un valor de  $m$  que maximiza el *throughput* (y por tanto también la *utilización del espectro*) que depende del tráfico primario.
- Si los usuarios primarios usan Asignación Secuencial, se puede interpretar el comportamiento del sistema como una combinación entre el sistema con Asignación Aleatoria y el sistema con Asignación Secuencial Compacta, y, además, para tráficos primarios altos, no se encuentran máximos definidos, por lo que a priori resulta más difícil optimizar un sistema con este tipo de asignación.
- Era necesario ignorar la restricción impuesta por el parámetro  $n$  para poder llevar a cabo este estudio, y, gracias a ello, se puede encontrar la mejor solución al problema cuando sí hay restricción de este parámetro, como se explica a continuación. Si se plantea el problema con un tráfico primario arbitrario, un valor óptimo de  $m$  igual a  $m_{\text{óptimo}}$  y un valor del parámetro  $n$  igual a  $n_{\text{max}}$ , el terminal secundario va a escanear canales hasta intentar llegar a  $m_{\text{óptimo}}$  canales escaneados. Si antes de llegar a ese valor se encuentran  $n_{\text{max}}$  canales libres, termina la fase de escaneo y se pasa a la de transmisión.
- Los resultados obtenidos de forma analítica y los obtenidos mediante simulación coinciden, por lo que quedan validados.

En la *Figura 4.29* se muestra el throughput obtenido por el valor óptimo de  $m$  en cada caso, en función del  $TO$  primario y para los tres tipos de asignación. Es decir, la figura muestra los valores máximos de throughput que se pueden obtener para cada valor del  $TO$ , lo que sirve para ver qué tipo de asignación es más favorable para el rendimiento de los usuarios secundarios. Al igual que en figuras anteriores, se mantiene  $N = 10$  y  $c = 4$ .



*Figura: 4.29. Throughput máximo vs  $TO$  para todas las asignaciones*

Se pueden extraer dos conclusiones importantes de la *Figura 4.29*:

Se corrobora que cuanto más tráfico hay en Asignación Secuencial, más se parece el comportamiento del sistema al de uno con Asignación Aleatoria. Se puede comprobar que cuando el tráfico es reducido, el throughput en Asignación Secuencial es el 96% del tráfico en Asignación Secuencial Compacta, mientras que si hay mucho tráfico este porcentaje baja hasta el 72% (más cercano al de Asignación Aleatoria que es un 63%).

Como era de esperar, los mejores resultados se obtienen con Asignación Secuencial Compacta, ya que, al concentrarse toda la ocupación primaria a un lado del espectro y sin huecos libres, deja usar más canales a los usuarios sin licencia.

# Capítulo 5

## Escaneo Inteligente

---

### 5.1 Objetivos

Partiendo de los resultados del capítulo anterior, en el que se estudia el comportamiento del protocolo ante los diferentes tipos de asignación, se plantea el siguiente capítulo, que tiene como objetivo presentar una serie de políticas de escaneo que optimicen el rendimiento del protocolo.

En primer lugar, se plantean algunas políticas de parada y se muestran los resultados mediante simulación.

En segundo lugar, se formula el problema de escaneo como un problema de *Optimal Stopping* o *Parada Óptima*, por lo que se plantea en primer lugar de forma analítica, para seguidamente proceder a su programación y simulación.

Debido a la complejidad computacional de este método, se plantea un método similar pero con menor coste, *k-stage look-ahead* [5].

Por último, se comparan todas las políticas expuestas y se extraen conclusiones.

Todas las simulaciones del capítulo se efectúan con valores de  $N = 10$  y  $c = 4$ .

### 5.2 Políticas de escaneo

A continuación, se detallan algunas posibles políticas de escaneo. Aunque algunas pueden parecer triviales, aplicadas a un sistema con un tipo de asignación concreto pueden ofrecer buenos resultados, incluso en algún caso, el mejor resultado posible.

#### 5.2.1 Escaneo hasta el final

Esta es la política más simple ya que el usuario secundario se limita a escanear todos los canales sin tener en cuenta ninguna consideración.



Es la mejor opción cuando los usuarios primarios usan Asignación. Si el tráfico primario es muy bajo, también ofrece buenos resultados sea cual sea la asignación primaria.

### 5.2.2 Escaneo hasta ocupado

Es una política también sencilla, ya que la única consideración que tiene que tener el terminal secundario es que tiene que parar cuando encuentre en canal ocupado.

Los casos extremos son: cuando, en el primer escaneo, se encuentra el canal ocupado, caso en el que no escanearía ningún canal más y tampoco podría transmitir por ninguno; y cuando todos los canales están libres, caso en el que escanearía todos los canales y podría transmitir también por todos ellos si no hay ninguna restricción de otro tipo.

Esta política es óptima cuando la asignación primaria es Secuencial Compacta, ya que, como se desarrolla en la *sección 4.4.3*, a partir del primer canal que se encuentre ocupado todos los demás también lo van a estar, por lo que interesa parar de escanear cuando se encuentre este primer canal ocupado.

Para Asignación Primaria Aleatoria no ofrece buenos resultados sea cual sea la carga del sistema, ya que en este caso el número de canales escaneados es aleatorio, como la ocupación de los canales.

Si la Asignación Primaria es Secuencial, se obtienen resultados aceptables para tráfico primarios medios o bajos, pero no es la política más recomendable.

### 5.2.2 Escaneo hasta $m$ óptimo

Esta política consiste en escanear hasta llegar al valor óptimo de  $m$ , es decir, el valor con el que se obtiene un throughput máximo.

Esta política, en general, ofrece buenos resultados independientemente de la asignación primaria.

Para la Asignación Secuencial ésta es la política óptima, y para el caso concreto de Asignación Aleatoria, como  $m$  óptimo siempre es  $N$ , esta política es equivalente a la de *Escaneo hasta el final*.

Ésta es la política más complicada de implementar ya que se requiere conocer a priori los valores óptimos de  $m$  en función del tráfico primario. Es necesario obtener una función entera que para cada valor de TO (Real) devuelva el valor óptimo de  $m$  (Entero).

$$f : \mathfrak{R} \rightarrow \mathbb{Z}$$

Además, dicha función sólo podrá devolver valores comprendidos en el intervalo  $[1, N]$ , que son los posibles valores del parámetro  $m$ .

Otra dificultad añadida es que, para poder aplicar la función anterior, es necesario estimar constantemente el tráfico primario. Dicha estimación se podría realizar mediante el escaneo de canales de los terminales secundarios, pero tiene el inconveniente de que si la estimación no es muy precisa tampoco lo será el valor de  $m$  obtenido, por lo que el sistema no funcionará bien del todo.

Para este estudio se han obtenido las relaciones entre TO y  $m$  óptimo de forma empírica.

### 5.2.3 Resultados de las políticas expuestas

A continuación se va a mostrar el rendimiento de las políticas expuestas en los apartados anteriores en función del tráfico primario ofrecido. De aquí se pueden extraer qué políticas son más ventajosas para cada tipo de asignación.

En primer lugar, se muestra la figura obtenida para la asignación aleatoria (*Figura 5.1*). Se puede observar cómo la mejor opción es escanear todos los canales y cómo los resultados obtenidos con la política *Escaneo hasta Ocupado* no son buenos por las razones indicadas anteriormente.

En el caso de Asignación Secuencial Compacta (*Figura 5.2*), se observa que los valores obtenidos con la política *Escaneo hasta Ocupado* son los mejores posibles. Con *Escaneo hasta  $m$  Óptimo* se obtienen buenos resultados, cercanos a los óptimos. Si se usa la política de *Escaneo hasta el final*, los resultados tienden a empeorar conforme aumenta el tráfico ofrecido primario.

