



# INSTITUTO POLITÉCNICO NACIONAL

UNIDAD PROFESIONAL INTERDISCIPLINARIA EN  
INGENIERÍA Y TECNOLOGÍAS AVANZADAS

## *Proyecto Terminal*

**“Modelado matemático, basado en cadenas de Markov, para  
servicios de video en vivo soportados por redes híbridas P2P-CDN”**

Que para obtener el título de

**“Ingeniero en Telemática”**

Presenta:

**Ulises Muñoz Ruiz  
José Manuel Ortiz Islas**

Asesores:

**Dr. Mario Eduardo Rivero Ángeles  
Dr. Noé Torres Cruz  
Dra. Iclia Villordo Jimenez**



Ciudad de México. Junio, 2023



# INSTITUTO POLITÉCNICO NACIONAL

UNIDAD PROFESIONAL INTERDISCIPLINARIA EN  
INGENIERÍA Y TECNOLOGÍAS AVANZADAS

## *Proyecto Terminal*

**“Modelado matemático, basado en cadenas de Markov, para servicios de video en vivo soportados por redes híbridas P2P-CDN”**

Que para obtener el título de:

**“Ingeniero en Telemática”**

P r e s e n t a:

Ulises Muñoz Ruiz  
José Manuel Ortiz Islas

Asesores:

---

Dr. Mario Eduardo  
Rivero Ángeles

---

Dr. Noé  
Torres Cruz

---

Dra. Iclia Villordo  
Jimenez

Presidente del Jurado:

Secretario del Jurado:



---

Dr. Víctor Barrera  
Figueroa

---

Dra. Cyntia Eugenia  
Enriquez Ortiz

# Resumen

*Live streaming* es la tecnología que permite difundir una señal de video a través de Internet en tiempo real. Al utilizar esta tecnología, el usuario puede visualizar videos de acuerdo con sus preferencias, sin embargo, en esta aplicación es difícil determinar la cantidad de recursos que la red debe asignar a cada usuario para mantener una Calidad de Servicio (QoS, por sus siglas en inglés) adecuada. Este problema es aún más complejo cuando se consideran implementaciones en las que los usuarios funcionan como pares (comparten entre ellos contenido previamente descargado) y a la vez tienen acceso al video mediante Redes de Distribución de Contenido (CDN, por sus siglas en inglés). Considerando lo anterior, en este proyecto se desarrolló un modelo matemático, basado en una Cadena de Markov, que describe el funcionamiento de la red y, por lo tanto, permite evaluar su desempeño de acuerdo con el comportamiento de los *downloaders*, el ancho de banda total que estos consumen, el ancho de banda que proveen los *peers* y el ancho de banda de los servidores *CDN*. Por otro lado, se propone una adecuación, para servicios de video en vivo, de algunos esquemas de asignación de recursos implementados en sistemas que proporcionan un servicio de video bajo demanda. Se espera, que la eficiencia de la adecuación de los esquemas supere la eficiencia de esquemas implementados en sistemas actuales, en términos de los anchos de banda consumidos dentro del sistema. Este proyecto no contempla la implementación del esquema propuesto, pero sí su evaluación con el modelo desarrollado y considerando parámetros reales de operación de las redes actuales.

**Palabras Clave:** Cadenas de *Markov*, Red Híbrida *CDN-P2P*, Transmisión en Vivo.

# Índice General

Resumen.....	3
Índice de Figuras.....	5
Índice de Tablas .....	6
Capítulo 1.....	7
Introducción .....	7
1.1    Introducción.....	7
1.2    Planteamiento del Problema.....	11
1.3    Propuesta de Solución .....	13
1.4    Alcances .....	15
1.5    Justificación.....	17
1.6    Metodología.....	18
1.7    Objetivos.....	20
1.7.1    Objetivo General.....	20
1.7.2    Objetivos Específicos .....	20
Capítulo 2.....	21
Estado del Arte .....	21
2.1    Estado del arte del modelo de servicios de video con cadenas de Markov ....	21
2.2    Estado del arte de esquemas de asignación de recursos para servicios de video	24
Capítulo 3.....	26
Marco Teórico.....	26
3.1    Video en Vivo .....	26
3.1.1    Formatos de codificación .....	26
3.1.2    Formatos de empaquetamiento .....	28
3.1.3    Servicios de video en vivo .....	28
3.1.4    Protocolos para la distribución de video en vivo por internet.....	31
3.2    Redes <i>CDN-P2P</i> .....	33
3.3    Cadenas de Markov.....	36
3.4    Servicios de video sobre redes híbridas <i>CDN-P2P</i> .....	37
Capítulo 4.....	41
Análisis.....	41
4.1    Análisis de la cadena de Markov .....	41
4.2    Análisis de modelos para servicios de video bajo demanda .....	48

4.2.1 Características de los VoD.....	48
4.2.2 Condiciones de Penuria y Abundancia para el Esquema de Asignación de Recursos Uniforme .....	51
4.2.3 Cadena de Markov para modelar servicios de VoD .....	52
4.3 Análisis esquemas de asignación de recursos para servicios de video bajo demanda .....	54
Capítulo 5.....	61
Diseño .....	61
5.1 Diseño de la cadena de Markov para servicios de video en vivo .....	61
5.2 Diseño del esquema de asignación de recursos para servicios de video en vivo 75	
Capítulo 6.....	81
Pruebas e Implementación.....	81
6.1 Rediseño de la cadena de Markov para servicios de video en vivo.....	94
Capítulo 7.....	116
Resultados.....	116
Conclusiones .....	124
Referencias.....	126

## Índice de Figuras

Figura 1. Capas de OTT .....	7
Figura 2. Aumento de personas en la plataforma twitch .....	9
Figura 3. Clasificación de plataformas Live Streaming.....	9
Figura 4. Descarga de una ventana de video.....	13
Figura 5. Diagrama a bloques del códec H264 [18].....	27
Figura 6. Diagrama a bloques del formato de codificación mpeg [20] .....	28
Figura 7. Escenario de video Edge-C3 en redes inalámbricas [21] .....	33
Figura 8. Clasificación de peers en un escenario CDN básico [6].....	35
Figura 9. Cadena de Markov Unidimensional para sistema con pérdidas sin cola .....	42
Figura 10. Transición en el estado de la cadena de Markov Unidimensional .....	42
Figura 11. Diagrama de flujo para dar solución por simulación a la cadena de Markov .....	44
Figura 12. Gráfica de solución por implementación de la cadena de Markov para $S=30$ , $\lambda = 4$ y $\mu = 0.5$ .....	47
Figura 13. Gráfica de solución por implementación de la cadena de Markov para $S=30$ , $\lambda = 6$ y $\mu = 0.5$ .....	47

Figura 14. Gráfica de solución por implementación de la cadena de Markov para $S=30$ , $\lambda = 8$ y $\mu = 0.5$ .....	47
Figura 15. Estructura de un archivo de video bajo demanda-VoD .....	49
Figura 16. Clasificación de usuarios dentro de una red P2P .....	50
Figura 17. Distribución de recursos en una red P2P .....	51
Figura 18. Cadena de Markov para un sistema de consumo de servicio de video bajo demanda (VoD) .....	53
Figura 19. Estructura de una hiperventana .....	61
Figura 20. Cadena de Markov de un sistema de transmisión de video en vivo .....	65
Figura 21. Diagrama de flujo de solución matemática para la cadena de Markov de un sistema de video en vivo .....	69
Figura 22. Diagrama para mín=ENTran ó EnTRep .....	73
Figura 23. Diagrama para mín=ENTAb.....	74
Figura 24. Obtención de recursos para la población en la ventana $i$ .....	79
Figura 25. Gráfica bidimensional de downloaders promedio en VoD bajo DU .....	82
Figura 26. Gráfica de numero de downloaders en VoD bajo DU.....	82
Figura 27. Gráfica de numero de seeds en VoD bajo DU .....	83
Figura 28. Gráfica poblaciones promedio de downloaders live stream DU .....	84
Figura 29. Gráfica bidimensional de downloaders promedio en livestraming utilizando el esquema DU.....	85
Figura 30. Gráfica downloaders promedio live streaming DU con TVI.....	86
Figura 31. Gráfica de downloaders promedio variando $\lambda$ .....	86
Figura 32. Gráfica de downloaders promedio variando $P_{\omega}$ tasa de producción.....	87
Figura 33. Gráfica de downloaders promedio variando $\theta_0$ .....	88
Figura 34. Ancho de banda que demanda el sistema .....	89
Figura 35. Tiempo promedio de descarga por ventana .....	90
Figura 36. Tiempo promedio de descarga ventana 0 .....	90
Figura 37. Poblaciones promedio de downloaders con nueva expresión .....	91
Figura 38. Downloaders promedio considerando el avance los usuarios en la ventana C .....	91
Figura 39. Ancho de banda por iteración proveniente de los servidores. ....	93
Figura 40.. Downloaders promedio en el sistema livestreaming .....	93
Figura 42. Replanteamiento de la cadena de Markov de un sistema de transmisión de video en vivo.....	95
Figura 43. Ancho de Banda consumido en la descarga de video .....	99

## Índice de Tablas

Tabla 1. Atributos de los videos [20] .....	29
Tabla 2. Características del video en vivo [21] [22] .....	30
Tabla 3. Características principales de esquemas de asignación de recursos investigados.....	57
Tabla 4. Casos de desconexión de un peer .....	64

# Capítulo 1

## Introducción

### 1.1 Introducción

En la actualidad existen dos formas de transmitir video mediante protocolos de Internet, *OTT* (*over the top*) e *IPTV* (*Internet Protocol Television*) [1]. Ambas tecnologías permiten distribuir contenido de video sobre la red, sin embargo, cada una de ellas lo realiza de manera distinta.

*OTT* tiene 2 enfoques: *VoD* (*Video on Demand*) y *live streaming*. En ambos casos se hace uso de servidores de Internet público para transmitir el contenido a todos los espectadores. Debido a que la calidad en la transmisión del video está sujeta a las condiciones normales de la red de internet del cliente, el archivo de video se divide en pequeños fragmentos con la finalidad de que el usuario los pueda recibir de manera cronológica, formar un *buffer* considerable y así poder reproducir el video sin complicaciones.

En *OTT* se habla de un servicio de 3 capas que garantizan una óptima experiencia de visualización de video sobre una red impredecible; la primera capa es la del video central, la segunda es la capa contenedora encargada de dividir el video en fragmentos y finalmente se presenta la capa de protocolo de transferencia basado en tiempo [1].

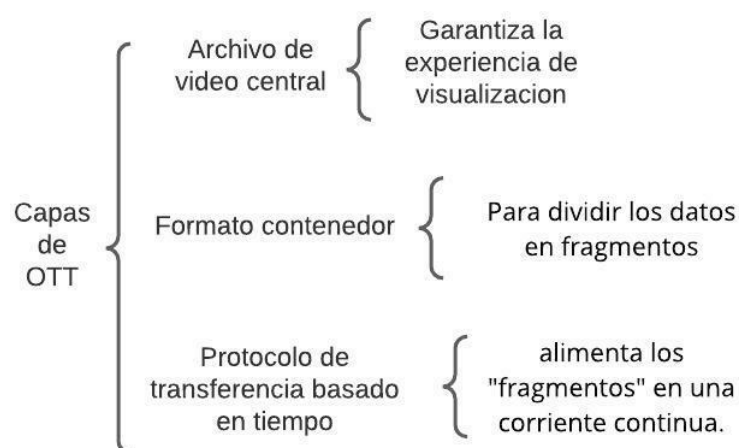


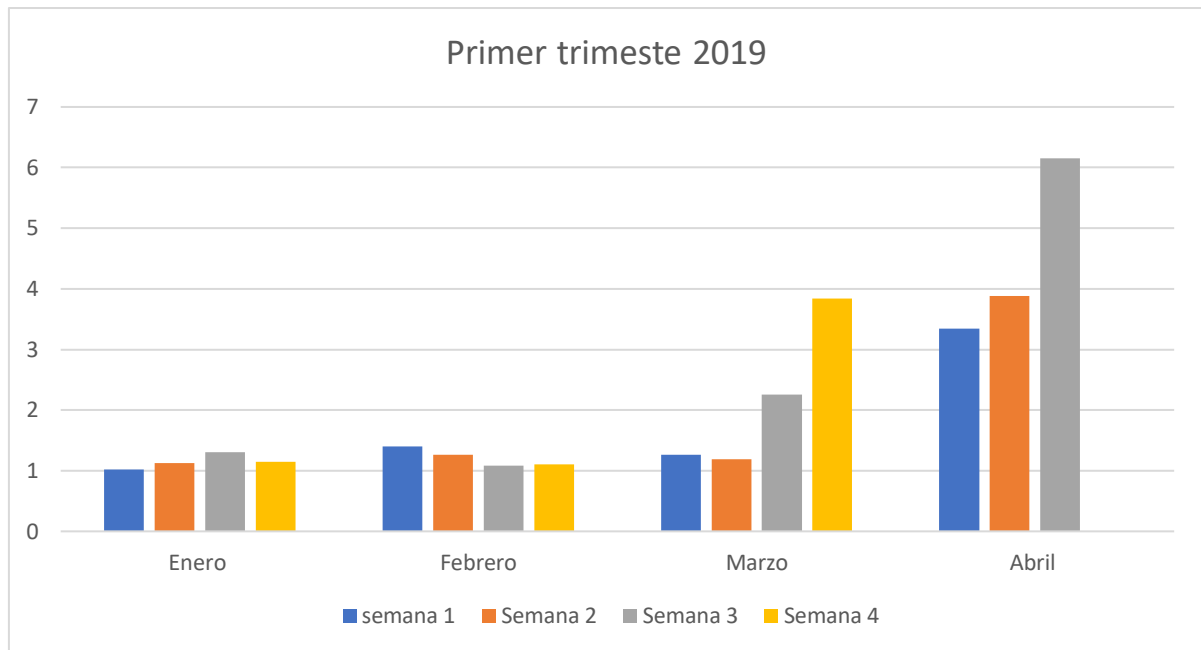
Figura 1. Capas de OTT

Por otro lado, *IPTV* está enfocado en hacer llegar el contenido disponible en Internet al sistema clásico de televisión, en cierto sentido se puede definir como un sistema híbrido de *OTT* y televisión por cable. Este tipo de transmisión, emplea una red *CDN* dedicada para garantizar una buena calidad de servicio, donde el video no viaja directamente a un navegador, sino que va desde un ruteador a un decodificador que permite su visualización en una TV. En este tipo de transmisión el proveedor de servicios es el responsable de proporcionar una *CDN* estable para no recurrir al servicio de proveedores terceros.

Una vez conocidas estas 2 formas de transmisión de video, el presente proyecto se enfoca en el servicio *OTT* dirigido a *live streaming*, con el propósito de desarrollar un modelo matemático que describa este tipo de sistemas y, por lo tanto, ayude a entender los fenómenos que afectan la calidad en el servicio. Además de lo anterior, también se plantea, como parte de este proyecto, definir un esquema de asignación de recursos que mejore el desempeño de propuestas publicadas con anterioridad.

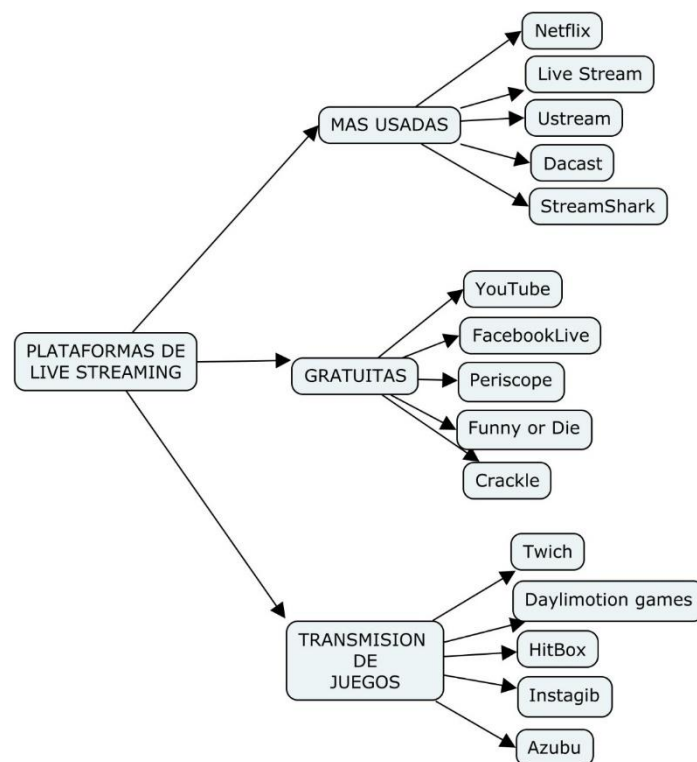
La difusión de video en vivo, mejor conocido como *live streaming*, tiene como función principal permitir el acceso remoto a los usuarios que deseen conectarse a la red para acceder al contenido en tiempo real. Esto permite a los usuarios visualizar diversos eventos masivos a distancia (por ejemplo, conciertos, premiaciones, etc.), cuando su ubicación geográfica no les permite estar presentes físicamente en dicho evento. Con el ritmo tecnológico actual, este servicio debe ser flexible y adaptativo frente a las nuevas generaciones de dispositivos y el creciente número de usuarios que desean ver contenido en vivo. En el tercer trimestre del año 2019 se observó un incremento del 91.8% en el uso de las plataformas de *live streaming*. En el gráfico de la Figura 2 se puede observar el aumento en miles de personas referente al uso de la plataforma de twitch en la categoría de música en vivo.





*Figura 2. Aumento de personas en la plataforma twitch*

Hoy en día existe un vasto número de plataformas que ofrecen este tipo de servicio, por ejemplo, las redes sociales que permiten llevar a cabo la transmisión de video en vivo. En la Figura 3 se pueden observar algunas de las plataformas que brindan servicio de video en vivo.



*Figura 3. Clasificación de plataformas Live Streaming*

Los servicios de *live streaming* pueden ser distribuidos por redes *CDN*. La arquitectura *CDN* (*Content Delivery Network*) cuenta con un servidor principal donde se encuentra alojado el contenido, posteriormente este servidor distribuye el contenido a diversos servidores que se encuentran establecidos en otras partes geográficas, para que los usuarios accedan al contenido desde los diferentes servidores, evitando con esto la congestión en el servidor principal, por esta razón, la *CDN* brinda estabilidad en la transmisión. Sin embargo, este tipo de red implica un costo elevado por el número de servidores involucrados.

Por otro lado, existe una red llamada *P2P* (*Peer to Peer*) en la cual los usuarios involucrados pueden operar como servidores y consumidores al mismo tiempo. En las redes *P2P* los usuarios son conocidos como pares (*peers*), ya que no solo descargan el contenido del archivo, sino que también están habilitados para compartirlo; este tipo de redes generan un costo menor debido a que no se emplean servidores fijos, sino que se aprovecha a cada uno de los *peers* y por ende se logra una mayor escalabilidad.

La arquitectura híbrida *CDN-P2P* es una tecnología empleada para el servicio de transmisión de video debido a que ofrece la escalabilidad de las redes *P2P* y la estabilidad de una red *CDN*. Algunos trabajos de investigación han tomado a bien implementar y evaluar esta arquitectura con el fin de probar que son eficientes en la distribución de video a gran escala sobre Internet [2]. En [3] se hace hincapié que la arquitectura *CDN-P2P* ha sido propuesta para sistemas de transmisión, con la finalidad de alcanzar la escalabilidad de una red *P2P*, así como los pequeños retardos y alto rendimiento de una red *CDN*. Por ejemplo, la plataforma *LiveSky* emplea este tipo de arquitectura.

En consideración con lo descrito anteriormente, el desarrollo de este proyecto se enfoca en emplear un sistema híbrido *CDN-P2P* para la transmisión de video en vivo, el cual brinda la estabilidad de una red *CDN* y al mismo tiempo la escalabilidad de una *P2P*.

El resto del documento se organiza de la siguiente forma: el Capítulo 1 está orientada a exponer el problema que se va a abordar en el proyecto, así como las limitaciones, justificaciones y objetivos de este. En el Capítulo 2 se abordan los trabajos relacionados con este proyecto. En el Capítulo 3, se proporciona la teoría necesaria para comprender la terminología empleada. Después, en el Capítulo 4 se realiza el análisis de problemáticas similares a la abordada en este proyecto. El Capítulo 5 proporciona el diseño de la solución a implementar para la cadena de Markov y el diseño del nuevo esquema de asignación de recursos bajo una distribución uniforme. Finalmente, se establecen las conclusiones generales de este trabajo de investigación.