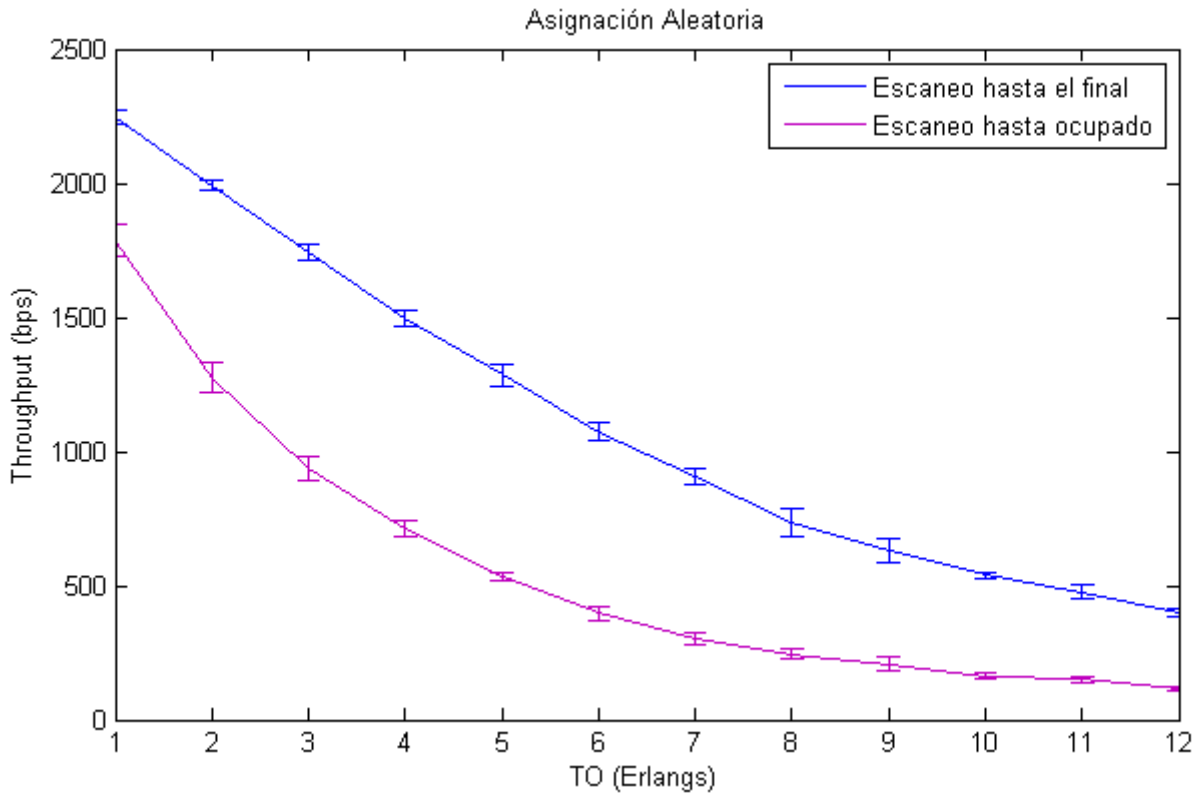


Figura: 5.1. Políticas de escaneo en Asignación Aleatoria

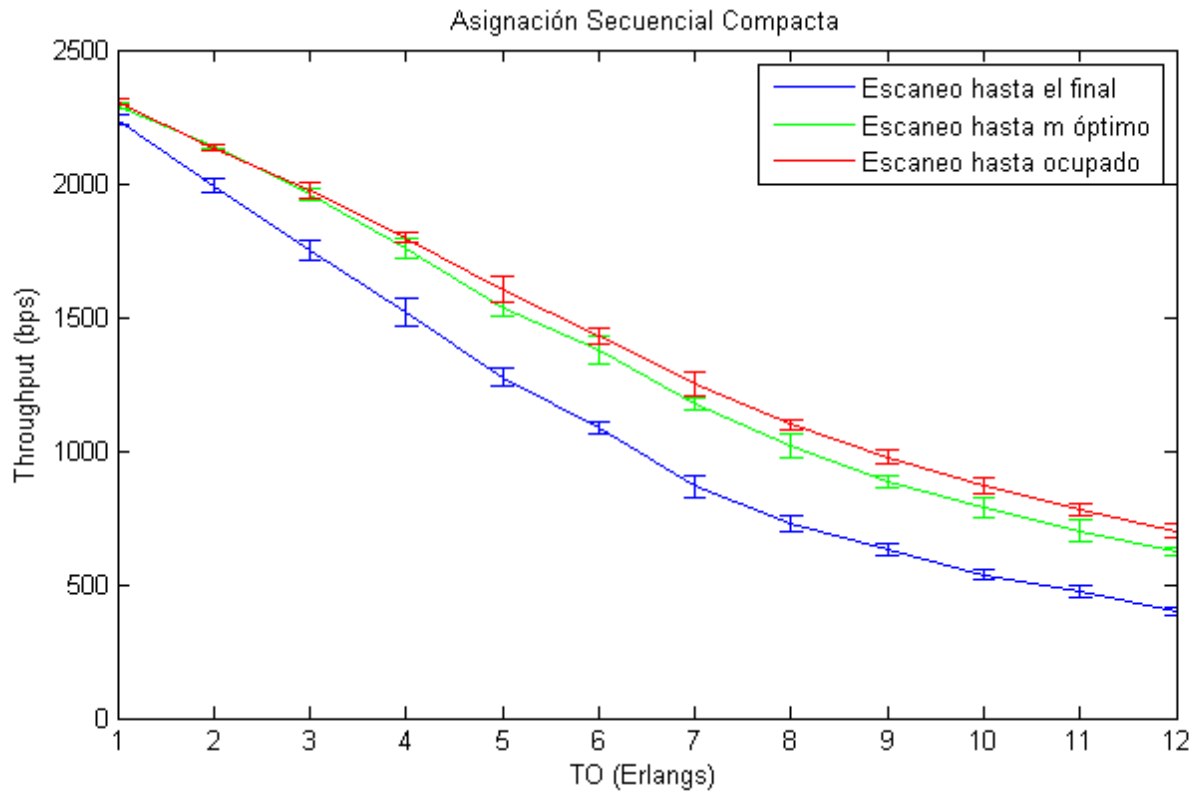


Figura: 5.2. Políticas de escaneo en Asignación Secuencial Compacta

Por último se aborda el caso de Asignación Secuencial (Figura 5.3). Se observa, por un lado, que la política *Escaneo hasta m Óptimo* es la mejor para este caso, ya que obtiene buenos resultados independientemente del tráfico. Para entender la tendencia de las otras dos políticas conviene aproximar el sistema a un sistema con Asignación Secuencial Compacta cuando tiene poco tráfico primario y a un sistema con Asignación Aleatoria cuando la ocupación primaria es mayor, como se explica en el capítulo anterior. Si se observan los extremos de la gráfica, cuando el tráfico primario es reducido se obtienen muy buenos resultados con *Escaneo hasta ocupado* (como ocurre en Asignación Secuencial Compacta), en cambio, conforme aumenta el tráfico los resultados obtenidos con esta política empeoran cada vez más, y con *Escaneo hasta el Final* se obtienen valores cercanos a los óptimos (semejanza a Asignación Aleatoria).

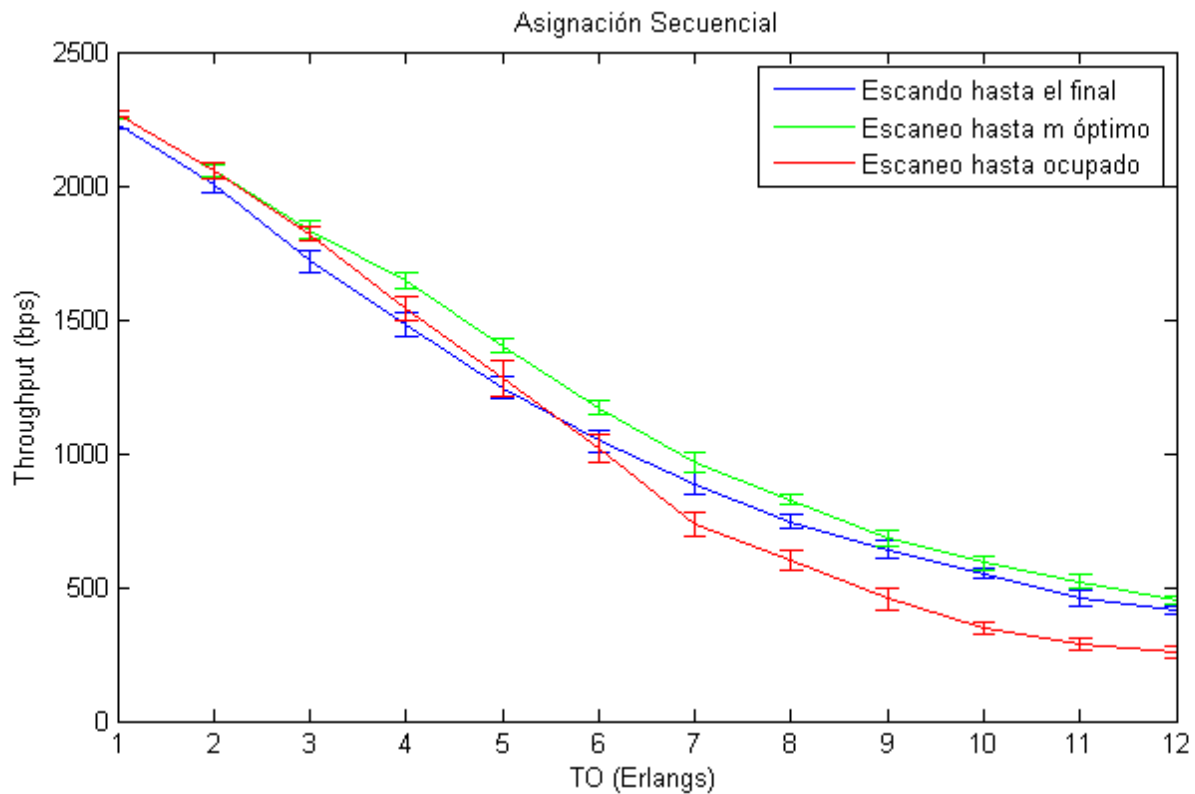


Figura: 5.3. Políticas de escaneo en Asignación Secuencial

El siguiente paso es buscar políticas o reglas de parada más complejas que se adapten mejor a cualquier situación del sistema. Esto lleva a que la decisión no se tome a priori, sino que se tome sobre la marcha en función de los resultados parciales obtenidos.

## 5.3 Optimal Stopping

### 5.3.1 Introducción

Al observar un fenómeno a medida que transcurre el tiempo, naturalmente, surge la pregunta de en qué momento este fenómeno alcanzará sus valores extremos. Además, surge el problema de detener dicho fenómeno al alcanzar estos niveles pero, en general, para esto no se cuenta con más información que los datos obtenidos por la observación de su evolución, por lo cual, la decisión de cuándo detenerlo tiene que tomarse sólo a partir de los valores pasados.

Es más, no se puede esperar que siempre se pueda detener un fenómeno en sus niveles extremos, pero es razonable poder hacerlo en promedio.

La teoría de *Optimal Stopping* es concebida para elegir el momento de realizar una acción basándose en la observación secuencial de variables aleatorias, a fin de obtener la máxima recompensa y minimizar los costes.

Históricamente el problema nace en el análisis de las observaciones secuenciales con “*theory of the sequential probability ratio test*” de Walt (1945) y en libros posteriores, “*Sequential Analysis*” (1947) and “*Statistical Decision Functions*” (1950). La perspectiva Bayesiana se trata en el artículo *Blackwell and Girshick* (1948). La generalización de análisis secuencial a problemas de parada pura sin estructura estadística vino de la mano de Snell (1952). Artículos en la década de los 60 de Chow (1961) and Robbins (1963) impulsaron el interés y el rápido crecimiento de la materia. El libro “*Great Expectations: The Theory of Optimal Stopping*” publicado por Chow, Robbins y Siegmund (1971), se suma a este desarrollo.

### 5.3.2 Formulación

El problema de parada óptima se define en dos puntos:

1. Una secuencia de variables aleatorias,  $X_1, X_2, \dots$ , cuya distribución conjunta se asume conocida.
2. Una secuencia de funciones reales de recompensa,

$$y_0, y_1(x_1), y_2(x_1, x_2), \dots, y_\infty(x_1, x_2, \dots)$$

Una vez conocidos estos dos puntos, el problema de parada óptima asociado se describe de la siguiente manera. La secuencia de  $X_1, X_2, \dots$ , puede observarse con tanta longitud como sea posible.

Para cada  $n = 1, 2, \dots$ , después de la observación  $X_1 = x_1, X_2 = x_2, \dots, X_n = x_n$  la decisión es parar y recibir la conocida recompensa  $y_n(x_1, x_2, \dots, x_n)$  o continuar con la observación  $X_{n+1}$ . Si la decisión es de no hacer ninguna observación, la recompensa es una constante con el valor de  $y_0$ ; si nunca se para de realizar observaciones, la recompensa recibida es  $y_\infty(x_1, x_2, \dots)$ .

El objetivo es escoger un tiempo para parar en el cual se espere que la recompensa sea máxima, lo cual da lugar a que la toma de decisiones sea aleatoria. Esto es, dado que el proceso alcanza el estado  $n$ , se ha observado  $X_1 = x_1, X_2 = x_2, \dots, X_n = x_n$ , entonces la probabilidad de parar depende de estas observaciones. Se denota esta probabilidad con  $\phi_n(x_1, x_2, \dots, x_n)$ . La regla de parada óptima se basa en una secuencia de esas funciones,

$$\phi = (\phi_0, \phi_1(x_1), \phi_2(x_1, x_2), \dots)$$

donde para cada  $n$  y  $x_1, x_2, \dots, x_n$ ,  $0 \leq \phi_n(x_1, x_2, \dots, x_n) \leq 1$ . La regla de parada se dice que no es aleatoria si cada  $\phi_n(x_1, x_2, \dots, x_n)$  es 0 ó 1. Así,  $\phi_0$  representa la probabilidad de que no se tome ninguna observación. Tomada la primera observación se obtiene  $X_1 = x_1$ , por tanto,  $\phi_1(x_1)$  representa la probabilidad de parar después de la primera observación etc. La regla de parada  $\phi$  y la secuencia de observaciones  $X = (X_1, X_2, \dots)$  determinan el valor aleatorio de  $N$  donde ocurrirá la parada,  $0 \leq N < \infty$  donde  $N = \infty$  si la parada nunca ocurre.

Un problema de parada óptima tiene un horizonte finito si el número de estadios en los que debe parar tiene un límite superior conocido. Si es obligatorio parar después de observar  $X_1, X_2, \dots, X_T$ , se dice que el problema tiene horizonte  $T$ . Los problemas de parada óptima con horizonte finito son un caso particular que se puede resolver por el método *Backward Induction* [4].

Como pase lo que pase se debe parar en el estadio  $T$ , primero se debe encontrar la regla de parada óptima en el estadio  $T-1$ . Así, conocida esta regla de parada ahora se debe encontrar la regla de parada óptima en el estadio  $T-2$ , y así sucesivamente hasta llegar al estado inicial. En particular se define:

$$V_T^{(T)} = y_T(x_1, x_2, \dots, x_T)$$

y por inducción para

$$V_j^{(T)} = \max\{y_j(x_1, \dots, x_j), E(V_{j+1}^{(T)}(x_1, \dots, x_j, X_{j+1}) | X_1 = x_1, \dots, X_j = x_j)\}$$

### 5.3.3 Optimal Stopping en el escaneo espectral

La decisión de escaneo del espectro puede ser formulada como un problema de parada óptima (*Optimal Stopping*). Se denota  $X_n$  como el estado del canal  $n$ -ésimo pudiendo tomar los valores 0 ó 1 (vacío u ocupado) y la probabilidad  $P(X_n = 1) = p$ . El valor esperado de  $X_n$  es  $u = E(X_n)$ . Se denota  $y_n$  como la función de recompensa de parar el escaneo y transmitir tras haber escaneado  $n$  canales.  $y_n$  es una función de la disponibilidad de los canales y depende de la tecnología de radio. Para considerar las limitaciones de la radio cognitiva se define  $b_n$ , que es el número máximo de canales libres por los que se puede transmitir.  $n_{max}$  es el número máximo de canales por los que se puede transmitir simultáneamente, según la limitación del hardware usado.

$$b_n(x_1, \dots, x_n) = \max\left\{\sum_{k=1}^n x_k, n_{max}\right\}$$

La función de recompensa se puede escribir como

$$y_n(x_1, \dots, x_n) = \frac{T}{T + t_s + nt} b_n(x_1, \dots, x_n) = \frac{c}{c + t_s + n} b_n(x_1, \dots, x_n)$$

donde  $c = \frac{T}{t}$ .  $y_n$  es en realidad la tasa efectiva de datos durante el intervalo de tiempo  $T$  después de parar de escanear y tomar la decisión de transmisión.