## 1.2 Planteamiento del Problema

Actualmente, las tendencias de distribución de contenido apuntan al servicio de transmisión de video en vivo como un sector de gran importancia en la vida diaria. Derivado de la situación que se vivió dentro de la sociedad en los últimos años debido a la pandemia SARS-Cov2 (covid-19), el aforo a eventos masivos era limitado para disminuir el riesgo de contagio, por lo cual la transmisión de contenido en vivo fue un factor importante que hasta hoy día resulta favorable. Y ahora estamos más acostumbrados a realizar actividades de forma remota sin necesidad de acudir físicamente. Tal es el caso de premiaciones, partidos de fútbol, conciertos, concursos, por mencionar algunos. Estos servicios hacen uso de la transmisión en vivo para que el contenido llegue a todos los usuarios que están interesados en este. De igual manera, algunas aplicaciones como *Facebook Live*, *Instagram Stories*, *Periscope*, *WhatsApp*, *Snapchat*, etc. [4], resultan innovadoras y han crecido rápidamente gracias a que transmiten contenido de este estilo.

Sin embargo, cuando se presenta una transmisión de video en vivo desde una plataforma *streaming*, debido a la alta demanda del video por parte de los espectadores, se presenta sobrecarga de peticiones a los servidores donde se aloja temporalmente el contenido, y se ocasiona el efecto cuello de botella en esta parte de la red. Esto afecta la velocidad de flujo y en consecuencia un retardo en la distribución del contenido, ya que no llega al mismo tiempo a todos los usuarios interesados en este. De igual forma, se afecta la experiencia de visualizar el contenido en tiempo real; para lo cual el uso de una red básica cliente-servidor resulta insuficiente [5] por el hecho de que no proporciona un alcance adecuado para proveer el servicio a todos los clientes que consumen el video en vivo.

La actual variedad de aplicaciones multimedia, en conjunto con el desarrollo acelerado de las comunicaciones móviles y la accesibilidad de múltiples dispositivos finales para visualizar contenido en vivo y bajo demanda, permiten emplear el ancho de banda del tráfico de datos presente en las redes subyacentes, con lo cual se modifica el modelo cliente-servidor tradicional incrementando el ancho de banda de subida de una red. [1]

En la literatura existen trabajos que han modelado servicios de video basados en cadenas de Markov. Estos modelos han probado ser muy flexibles y al compararse con otros métodos de modelado han mostrado ser muy precisos. Desafortunadamente, estos desarrollos se han concentrado en servicios de video bajo demanda [5] [6] [7]. Hasta este punto de la investigación se conoce que el modelado con cadenas de Markov aún no ha sido aplicado a servicios de video en vivo.

Cabe mencionar que estos modelos utilizados para VoD no pueden ser aplicables directamente a *live streaming* debido a que:

- ✚ En VoD los usuarios tienen poca sincronización entre ellos, puesto que el contenido ya se ha generado con anterioridad y los usuarios no esperan una experiencia en tiempo real. En cambio, en *live streaming*, los usuarios están altamente sincronizados. Como consecuencia, los procesos de arribos en cada caso son significativamente diferentes.
- ✚ Otra consecuencia de la sincronización en *live streaming* es que los usuarios están interesados únicamente en fragmentos del video de lo que está ocurriendo en tiempo real, mientras que en VoD se pueden tener usuarios interesados en intervalos muy diversos del video. Por lo tanto, las poblaciones que pueden compartir recursos son muy diferentes en cada caso.
- ✚ En VoD las tasas de descarga pueden ser significativamente más altas que las tasas de reproducción, porque el contenido ya ha sido almacenado. Lo anterior permite al usuario de VoD prealmacenar gran parte del contenido. Por otro lado, en *live streaming*, este prealmacenamiento intensivo no es factible, lo que, de nueva cuenta, afecta al modelo e incluso a los esquemas de enfocados en compartir contenido.
- ✚ En VoD el video es estático, por lo tanto, las tasas de descarga y reproducción se mantienen de cierto modo constantes, sin embargo, en *live streaming* la tasa de producción está altamente ligada con la velocidad a la que se genera el video (ocurre el evento) y en consecuencia el usuario debe descargar el contenido a esa velocidad para mantenerse en tiempo real.

Debido a que, en la transmisión de video en vivo, el contenido se genera de manera simultánea a su visualización, se pueden presentar diversos congelamientos en la reproducción, en consecuencia, se pierde la QoE de visualizar el contenido en tiempo real. Esto se atribuye a múltiples factores, por ejemplo, que no existan los *peers* suficientes para distribuir el contenido a los *peers* que se conecten posteriormente al sistema.

En este caso, el servidor principal se ve obligado a proporcionar el contenido a todos los *peers* que arriben al sistema, sin embargo, este hecho puede incrementar el costo de la transmisión de video afectando inherentemente al usuario final. Por otro lado, si el video se pausa de forma continua, los usuarios pierden el interés en visualizarlo y, por lo tanto, dejan de seguirlo, esto genera pérdidas a las plataformas que distribuyen el contenido.

Ante este escenario se plantea la siguiente pregunta: ¿Cómo desarrollar un modelo, basado en una cadena de Markov, que represente el comportamiento de una población de *peers* que consume el servicio de video en vivo y que además permita el diseño de un nuevo esquema de asignación de recursos?

### 1.3 Propuesta de Solución

Con este proyecto lo que se busca es modelar un servicio de video en vivo que se distribuye a través de una red híbrida *P2P-CDN* para conocer el comportamiento de los *peers* una vez que se encuentran conectados al sistema y comienzan el proceso de descarga del contenido de una transmisión en vivo.

El video con el cual se trabaja en este modelo es el conjunto de señales de audio y video empaquetados en un solo archivo, debido a que se desea que el usuario tenga la mayor calidad de servicio y experiencia en la visualización del contenido. Tomando en cuenta que el archivo de video se genera en pequeños fragmentos llamados *chunks*, se propone agruparlos en secciones más grandes a las cuales denominaremos ventanas.

Esta agrupación se realiza debido a que en este proyecto se plantea que un *peer* pueda descargar una ventana de múltiples *peers* habilitados para compartirle ese contenido, es decir, cuando un *peer* desee descargar una ventana, este podrá descargar fragmentos de esta desde diferentes *peers* como se muestra en la Figura 4.

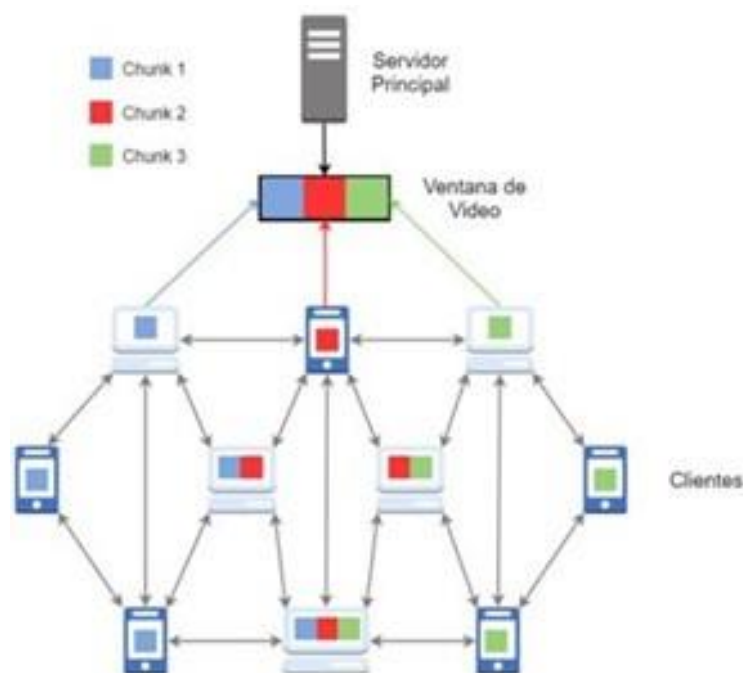


Figura 4. Descarga de una ventana de video

La descarga de las ventanas que componen al video se realizará de acuerdo con el esquema de asignación de recursos que se propone en el presente trabajo porque con esto se espera reducir el ancho de banda demandado a los servidores y que el ancho de banda demanda por los *peers* para descargar el video dentro del sistema sea atendido en su mayoría por el ancho de banda de subida de la red *P2P*.

Una vez que se tiene la segmentación de los fragmentos del video, los usuarios se agrupan de acuerdo con la ventana que se encuentren descargando en ese instante, de esta manera se procede a formar poblaciones de *peers* por ventana de video. Una población de *peers* se refiere a los usuarios que están descargando la misma ventana del video en un mismo instante, la población recibe el nombre de acuerdo con la ventana de descarga, es decir, si se encuentran descargando la ventana cero, la población de *peers* en esta ventana se llamará población  $X_0$ .

Dentro del sistema desarrollado existen cuatro sucesos principales que tienen ocurrencia dependiente del comportamiento de los *peers* y los cuales afectan directamente al estado de la cadena de Markov. Los sucesos o eventos son los siguientes:

- a) La llegada o conexión de un nuevo usuario al sistema. Este suceso será cuantificado por una tasa de conexión llamada  $\lambda$ .
- b) Transferencia de un *peer* a la ventana inmediata superior. La cual ocurre cuando un *peer* en la población  $X_i$  termina de descargar el contenido de la ventana  $i$  y comienza la descarga de la ventana  $i + 1$ . Este suceso ocurre a una tasa  $(\tau_i)$
- c) Transferencia de un *peer* a la ventana inmediata inferior. Este suceso ocurre cuando un *peer* no descarga el video a la misma tasa que se produce el contenido y por ende ingresa una nueva ventana de video a la hiperventana, provocando un corrimiento en la posición de las ventanas. Es decir, si un *peer* está descargando la ventana  $i$  cambiará a la ventana  $i - 1$ , este suceso es cuantificado por la tasa  $(P_\omega)$ .
- d) Desconexión del sistema. Este suceso ocurre cuando por alguna circunstancia alguno de los *peer* dentro del sistema sale de la transmisión en tiempo real antes que esta finalice. Este suceso será estimado por la tasa de desconexión  $\theta$ .

Por otro lado, el esquema de asignación de recursos definido en este proyecto tiene como principal objetivo aprovechar al máximo los recursos provenientes de la red *P2P*, para lo cual se establece que los *peers* podrán obtener recursos para descargar el video de *peers* que tengan un mayor progreso en la descarga del contenido en vivo.

El diseño del esquema de asignación de recursos es el resultado del análisis de esquemas reportados en la literatura. Del cúmulo de esquemas investigados se eligen a los esquemas de asignación de recursos  $Q$  ventanas hacia atrás y DVPG para ser analizados y plantear en cuál de ellos es conveniente realizar adecuaciones para los servicios de video en vivo, adicional al esquema de asignación de recursos de distribución uniforme (DU).

En primera instancia, el esquema desarrollado se realizará siguiendo el eje de funcionamiento del esquema de asignación de recursos de distribución uniforme,

el cual indica que un usuario puede obtener recursos de usuarios ubicados en ventanas superiores a la que este está descargando.

En ese mismo sentido, se obtendrá una expresión de acuerdo con el esquema de distribución uniforme siguiendo los parámetros y limitaciones propios de un servicio de video en vivo. De forma general, se expresa, abiertamente, que el esquema de distribución uniforme no presenta una asignación de recursos óptima, debido a que los usuarios reciben los recursos en función del tamaño de su población.

Lo descrito anteriormente puede generar que algunas poblaciones de *peers* tengan recursos disponibles de sobra, mientras que otras poblaciones operen bajo condición de penuria en cuanto a recursos se refiere. A pesar de esto, el análisis basado en la distribución uniforme resulta de gran ayuda para comprender como se puede delimitar el acceso a los recursos de tal forma que el desempeño del sistema sea óptimo.

Además, este análisis facilita la comprensión de la asignación de recursos. Es decir, al estudiar el esquema de asignación de recursos de distribución uniforme y aplicarlo a servicios de video en vivo se comprende la forma en que los recursos son distribuidos dentro de la red y la razón de esta distribución.

De manera preliminar, el diseño del esquema de asignación de recursos para servicios de video en vivo también se realiza tomando como base los esquemas de asignación de recursos: Esquema Q ventanas hacia atrás y DVPG.

A lo largo del desarrollo del presente proyecto y una vez analizados los esquemas de asignación de recursos, se establecen las bases que se tomaron para el desarrollo de las expresiones descritas en este documento.

Debido a que la simulación de este tipo de sistemas es realmente compleja y se requiere de diversos equipos tecnológicos para validar los resultados, el proyecto se limita a evaluar matemáticamente por implementación la cadena de Markov que representa al sistema y a evaluar de la misma forma el esquema de asignación de recursos propuesto para servicios de video en vivo. Dichas evaluaciones se realizarán tomando parámetros de operación utilizados en sistemas de distribución de video reales.

## 1.4 Alcances

Este proyecto se limita a definir una cadena de Markov que represente el comportamiento de los usuarios en el proceso de descarga a lo largo de una transmisión de video en vivo a través de una red híbrida *P2P-CDN*. En dicha cadena se observa de igual forma la relación que tienen los usuarios con el archivo de video y la composición de este. Finalmente, en la cadena se muestran

los sucesos que generan un cambio en su estado al momento de ocurrir, así como la tasa a la cual ocurre alguno de estos.

El desarrollo de la cadena se realiza de forma cuasi paralela al diseño de un nuevo esquema de asignación de recursos. En primera instancia se considera al esquema de asignación de recursos tradicional uniforme para plasmar el comportamiento de los usuarios y la forma en la cual estos consumen recursos para poder descargar el video en vivo.

Posteriormente, se plantea trabajar la cadena definida bajo el esquema uniforme adecuado a servicios de video en vivo. Finalmente, se propone un nuevo esquema de asignación de recursos diseñado de acuerdo con esquemas de asignación de recursos que se han investigado hasta el momento, pero con la diferencia de abstraer el método que emplean para asignar los recursos y adaptarlo al concepto de *live streaming*.

El estado de la cadena de Markov definida para servicios de video en vivo considera que un instante ocurre un único suceso por parte de los usuarios (conexión, desconexión o transferencia).

Las evaluaciones por implementación de la cadena de Markov y del nuevo esquema de asignación de recursos que se propondrá en este proyecto se realizarán empleando parámetros (tasa de subida, tasa de descarga, tasa de desconexión, etc.) que se han reportado en trabajos de investigación relacionados con los servicios de video y los esquemas de asignación de recursos.

La solución matemática de la cadena de Markov y del esquema de asignación de recursos se implementan en el ambiente de desarrollo Matlab, con la finalidad de obtener la cantidad de usuarios promedio situada en cada una de las ventanas que componen al video contenido dentro de la hiperventana.

Estos resultados son traducidos a tasas de subida y descarga requeridos por el sistema para la distribución de contenido en vivo y de esta forma conocer el desempeño general y las condiciones que presenta el sistema.

Debido a que un archivo de video se produce por pequeños *frames*, a los cuales se les denomina *chunk* y estos se agrupan en ventanas, un usuario conectado al sistema, conocido a partir de su conexión como *peer*, podrá obtener *chunks* de múltiples *peers* que tengan la capacidad de compartir el contenido.

El sistema que se desarrolla considera que, al momento de la conexión, un *peer* comienza la descarga del contenido en la ventana 0 de la hiperventana con la finalidad de que este cree *buffer* y así evitar congelamientos en el proceso de descarga del video y pierda sincronía con la producción del video.



Este proyecto no considera la implementación de un simulador para este modelo, sin embargo, se presentarán las estadísticas de anchos de banda consumidos y el desempeño del sistema, como ya se ha mencionado en función de parámetros en sistemas reales.

## 1.5 Justificación

A raíz de la pandemia por SARS-CoV-2 el consumo de servicios de *streaming* y de visitas virtuales presento un incremento respecto al consumo de *live streaming* en años anteriores [8]. Actualmente, múltiples plataformas de distribución de video y redes sociales disponen de este servicio para compartir eventos del día a día en tiempo real. Este servicio permite tener acceso a contenido generado de forma simultánea a su distribución dentro de la red, desde cualquier ubicación geográfica del planeta. *Live streaming* es un servicio que se distribuye en redes del tipo tradicional (cliente-servidor) y redes híbridas (*P2P-CDN*).

Existen diversos modelos para servicios de video bajo demanda (*VoD*), sin embargo, no se tiene conocimiento acerca de un modelo que represente los servicios de video en vivo. Por esta razón es que se ha decidido realizar el modelado de este tipo de servicio mediante cadenas de Markov. El análisis de los eventos, sucesos y comportamiento por parte de los usuarios en el consumo de *live streaming* es análogo a lo que sucede en el ambiente del video bajo demanda, pero con la diferencia de que *live streaming* es un archivo dinámico y no estático como el caso de *VoD*.

Por otra parte, también existen múltiples esquemas de asignación de recursos aplicados a *VoD* y no se conoce algún esquema propio para video en vivo. En este proyecto se han resaltado los esquemas con un óptimo desempeño para servicios de video bajo demanda con el fin de conocer los parámetros que emplean para la distribución de recursos entre los usuarios y analizar el desempeño que presentan. Este análisis permitió desarrollar un esquema de asignación de recursos funcional con un óptimo desempeño para servicios de video en vivo.

Por lo descrito anteriormente es que se decidió desarrollar en este proyecto el modelado de servicios de video en vivo mediante una cadena de Markov y desarrollar un esquema de asignación de recursos propio para los servicios de video en vivo sobre una red híbrida *P2P-CDN*. El modelado mediante cadenas de Markov permitió conocer el comportamiento de los usuarios al momento de consumir un servicio de video y su repercusión en la dinámica del sistema que lo transmite.

A su vez, este modelo permite conocer las tasas de operación del sistema y su relación con el desempeño general del mismo. Así mismo, se podrán obtener

estadísticas de consumo y desempeño del sistema en base a los anchos de banda que obtenidos de los *peers* y los servidores.

Por otro lado, este esquema también busca obtener mayor proporción de recursos de la red *P2P* y disminuir el consumo de recursos provenientes de la red *CDN*.

## 1.6 Metodología

La metodología que se optó implementar para el desarrollo del proyecto es *SCRUM*, debido a que esta se caracteriza por ser una metodología basada en una estructura de desarrollo incremental, lo cual quiere decir, que cualquier ciclo de desarrollo del producto y/o servicio se descompone en pequeños proyectos divididos en distintas etapas: análisis, desarrollo, implementación y evaluación. En la etapa de desarrollo se realizan interacciones del proceso o *sprint*, es decir, entregas regulares y parciales del producto final.

De acuerdo con la complejidad de este proyecto y el número de desarrolladores, esta metodología se adapta de forma adecuada, ya que permite abordar proyectos complejos que exigen una flexibilidad y una rapidez esencial al momento de ejecutar los resultados.

*SCRUM* está orientada a gestionar y normalizar los errores que se puedan producir en procesos de desarrollo de larga duración, a través de, reuniones frecuentes para asegurar el cumplimiento de los objetivos establecidos. Para este proyecto esta característica ha sido fundamental al momento de programar reuniones para debatir conceptos, establecer técnicas de trabajo, definir objetivos e intercambiar experiencias personales y de esta forma fomentar la proactividad.

Finalmente, *SCRUM* aporta al proyecto: innovación, flexibilidad, competitividad y productividad.

Siguiendo la metodología antes menciona se han trazado las siguientes fases para el desarrollo de este proyecto:

### Análisis

En esta fase del proyecto se realizaron las siguientes actividades:

- Investigación sobre los conceptos clave de archivos de video
- Investigación sobre las cadenas de Markov
- Investigación sobre conceptos de arquitecturas de red
- Investigación acerca del modelado de servicios de video mediante cadenas de Markov
- Investigación de esquemas de asignación de recursos

- Análisis de modelos para servicios de video bajo demanda (VoD)
- Análisis de esquemas de asignación de recursos
- Análisis de una cadena de Markov unidimensional

Con base en las actividades antes mencionadas se redactaron las siguientes secciones del presente documento: Introducción, planteamiento del problema, propuesta de solución, alcances, objetivos y marco teórico.

## **Diseño**

En esta etapa del proyecto se realizaron las siguientes actividades:

- Diseño de la cadena de Markov
- Diseño de la solución matemática de la cadena
- Bosquejo del nuevo esquema de asignación de recursos

Para el diseño de la cadena de Markov que representa los servicios de video en vivo distribuidos sobre una red híbrida *P2P-CDN*, se definieron, en primer lugar, los sucesos (comportamiento de los *peers*) que afectan directamente al estado de la cadena y producen un cambio en este.

Posteriormente, se definió la estructura que debe tener el archivo de video y la clasificación de los *peers* una vez que están conectados al sistema. Además, se estableció la nomenclatura que representa las tasas de ocurrencia de un suceso por parte de los usuarios.

Finalmente, se identificaron los cambios que debe tener el estado de la cadena de Markov una vez ocurrido un suceso.

En el diagramado de la solución matemática se describe el procedimiento a seguir para evaluar por implementación la cadena de Markov. Es decir, se estableció el algoritmo a implementar para simular la ocurrencia de sucesos por parte de los usuarios que se encuentran dentro de un sistema de servicio de video en vivo y así obtener estadísticas de consumo y desempeño de este sistema basado en las transiciones del estado de la cadena.

Para el bosquejo del nuevo esquema de asignación de recursos se analizaron distintos esquemas de asignación de recursos enfocados al servicio de video bajo demanda y se identificaron las principales diferencias entre este servicio y el de *live streaming* para de esta forma poder establecer un esquema a fin para el servicio de video en vivo.

## **Implementación**

En esta etapa del proyecto se desarrollaron las siguientes actividades:

- Implementación del diagrama de flujo de la solución matemática de la cadena de Markov: esta actividad se refiere a la implementación de los eventos por parte de los usuarios (conexión, desconexión o transferencia)

en un IDE que permitió obtener estadísticas de consumo y desempeño significativas para la solución de la cadena y el diseño de un nuevo esquema de asignación de recursos.

- Evaluación del nuevo esquema de asignación de recursos
- Evaluación del esquema propuesto en el proyecto frente a esquemas desarrollados en otros trabajos de investigación.
- Corrección de errores de implementación como lógica de programación y sintaxis.

## Evaluación

Finalmente, en esta etapa, se realizarán pruebas de las evaluaciones por implementaciones de la cadena de Markov y el nuevo esquema de asignación de recursos para obtener estadísticas del funcionamiento del sistema de servicio en vivo. De acuerdo con los resultados obtenidos en esta etapa, se realizaron modificaciones, en caso de ser necesarias, para obtener los resultados del ancho de banda de los *peers* mayor al del servidor.

## 1.7 Objetivos

### 1.7.1 Objetivo General

Desarrollar un modelo matemático basado en una cadena de Markov, con la finalidad de diseñar un esquema eficiente de asignación de recursos para un sistema de video en vivo, soportado por redes híbridas *CDN-P2P*.

### 1.7.2 Objetivos Específicos

Plantear una cadena de Markov basada en las conexiones y desconexiones de los *peers* de cada una de las ventanas del video, para modelar el comportamiento de los múltiples usuarios en el sistema.

Evaluar la cadena de Markov dentro del modelo híbrido *CDN-P2P* mediante la implementación de métodos numéricos, con la finalidad de estimar la demanda de descarga y la cantidad de recursos de subida disponibles en el sistema.

Diseñar y evaluar un nuevo esquema de asignación de recursos, basado en la solución de la cadena de Markov, con el propósito de mejorar el desempeño del esquema de asignación uniforme tradicional.

# Capítulo 2

## Estado del Arte

En este capítulo se presentan diversas investigaciones relacionadas con el análisis de servicios de video, de manera particular servicios de video bajo demanda (*VoD*), y al análisis de esquemas de asignación de recursos dentro de sistemas para este tipo de servicios.

### 2.1 Estado del arte del modelo de servicios de video con cadenas de Markov

En esta sección se introducen investigaciones que muestran gran relación con el análisis de servicios de video bajo *demand* (*VoD*) mediante cadenas de Markov y que son tomados como base para el desarrollo del análisis de servicios de video en vivo mediante cadenas de Markov en el presente proyecto. Esto debido a que hasta este momento no se han encontrado en la literatura modelos basados en cadenas de Markov para servicios de video en vivo.

En [9] los autores rescatan las principales características de las redes *CDN* y *P2P* de manera individual y de forma híbrida. En este trabajo se realiza un modelo híbrido *CDN-P2P* sobre la web con la finalidad de disminuir las peticiones realizadas a los servidores *CDN*; se menciona que el *streaming* de video en vivo es una tecnología que permite a un usuario denominado emisor generar un video que se transmite a otros usuarios en tiempo real.

En este trabajo se mencionan algunas ventajas de las redes *CDN* y *P2P*. De la *CDN* explican que tiene una excelente calidad en el servicio y costos de implementación y mantenimiento relativamente altos. Por otro lado, *P2P* tiene una mejor escalabilidad y menores costos de implementación. El modelo propuesto en este trabajo tiene 2 capas fundamentales, la capa *CDN* y la capa *P2P*. En ese modelo el usuario final se sitúa directamente en la capa *P2P* y en caso de no encontrar los recursos necesarios en su nivel los solicitará al nivel de arriba (*CDN*).

En [10] se compara y evalúa el rendimiento de la red *CDN* frente a la red *P2P*; el punto clave de la *CDN* es replicar el contenido del servidor original en un caché local y de ahí distribuir el contenido a los clientes. Mientras que en la arquitectura *P2P* la distribución se aplica del lado de la red del cliente, ya que estos se

convierten en asociados activos al proporcionar el contenido que están recibiendo a otros clientes.

En [11] los autores desarrollan un modelo de transmisión de video sobre una red híbrida *CDN-P2P* para lograr estabilidad y escalabilidad en la difusión de contenido. Aunado a esto, mencionan que el modelo cuenta también con otros componentes por sus siglas en inglés *SDN* (redes definidas por software), *NSP* (proveedores de servicio de red), *VCP* (proveedor de contenido de video) que permiten tener mejores parámetros en la visualización, la experiencia y el servicio. El modelo al ser “basado en *SDN*” permite que todos los usuarios puedan aprovechar los recursos disponibles en la *CDN-P2P* de forma inteligente para tener mejor calidad de experiencia y de servicio. Por otro lado, con *NSP* y *VCP* se logra tener mayores recursos en la periferia de un usuario y con ello se logra un mejor rendimiento en la red. De igual forma, en este trabajo los autores mencionan los algoritmos empleados para clasificar a los *peers* en diversos grupos y con esto verificar que los recursos sean asignados equitativamente.

En [12] se proporciona un sistema de transmisión híbrido (Video en vivo y *VoD*), basado en codificación de video escalable *SVC* (*Scalable Video Coding*). El sistema consiste en un transcodificador en línea. En este trabajo se propone un servidor de *streaming* basado en *SVC* para hacer paquetes tipo *RTP* y distribuirlos dentro de la red. El servidor *P2P* recibe los paquetes desde el servidor *streaming* y se encarga de distribuirlos a los *peers* (empleado en el sistema en vivo). Además, existe un *tracker* con el fin de mantener la información y el estado de todos los *peers*, para clasificarlos en *seed peer* y *common peer* dependiendo de la carga y descarga que estén realizando. Finalmente, se menciona que el reproductor *SVC* únicamente está disponible para PC's desarrolladas como un complemento en ActiveX y teléfonos inteligentes. Con este modelo se logra reducir la probabilidad de pérdida y retraso de paquetes.

En [7] se establece un modelo para la distribución de video bajo demanda en redes heterogéneas 5G. En este trabajo se aborda un modelo fluido (sistema de ecuaciones) que representa el comportamiento de los *peers* conectados al servicio. Para analizar el comportamiento de los *peers* se tomaron en cuenta parámetros como duración del video, ancho de banda de los servidores, tiempo de permanencia, entre otros. Estos datos permitieron evaluar el esquema para observar si la asignación de recursos se realizaba de forma uniforme y equitativa para todos los usuarios. Los resultados fueron comparados con resultados obtenidos en otras investigaciones que utilizan cadenas de Markov para analizar el compartimiento de los *peers* en esos sistemas. De igual forma, en este trabajo se establece que si el modelo es exitoso se puede generar un modelo que reduzca el número de recursos utilizados para su funcionamiento. Los resultados que se obtuvieron se consideran veraces por la similitud que tienen con los resultados basados en cadenas de Markov de otras investigaciones basadas en

el funcionamiento real que muestra la plataforma de distribución de contenido YouTube.

En [6] los autores presentan un *framework* analítico para modelar el servicio de video bajo demanda (*VoD*) a través de una red híbrida *CDN-P2P*, su trabajo está basado principalmente en un modelo fluido y en cadenas de Markov. El modelo fluido permite a los autores comprender el comportamiento del sistema en términos de retrasos por parte de los usuarios, congelamiento en la transmisión del contenido, desfase en la reproducción de una ventana y la descarga de la ventana posterior, la clasificación de los *peers*, etc. Por su parte, la cadena de Markov se emplea para conocer la tasa de subida y bajada que presenta el sistema, es decir, la cantidad de *peers* que se encuentran en cada grupo ( $j, k$ ), lo que quiere decir que dichos *peers*, están reproduciendo la ventana  $j$  y descargando la ventana  $k$ . Al final del trabajo, los autores muestran los resultados numéricos obtenidos basándose en diferentes parámetros QoS.

En [13] el autor propone un nuevo esquema de asignación de recursos llamado Distribución por Ventanas Priorizadas (DVP). El esquema propuesto en este trabajo demostró ser más eficiente comparado con otros esquemas de asignación de recursos, esto al reducir el ancho de banda solicitado a los servidores. Este trabajo se enfoca en analizar un archivo de video segmentado en ventanas de igual tamaño (red heterogénea) mediante un modelo fluido y complementar con un análisis mediante cadenas de Markov. Para realizar dichos análisis, el autor se basa en diversos parámetros de QoS y QoE. Cabe mencionar que el video analizado es un video bajo demanda (*VoD*) a través de una red *P2P*. Finalmente, el autor reporta los resultados obtenidos a partir de diferentes parámetros QoS y QoE de entrada.

En [5] se menciona que las redes *P2P* son una tecnología clave para la distribución de contenido de video dentro de las próximas generaciones de comunicaciones inalámbricas, incluyendo la quinta generación de sistemas móviles denominada 5G. En este trabajo se fragmenta el contenido de video en ventanas que a su vez están compuestas por partes más pequeñas llamadas *chunks*, además se dice que el tamaño de la primera ventana puede ser diferente a la de las demás que componen el archivo de video, con la finalidad de obtener un equilibrio entre el retraso inicial y la duración de las pausas.

## 2.2 Estado del arte de esquemas de asignación de recursos para servicios de video

En esta sección se presentan trabajos de investigación en los que se ha abordado la asignación de recursos para servicios de video bajo demanda (VoD) con la finalidad de conocer el desempeño y funcionamiento de dichos esquemas.

En [7] el autor menciona que los recursos disponibles en una red deben ser asignados correctamente para obtener un buen desempeño del sistema para la descarga y visualización de un video. En este trabajo se implementa una red heterogénea, lo cual indica que todos los usuarios conectados a la red tendrán la misma oportunidad de acceder a los recursos disponibles. Sin embargo, esto implica que los recursos se distribuyen de acuerdo con el esquema de asignación de recursos uniforme.

Por esa razón, en este trabajo el autor propone un esquema de asignación de recursos diferente, al cual llamó esquema de *Q ventanas hacia atrás*. Este esquema establece que todos los *peers* que deseen descargar y reproducir un video tendrán la misma oportunidad de consumir recursos de *peers* en *Q* ventanas hacia adelante y del servidor principal. Con esto se garantiza una correcta distribución de recursos y se evita que algunos usuarios se encuentren en sobre abundancia mientras que otros están en penuria. El parámetro llamado *Q*, es definido como la cantidad de ventanas superiores desde donde un usuario podrá consumir recursos.

En [14] el autor menciona las ventajas que ofrece una red híbrida *CDN-P2P* para la transmisión, descarga y reproducción de un video bajo demanda (VoD). En este trabajo se realizan diversas comparaciones de esquemas de asignación de recursos con el fin de obtener puntos de mejora y así proponer un nuevo esquema de asignación de recursos. Como resultado de esta investigación se tiene un esquema de asignación de recursos que tiene como parámetro a *Q*, que es la cantidad de ventanas hacia arriba que se tomarán en cuenta para obtener recursos, es decir, si un *peer* desea descargar la ventana *i*, los usuarios que le podrán proporcionar recursos son los usuarios que estén descargando la ventana *j* siempre y cuando  $i < j < i + Q$ . Por otro lado, este esquema es iterativo, es decir, si un video está constituido por *N* ventanas, el parámetro *Q* deberá cambiar en función de  $i + Q < N - 1$ .

En [6] los autores hacen una evaluación de diferentes esquemas de asignación de recursos presentes en la distribución de contenido sobre redes *CDN*, *P2P* y redes híbridas *CDN-P2P* con el fin de obtener el desempeño que estos esquemas muestran frente a la distribución de video bajo demanda y así proponer un nuevo esquema de asignación de recursos con base en el modelo fluido que describe el autor, en el cual se explica el comportamiento de los *peers* y las reglas que se deben seguir para una correcta descarga y reproducción del video.

El autor menciona que en diversos esquemas únicamente basta con que un *peer* desee descargar el contenido de una ventana *i* y el grupo de *peers* en ventanas



superiores, es decir, en ventanas  $j > i$  le proporcionarán recursos, sin embargo, también menciona que esta práctica inevitablemente genera una disparidad y provoca que algunos *peers* tengan más recursos disponibles para consumir que otros. En el esquema que propone, los recursos provenientes de los servidores *CDN* y de los demás *peers* conectados al sistema se asignan de manera equitativa. En este esquema se establece un parámetro de prioridad denotado por  $\varepsilon$ , el cual controla la cantidad de recursos asignada a un grupo de *peers* perteneciente a una ventana  $i$ . Finalmente, el autor establece que este esquema es mejor en comparación con el esquema uniforme y el esquema PWA.

En [13] el autor un nuevo esquema de asignación de recursos llamado Distribución por Ventanas Priorizadas (DVP). La idea principal del esquema propuesto en este trabajo es asignar recursos con mayor prioridad para la descarga de las partes del video que se encuentran en penuria, específicamente a las ventanas altas.

Dentro de la asignación de los recursos por priorización de ventana existen dos tipos de recursos: los provenientes de los servidores y los provenientes de los *seeds*. El uso de DPV para asignar los recursos de los *downloaders* provee un esquema más flexible y eficiente, ya que propicia que los *peers* en una ventana en específico aprovechen los recursos de *downloaders* en ventanas inmediatas superiores.

En este esquema, los recursos de los *downloaders* en la ventana  $i$  son inútiles para *peers* con progresos de descarga mayores, es decir, situados en ventanas  $j$ ,  $j > i$ . Finalmente, el autor realiza una generalización del esquema DPV y lo llama DVPG, esto con el fin de analizar de forma conjunta los recursos provenientes tanto de *seeds* como de *downloaders* (*peers*).

En [15] los autores analizan un archivo de video bajo demanda que es dividido en ventanas y consumido por diversos usuarios conectados al sistema. Estos usuarios reciben el nombre de *peers* y requieren consumir recursos para descargar y reproducir el video. En este artículo el autor propone un esquema de asignación de recursos llamado *INUA* (*Immediate-neighbors Uniform Allocation*).

El esquema *INUA* establece que un *peer* podrá obtener recursos de *peers* con un progreso de descarga mayor, es decir, un *peer* que se encuentra descargando la ventana  $j$ , podrá obtener recursos de usuarios en ventanas  $i$ ,  $i > j$ . Al conjunto de ventanas con mayor progreso de descarga se le llama  $U_j$  y es de longitud  $M_j$ . Por lo tanto, las ventanas  $i$  que pueden proporcionar recursos están acotadas por los límites  $j + 1 < i < j + M_j$ .

# Capítulo 3

## Marco Teórico

En esta sección se describe al video en vivo, así como los diversos formatos empleados para la codificación del contenido. De igual forma, se muestran los formatos utilizados para el empaquetamiento de este. Enseguida se muestran los protocolos para la distribución de video en vivo por Internet. Y finalmente se enuncia la manera en la cual operan las redes híbridas *CDN-P2P* en los servicios de video en vivo.

### 3.1 Video en Vivo

El análisis de un archivo de video que se realiza en esta sección tiene como objetivo comprender la estructura, característica y distribución de un video. De igual forma se mencionan algunas plataformas que permiten la distribución de contenido y la manera en la cual se realiza esta distribución para que los usuarios finales tengan acceso a este contenido.

Una vez analizados los puntos anteriores, surge una pauta para delimitar y definir las características del archivo de video que será el objeto de estudio para realizar el modelado de un sistema de distribución de video en tiempo real. Además de establecer la forma en que el video será “distribuido” entre los diversos usuarios que se encuentran dentro del sistema a modelar en este trabajo.

#### 3.1.1 Formatos de codificación

En la transmisión de contenido en vivo, se usa como principal herramienta a un *códec*, que es el acrónimo de codificador-decodificador. Esto se debe a que la señal de video necesita ser codificada en un formato de *streaming* que permita tener las menores pérdidas de calidad y sonido posibles. Los *códecs* realizan la codificación *streaming* para comprimir el contenido a emitir y descifrarlo una vez recibido en los diferentes dispositivos móviles para la reproducción.

A continuación, se describen algunos de los formatos de codificación que permiten la transmisión de audio y video en un solo paquete a través de una red. Información obtenida a partir de [16] [17].

Mjpeg: Un video no es más que una sucesión de imágenes en movimiento. Si se comprimen todas las imágenes en formato JPEG se obtendría el formato

MJPEG, o Motion JPG. Con este formato se logra una buena compresión con respecto al original.

H264: es un formato de codificación de video para grabar y distribuir señales de video FullHD y audio. Fue desarrollado y mantenido por el ITU-T Video Coding Experts Group (VCEG) con el ISO/IEC JTC1 Moving Picture Experts Group (MPEG). Empleado normalmente para grabación, compresión y distribución de contenidos de vídeo, el formato H.264 es un método de transmisión de video compatible con redes de datos, que suministra imágenes de alta calidad sin consumir demasiado ancho de banda. En la Figura 5 se puede observar el proceso de codificación empleando H264.

Mpeg4: Es una evolución de MPEG. Es más avanzado y ofrece numerosas opciones de configuración y muy alta calidad. Sin embargo, su edición exige gran potencia en hardware.

VP8 (Formato de Compresión de Video o Especificación de Compresión de Video): Es una especificación para codificar y decodificar video de alta definición tanto en archivo como en flujo de bits para visualizar.

VP9: Es de código abierto y libre de regalías. Fue desarrollado por *Google* como sucesor del VP8, la alternativa moderadamente exitosa al H.264. Durante su desarrollo VP9 fue apodado “NGOV” (*Next Gen Open Video*) y *Google* ya ha integrado el soporte en el navegador *Chrome* y en *YouTube*.

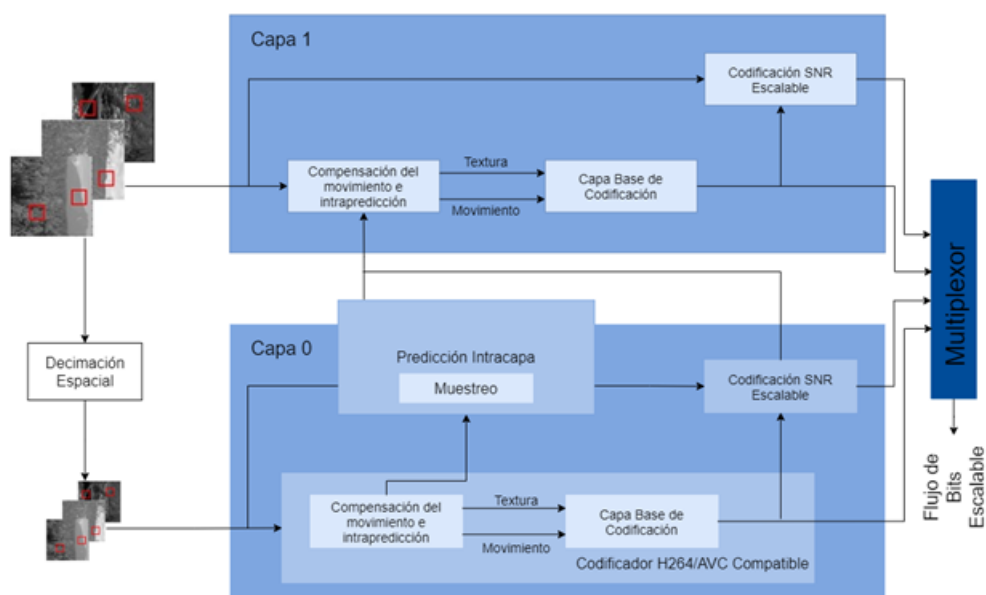


Figura 5. Diagrama a bloques del códec H264 [18]

### 3.1.2 Formatos de empaquetamiento

A continuación, se mencionan los formatos de empaquetado más usados dentro de la distribución de video y audio en vivo [19]

.avi: Es el empaquetado multimedia con mayor utilidad como archivo de vídeo. Significa audio video interleave, es decir video y audio entremezclados.