Se asume que el número máximo de canales que un usuario puede escanear antes de tomar la decisión de parada es, como máximo,  $K$  ( $K \leq N$ ). Lo que significa que se trata

de un problema de horizonte finito que se puede resolver mediante *Backward Induction*, como se muestra a continuación:

$$V_K^{(K)}(x_1, \dots, x_K) = y_K(x_1, \dots, x_K) = \frac{c}{c + t_s + K} b_K(x_1, \dots, x_K)$$

entonces,

$$E(V_K^{(K)}(x_1, \dots, x_{K-1}, X_K) | X_1 = x_1, \dots, X_{k-1} = x_{k-1}) = \frac{c}{c + t_s + K} [p \cdot b_K(x_1, \dots, x_{K-1}, 1) + q \cdot b_K(x_1, \dots, x_{K-1}, 0)]$$

donde  $p$  y  $q$  son las probabilidades de  $X_k = 1$  y  $X_k = 0$ , respectivamente; e inductivamente para  $n = K - 1$  hacia atrás hasta  $n = 2$ .

$$V_n^{(K)}(x_1, \dots, x_n) = \max\{y_n(x_1, \dots, x_n), E(V_{n+1}^{(K)}(x_1, \dots, X_{n+1}) | X_1 = x_1, \dots, X_n = x_n)\}$$

$$E(V_n^{(K)}(x_1, \dots, x_{n-1}, X_n) | X_1 = x_1, \dots, X_{n-1} = x_{n-1}) = p \cdot V_n^{(K)}(x_1, \dots, x_{n-1}, 1) + q \cdot V_n^{(K)}(x_1, \dots, x_{n-1}, 0)$$

Obviamente se debe escanear al principio obteniendo como resultado  $x_1$ , ya que  $y_0 = 0$  y  $E(V_1) \geq 0$ . Entonces, se compara  $y_1$  con  $E(V_2)$ , se toma la decisión y se continua así sucesivamente. En cada estadio,  $\{E(V_n)\}$  define la regla de parada óptima.

### 5.3.4 Interpretación mediante grafos

Para mostrar de forma más clara la resolución del problema de parada óptima mediante el método *Backward Induction* se va a ilustrar mediante grafos, concretamente árboles, la evolución del algoritmo.

Se propone un sistema con cinco canales ( $N = 5$ ). Como se ha descrito anteriormente antes de tomar ninguna decisión se escanea el primer canal y se obtiene  $X_1 = x_1$ .



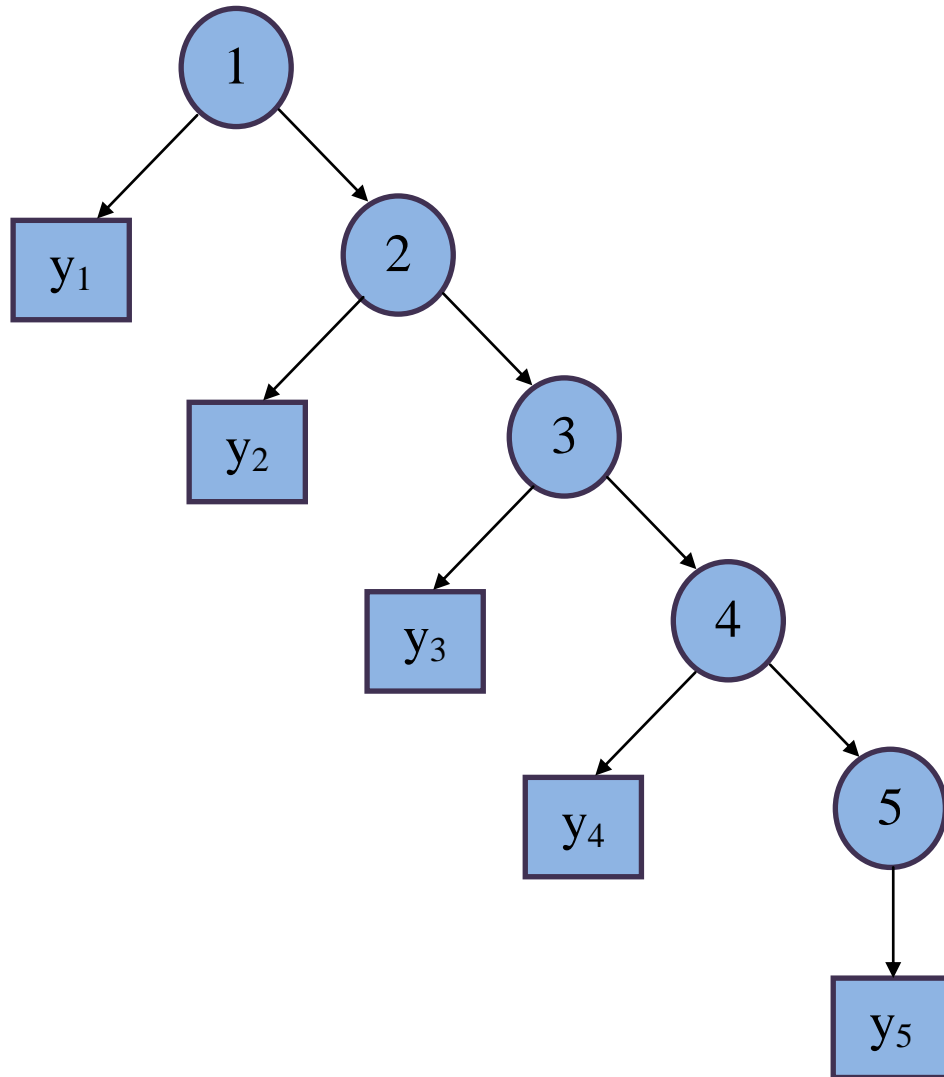


Figura: 5.4. Árbol del algoritmo recursivo

Al escanear el primer canal el algoritmo se encuentra en el *nodo1*. Ahora debe elegir una de sus dos ramas, por un lado tiene la función de recompensa  $y_1$  y por otro lado tiene el *nodo2*, que es la recompensa si siguiera escaneando. Como esta recompensa no se puede saber porque aún no se han escaneado los canales, se estimará.

En este punto sólo quede elegir la mejor opción, es decir, la que ofrezca una mayor recompensa, pero para ello es necesario saber cuál es la recompensa estimada si se escanea el *nodo2*.

Para obtener la recompensa estimada si se escaneara el *nodo2* es necesario saber el máximo entre la función recompensa  $y_2$  y la recompensa obtenida si se siguiera escaneando el *nodo3*, y así recursivamente hasta recorrer todo el árbol.

Es importante destacar que, para poder estimar las funciones recompensa, es necesario conocer la probabilidad de que cada uno de los canales esté ocupado. Dicha probabilidad varía dependiendo del tipo de asignación y el tráfico primario.

Este recorrido del árbol se puede expresar también mediante la particularización de las expresiones mostradas en la *sección 5.3.3*.

$$V_1^{(5)} = \max\{y_1(x_1), E(V_2^{(5)}(x_1, X_2)|X_1 = x_1)\}$$

$$V_2^{(5)} = \max\{y_2(x_1, x_2), E(V_3^{(5)}(x_1, x_2, X_3)|X_1 = x_1, X_2 = x_2)\}$$

$$V_3^{(5)} = \max\{y_3(x_1, x_2, x_3), E(V_4^{(5)}(x_1, x_2, x_3, X_4)|X_1 = x_1, X_2 = x_2, X_3 = x_3)\}$$

$$V_4^{(5)} = \max\{y_4(x_1, \dots, x_4), E(V_5^{(5)}(x_1, \dots, x_4, X_5)|X_1 = x_1, \dots, X_4 = x_4)\}$$

$$V_5^{(5)} = y_5(x_1, \dots, x_5)$$

Siguiendo con el ejemplo, si la opción de escanear el segundo canal fuera ventajosa, al escanearse se obtiene  $X_2 = x_2$ , y se plantea un árbol similar al anterior, pero con la diferencia de que el primer nodo ahora es *nodo2*, el resto de nodos quedan igual. Y así sigue el proceso mientras se sigan escaneando canales, en cada iteración se quita el primer nodo del árbol. En el caso de que se escanearan todos los canales, en la última iteración sólo quedaría el *nodo5*, el cual sólo tiene la opción de obtener lo que le ofrece su función de recompensa, por lo que ahí termina el algoritmo.