.mov: Empaquetado por defecto de *QuickTime* y productos *Apple*. Otras extensiones recientes compatibles con .mov son: .mp4, .m4a, .3gp, .mj2.

.mpeg: Empaquetado ampliamente utilizado, por ejemplo, en la emisión TDT. Otras extensiones compatibles son .MOD y .mpg. En la Figura 6 se muestra el proceso de empaquetado .mpeg.

.vob: Formato mpeg2 con opción de encriptado para su uso en el DVD.

.flv: Formato propio de las animaciones *flash*.

.ogg: Empaquetado tradicionalmente usado para el audio vorbis.

.mkv, .matroska: Empaquetado especial para alta densidad de datos en HD.

.webm, .WebM: Formato de empaquetado libre de última generación

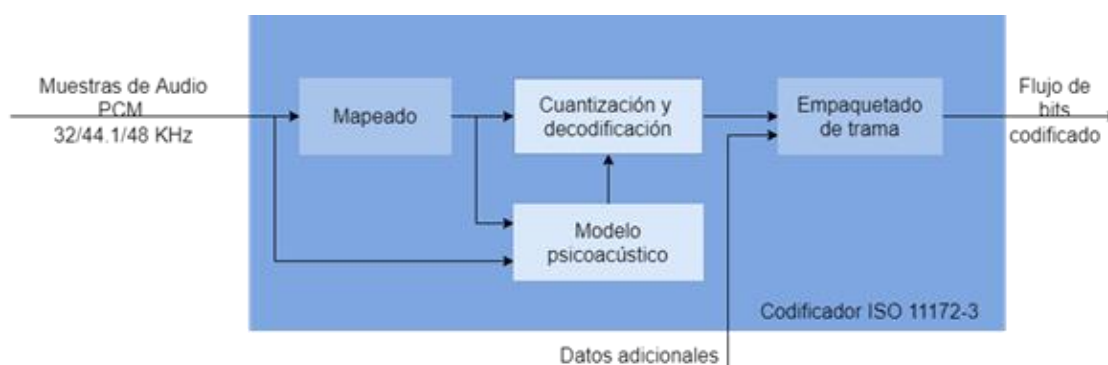


Figura 6. Diagrama a bloques del formato de codificación mpeg [20]

### 3.1.3 Servicios de video en vivo

Un *stream* de contenido multimedia, generalmente llamado video, está compuesto por varios *streams* simples que poseen un tipo de contenido específico (audio, video o subtítulos). Cada *stream* simple es codificado por un *códec* en un formato específico como h264, mp4 o mjpeg para el caso de la señal de video. En el caso de la señal de audio existen formatos como Mp3, aac, ogg, pm\_s16le.

Los contenidos de un *stream* de multimedia son empaquetados en un nuevo formato llamado contenedor. Como se mencionó en la sección 3.1.2 algunos ejemplos de contenedor son mov, avi, mpeg, flv, vob son algunos de estos. El objeto de emplear un *container* es lograr sincronía entre los contenidos. Una vez realizado el empaquetado el *stream* de multimedia se puede almacenar como archivo en un servidor o se puede transmitir en directo.

Los archivos de video comúnmente son producidos y distribuidos por una amplia gama de equipos de usuario (UE), tales como teléfonos celulares, dispositivos inteligentes, o dispositivos conectados al Internet de las cosas (*IoT*). Aunado a esto, durante el proceso de producción de videos, se pueden encontrar diferentes tipos de contenido de video, como películas, anuncios, aplicaciones de realidad aumentada, videos de seguridad, etc. En la Tabla 1 se enlistan las características principales de los archivos de video y en la Tabla 2 las características principales de un video en vivo.

Característica	Descripción
Espacio de Color	Modelo matemático que asigna una representación del color a su equivalente perceptivo.
Resolución	El número de píxeles en una ventana de video, expresada como $N * M$ píxeles, donde N y M representan el número de columnas y filas respectivamente, en la matriz de píxeles.
Proporcionalidad	Relación entre el ancho y alto de la ventana de video.
Tasa de Ventana	El número de Ventanas por segundo(fps) de un video.
Tasa de bit	El número de bits necesario para representar un segundo de un video codificado.
Calidad de Video	Fidelidad de un video codificado respecto a su versión original, la cual es medida subjetivamente o mediante una métrica objetiva (PSNR, VQM y VMAF)
Esquema de Codificación	Formato de codificación de video y parámetros de codificación.

*Tabla 1. Atributos de los videos [21]*

Característica	Descripción
Tiempo real	El evento es capturado a través de un dispositivo y transmitido por la red al mismo tiempo.
Interacción en tiempo real	En caso de que los usuarios puedan realizar preguntas y/u observaciones, estas ocurren a lo largo de la transmisión en vivo.

Sentido de pertenencia	La audiencia espectadora experimenta sensación de exclusividad o pertenencia al grupo que tiene acceso al contenido.
Base para VoD	El video capturado en tiempo real se convierte en VoD si se almacena y los usuarios acceden a este tiempo después de su grabación.
Edición	El video capturado no tiene ediciones gráficas o de sonido, es decir, se transmite tal cual se graba.
Manejo	Los usuarios no pueden manipular el video en cuanto pausarlo, adelantar o retrasarlo se refiere.
Fases	Consta de 3 fases principales: grabación, procesamiento y distribución.

*Tabla 2. Características del video en vivo [22] [23]*

Los servicios de video distribuidos a través de la Internet son agrupados en dos principales categorías, los servicios bajo demanda y los servicios de *live streaming*.

La distribución de contenido multimedia hace referencia al proceso por el cual se distribuyen los servicios, como películas, clips de video y/o transmisiones en vivo sobre una red, ya sea en tiempo real o no. Existen 2 métodos principales de dispersión de contenido multimedia sobre una red, *downloading* y *streaming*. Por objetivos del proyecto el *streaming* y una parte de *downloading* marcarán uno de los ejes principales de investigación. [4]

*Downloading (progressive downloading)*: es el procedimiento donde el usuario puede visualizar el contenido multimedia mientras lo está recibiendo de otro dispositivo.

*Streaming*: a diferencia de *downloading*, en este método de distribución se cuenta con un servidor *streaming* especializado, el cual distribuye el contenido multimedia a los diversos usuarios conectados. La distribución se realiza considerando los recursos específicos por asignar a cada uno de los usuarios en función de su solicitud. En este método no es necesario que el archivo de video sea descargado en el dispositivo final del usuario.

El presente proyecto está enfocado en el servicio de *live streaming*. Plataformas como *Sling TV*, *DirecTV Now*, *YouTube TV*, y *Live TV de Hulu* [24] son las encargadas de transmitir contenido en vivo.

### 3.1.4 Protocolos para la distribución de video en vivo por internet

En la transmisión de video en vivo diversas tecnologías trabajan colectivamente para lograr la distribución del contenido sobre una red. Esto se debe a que el *streaming* implica diversas etapas. En primer lugar, se capta el contenido desde el dispositivo donde se produce, posteriormente este contenido entra en la etapa de codificación para así poder ser almacenado y transmitido a los diversos espectadores de manera simultánea.

Un protocolo de transmisión es la tecnología empleada para transportar archivos de video a través de Internet. [25]

Anteriormente el video disponible en la red era entregado por el protocolo RTMP, que es el protocolo de mensajería en tiempo real. Este protocolo es considerado como un estándar basado en *Flash* para *live streaming*. Actualmente aún es utilizado para el envío de video desde el codificador RTMP a una plataforma de video en línea.

Sin embargo, el protocolo RTMP basado en *Flash* ya no es apropiado para entregar video a los usuarios. Esto debido a que el complemento *Flash* ha perdido importancia debido a que los dispositivos que admiten este protocolo son cada vez menos. En los últimos diez años el protocolo RTMP ha sido reemplazado paulatinamente por el protocolo HLS [25]. De forma paralela el protocolo MPEG-DASH ha tomado un lugar importante en *live streaming*.

MPEG-DASH es el protocolo de transmisión utilizado con mayor frecuencia en los últimos días. Este protocolo fue creado como una opción frente a la fragmentación que sufrió el mercado de transmisión de video causada por la competencia de HLS de *Apple* con diversos protocolos de transmisión de video. Esto obligó a las organizaciones de estándares a desarrollar MPEG-DASH como un protocolo de transmisión estándar de código abierto, al igual que el protocolo de transmisión HLS.

MPEG-DASH es un método de tasa de bits adaptativo para la transmisión de video. Por otro lado, ha mostrado ser compatible con la publicidad y la tecnología, razón por la cual debe estar a la vanguardia. MPEG-DASH es compatible con DRM, entrega HTTP, transmisión de baja latencia y otras características de *live streaming*.

Dispositivos *Android*, *iOS*, *Windows*, *Mac*, *Linux*, *Chrome OS* por mencionar algunos, trabajan con los protocolos HLS y MPEG-DASH. Sin embargo, este último no es compatible con el navegador *Safari* móvil, por lo tanto, no es admitido por la mayoría de los usuarios *iPhone*, *iPad* y *AppleTV* ya que utilizan el navegador *Safari* como buscador predeterminado.

HLS y MPEG-DASH son protocolos de velocidad de bits adaptables. Lo cual se traduce, en que los usuarios reciben automáticamente el video con mejor calidad a razón de la calidad de conexión a Internet que poseen. En un escenario ideal este hecho proporciona una experiencia de visualización estable y de alta calidad a los espectadores, mientras que se reduce el almacenamiento en búfer y el retraso.

A medida que el tráfico de video en las redes celulares crece exponencialmente, los operadores de redes móviles (ORM) aplican tecnologías de quinta generación (5G) de redes de comunicaciones. Esto con la finalidad de satisfacer los requisitos de calidad de servicio y de experiencia en las aplicaciones multimedia. El objetivo principal de la implementación de estas tecnologías es ofrecer servicios multimedia de alta velocidad de datos, baja latencia y fiables en comunicaciones móviles de banda ancha y de baja latencia

Lo descrito anteriormente se logra introduciendo la computación de borde multiacceso (MEC), que integra tecnologías de computación en la nube y de redes inalámbricas. La idea principal de la MEC es asignar recursos informáticos a los usuarios finales dentro de la red de acceso de radio (RAN). El creciente interés por aprovechar los recursos de borde para ofrecer mejores experiencias multimedia ha provocado que surjan múltiples plataformas MEC comerciales. [21]

Las capacidades de almacenamiento de contenido en caché de las redes centradas en la información (ICN) se han combinado con MEC. Esto con el propósito de proporcionar capacidades integradas de almacenamiento en caché, computación y comunicación (edge-C3). En las aplicaciones multimedia, el edge-C3 es capaz de procesar y almacenar simultáneamente contenido de video en caché para proporcionar a los usuarios servicios de baja latencia y gran ancho de banda. Este mecanismo se muestra en la Figura 7, donde se puede observar que los UE cuentan con capacidades de computación y almacenamiento de mayor potencia. Esto les permite participar también en el edge-C3. Además, el crowdsourcing móvil y la comunicación de dispositivo a dispositivo (*D2D*) permiten que los equipos de los usuarios adyacentes puedan compartir recursos entre sí, lo que reduce la congestión de la red y los recursos que deben utilizarse en los servidores de borde.



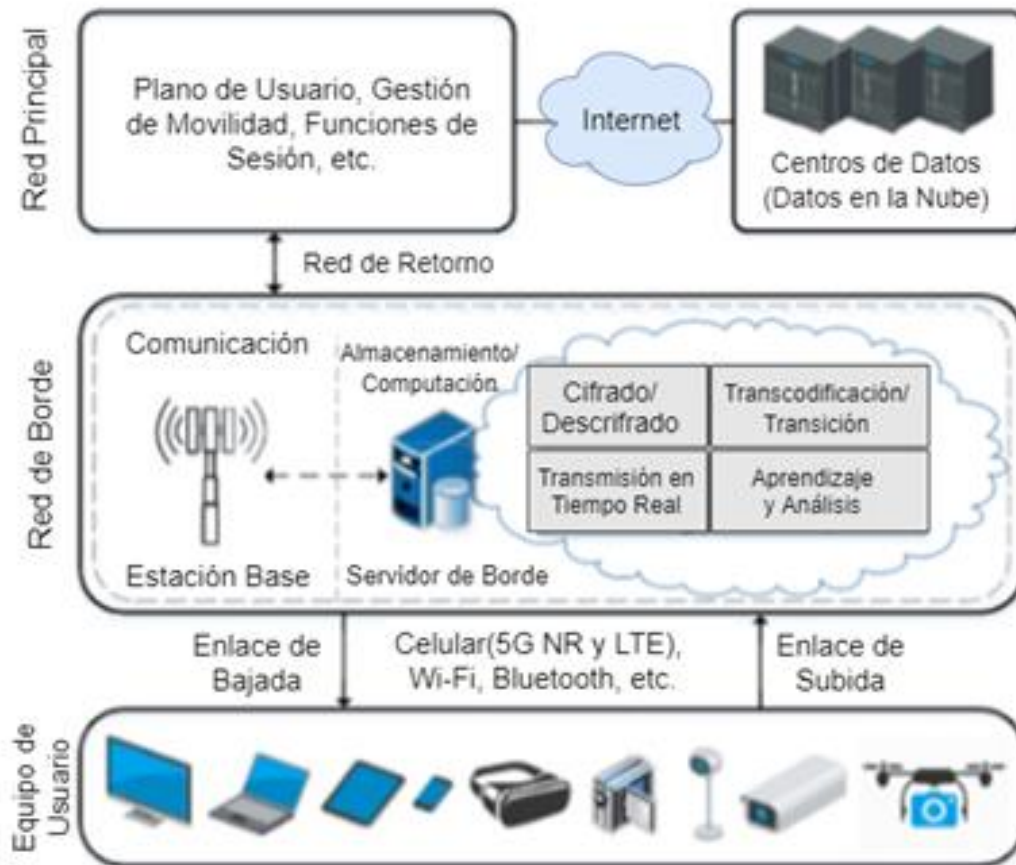


Figura 7. Escenario de video Edge-C3 en redes inalámbricas [21]

## 3.2 Redes CDN-P2P

Las redes *P2P* (*Peer to Peer*) son utilizadas para incrementar la capacidad de los sistemas. En comparación con los sistemas convencionales cliente-servidor, las redes *P2P* reducen el tráfico de peticiones a los servidores gracias al trabajo colaborativo realizado por los nodos. En este sentido las redes *P2P* resultan ser más escalables, es decir, su capacidad aumenta conforme aumenta la cantidad de usuarios conectados al sistema.

Como se ha mencionado las redes *P2P* han demostrado ser más escalables, sin embargo, su rendimiento disminuye por factores asociados al comportamiento de los usuarios tal como *freeriding* (*peers* no cooperativos) y *churning* (desconexión inesperada de *peers* habilitados para distribuir el contenido). [5]

En un inicio las redes *P2P* surgieron como un sistema de distribución de archivos, sin embargo, en años recientes el análisis de este tipo de redes para la distribución de video ha incrementado. En este tipo de servicio es importante

resaltar que la reproducción de un video es iniciada a pesar de que su descarga esté en curso.

Las redes *P2P* representan una tecnología estrechamente relacionada con la distribución de video en la quinta generación de sistemas móviles “5G”. Las redes *P2P* tienen como base principal al protocolo *BitTorrent* [5], el cual tiene como principal objetivo dividir un archivo de video en diversos segmentos denominados *chunks*. Estos pequeños fragmentos de video permiten que los diversos *peers* en el sistema descarguen un archivo de video intercambiando *chunks* entre sí.

En las redes *P2P*, el protocolo *BitTorrent* separa a los *peers* dentro del sistema en dos grupos: *leechers*, *peers* que contienen una porción del archivo de video. Y *seeders*, *peers* que poseen el archivo de video por completo y permanecen en el sistema con el fin de compartir sus recursos con otros *peers*. Tanto los *seeders* como los *leechers* comparten sus recursos con otros *leechers*.

Dentro de una red *P2P*, cuando un *peer* se une al sistema para descargar el video, se debe contactar con un nodo llamado *tracker*, cuya función principal es crear una lista de los *peers* que poseen una porción o el video en su totalidad. Posteriormente el *tracker* muestra una lista de los *peers* habilitados para compartir sus recursos con el usuario que llegó. Finalmente, el usuario se pone en contacto con los *peers* en la lista y establece que fragmentos de video descargará de cada *peer* con el que esté conectado.

Por otra parte, dentro de las redes de distribución de contenidos (*CDN*), los servidores son proporcionados por diversos proveedores de servicios de Internet (*ISP*). Estos servidores permiten a los usuarios finales acceder a los diferentes contenidos disponibles en la red, reduciendo el retraso y mejorando la calidad del servicio (*QoS*); desgraciadamente la cantidad de tráfico de video cada vez es mayor, por lo tanto, las *CDN* se enfrentan con problemas de escalabilidad debido al costo de su implementación. [5]

En este contexto, los sistemas híbridos *CDN-P2P* muestran ser eficaces por la capacidad de escalabilidad de las redes *P2P* y la estabilidad proporcionada por la infraestructura fija de las redes *CDN*. Esto se debe a que las redes *CDN-P2P* consideran las características de los archivos de vídeo (la popularidad, la tasa de codificación y el tamaño), las propiedades de las redes *CDN* (capacidad de carga), las propiedades de las redes *P2P* (tasas de datos de carga y descarga, el tiempo de permanencia), y los esquemas implementados para asignar los recursos disponibles. [6]

En trabajos como [5] y [6] agrupan un número determinado de *chunks* en segmentos de video relativamente más grandes llamados ventanas. En consecuencia, clasifican a los *peers* de acuerdo con la ventana de video que se

encuentran descargando. En la Figura 8 se puede observar la clasificación de los *peers*, que encuentran dentro de una red *CDN*, mediante un plano cartesiano.