### 5.3.5 Implementación

Para la programación de la solución al problema de parada óptima mediante el método de *Backward Induction* se han diseñado dos funciones cuyos detalles se exponen a continuación. Éste es un método de programación dinámica, por lo que es necesario resolverlo mediante un algoritmo recursivo.

La primera función llamada *bi\_decision* es la encargada de recibir todos los datos necesarios, llamar a la función que recorrerá el árbol de forma recursiva y devolver 1 ó 0 si conviene seguir escaneando o no, respectivamente.

Definición:

```
escaneo = bi_decision(estadio, vector_p, vector_lib, datos)
```

*escaneo* = 0 => se para de escanear; 1 => se continua escaneando.

*estadio* = n° de canales que se han escaneado hasta el momento de llamar a esta función.

*vector\_p* = Vector con las probabilidades de ocupación primaria de los canales.

*vector\_lib* = Vector con los resultados de los escaneos realizados hasta el momento, 1=> canal ocupado; 0 => canal libre.

*datos* = Vector con los valores necesarios para el cálculo de throughput, [*T*, *t<sub>s</sub>*, *t*].

Código:

```
function escaneo = bi_decision(estadio, vector_p, vector_lib, datos)

    vector_lib=concatenar([vector_lib(1:estadio)==0 (1-vector_p(estadio+1:end))]);

    tp=sum(vector_lib(1:estadio))*datos(1)/(datos(2)+datos(1)+datos(3)*estadio);

    if (estadio==length(vector_p))
        escaneo=0;
    elseif (tp<bi_recorrido(estadio, estadio+1, vector_lib, datos))
        escaneo=1;
    else
        escaneo=0;
    end
end
```

En la primera línea se añade, para los canales que aún no han sido escaneados, la probabilidad de que estén libres para realizar futuras estimaciones. En la siguiente línea se calcula el throughput que se obtiene si se para de escanear en ese momento. Más

adelante se compara el throughput con el que se obtendría en el estado siguiente. Para obtener esa estimación se llama a la siguiente función, *bi\_recorrido*.

La función *bi\_recorrido* es la que mediante llamadas recursivas recorre todo el árbol para obtener el valor máximo del throughput en la estimación.

Definición:

```
tpmax = bi_recorrido(estadio, nodo, vector_lib, datos)
```

*tpmax* = throughput máximo estimado en el nodo *nodo*.

*nodo* = n° de nodo en el que se encuentra el algoritmo en una determinada llamada.

Los demás parámetros tienen la misma definición que en la función anterior.

Código:

```
function tpmax = bi_recorrido(estadio, nodo, vector_lib, datos)

    tp=sum(vector_lib(1:nodo))*datos(1)/(datos(2)+datos(1)+datos(3)*nodo);

    if(nodo==length(vector_lib))           //caso base
        tpmax=tp;
    else                                     //llamada recursiva
        tpmax=max(tp, bi_recorrido(estadio, nodo+1, vector_lib, datos));
    end

end
```

Es importante destacar que la variable *tp* en esta función en realidad representa un throughput estimado, ya que no sólo usa la información de los canales escaneados sino también la probabilidad de que los canales estén libres.

Seguidamente se presenta la estructura del algoritmo recursivo, un caso base y la llamada recursiva.

### 5.3.6 Complejidad Computacional

Es necesario indagar en la complejidad del algoritmo anteriormente descrito ya que ante una complejidad elevada no todos los dispositivos hardware estarán en disposición de poder ejecutarlo.

Se va a considerar como una operación de cálculo relevante dentro del algoritmo el cálculo del throughput, es decir, se desprecian todas las demás operaciones y se calcula la complejidad del algoritmo en función del número de veces que hay que calcular el throughput.

Partiendo de un caso particular, en el que, como en el caso anterior, se toma  $N = 5$ , se puede comprobar que en el primer estadio se calcula cinco veces el *throughput*, en el segundo estadio se calcula cuatro veces, y así sucesivamente hasta llegar al quinto estadio que se calcula una vez. Si se generaliza este cálculo para  $N$  canales y se calcula el número de operaciones se obtiene lo siguiente:

$$N_o = \sum_{i=1}^N i = \frac{N(1+N)}{2}$$

Esta expresión es proporcional a  $N^2$ , por lo que el orden de complejidad del algoritmo es  $N^2$ ,  $O(N^2)$ .

Con este orden de complejidad, cuando el número de canales es pequeño, el número de operaciones también lo es, pero al aumentar el número de canales, el número de operaciones crece de forma exponencial, por lo que esto no es abarcable para dispositivos con limitaciones hardware, que eran el objetivo de este estudio.

Para solventar este inconveniente es necesario buscar otra alternativa a la presente solución del problema de parada óptima que presente un menor orden de complejidad computacional.

## 5.4 *k*-stage look-ahead

Este método se propone como solución alternativa al *Optimal Stopping* con un coste computacional más bajo, asequible para dispositivos con limitaciones hardware.

Se basa en un truncamiento de la solución completa (*Backward Induction*) desarrollada anteriormente, lo que hace que se obtengan resultados parecidos que se acercan más o menos a la solución completa en función de *k*.

La regla de parada *k*-stage look-ahead se describe por el estadio de parada,

$$N_k = \min\{n \geq 0 : y_n(x_1, \dots, x_n) \geq E(V_{n+1}^{(n+k)}(x_1, \dots, X_{n+1}, \dots, X_{n+k}) | X_1 = x_1, \dots, X_n = x_n)\}.$$

*k*-stage look-ahead es la regla de parada que se detiene o continúa de acuerdo con la regla de parada óptima, truncando *k* estados hacia adelante. Dicho de otro modo, en el estadio *n* si la regla óptima truncando en el estado *n + k* es continuar, *k*-stage look-ahead continúa; si no es así para.

De igual modo que la solución completa, esta versión truncada se puede entender mediante un árbol, pero que esta vez sólo tendrá *k + 1* nodos (cada nodo *n* tendrá su respectiva función de recompensa  $y_n$ ). Si el algoritmo se encuentra en el estadio *n*, el árbol tendrá el nodo *n*, el nodo *n + 1*, y así hasta el nodo *n + k* (en el caso de que esos nodos existan). Así, se aplicará el mismo método que en la solución completa pero con este árbol truncado.

Una propiedad importante de estas reglas de parada es que, si existe la regla de parada óptima y *k*-stage look-ahead indica continuar, lo óptimo siempre es continuar.

La más simple de estas reglas es *1*-stage look-ahead,

$$N_1 = \min\{n \geq 0 : y_n(x_1, \dots, x_n) \geq E(V_{n+1}^{(n+1)}(x_1, \dots, x_n, X_{n+1}) | X_1 = x_1, \dots, X_n = x_n)\}$$

dicho en palabras,  $N_1$  es el estadio en el que se para y es el valor mínimo de *n* en el que el valor devuelto por la función de recompensa es mayor que la recompensa estimada cuando se pasa el siguiente estadio y entonces se para.

*1*-stage look-ahead es la regla de parada que se evaluará en el presente trabajo debido a su simplicidad y la proximidad de sus resultados a la parada óptima. Además se puede

demostrar que, si el problema es monótono, esta regla de parada encuentra la solución óptima en problemas de horizonte finito.

En la práctica, aplicar esta regla de parada equivale a decir que *el usuario secundario parará de escanear cuando el resultado de la función de recompensa parando en ese estado sea mayor que el resultado de la función de recompensa que se estima de escanear otro canal y entonces parar.*

## 5.4 Resultados

A continuación se muestran y comparan los resultados de todas las políticas de escaneo propuestas en el capítulo.

En primer lugar, se comparan los resultados obtenidos con *Optimal Stopping* y con *1-stage look-ahead*. Como se aprecia en las figuras, los resultados obtenidos para cualquiera de las asignaciones apenas varían, es decir, la diferencia de rendimiento entre estos dos métodos, en los casos concretos que se proponen en el presente trabajo, es casi nula. Obviamente, antes estos resultados, se opta por elegir el método *1-stage look-ahead* que ofrece una menor complejidad computacional.