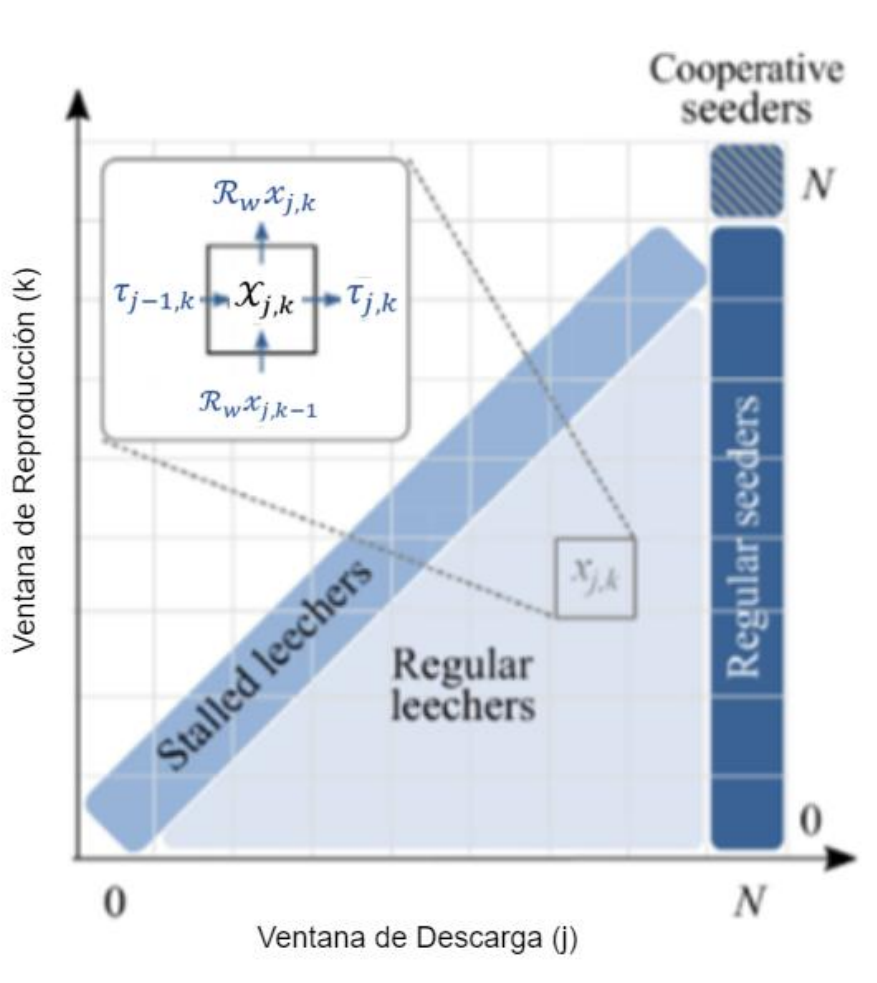


Figura 8. Clasificación de *peers* en un escenario *CDN* básico [6]

En las redes híbridas *CDN-P2P* los recursos (tasa de subida) disponibles de las redes *CDN* y *P2P* se suman para atender las peticiones de descarga. Estos recursos son distribuidos acorde con un esquema de asignación de recursos dado.

El esquema de asignación de recursos permite obtener la cantidad de tasa de subida que proporciona la *CDN* y la cantidad que provee la *P2P* con el objetivo de satisfacer la demanda de descarga para un servicio de video. Estas tasas de datos están directamente relacionadas con el costo operativo de la *CDN* y con la cantidad de recursos que deben proceder de la red *P2P* (energía y el ancho de banda de subida).

### 3.3 Cadenas de Markov

En las secciones anteriores se ha mencionado el modelado de sistemas de distribución de video a través de diferentes métodos, uno de ellos es el modelado de un sistema mediante cadenas de Markov. Este método ha sido seleccionado en el presente trabajo debido a que permite evaluar y/o simular el comportamiento que presentan los usuarios dentro de un sistema de distribución de video.

El estado de la cadena de Markov, al ser esta de tiempo discreto, permite emular los sucesos que pueden tener ocurrencia dado el comportamiento de un usuario y con esto determinar el comportamiento general del sistema.

Generalmente el comportamiento de un sistema, físico o matemático se puede representar describiendo los diferentes estados que puede ocupar e indicando cómo es el cambio entre estos estados. Suponiendo que el sistema que se modela ocupa uno y sólo un estado en cualquier momento del tiempo y su evolución se representa mediante transiciones de estado a estado, se puede decir, que estas transiciones se producen de forma instantánea, es decir, que el paso de un estado a otro no consume tiempo. Si la evolución futura del sistema depende únicamente de su estado actual y no de su historia pasada, el sistema puede representarse mediante un proceso de Markov. Las aplicaciones de los procesos de Markov se pueden encontrar ampliamente en las ciencias biológicas, físicas y sociales, así como en los negocios y la ingeniería (Libro probabilidad).

Un proceso de Markov es un tipo especial de proceso estocástico. Un proceso estocástico se define como una familia de variables aleatorias  $\{X(t), t \in T\}$ . En otras palabras, cada  $X(t)$  es una variable aleatoria y se define en un espacio de probabilidad. El parámetro suele representar el tiempo, por lo que  $X(t)$  denota el valor asumido por la variable aleatoria en el momento  $t$ .  $T$  se denomina conjunto de índices o espacio de parámetros y es un subconjunto de  $(-\infty, \infty)$ . Si el conjunto de índices es discreto, por ejemplo,  $T = \{0, 1, 2, \dots\}$ , entonces tenemos un proceso estocástico de parámetros discretos; de lo contrario, si  $T$  es continuo, por ejemplo,  $T = \{t: 0 \leq t \leq \infty\}$ , llamamos al proceso un proceso estocástico de parámetros continuos.

Los valores que asumen las variables aleatorias  $X(t)$  se denominan estados. El conjunto de todos los estados posibles forma el espacio de estados del proceso y éste puede ser discreto o continuo. Si el espacio de estados es discreto, el proceso se denomina cadena y los estados suelen identificarse con el conjunto de números naturales  $\{0, 1, 2, \dots\}$ , o un subconjunto de este. Un ejemplo de espacio de estado discreto es el número de clientes de un servicio, mientras que un ejemplo de espacio de estado continuo es el tiempo de espera de un cliente.

Para una cadena de Markov de tiempo discreto, su estado se encuentra definido en un conjunto discreto, pero infinito, de tiempos. Las transiciones de un estado a otro sólo pueden ocurrir, o no ocurrir, en estos instantes de tiempo. Por tanto, se puede representar, sin pérdida de generalidad, el conjunto de índices discretos  $T$  del proceso estocástico subyacente con el conjunto de números naturales  $\{0,1,2,\dots\}$ . Las observaciones sucesivas definen las variables aleatorias  $X_0, X_1, \dots, X_n, \dots$  en los pasos de tiempo  $0, 1, \dots, n, \dots$ , respectivamente. Formalmente, una cadena de Markov de tiempo discreto  $\{X_n, n = 0, 1, 2, \dots\}$  es un proceso estocástico que satisface la siguiente relación, denominada propiedad de Markov [26]:

Para todos los números naturales  $n$  y todos los estados  $x_n$

$$Prob\{X_{n+1} = x_{n+1} | X_n = x_n, X_{n-1} = x_{n-1}, \dots, X_0 = x_0\}$$

Así, el hecho de que el sistema esté en el estado  $x_0$  en el paso de tiempo 0, en el estado  $x_1$  en el paso de tiempo 1, y así sucesivamente, hasta el hecho de que esté en el estado  $x_{n-1}$  en el paso de tiempo  $n - 1$  es completamente irrelevante. El estado en el que se encuentra el sistema en el paso de tiempo  $n + 1$  sólo depende de dónde se encuentre en el paso de tiempo  $n$ . El hecho de que la cadena de Markov se encuentre en el estado  $x_n$  en el paso de tiempo,  $n$  es la suma total de toda la información relativa a la historia de la cadena que es relevante para su evolución futura.

En la cadena de Markov, la condición de abundancia (o penuria) depende del estado del sistema en el tiempo de simulación  $t$ .

Puesto que este estado es dinámico, no se puede asegurar que en cualquier instante de tiempo la cadena esté en abundancia o penuria, sino que una u otra condición tienen cierta probabilidad de ocurrir. Si se supone un punto de operación, tal que, según el modelo fluido, garantice abundancia, pero que está relativamente cerca del punto crítico; entonces, durante la evaluación de la cadena de Markov, existe una probabilidad no despreciable de que el sistema caiga en penuria durante algunos intervalos de tiempo, ya que las condiciones de abundancia en el modelo fluido sólo están dadas en términos de los valores promedio  $X_j$ . Lo anterior provoca que se reduzca la tasa de transición hacia ventanas superiores durante estos intervalos.

### 3.4 Servicios de video sobre redes híbridas *CDN-P2P*

Anteriormente se mencionó que parte esencial del modelado que se realiza en este trabajo tiene un interés particular por analizar y definir la estructura del archivo de video que se trabajará en este proyecto. Independientemente del tipo de video (bajo demanda o tiempo real) el empaquetamiento y distribución son similares, sin embargo, en este trabajo se analiza y modela el comportamiento

de un sistema de distribución de video en vivo. Por lo tanto, en esta sección se enfatiza en el video en tiempo real.

El video en vivo a través de internet, incluidos los servicios *OTT* e *IPTV*, han incrementado su polaridad en los últimos años [11]. Estos servicios se basan en el modelo cliente-servidor que utiliza la transmisión adaptativa del protocolo de transmisión de hipertexto (*HTTP*).

Los servicios de video *OTT* comúnmente emplean redes de distribución *CDN* que almacenan el contenido en caché en múltiples ubicaciones geográficas, conocidas como puntos de presencia (*PoP*). *CDN* no solo facilita la carga en el servidor de origen, sino que también reduce la latencia. Por otro lado, los servicios de *IPTV* emplean la transmisión de multidifusión del protocolo de internet *IP* a través de redes privadas habilitadas para multidifusión, donde una única transmisión es originada en el servidor y se entrega a varios clientes. Sin embargo, el protocolo de multidifusión de *IP* no se admite en la internet abierta, por lo tanto, no está disponible para los servicios de video.

Se han propuesto sistemas híbridos *CDN-P2P* para ofrecer un servicio *CDN* escalable al mismo tiempo que se reduce su ancho de banda y los costos del servidor. Los sistemas más recientes se centran en servicios *CDN-P2P* entregados a navegadores a través de canales de datos *WebRTC* sin depender de complementos de terceros o software de propietarios.

Las principales aplicaciones *CDN-P2P* basadas en *WebRTC* incluyen *SwarmCDN*, *PeerCDN* y *Sharefest* que a su vez condujeron a soluciones comerciales, como *Swarmify*, *Peer5* y *Streamroot*. En estos sistemas, los clientes que solicitan contenido son redirigidos a un servidor centralizado (*tracker*) que devuelve una lista de *peers* con el contenido deseado. Sin embargo, el intercambio de contenido entre *peers* no es administrado por el servidor central y estos sistemas pueden no ser compatibles con *NSP*. Un componente importante de los sistemas *P2P* es la formación de grupos.

Existe una serie de diferentes métodos de agrupamiento en la literatura, que se pueden aplicar a la formación de grupos *P2P* basados en mallas.

La aplicación *CDN* es un servicio que se ejecuta en el nodo de cómputo del centro de datos de *NSP Edge*. Extrae contenido de video en vivo desde el servidor de origen al centro de datos *NSP Edge*. Solo un pequeño subconjunto de pares recibe contenido en vivo desde el nodo *CDN*. Todas las demás solicitudes de contenido se redirigen a la aplicación *P2P*.

El control del servicio *P2P-CDN* que se propone en [11] consta de dos fases:

- i) La fase de inicio, la aplicación *P2P* envía una serie de fragmentos a los *peers* a través de un árbol de multidifusión para garantizar que cada usuario almacena en búfer un número predeterminado de

fragmentos antes de comenzar la reproducción. Se prefiere la multidifusión de *IP* administrada por el controlador *SDN* en la fase de inicio, ya que se sabe que la multidifusión de *IP* tiene un retardo bajo y no se espera un abandono de *peers* durante el corto período de inicio.

- ii) En la fase posterior a la reproducción, la aplicación *P2P* realiza una programación de fragmentos pseudo central basada en malla para todos los *peers* en cada grupo *P2P*. En esta etapa se usa la transmisión *P2P* basada en malla porque es más resistente a la rotación de *peers*. En los sistemas *P2P* de última generación, los *peers* realizan la programación de forma distribuida, lo que conduce a un uso ineficiente de los recursos de la red a medida que los *peers* se favorecen a sí mismos. Se propone una programación pseudo centralizada de fragmentos, realizados en los centros de datos *NSP-edge* para todos los *peers* dentro de cada grupo *P2P*, que sea justa para todos y utilice los recursos de la red de manera más eficiente. El procedimiento es escalable, ya que el número de *peers* en un grupo es limitado.

La fase de inicio continúa hasta que el servidor *CDN* inserta un número predeterminado de fragmentos a lo largo del árbol de multidifusión.

Los *peers* inician la reproducción y entran en la fase posterior a la reproducción tan pronto como almacenan estos fragmentos en búfer. La demora máxima de inicio se calcula cuando todos los *peers* del grupo inician la reproducción. La programación *P2P* basada en malla, posterior a la reproducción, se lleva a cabo en intervalos de tiempo fijos para minimizar la sobrecarga de señalización entre la aplicación *P2P* y los *peers*.

Como se puede notar en este capítulo tercero, en específico en esta última sección se habla de la distribución del video en tiempo real en aplicaciones que utilizan las redes híbridas *P2P-CDN* por las ventajas de estabilidad y escalabilidad que estas ofrecen. Si bien, se habla de forma específica acerca de las aplicaciones que distribuyen video en vivo y la manera en que estas lo hacen, el objetivo de este proyecto no está enfocado en simular una distribución real del video.

El modelado de un sistema de distribución de video en tiempo real abordado en este proyecto únicamente se centra en realizar un supuesto de distribución entre los distintos usuarios del sistema. A su vez el presente trabajo no considera la existencia de un *tracker* que proporciona la lista de usuarios con recursos disponibles para atender la demanda de descarga de otros usuarios (hablando de la aplicación *P2P*), por otra parte, en cuanto a la red *CDN* no se están

consumiendo recursos de servidores reales, sino más bien se toman datos publicados en artículos de investigación que abordan la distribución de video.

El esquema de asignación de recursos en este proyecto representa de cierto modo como se realiza la distribución del video entre los pares dentro del sistema y a su vez refleja el consumo de ancho de banda de los servidores *CDN*.

Por otro lado, el modelado de un sistema de distribución de video mediante una cadena de Markov permite conocer cómo es que los usuarios interactúan con el archivo de video. De ahí, se puede definir el cambio que se genera en el estado de la cadena de Markov a partir de un suceso ocurrido (comportamiento del usuario) y finalmente se puede analizar un comportamiento general del sistema evaluando por implementación este modelo.

En los capítulos siguientes se detalla a fondo la estructura y característica del video empleado para el modelado de un sistema de distribución de video en vivo sobre una red híbrida *P2P-CDN*, así como la asignación de recursos que se realiza para que los usuarios dentro de este sistema “descarguen” el archivo de video.



# Capítulo 4

## Análisis

En este capítulo se abordaron tres tópicos principales que son base teórica para el desarrollo del presente proyecto. El primero es el análisis del modelado de un sistema de consumo de recursos mediante una cadena de Markov, con la finalidad de conocer las expresiones y la solución por simulación de una cadena de este estilo. El segundo es el análisis de un modelo de servicio de video bajo demanda mediante cadenas de Markov, en este sentido se representa el consumo de recursos en un servicio de VoD así como la caracterización del comportamiento de los usuarios. Finalmente, el tercer tema es referente al análisis del funcionamiento de esquemas de asignación de recursos reportados en trabajos relacionados, con el objetivo de conocer la forma en que los recursos son asignados dentro de un sistema y los parámetros que influyen en dicha asignación.

### 4.1 Análisis de la cadena de Markov

En esta sección se realizó el análisis de una cadena de Markov unidimensional que representa a un sistema con pérdidas, arribos markovianos, número finito de servidores y sin cola. Con la finalidad de conocer la evolución en el estado de una cadena de Markov al ocurrir un arribo o pérdida y la forma en que se resuelve una cadena de Markov por simulación.

Dicha cadena representa a un sistema con  $S$  servidores disponibles para atender las peticiones de arribo de usuarios que ocurren a tasa  $\lambda$ . En este sistema se conoce el tiempo promedio de duración de un servicio ( $\mu$ ) y por ende se puede calcular la tasa de finalización de un servicio ( $\frac{1}{\mu}$ ). La cadena de Markov puede transitar entre el estado 0 y el estado  $S$ , es decir, los posibles estados de esta cadena son  $S+1$  estados.

La cadena de Markov se encuentra en estado cero cuando no existen servicios en atención dentro del sistema y se encuentra en estado  $S$  cuando todos los servidores disponibles se encuentran atendiendo a un servicio.

La cadena de Markov analizada considera que existe independencia entre los eventos que pueden ocurrir en el sistema, es decir, en un instante de tiempo ocurre un único evento (petición de arribo o finalización de servicio). Se dice que

la cadena es unidimensional debido a que los estados del sistema son identificados por una sola variable.

Las tasas que provocan una transición en el estado de la cadena son las siguientes:

$\lambda$ : Representa la velocidad promedio a la que ocurren las peticiones de arribo al sistema y las transiciones del estado  $i$  al estado  $i + 1$  en la cadena de Markov.

$\left(\frac{j}{\mu}\right)$ : Representa la velocidad promedio a la cual finaliza un conjunto de  $j$  servicios dentro del sistema. Esta tasa indica la velocidad a la que el estado de la cadena transita del estado  $j$  al estado  $j - 1$  y es el resultado de dividir el estado actual de la cadena  $j$  por la duración promedio de un servicio  $\mu$ .

En la cadena de Markov de la Figura 9 se representa al sistema de pérdidas sin cola, la transición entre los posibles estados y las tasas a las cuales ocurren dichas transiciones en el estado de la cadena.

En la Figura 9 se puede observar que los estados de la cadena se encuentran definidos de 0 a  $S$ , donde  $S$  es el número de servidores que atienden las peticiones de servicio y también representa el estado máximo de la cadena. De igual forma, en la Figura 9 se observa la transición entre los posibles estados y la tasa a la cual ocurre.

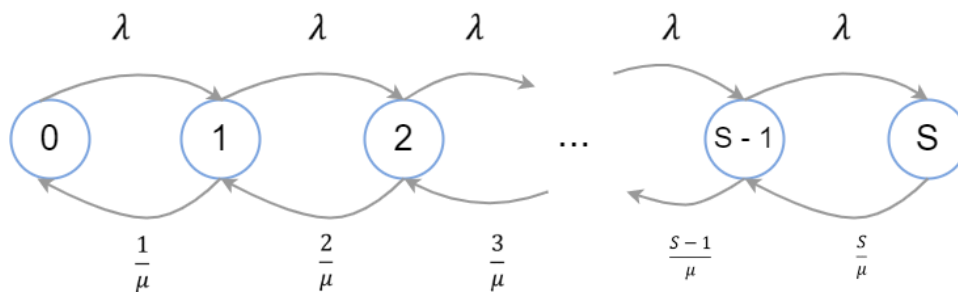


Figura 9. Cadena de Markov Unidimensional para sistema con pérdidas sin cola

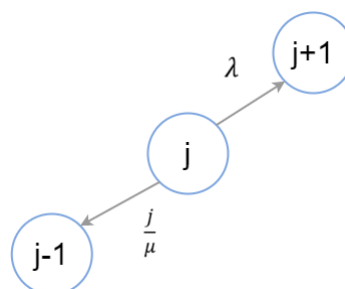


Figura 10. Transición en el estado de la cadena de Markov Unidimensional

En la Figura 10 se representan las posibles transiciones en el estado de la cadena de Markov de un sistema con pérdidas asociadas a su correspondiente tasa de ocurrencia, es decir, la cadena transita al estado superior ( $j + 1$ ) a tasa  $\lambda$  o transita al estado inferior ( $j - 1$ ) a tasa  $\frac{j}{\mu}$ .

Las transiciones de estado son eventos discretos, es decir, en un instante de tiempo sólo puede ocurrir una transición al estado superior o una transición al estado inferior. Estos eventos se implementan generando una variable aleatoria con distribución exponencial negativa con su respectiva tasa y se hace un acumulado del tiempo de estancia por estado para de esta forma obtener estadísticas de consumo.

Una vez analizada la cadena de Markov que modela un sistema con pérdidas y sin cola se realizó un diagrama de flujo ilustrado en la Figura 11, para dar solución por medio de simulación a la cadena de Markov y finalmente se implementó esta simulación en el IDE Matlab para obtener estadísticas de consumo y desempeño del sistema.

La implementación inicia definiendo los parámetros de entrada, es decir, las tasas que modificarán la posición del usuario en el estado de la cadena. Las variables más notables son: tasa de arribo de usuarios ( $\lambda$ ), duración promedio de un servicio ( $\mu$ ) y número de servidores ( $S$ ).

Posteriormente, se definen los siguientes vectores de longitud  $S + 1$ : tiempos de estancia por estado (*tiempo*) y arribos por estado (*contador*).

En el caso de que la cadena se encuentre en estado cero únicamente se genera una variable aleatoria con distribución exponencial negativa a partir de la tasa de arribo de usuarios ( $\lambda$ ) y se acumula su valor en tiempo de estancia en la posición 1 y se incrementa en una unidad al contador de arribos del estado 0.

En caso de que la cadena se encuentre en otro estado, contenido en  $[1, S]$ , se generan 2 variables exponenciales negativas (A y B) a partir de la tasa de arribo de usuarios ( $\lambda$ ) y la duración promedio de un servicio ( $\mu$ ) respectivamente. Posteriormente se elegirá la de menor valor para determinar qué evento ocurrió (petición de arribo o finalización de servicio) y se tienen los siguientes 2 casos:

\* Petición de arribo: Se almacena el valor de A en tiempo de estancia por estado en la posición (*EA: estado anterior*) y el contador de arribos por estado se incrementa en una unidad en la posición ( $EA + 1$ ).

\* Finalización de servicio: Se almacena el valor de B en tiempo de estancia por estado en la posición (*EA*).

La implementación de la solución matemática se realizó conforme al diagrama de flujo de la Figura 11.

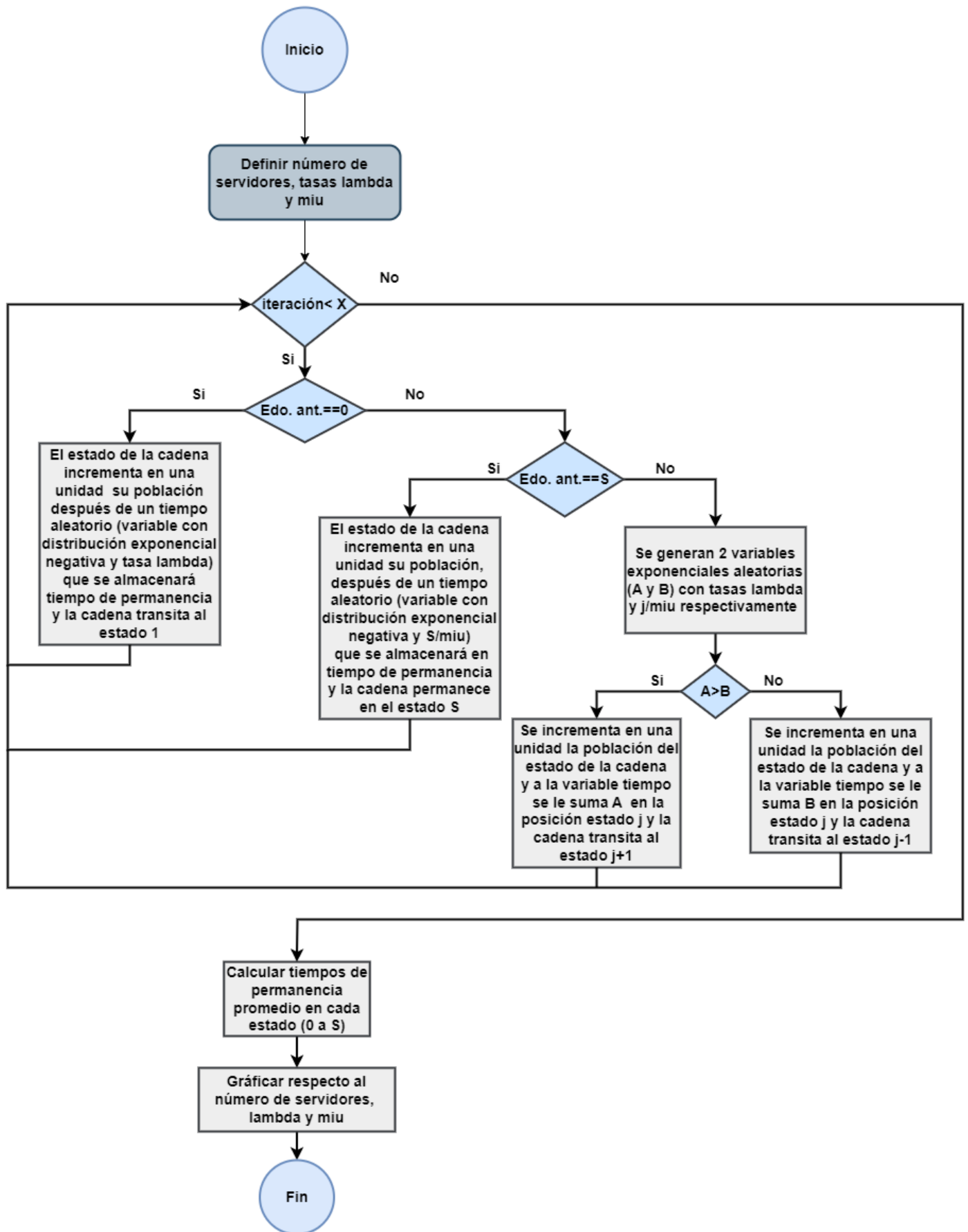


Figura 11. Diagrama de flujo para dar solución por simulación a la cadena de Markov

La cadena de Markov que representa un sistema con pérdidas sin cola se solucionó por dos métodos: el analítico y el numérico.

La solución numérica se llevó a cabo por dos métodos: tiempo promedio de estancia por estado y cantidad de arribos por estado. En ambos métodos se generan dos variables aleatorias (A y B) con distribución exponencial negativa. La variable A se genera con la tasa promedio de arribo de usuarios ( $\lambda$ ) y B se genera con la tasa promedio de finalización de servicio ( $\frac{j}{\mu}$ ).

De entre A y B se escoge la de menor valor con la finalidad de conocer que suceso ocurrió. A indica que ha ocurrido un arribo dentro del sistema y, por lo tanto, el estado de la cadena transita de  $j$  a  $j + 1$ . Por el contrario, B indica que un servicio que estaba en curso ha finalizado y el estado de la cadena transita de  $j$  a  $j - 1$ .

Existen dos casos borde en este sistema, el estado 0 y el estado S. 0 es el mínimo estado en que puede estar el sistema y por lo tanto cuando el sistema se encuentra en este únicamente se puede generar una variable A. Por su parte, el estado S es el estado máximo y cuando el sistema está en estado S únicamente se puede generar una variable S.

Finalmente, el cálculo de la probabilidad de que el estado se encuentre en estado  $j$  es el que se calcula de manera específica según el método empleado.

- Tiempo promedio por estado: Un vector contador que almacena el tiempo promedio de estancia por estado (valor de la variable mínima entre A y B) se incrementa en la posición  $j$  según la variable mínima. Finalmente se obtienen el tiempo de estancia promedio final por estado al dividir el tiempo almacenado en cada estado entre el tiempo total de simulación. Este valor promedio indica la probabilidad de que el sistema se encuentre en estado  $j$  y se grafica en función de las variables de entrada.
- Arribos por estado: Un vector de poblaciones se incrementa en una unidad en la posición  $i$  de acuerdo con el valor mínimo entre las variables A y B. En caso de que la variable A haya sido el mínimo (petición de arribo) se incrementa en la posición  $i = j + 1$ . En caso contrario, si B es el mínimo se incrementa en una unidad en la posición  $i = j - 1$  porque ocurrió la finalización de un servicio. Finalmente estas poblaciones por estado son divididas por el tiempo total de simulación y así se obtiene la probabilidad de que el sistema se encuentre en estado  $j$ .

Estos arribos permiten conocer en qué medida se realizó el cambio en el estado del sistema y son graficados en función de las variables de entrada.

La solución analítica, por su parte, se realizó mediante la expresión Erlang-B, que permite conocer la probabilidad, que depende del producto de la tasa de arribo de usuarios ( $\lambda$ ) y la duración promedio de un servicio( $\mu$ ), de que un sistema se encuentre en estado  $j$ .

$$P_j = \frac{\frac{(\lambda\mu)^j}{j!}}{\sum_{k=0}^S \frac{(\lambda\mu)^k}{k!}}$$

### *Ecuación Erlang-B*

Una vez obtenidos los resultados de la implementación se compararon gráficamente las curvas obtenidas con las soluciones numéricas (tiempo promedio por estado y arribos por estado) con la curva obtenida a partir de la solución analítica como se muestra en las Figuras 12-14.

En dichas gráficas se observa que las curvas de probabilidad de que el sistema se encuentre en estado  $j$  tienen valores aproximado en los tres casos considerando que el valor promedio de probabilidad varía de acuerdo con el producto de la tasa de arribos y la duración promedio de servicio.

De dicha comparación se puede rescatar que el método que muestra valores de probabilidad de estancia en el estado  $j$  con mayor proximidad a los valores de probabilidad mediante solución analítica es la solución numérica a partir de los arribos por estado, por lo tanto, a partir de ahora se retoma este método para realizar las evaluaciones de las próximas cadenas de Markov.

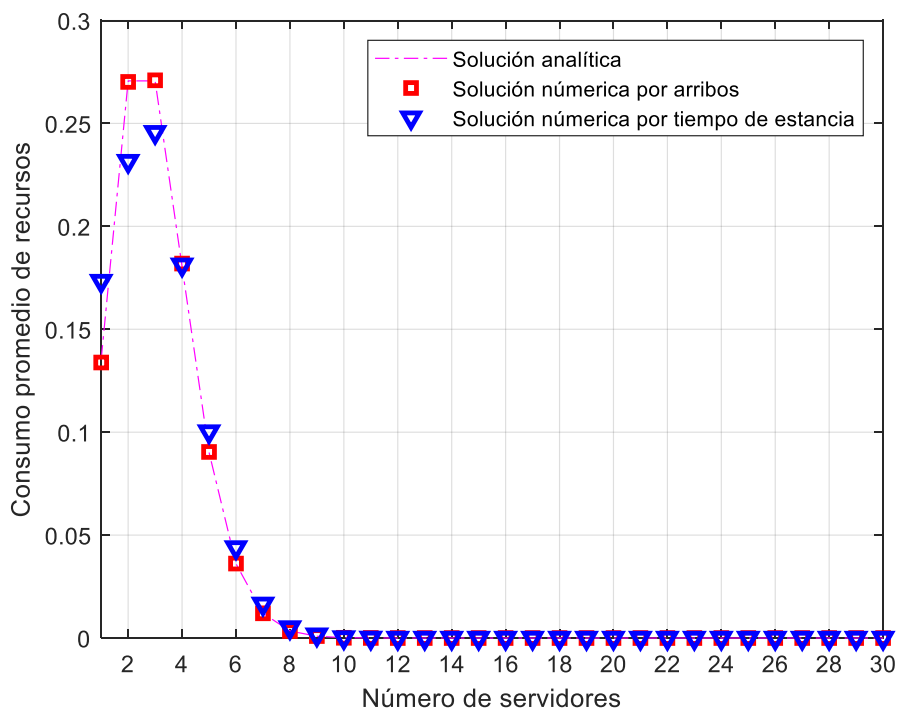


Figura 12. Gráfica de solución por implementación de la cadena de Markov para  $S=30$ ,  $\lambda = 4$  y  $\mu = 0.5$

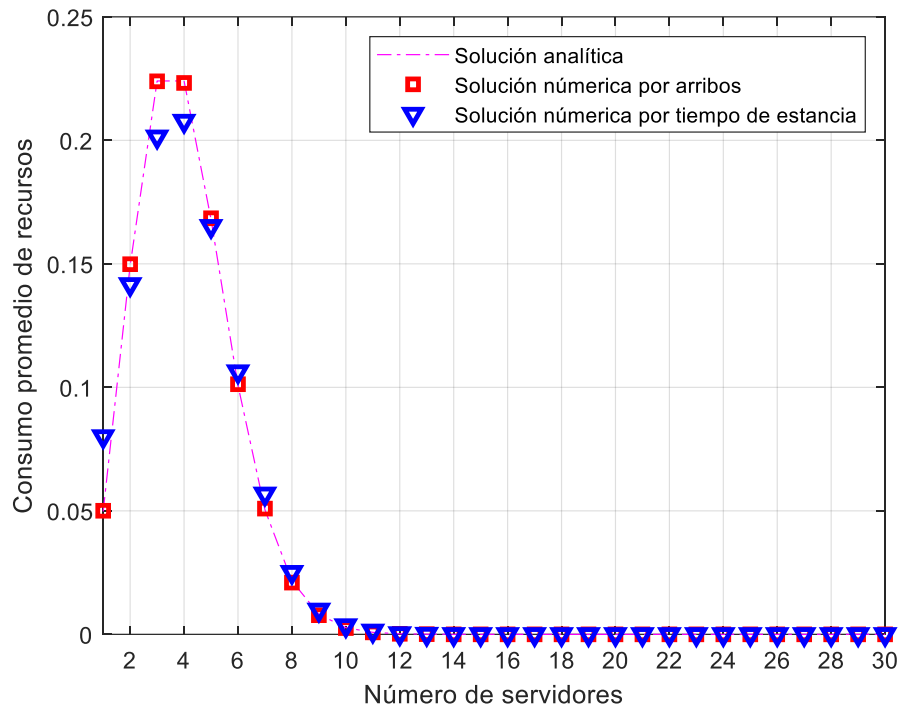


Figura 13. Gráfica de solución por implementación de la cadena de Markov para  $S=30$ ,  $\lambda = 6$  y  $\mu = 0.5$

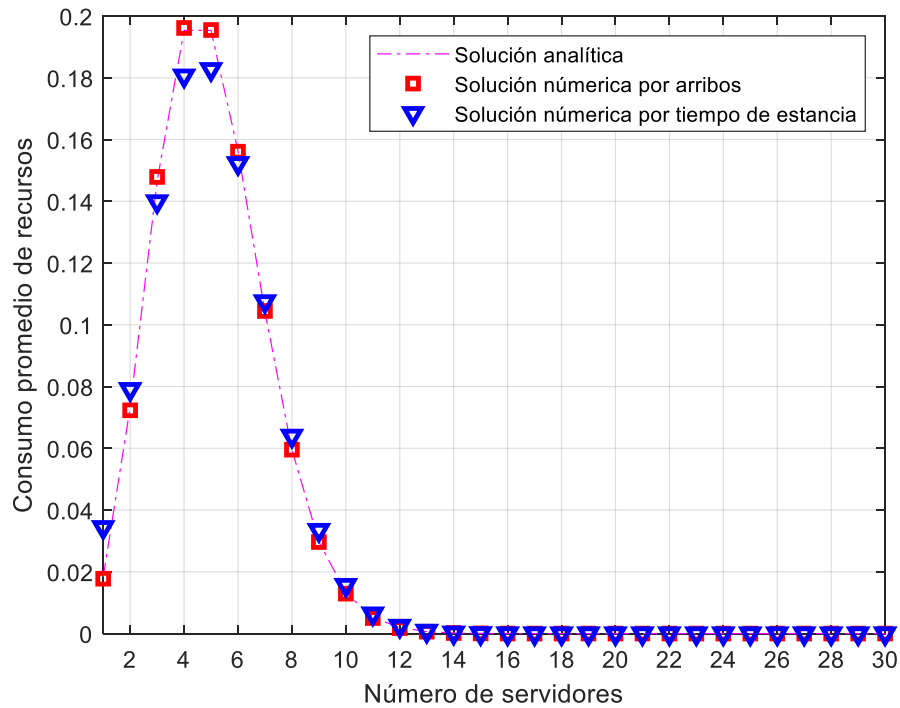


Figura 14. Gráfica de solución por implementación de la cadena de Markov para  $S=30$ ,  $\lambda = 8$  y  $\mu = 0.5$

## 4.2 Análisis de modelos para servicios de video bajo demanda

En esta sección se aborda el análisis de las características que tiene la estructura de un archivo de video bajo demanda y el modelado del comportamiento que muestran los usuarios una vez que se encuentran dentro del sistema consumiendo este servicio. Este análisis es parte medular de la investigación desarrollada en este proyecto debido a que permite comparar el video bajo demanda frente al video en vivo y así establecer la estructura del video en vivo, ya que se trata de servicios diferentes.

Por otro lado, este análisis permite conocer los sucesos que tienen ocurrencia en un sistema de video bajo demanda, y por lo tanto, tomando en cuenta que la estructura de un video en vivo es diferente se pueden establecer los sucesos que tienen ocurrencia a partir del comportamiento de los usuarios en un sistema de servicio de video en vivo.

### 4.2.1 Características de los VoD

El video bajo de manda *VoD* (*Video on Demand*) tiene dos vertientes. La primera, que es un archivo multimedia que tiene como base un video en vivo, es decir, si la captura de un evento en tiempo real es almacenado una vez finalizado y visualizado por los usuarios posteriormente se cataloga como video bajo demanda. La segunda es que un *VoD* es un archivo multimedia diseñado, capturado, editado y almacenado antes de que los usuarios puedan visualizarlo.

Este archivo es previamente almacenado dentro de un servidor de tal manera que los usuarios puedan acceder a través de internet a este en el momento que lo requieran como se menciona en la sección 2.2 del Capítulo 2. En los servicios de *VoD* el usuario tiene la capacidad de elegir el video de su preferencia. En comparación con los servicios de video en vivo, el video bajo demanda puede ser manipulado por el usuario, es decir, el usuario puede reproducir, adelantar, retrasar o pausar el video cuando el usuario así lo requiera, estas acciones no las puede realizar un usuario cuando consume un servicio de video en vivo.

La naturaleza de un archivo de video implica que éste sea producido en pequeños segmentos o conjuntos de *frames* de corta duración llamados *chunks*. Debido a que en el caso de *VoD*, la duración total del video es finita y por lo tanto se conoce el número de *chunks* que componen al video, en trabajos como [13], se opta por agrupar una cantidad finita de *chunks* ( $n$ ) en un segmento de video llamado ventana.

Las  $N$  ventanas que componen al video son de igual tamaño. Es decir, un archivo de video bajo demanda, está compuesto por  $N$  ventanas y éstas a su vez están



compuestas por  $n$  *chunks*, que siguen el flujo de la estructura del VoD como se observa en la Figura 15.



Figura 15. Estructura de un archivo de video bajo demanda-VoD

La agrupación de *chunks* en ventanas se realiza con la finalidad de clasificar a los *peers* dentro del sistema de acuerdo con el número de ventana que están descargando. Por otro lado, esta agrupación permite distribuir de forma eficiente el contenido entre todos los *peers* que solicitan el VoD dentro de un sistema P2P.

En la evaluación de modelos basados en cadenas de Markov para servicios de VoD se consideran características del servicio como la tasa de codificación, la popularidad del video, los atributos de la red, la capacidad de tasa de carga de datos de los servidores y *peers*. Al analizar el modelo propuesto en [5] se tiene la siguiente clasificación de usuarios que consumen un servicio VoD:

- *Peers*: son aquellos usuarios interesados en conectarse a un sistema para descargar/visualizar el archivo de video que se distribuye en este.
- *Seeds*: son *peers* situados en la ventana  $N$  y poseen el archivo de video en su totalidad almacenado en su *buffer*, por lo tanto, pueden compartir el archivo de video con *peers* en ventanas  $i$ ,  $0 \leq i \leq N - 1$
- *Downloaders*: son *peers* situados en la ventana  $i$ ,  $0 \leq i \leq N - 1$  que tienen almacenado en su *buffer* únicamente una fracción del archivo de video y por ende es la porción de video que pueden compartir con *peers* en ventanas  $j$ ,  $j < i$ .

De manera general tanto *seeds* como *downloaders* son *peers* conectados al sistema. Un *seed* es un *peer* que ha finalizado el proceso de descarga del archivo, pero permanece en el sistema para compartir sus recursos con otros *peers*. En la Figura 16 se observa la distribución y posición de estos usuarios dentro de la red.