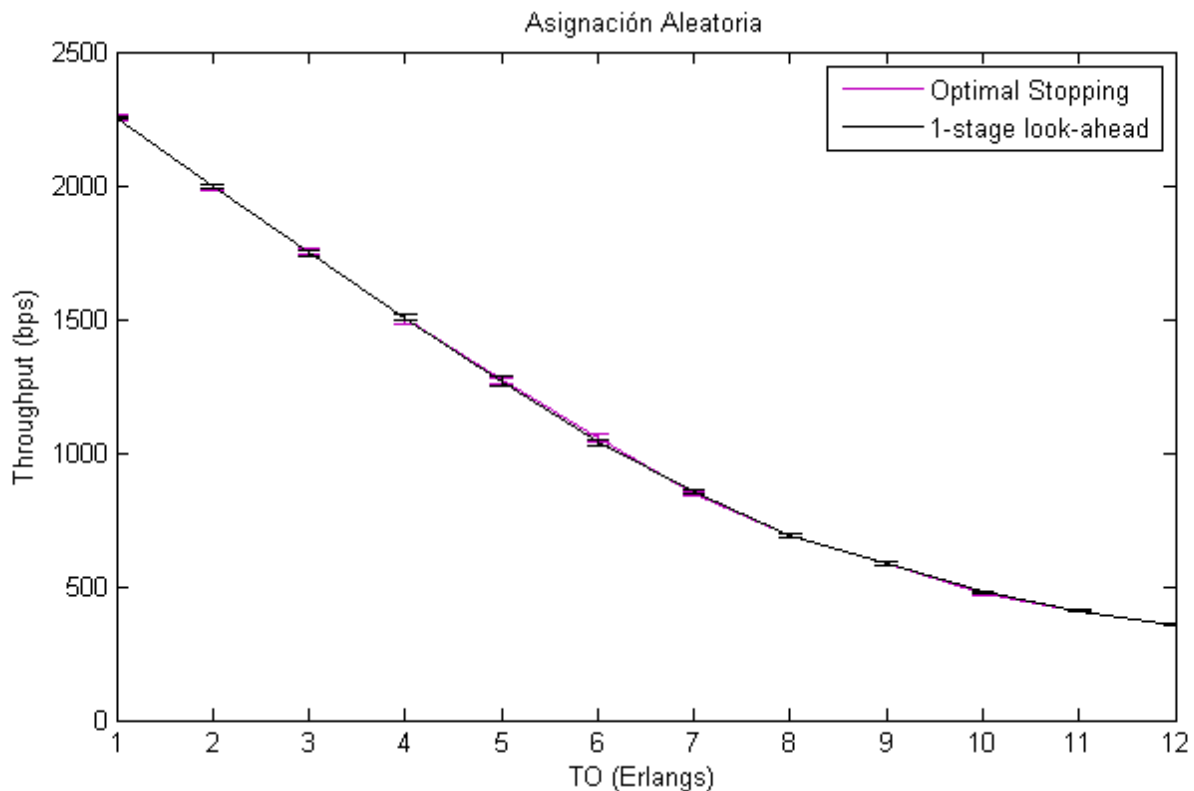


Figura: 5.5. *Optimal Stopping* y *1-stage look-ahead* en Asignación Aleatoria

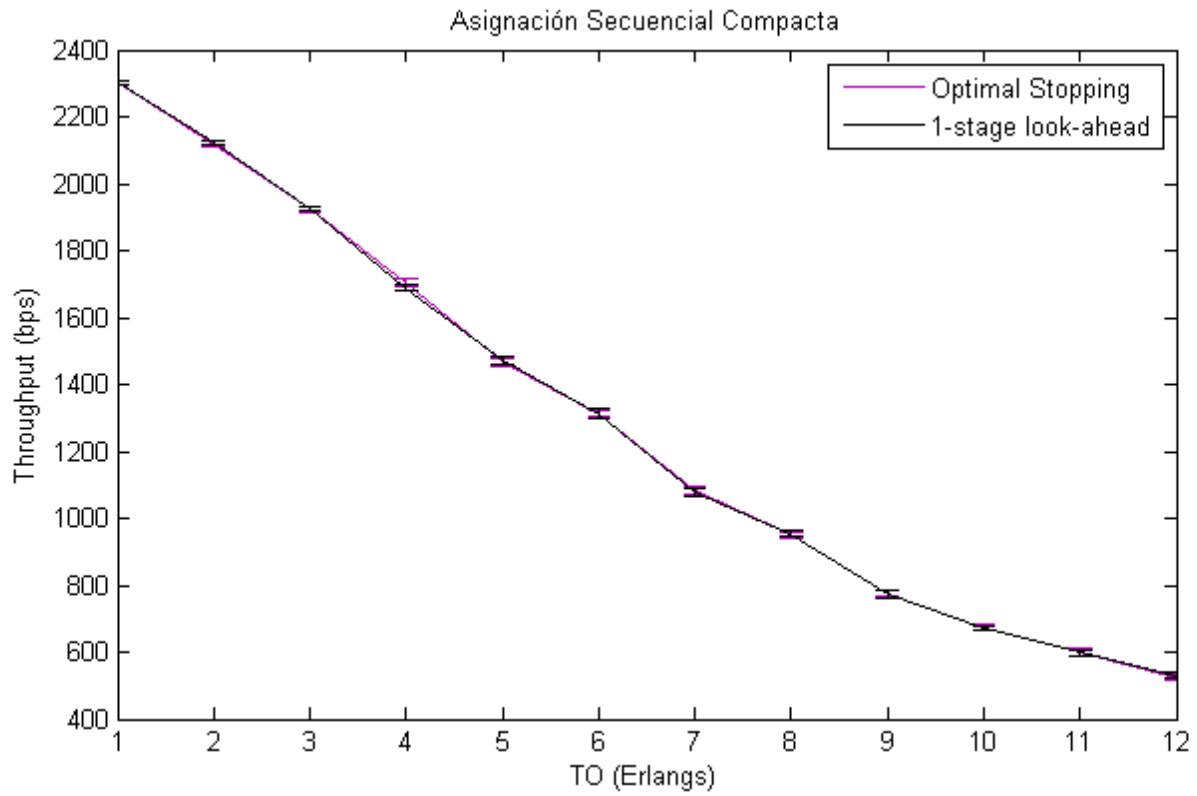


Figura: 5.6. *Optimal Stopping* y *1-stage look-ahead* en *Asignación Secuencial Compacta*