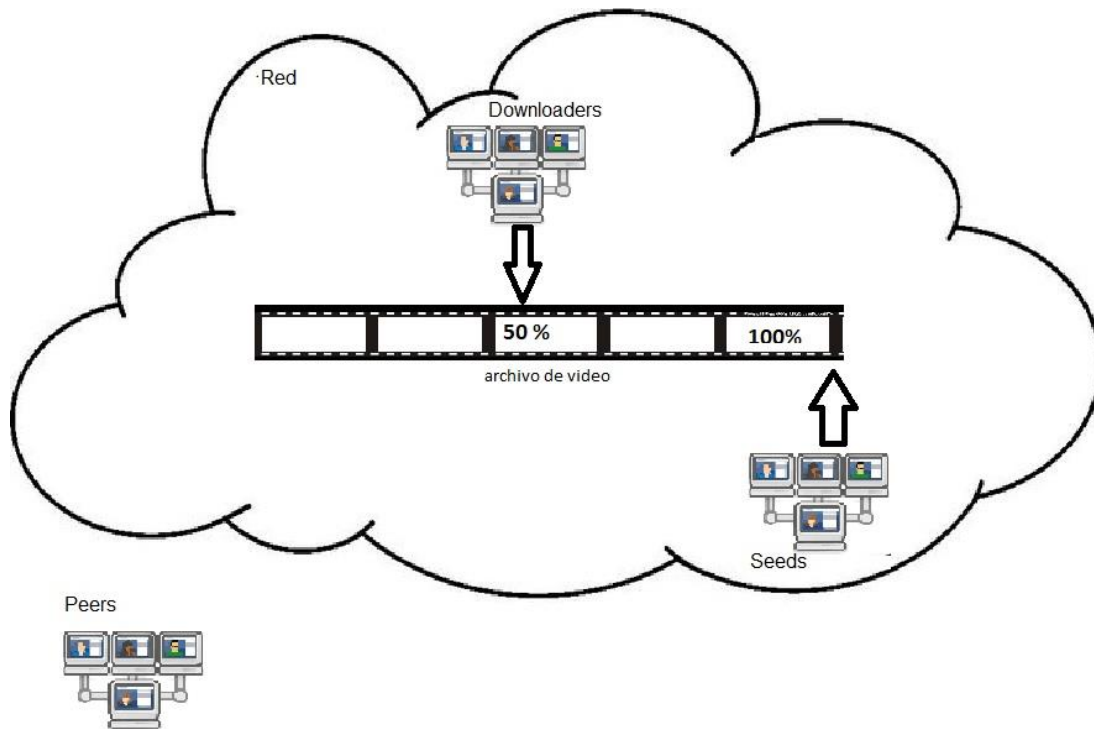


Figura 16. Clasificación de usuarios dentro de una red P2P

Dentro de un sistema híbrido *P2P-CDN* los recursos son proporcionados por los *peers* conectados al sistema y por los servidores *CDN*. Por lo tanto, los *peers* que consumen recursos para descargar el contenido en este tipo de sistemas también pueden atender a otros *peers* que necesiten recursos para continuar con la descarga del video. Un *peer* tiene la capacidad de compartir sus recursos con *peers* que presentan un menor progreso en la descarga del video, es decir, un *peer* que se encuentra descargando la ventana  $i$ , tiene almacenadas en su *buffer* las ventanas  $[0, i - 1]$ .

Lo anterior indica entonces, que este *peer* puede proporcionar estos recursos a *peers* que deseen descargar las ventanas  $[0, i - 1]$ . Este *peer* no puede atender a *peers* descargando ventanas superiores a la ventana  $i$  puesto que aún no cuenta con estos recursos. Esta forma de asignar recursos se conoce como distribución uniforme y no se asignan únicamente los recursos de un *peer* en particular sino que se asignan los recursos proporcionados por la población de *peers* que se encuentran descargando una ventana en específico como se observa en la Figura 17.

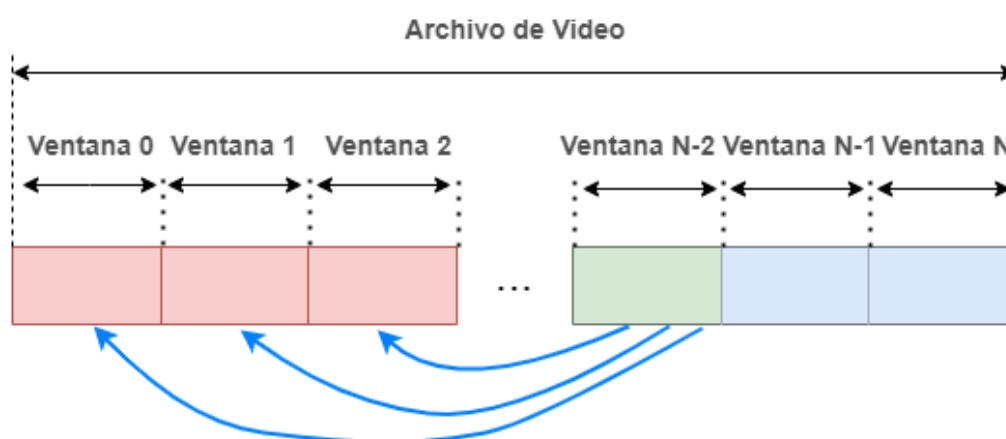


Figura 17. Distribución de recursos en una red P2P

En la Figura 17 se puede observar cómo es que las poblaciones de *peers* de las ventanas  $[0, N - 3]$  son atendidas por la población en la ventana  $N - 2$ . De forma análoga estas poblaciones también reciben recursos proporcionados por las poblaciones  $N - 1$  y  $N$ .

#### 4.2.2 Condiciones de Penuria y Abundancia para el Esquema de Asignación de Recursos Uniforme

En el Esquema de Asignación de Recursos con Distribución Uniforme los recursos que proveen los pares en la población ( $X_i$ ) de una ventana  $i$  son asignados de manera uniforme entre todas las poblaciones ( $X_j$ ) de *peers* situadas en ventanas  $j, j < i$  como se muestra en la Figura 17. La cantidad de recursos con la que una población es atendida va en función del tamaño de dicha población.

Sin importar el tipo de servicio que se distribuya dentro de un sistema ni el esquema de asignación de recursos que se emplee existen dos condiciones en las cuales se puede encontrar el sistema. Estas condiciones son la condición de abundancia y la condición de penuria.

En el caso del Esquema de Asignación de Recursos Uniforme, si la cantidad de recursos de subida con los cuales puede atender la población  $X_i$  a poblaciones  $X_j$  con  $j < i$  son mayores a los recursos de descarga demandados por las poblaciones  $X_j$  se entiende que el sistema opera en condición de abundancia.

En el caso contrario, cuando los recursos de subida de la población  $X_i$  son menores a los recursos de descarga demandados por las poblaciones  $X_j$  con  $j < i$ , se dice que el sistema se encuentra en condición de penuria.

En el Esquema de Asignación de recursos Uniforme la tasa de subida proporcionada por los servidores *CDN* es distribuido uniformemente entre todas las poblaciones  $X_i, 0 \leq i \leq N$ .

Los recursos con los cuales es atendida la demanda de descarga de una población  $X_i$  es el conjunto (denotado por  $\tau_i$ ) de la proporción que le puede proveer las poblaciones  $X_k, k > i$  y la proporción que le pueden proveer los servidores *CDN*. Como se había mencionado anteriormente, si la tasa de descarga demandada por una población  $X_i$  es menor a  $\tau_i$  se dice que el sistema está en abundancia. En caso contrario, cuando  $\tau_i$  es menor que la tasa de descarga demandada por una población  $X_i$  el sistema se encuentra en condición de penuria.

#### 4.2.3 Cadena de Markov para modelar servicios de VoD

En la Figura 18 se observa la cadena de Markov asociada a un sistema de servicio de video bajo demanda (*VoD*). Mediante esa cadena de Markov se representa el comportamiento que tienen los usuarios una vez que se encuentran conectados al sistema y consumiendo el contenido. De manera general, en los servicios de video bajo demanda se identifican tres principales sucesos que pueden ocasionar una transición en el estado de la cadena de Markov.

El estado de la cadena es representado con un vector  $(\{X_0, X_1, X_2, \dots, X_{N-2}, X_{N-1}, X_N\})$  de longitud igual al número de ventanas ( $N$ ) en que se divide el video como se muestra en el ovalo azul de la Figura 18. Este vector contiene las poblaciones  $X_i, 0 \leq i \leq N$  que están descargando cada una de las ventanas que componen al archivo de video.

La transición en el estado de la cadena se puede generar una vez que ocurra alguno de los tres sucesos para *VoD* (arribo de un usuario al sistema, el abandono del sistema por parte de un usuario o la transferencia de un usuario a la ventana superior inmediata).

- El arribo de un usuario al sistema es medido a través de la tasa  $\lambda$ , la cual indica la velocidad promedio a la cual nuevos usuarios arriban al sistema. El arribo de un nuevo usuario es generado en la ventana 0, debido a que de manera natural un usuario comienza a visualizar el video desde el inicio.
- El abandono de un usuario al sistema tiene dos vertientes: un usuario que se encuentra descargando una ventana en el intervalo  $[0, N - 1]$  abandona el sistema a tasa  $\theta$ , que indica la velocidad promedio a la cual un usuario abandona el sistema, este abandono se puede generar por diversas circunstancias (fallas en la conexión a Internet, fallas en sus dispositivos, pérdida de interés en el contenido, etc.).

La otra vertiente es cuando el usuario pertenece a la ventana  $N$ , estos usuarios abandonarán el sistema a una tasa promedio  $\gamma$ . Lo anterior, se debe a que estos usuarios han completado la descarga del video en su totalidad y permanecen en el sistema para atender a otros usuarios que necesiten recursos para continuar su descarga. De igual forma, estos usuarios pueden permanecer en el sistema para terminar de visualizar el video y por lo tanto son fuente de recursos.

- La transferencia de un usuario a la ventana superior inmediata se genera cuando un usuario ha terminado de descargar la ventana  $i$  y desea comenzar a descargar la ventana superior inmediata superior  $(i + 1)$ . Esta transferencia ocurre a tasa  $\tau_i$ , que indica la velocidad promedio a la cual un usuario es transferido a la ventana superior inmediata.

La transferencia de un usuario no existe para la población en la ventana  $N$ , debido a que esta ventana es la última que componen al archivo de video y por lo tanto no existe otra ventana que descargar.

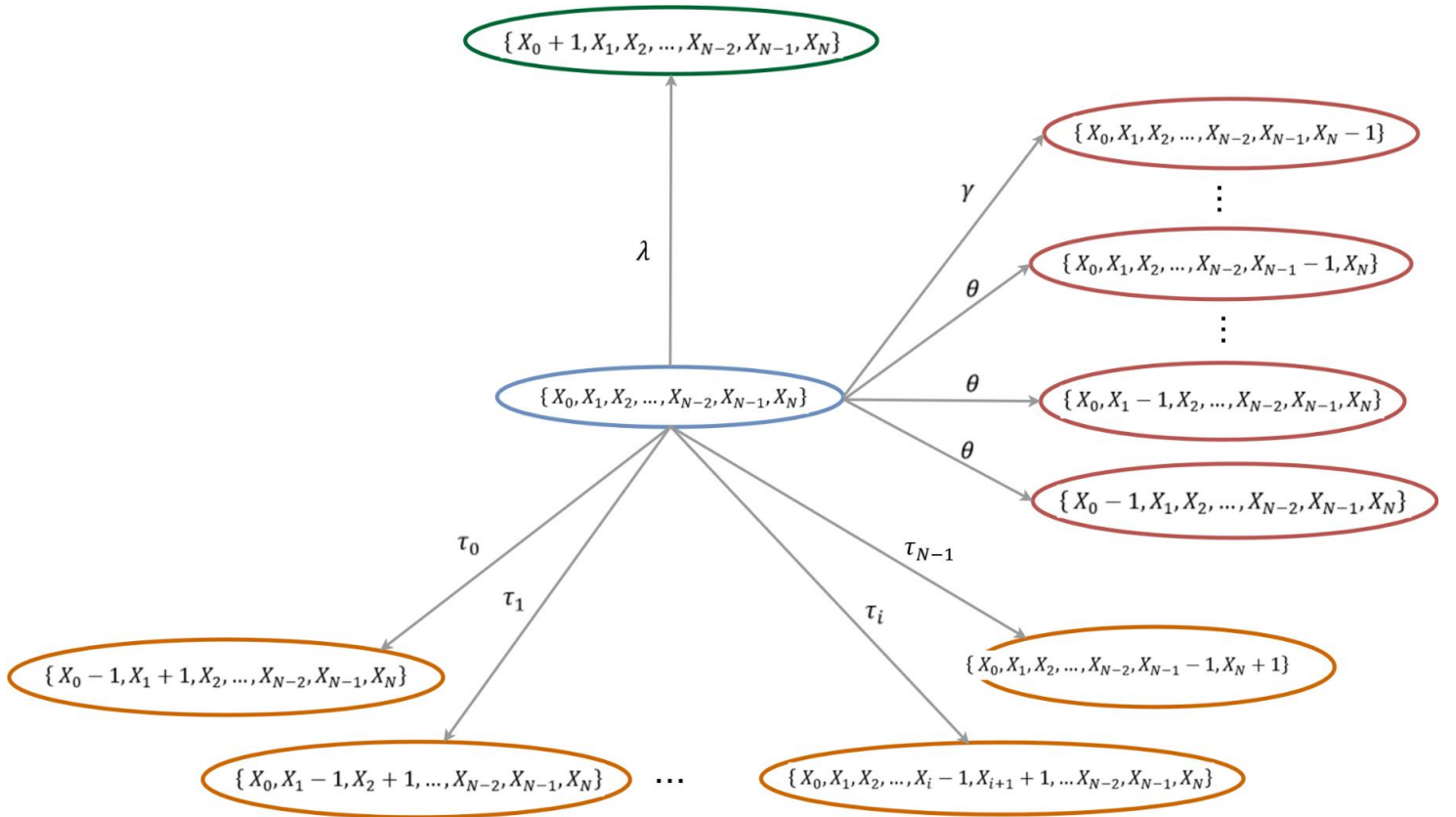


Figura 18. Cadena de Markov para un sistema de consumo de servicio de video bajo demanda (VoD)

### 4.3 Análisis esquemas de asignación de recursos para servicios de video bajo demanda

En esta sección se realizó el análisis de los esquemas de asignación de recursos reportados en los trabajos de investigación relacionados a este proyecto. En primer lugar, se describe a cada uno ellos con la finalidad de conocer sus principales características y funcionamiento para poder seleccionar algunos de los esquemas descritos, que pueden adecuarse para los servicios de video en vivo, y que serán tomados como base para realizar el bosquejo del esquema de asignación de recursos a proponer en el presente proyecto.

En el siguiente listado se reportan los esquemas de asignación de recursos para servicios de video bajo demanda (VoD) que presentan mayor relación con el presente proyecto.

- Esquema de Distribución Uniforme (DU): este esquema consiste en distribuir los recursos disponibles de la población de usuarios en ventana  $i$  de manera uniforme entre todas las poblaciones de usuarios que han arribado al sistema, es decir, los recursos son asignados de igual manera entre las diversas poblaciones en ventanas  $j, j < i$  que están consumiendo el archivo de video.

En este esquema el parámetro para asignar los recursos ofrecidos por *seeds*, *downloaders* y servidores *CDN* es que los usuarios pertenecientes a las ventanas 0 a  $N - 1$  son atendidos por usuarios que presentan un mayor progreso en su descarga.

- Esquema Q ventanas hacia atrás: este esquema al igual que el esquema DU consiste en atender con los recursos disponibles de la población de una ventana  $i$  de manera uniforme entre todas las poblaciones de usuarios posicionados en ventanas  $j, j < i$ .

La principal diferencia entre Q ventanas hacia atrás y DU es la definición de un parámetro de control llamado Q. Dicho parámetro limita las ventanas inferiores a las cuales se atenderá, ya que Q representa el número de ventanas inferiores que tendrán acceso a los recursos de una ventana.

El objetivo principal de este parámetro es atender con mayor prioridad a las ventanas inmediatas inferiores y generar de esta forma una asignación de recursos más equitativa.

En este esquema al igual que en DU los *downloaders* son atendidos por los servidores *CDN*, *seeds* y con recursos de otros *downloaders*.

- Esquema Distribución por Priorización de Ventanas (DPV): este esquema consiste en asignar los recursos provenientes de *seeds* y servidores *CDN* con mayor prioridad a *downloaders* que se encuentran descargando ventanas superiores (cercanas a la ventana  $N$ ).

En este esquema se define un parámetro  $\varepsilon$  para establecer la prioridad con la cuál serán asignados los recursos de la población en la ventana  $N$  y los recursos *CDN* a las poblaciones en ventanas inferiores a la ventana  $N$ . Esto con la finalidad de que los recursos se distribuyan de una manera más equitativa y el sistema en general se encuentre en condición de abundancia.

- Esquema de Generalización de Distribución por Priorización de Ventanas (GDPV): este esquema es una versión generalizada del esquema DPV y al igual que DU, consiste en asignar los recursos disponibles para atender de la población en una ventana  $i$  entre los *downloaders* posicionados en ventanas inferiores a la ventana  $i$ .

A diferencia de DU, en este esquema se establece un parámetro de prioridad denotado por  $\varepsilon$ . Este parámetro define el grado de prioridad con el que se atenderá a *downloaders* situados en ventanas inferiores a la ventana de donde se obtienen los recursos. Es decir,  $\varepsilon$  define una alta prioridad a ventanas  $i$  con mayor cercanía a la ventana  $N$  y provoca que la población de una ventana  $i$  atienda con mayor prioridad a ventanas inmediatas inferiores para propiciar una mayor colaboración entre todos los *downloaders* conectados al sistema.

La principal diferencia que existe entre DPV y GDPV, es que en DPV únicamente los *seeds* y servidores *CDN* pueden proporcionar recursos a ventanas inferiores mientras que en GDPV se considera también a los recursos provenientes de los *downloaders*.

- Esquema *Immediate-Neighbors Uniform Allocation (INUA)*: este esquema establece que un *downloader* puede seleccionar aleatoriamente los recursos de *peers* con mayor progreso en su descarga, es decir, un *downloader* puede seleccionar a los *peers* en ventanas superiores que le proporcionarán recursos.

Sin embargo, la selección se debe realizar bajo la condición  $M = QX$ . Esta condición limita el número de *peers* en ventanas hacia adelante que pueden atender a *downloaders* en ventanas inferiores. En dicha condición  $M$  representa los *peers* seleccionados para proporcionar recursos,  $Q$  el número de ventanas hacia adelante y  $X$  la cantidad de *peers* en las ventanas superiores.

La Tabla 3 muestra una comparación entre los esquemas de asignación de recursos analizados previamente.

<b>Esquema</b>	<b>Parámetros</b>	<b>Ventajas</b>	<b>Desventajas</b>
<b>Distribución uniforme (DU)</b>		Los recursos se distribuyen de forma homogénea en proporción al tamaño de la población en una ventana.	Todas las ventanas desde 0 a $j-1$ se consideran inferiores y por lo tanto se deben atender con la misma prioridad.
<b>Q ventanas hacia atrás</b>	Q número de ventanas inferiores	Los recursos se distribuyen de forma homogénea de acuerdo con el tamaño de la población en la ventana y el limitando el número de ventanas inferiores.	Al ser Q un parámetro estático se debe definir bien los límites de los bordes del video (ventanas 0 y N).  Existe la probabilidad de caer en condiciones de penuria si dentro del rango de ventanas no hay disponibles suficientes recursos.
<b>DPV</b>	$\varepsilon$ grado de prioridad	Organización por orden de prioridad a las ventanas	Únicamente los <i>seeds</i> y servidores <i>CDN</i> pueden compartir recursos.
<b>GDPV</b>	$\varepsilon$ grado de prioridad	Organiza las ventanas en orden de prioridades.  Atiende primero a las ventanas con la prioridad más alta.	Dificultad de implementación por la función de prioridad que se utiliza.  Existe posibilidad de generar condición de penuria.
<b>INUA</b>	$M = QX$	Un usuario puede seleccionar quien	No hay homogeneidad al momento de



		le proporcione recursos.	compartir los recursos.
--	--	--------------------------	-------------------------

*Tabla 3. Características principales de esquemas de asignación de recursos investigados*

Como resultado de analizar cada uno de los esquemas de asignación de recursos reportados en la Tabla 3 se decidió tomar como referencia a los esquemas:

- Esquema de asignación de recursos de distribución uniforme (DU)
- Esquema Q ventanas hacia atrás
- Esquema GDPV

Esto gracias a que, a pesar de que son esquemas para servicios de video bajo demanda (VoD), tienen características similares que se pueden adaptar y mejorar para servicios de video en vivo.

Con base en el análisis previo de los esquemas de asignación de recursos seleccionados se realizó un pseudocódigo que permite comprender el funcionamiento de cada uno de estos.

De forma general se identificó que todos los esquemas presentan el mismo comportamiento una vez que se ha determinado cuales son los *peers* de donde se obtendrán los recursos que utiliza un *downloader* para descargar el archivo de video y por lo tanto son similares a partir de esta condición.