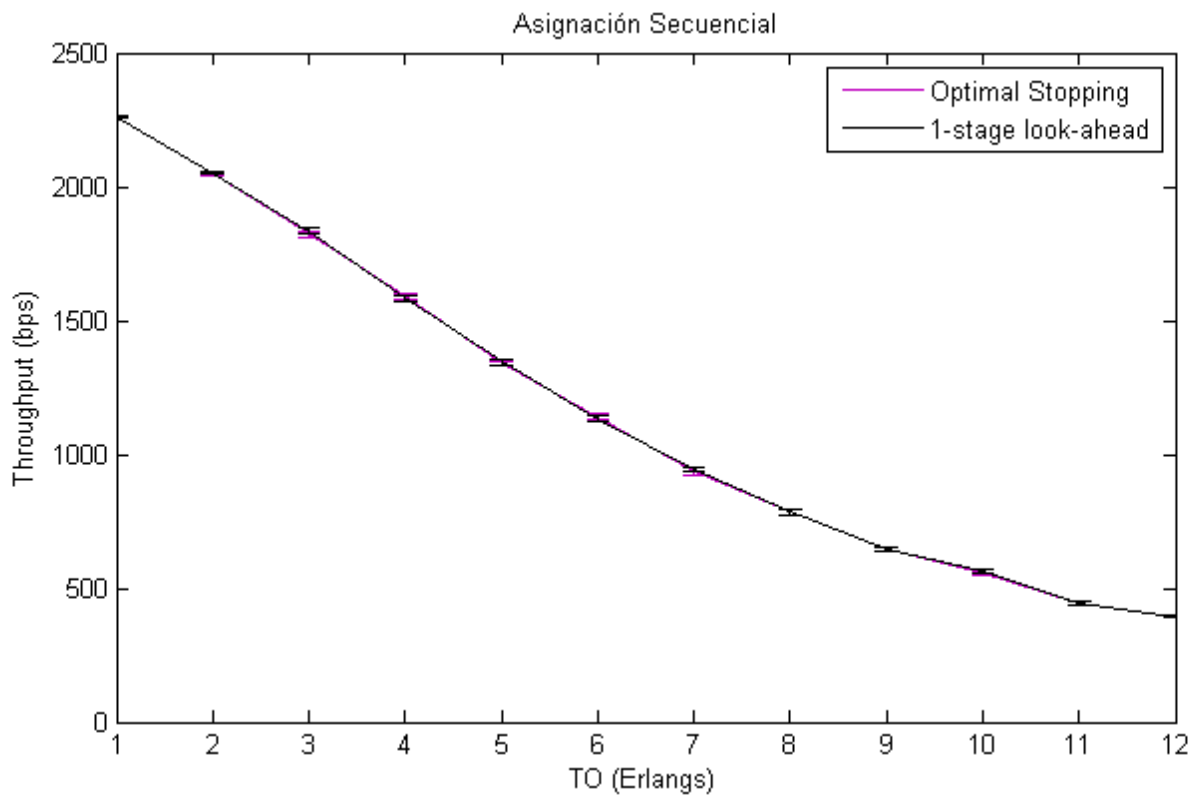


Figura: 5.7. *Optimal Stopping* y *1-stage look-ahead* en *Asignación Secuencial*



Por último, sólo queda comparar el rendimiento ofrecido por el método *1-stage look-ahead* con los demás propuestos en el capítulo y comentar cada caso concreto.

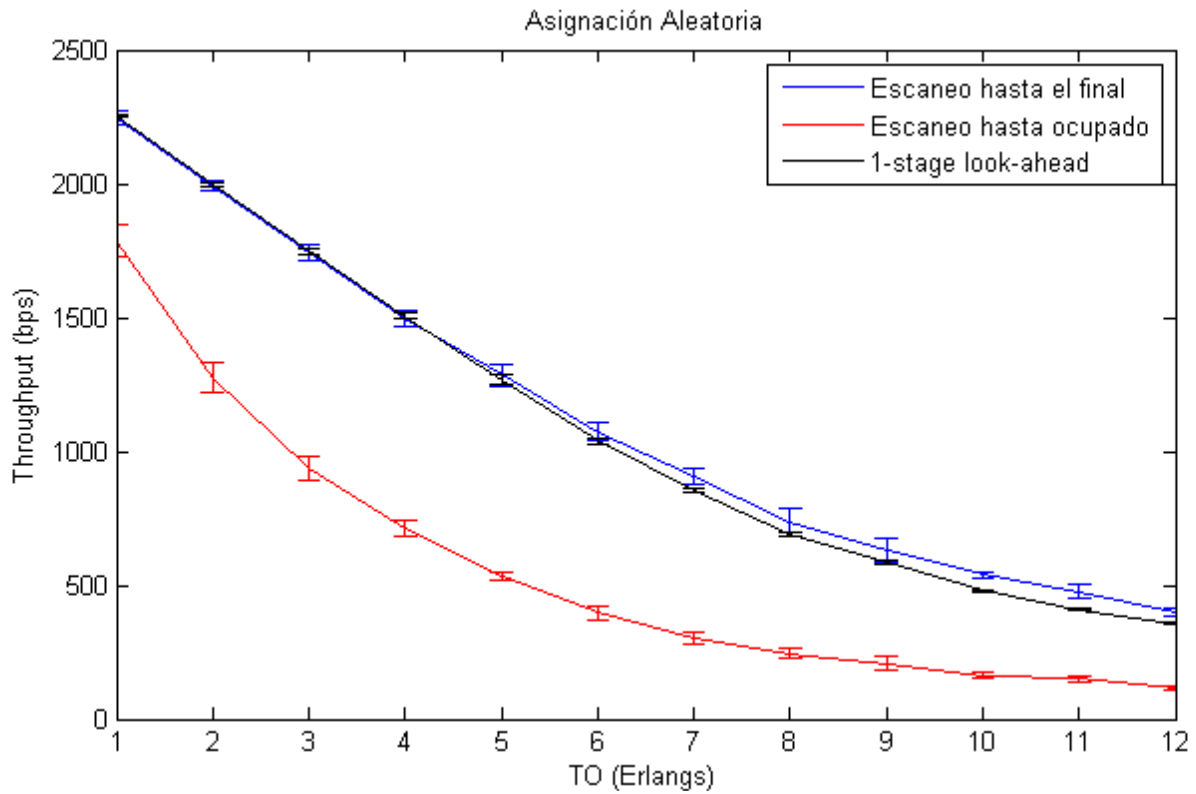


Figura: 5.8. Todas las políticas en Asignación Aleatoria

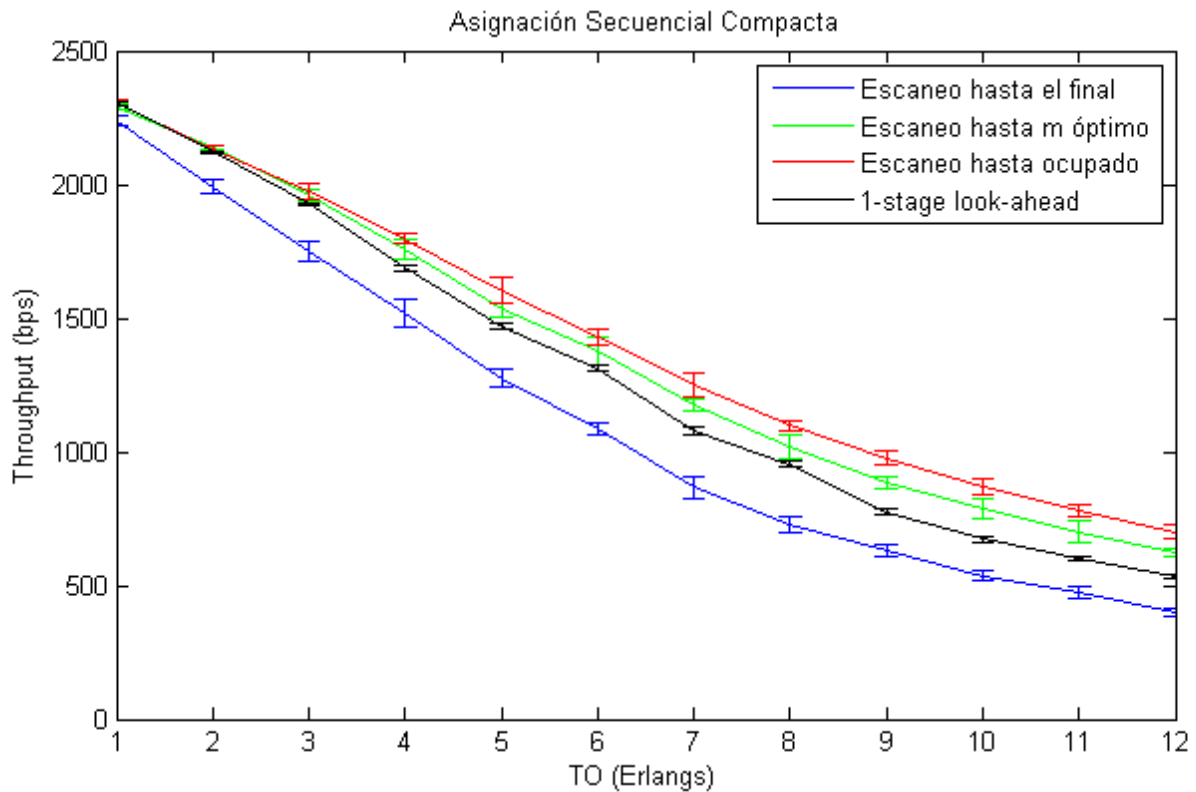


Figura: 5.9. Todas las políticas en Asignación Secuencial Compacta

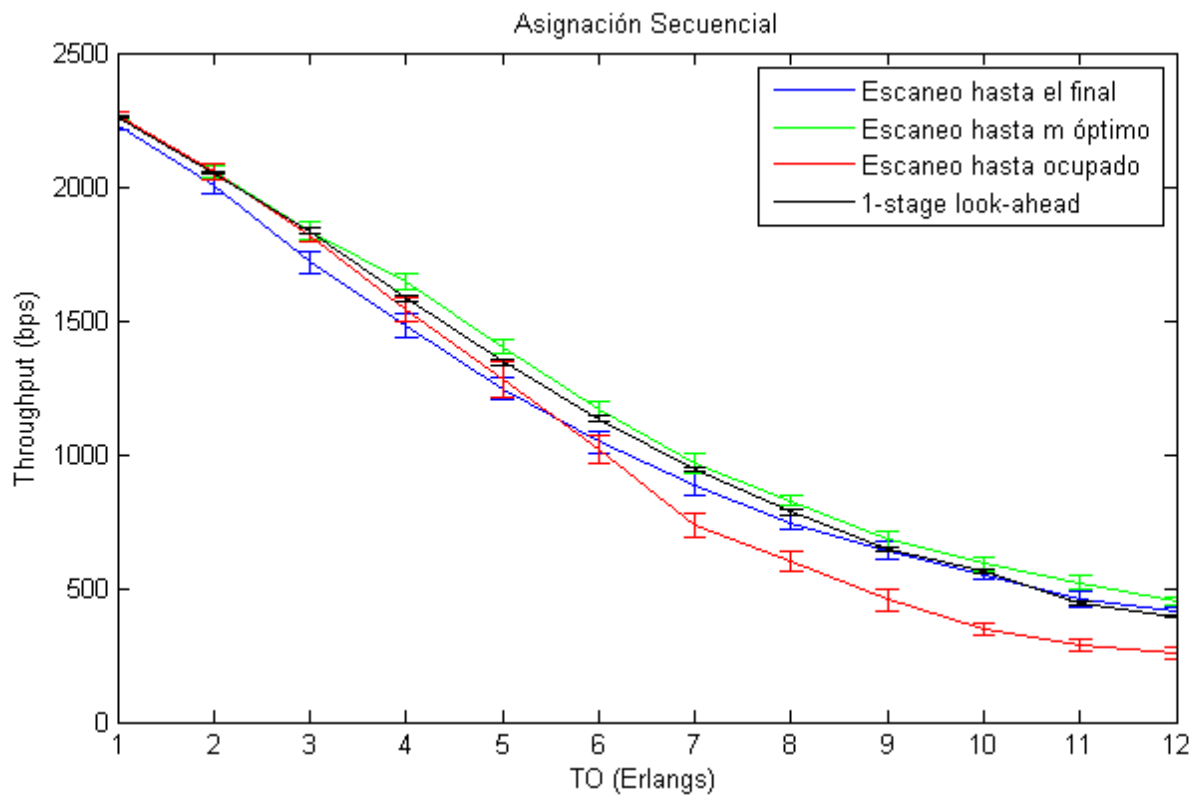


Figura: 5.10. Todas las políticas en Asignación Secuencial

De las anteriores figuras se pueden extraer las siguientes conclusiones:

- Para Asignación Aleatoria, la mejor política es la de *Escaneo hasta el final*, y con *1-stage look-ahead* se obtienen resultados similares, aunque empeoran ligeramente al aumentar el tráfico.
- Para Asignación Secuencial Compacta, la mejor política es *Escaneo hasta ocupado*; *1-stage look-ahead* no ofrece resultados tan buenos.
- Para Asignación Secuencial, la mejor política es *Escaneo hasta m óptimo*, aunque *1-stage look-ahead* se acerca mucho a esta política, por lo que también es una buena opción.

A la hora de elegir una política u otra es conveniente tener en cuenta los costes asociados a cada una de ellas, y no sólo los resultados obtenidos.

Como resumen, las más simples son *Escaneo hasta el final* y *Escaneo hasta ocupado*, ya que no necesitan ninguna información adicional del estado del sistema.

*Escaneo hasta  $m$  óptimo* necesita estimar el valor del *TO* y de ahí calcular el valor de *m* óptimo, lo cual supone una carga adicional.

Por último, *1-stage look-ahead* necesita calcular la probabilidad de ocupación de los canales, y luego estimar el valor del throughput en un hipotético futuro escaneo.

# Capítulo 6

## Conclusiones

---

Como se plantea en la introducción, se presenta el problema de un espectro regulado por licencias, cuyos usuarios (usuarios primarios) tienen acceso total y sin restricciones al medio. Para usuarios sin licencia (usuarios secundarios), la única forma que tienen de establecer comunicaciones inalámbricas es mediante redes inteligentes de acceso dinámico al espectro radioeléctrico. Estas redes tienen como objetivo acceder de forma oportunista a las sub-bandas libres (oportunidades espectrales) que no estén siendo usadas por los usuarios con licencia. Otro objetivo también muy importante de estas redes es no interferir o interferir de forma mínima en las comunicaciones de usuarios primarios.

En el escenario planteado, los usuarios primarios usan un esquema de asignación centralizado, por lo que los canales son asignados por una entidad central. Por el contrario, los usuarios secundarios usan un protocolo distribuido formando redes ad-hoc.

Para dicho propósito se usa el protocolo HC-MAC [1]. Una de sus principales características es que tiene en consideración las limitaciones hardware de los dispositivos de radio, por lo que su funcionamiento básico se divide en tres fases secuenciales: contienda, escaneo y transmisión.

Por ser un protocolo distribuido es necesaria una fase de contienda para la negociación entre nodos. Después de esta fase se realiza un escaneo secuencial de canales para posteriormente transmitir de forma simultánea por todos los canales libres encontrados, teniendo siempre en cuenta las limitaciones hardware del dispositivo.

De aquí surge el problema de cuántos canales es mejor escanear y en función de qué parámetros. La solución a este problema se busca mediante una política de escaneo.

En HC-MAC se resuelve el problema planteándolo como un problema de Optimal Stopping. En el presente trabajo se han estudiado otras políticas de escaneo además de

la propuesta en HC-MAC y se han evaluado en distintos tipos de asignación primaria no contemplados en [1].

Estudiando los tres tipos de asignación planteados (Aleatoria, Secuencial, Secuencial Compacta), se observa que el rendimiento del sistema secundario es muy diferente para cada una de ellos, por lo que resulta necesario tener en cuenta este factor a la hora de elegir una política de escaneo.

La resolución del problema de escaneo de canales mediante Optimal Stopping ofrece buenos resultados en casi cualquier tipo de asignación, por lo que, en general, es una buena opción. Por el contrario, si se conoce el tipo de asignación primaria, la elección de la política de escaneo se puede hacer de forma mucho más eficaz. Por ejemplo, si se conoce a priori que la asignación primaria es aleatoria, la política de escaneo óptima en este caso es Escaneo hasta el final ya que esta política es más sencilla de implementar y supone un menor coste computacional que Optimal Stopping.

En resumen, se demuestra que cuanto más compacta sea la asignación de los usuarios primarios, mejor será el rendimiento obtenido por una red cognitiva ad-hoc con la que comparta espectro. Por otro lado, el conocimiento del tipo de asignación primaria por parte de los usuarios secundarios favorece la toma de decisiones de éstos, por lo que también mejora el rendimiento del sistema secundario.

# Bibliografía

---

[1] **J. Jia, Q. Zhang, and X. Shen**, "*HC-MAC: A Hardware-constrained Cognitive MAC for Efficient Spectrum Management*", in IEEE JSAC special issue on "cognitive radio - Theory and Applications", Vol. 26, Issue 1, Jan. 2008.

[2] **T. Ferguson**, "*Chapter 1. Stopping Rule Problems*" available at: <http://www.math.ucla.edu/tom/Stopping/Contents.html>.

[3] **C. Cordeiro, K. Challapali, D. Birru and S. Shankar**, "*IEEE 802.22: The First Worldwide Wireless Standard based on Cognitive Radios*" Proc. First IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks, November 2005.

[4] **T. Ferguson**, "*Chapter 2. Finite Horizon Problems.*" available at: <http://www.math.ucla.edu/tom/Stopping/Contents.html>.

[5] **T. Ferguson**, "*Chapter 5. Monotone Stopping Rule Problems.*" available at: <http://www.math.ucla.edu/tom/Stopping/Contents.html>.