## Esquema de Asignación de Recursos Uniforme

### Inicio

- 1.- Llega un usuario nuevo al sistema y se posiciona en la población que está descargando la ventana  $i$
- 2.- Se define  $k = i + 1$
- 3.- **Repetición de Recolección**  
Se recolectan todos los *seeds* y *downloaders* de las poblaciones que se encuentren con las ventanas  $i + 1 \leq k \leq N$   
**Fin de Repetición**
- 4.- Listado de *peers* con recursos disponibles dentro de la población correspondiente
- 5.- ¿La tasa real de descarga es igual a la ideal de descarga?  
**Si:** Almacenar los *peers* que asignan recursos y la cantidad de recursos  
**No:** Se asignan recursos, ¿aún hay *peers* en la lista?  
**Si:** Retornar a paso 4

- No:** Buscar recursos en el *CDN* y repetir condición **Si** del paso 5
- 6.- Generar estadísticas

Fin

### Esquema Q ventanas hacia atrás

Inicio

- 1.- Llega una población nueva de usuarios al sistema en la ventana  $i$
- 2.- Se define  $Q$  que es el límite de ventanas hacia atrás
- 3.- **Repetición de Recolección**  
Se recolectan todos los *peers* de las poblaciones en ventanas  $i + 1 \leq k \leq i + Q$   
**Fin de Repetición**
- 4.- Listado las poblaciones con recursos disponibles
- 5.- ¿La tasa real de descarga es igual a la ideal de descarga?

**Si:** Almacenar las poblaciones de *peers* que asignan recursos y la cantidad de recursos

**No:** Se asignan recursos, ¿aún hay poblaciones nuevas en la lista?

**Si:** Retornar a paso 4

- No:** Buscar recursos en el *CDN* y repetir condición **Si** del paso 5
- 6.- Generar estadísticas

Fin

### Esquema DVPG

Inicio

- 1.- Llega un usuario nuevo al sistema a la ventana  $i$
- 2.- Se define  $\varepsilon$  que es el parámetro de control
- 3.- **Repetición de Recolección**  
Se recolectan todos *peers* que presenten con un índice  $(j + 1)^\varepsilon$  alto  
**Fin de Repetición**
- 4.- Listado de *peers* con recursos disponibles
- 5.- ¿La tasa real de descarga es igual a la ideal de descarga?

**Si:** Almacenar los *peers* que asignan recursos y la cantidad de recursos

**No:** Se asignan recursos, ¿aún hay *peers* en la lista?

**Si:** Retornar a paso 4

- No:** Buscar recursos en el *CDN* y repetir condición **Si** del paso 5
- 6.- Generar estadísticas

Fin

Aunado a esto se analizaron las expresiones para asignar recursos de los esquemas Q ventanas hacia atrás y DVPG para encontrar las diferencias para servicios de video bajo demanda y servicios de video en vivo.

### **Esquema Q ventanas hacia atrás**

La siguiente expresión determina la tasa promedio necesaria para descargar una ventana del video:

$$c_{\omega} \leq \mu_{\omega} M \left( \sum_{j=j+1}^{\min(j+q, N-1)} \frac{X_{JM}}{\sum_{k=\max(0, j-q)}^{j-1} x_{kM} + x_{kF}} + \frac{Y_M}{\sum_{k=N-q-1}^{N-1} x_{kM} + x_{kF}} \right) + \mu_{\omega} F \left( \sum_{j=j+1}^{\min(j+q, N-1)} \frac{X_{JF}}{\sum_{k=\max(0, j-q)}^{j-1} x_{kM} + x_{kF}} + \frac{Y_F}{\sum_{k=N-q-1}^{N-1} x_{kM} + x_{kF}} \right) + \frac{v}{X}$$

En primer lugar, el esquema de asignación de recursos a proponer en el presente proyecto no considera 2 regiones o zonas donde están distribuidos los *peers* dentro del sistema, por lo tanto, no se tomará a *M* y *F* como variables diferentes.

Debido a que no existen usuarios del tipo *seed* el esquema prescindirá del término *Y*, que representa los recursos provenientes de *peers* conectados a la ventana *c*.

El término  $\frac{v}{X}$  para el desarrollo de este proyecto va a representar los recursos que proporciona la red *CDN*.

Finalmente, considerando que el número de ventanas queda restringido por el parámetro *c*, los límites en la suma de los recursos cambiarán de forma significativa y el parámetro Q (número de ventanas hacia atrás) deberá ser  $0 < Q \leq c$ .

### **Esquema GDPV**

La siguiente expresión determina la tasa promedio necesaria para descargar una ventana del video:

$$c_{\omega} \leq (j+1)^{\varepsilon} \left( \sum_{j=j+1}^N \frac{\mu_{\omega} X_j}{V_{j-1}^*} + \frac{v}{V_{N-1}^*} \right), \text{ para } 0 \leq j \leq N-1$$

La expresión anterior se puede adecuar de manera correcta a los servicios de video en vivo debido a que considera tanto recursos de la red *P2P* como de la red *CDN*, además de que considera a todos los *peers* dentro del sistema para que reciban recursos de manera equitativa o correspondiente a su progreso de descarga.

Sin embargo, se debe limitar la expresión anterior al caso de los videos en vivo. Debido a que en una transmisión de video en directo no existen usuarios del tipo

*seed* los recursos dentro la red provienen exclusivamente de *peers* conectados a ventanas  $i, 0 \leq i \leq c$  y de los servidores *CDN*.

Por lo mencionado anteriormente el límite de la suma no llegará hasta  $N$  sino que se cambiará esta variable por una  $c$ , que identifica a la ventana actual (*currently*). A su vez en el presente proyecto se busca que los usuarios más cercanos a la ventana  $c$  sean los que reciban con mayor prioridad los recursos provenientes de los servidores *CDN*. Por lo tanto, el término  $\frac{v}{V_{N-1}^*}$ , seguramente será restringido a poblaciones cercanas a la ventana  $c$ .

El valor de la variable  $j$  siempre debe corresponder al índice de la ventana a donde pertenece la población que va a recibir los recursos y por lo tanto se debe limitar a  $0 \leq j \leq c - 1$ .

Finalmente, el parámetro de priorización  $\varepsilon$  para el caso de este proyecto puede ser cualquier número real e indicará la prioridad con que una población de *peers* puede acceder a los recursos ofrecidos por las poblaciones en ventanas superiores.

Como se había mencionado anteriormente el esquema de asignación de recursos que se propondrá en este proyecto puede ser la fusión de los 2 esquemas mencionados anteriormente, con los cual quizá vuelvan a cambiar los límites de la suma.

# Capítulo 5

## Diseño

En este capítulo se describe el diseño y estructura de la cadena de Markov que representa a los servicios de video en vivo, así como el diseño del esquema de asignación de recursos que se ha propuesto para este tipo de servicios.

### 5.1 Diseño de la cadena de Markov para servicios de video en vivo

En esta sección se describe la cadena de Markov que representa el comportamiento de los usuarios una vez que se conectan a un sistema de servicio de video en vivo, de igual forma se explican los posibles cambios que se pueden generar en el estado de la cadena de Markov dado un suceso.

En este proyecto se supone la distribución de un archivo de video en vivo sobre una red híbrida *P2P-CDN*. Como se ha mencionado anteriormente, los videos son generados por segmentos pequeños llamados *frames*, en este ámbito los denominaremos *chunks*, que es la unidad indivisible de un video. Cabe mencionar, que en el presente trabajo se modela únicamente el proceso de descarga del archivo de video que realizan los *peers* dentro del sistema.

Los *peers* conectados al sistema descargan el archivo de video *chunk* a *chunk*, sin embargo, con el objeto de no tener una gran cantidad de poblaciones de *peers* descargando el archivo de video, se agrupan *n chunks* en segmentos de video más grandes llamados ventanas. El tamaño de las ventanas influye en la manera en que se distribuyen los recursos entre *peers*, pues al haber ventanas más grandes, la distribución se realiza entre un menor número de grupos de *peers* (poblaciones). Se espera que cada uno de estos grupos sea de un tamaño considerable para una distribución eficiente. Lo anterior se representa en la Figura 19.

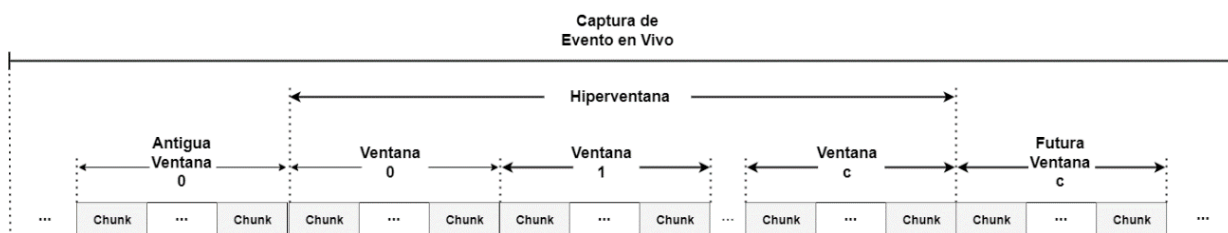


Figura 19. Estructura de una hiperventana

En el caso de video bajo demanda (VoD) se conoce la duración total del archivo de video y por lo tanto se divide en  $N$  ventanas de igual tamaño ( $n$  *chunks*) al momento de distribuirlo entre los usuarios. Sin embargo, para el caso de video en vivo no se conoce la duración total del archivo de video puesto que no se sabe con exactitud el tiempo que durará la captura del evento en tiempo real.

Considerando lo anterior, se optó por definir un contenedor llamado hiperventana, este contenedor es de longitud  $c$ , es decir, la hiperventana contiene de la ventana 0 a la ventana  $c$  y almacena las ventanas en tiempo real. La ventana 0, es la última ventana considerada para que el usuario visualice el evento en tiempo real, por otro lado, la ventana  $c$  representa el fragmento de video correspondiente al evento capturado en tiempo real.

La hiperventana siempre es de longitud  $C$ , sin embargo, no es un archivo estático, sino que su contenido cambia de manera simultánea a la producción del archivo de video. Es decir, cuando se captura una nueva ventana de video, la última ventana contenida en la hiperventana (ventana 0) sale de este contenedor, las demás ventanas se recorren y son reenumeradas para satisfacer la condición de que la hiperventana contiene las ventanas 0 a  $c$ .

En la Figura 19 se puede observar la estructura de la hiperventana compuesta por  $C$  ventanas. Las ventanas son identificadas con un subíndice  $i$ , para  $0 \leq i \leq c$ . A su vez dichas ventanas están conformadas por  $n$  *chunks*.

$c$ : Representa el índice de la ventana de video que se produce de forma simultánea a la captura del evento en tiempo real.

0: Representa el índice de la ventana con el máximo retardo considerado respecto a la ventana  $c$ , es decir, es la última ventana que aún se considera como visualización en tiempo real.

Como se mencionó anteriormente, los *peers* conectados al sistema de transmisión de video en vivo, son agrupado en poblaciones y dichas poblaciones se clasifican de acuerdo con la ventana que se encuentran descargando. En el modelo desarrollado para este proyecto se representa el comportamiento (variación en el tamaño) de cada una de las poblaciones en las ventanas del video en vez de representar el comportamiento individual de cada uno de los *peers*, con la finalidad de simplificar el modelo. La clasificación mencionada es la siguiente:

$X_i$ : Representa la población de *peers* que se encuentra descargando y la ventana  $i$ . Para  $i \in [0, 1, 2, 3, \dots, c - 1]$ .

$X_c$ : Representa la población de *peers* que se encuentra en espera de la producción de una nueva ventana de video.

Con base en el análisis y revisión de los modelos para el consumo de servicio de VoD reportados en la literatura y al análisis de la cadena unidimensional (Erlang-B), se pudo obtener el comportamiento que tienen los usuarios en la visualización de un video y así identificar los sucesos que pueden ocurrir en la transmisión del video en vivo para comprender como estos influyen en la redimensión de las poblaciones de cada una de las  $N$  ventanas que componen al archivo de video.

Los principales sucesos que se identificaron a lo largo de la descarga y reproducción de un video bajo demanda son:

- **Conexión de un *peer*:** Sucede cuando un *peer* se conecta al sistema para iniciar el proceso de descarga/reproducción del video. En este caso, el *peer* puede iniciar este proceso en cualquiera ventana del archivo de video. El *peer* deberá recibir los recursos de acuerdo con el esquema de asignación de para poder descargar el video y posteriormente él poder compartir recursos con *peers* que inicien la descarga del video tiempo después o que se sitúan en ventanas inferiores. De acuerdo con el esquema de asignación de recursos el *peer* no recibirá necesariamente el 100% de los recursos que necesita directamente de un servidor *CDN*, sino que puede recibir algún porcentaje de este y otro tanto de algunos *peers* situados en ventanas superiores. Cuando sucede una conexión a la ventana  $i$  su población se incrementa en uno.
- **Arribo de un *peer*:** Sucede cuando un *peer* se conecta al sistema para comenzar la descarga/reproducción del video desde la ventana 0. En este caso el *peer* recibirá recursos de acuerdo con el esquema de asignación de recursos, con la finalidad de recibir porciones de recursos que necesita desde diferentes *peers* habilitados para compartir recursos y desde los servidores *CDN*. En este caso la población de la ventana 0, se incrementa en 1.
- **Transferencia de un *peer* a la ventana superior:** Ocurre cuando un *peer* que estaba descargando la ventana  $i$  comienza a descargar la ventana  $i + 1$ . Por lo tanto, ya no forma parte de la población de la ventana  $i$  ahora es parte de la población que descarga la ventana  $i + 1$ . Por lo tanto, la población  $i$  se reduce en 1 y la población  $i + 1$  se incrementa en 1.
- **Desconexión de un *peer*:** Por diversas razones un *peer* puede interrumpir la visualización de un video (mala conexión a internet, pérdida de interés en el contenido, fallas en su dispositivo, etc.) antes de que termine la descarga/reproducción del contenido en su totalidad. Por lo tanto, si un usuario que se encuentra descargando la ventana  $i$ , se desconecta, se decrementará en 1 la población de esa ventana.

De forma general, cualquier población de una ventana contenida en  $[0, N]$  se altera por la transferencia, conexión o desconexión de un *peer*. Sin embargo, el arribo de un *peer* únicamente ocurre en la ventana 0, que representa a la primera ventana del archivo de video.

La desconexión de un *peer* contempla los casos reportados en la Tabla 4 con el fin de comprender la estructura de la distribución del video.

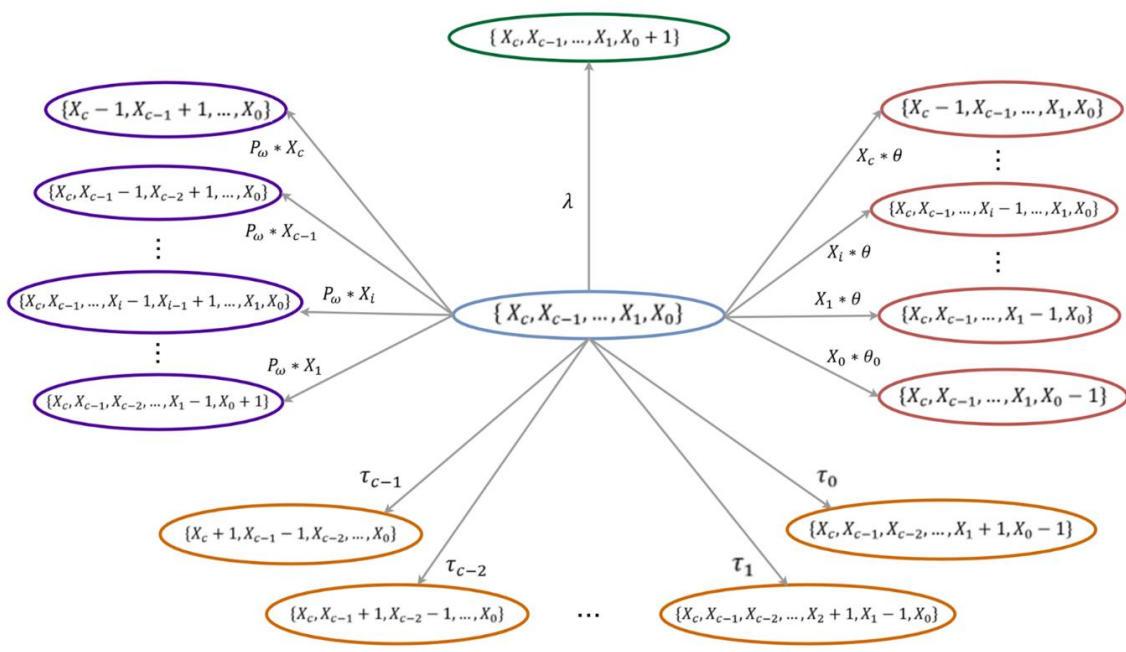
Caso	Descripción
Primero	El <i>peer</i> que se desconectó estaba conectado directamente al servidor <i>CDN</i> y no era punto de acceso para un <i>peer</i> que llegó después de iniciada la transmisión.
Segundo	El <i>peer</i> que abandona la conexión estaba conectado al servidor <i>CDN</i> y a su vez tenía a otro <i>peer</i> conectado a él.
Tercero	El <i>peer</i> que abandona la transmisión estaba conectado a otro <i>peer</i> y no tenía ningún <i>peer</i> conectado a él.
Cuarto	El <i>peer</i> que abandona el sistema estaba conectado a otro <i>peer</i> y a su vez tenía otro <i>peer</i> conectado a él.

*Tabla 4. Casos de desconexión de un peer*

Tomando en cuenta lo descrito anteriormente se realizó el diseño de una cadena de Markov para servicios de video en vivo. Los sucesos que se establecieron para desarrollar el modelo son la conexión, desconexión, transferencia a la ventana superior inmediata y transferencia a la ventana inferior inmediata. Dichos sucesos son los que generan un cambio o transición en el estado de la cadena de Markov (modelo) que representa un sistema de transmisión de video en vivo. Más adelante se explican a detalle los sucesos.

El sistema de transmisión de contenido en tiempo real se representa con la siguiente cadena de Markov:





*Figura 20. Cadena de Markov de un sistema de transmisión de video en vivo*

En la Figura 20, se observan los sucesos que generan una transición en el estado de la cadena y el cambio en el estado de la cadena.

En el centro de la Figura 20, se observa al vector  $\{X_c, X_{c-1}, \dots, X_1, X_0\}$ , que representa al estado de la cadena de Markov para servicios de video en vivo. Este vector contiene las poblaciones de *peers* en cada una de las ventanas contenidas en la hiperventana de un archivo de video (desde la ventana 0 hasta la ventana  $c$ ).

Este vector ilustra un estado general de la cadena de Markov, es decir, la cadena se encuentra en este estado en cualquier instante de la transmisión de video en vivo. Cada una de las poblaciones  $X_i$ , contenidas en este vector estado, tiene un valor aleatorio.

En el sistema a modelar para el proyecto se considera que los sucesos son discretos, es decir, ocurre un único evento a la vez (conexión de un usuario, transferencia de un *peer* a la ventana superior inmediata, transferencia de un *peer* a la ventana inferior inmediata o desconexión de un *peer* antes de terminar la transmisión).

A continuación, se describen los sucesos que producen un cambio en alguna población de *peers* perteneciente a una ventana del archivo de video contenida en la hiperventana y por ende el estado de la cadena de Markov transita.

Conexión de un nuevo usuario al sistema: Una vez que inicia la transmisión de video en vivo, diversos usuarios se pueden conectar a la transmisión para visualizar la distribución de contenido en tiempo real. Dicha conexión se realiza a tasa  $\lambda$ , que representa la tasa de conexión de un usuario en general del sistema.

En este proyecto se supone que el usuario se debe conectar a la ventana 0 de la hiperventana para comenzar a descargar el video, crear *buffer* y así evitar congelamientos en la descarga del contenido.

Al conectarse un usuario al sistema, el estado general de la cadena de Markov transita del estado  $\{X_c, X_{c-1}, \dots, X_1, X_0\}$  al estado  $\{X_c, X_{c-1}, \dots, X_1, X_0 + 1\}$ . Este evento es plasmado en la parte superior de la Figura 20.

Transferencia de un *peer* a la ventana inmediata superior: Cualquier población de *peers* correspondiente a una ventana  $i$ ,  $0 \leq c$  contenida en la hiperventana se modifica al generarse la transferencia de un *peer* a la ventana superior inmediata, esta transferencia ocurre a tasa  $\tau_i$ , es decir, una vez que un usuario termina de descargar la ventana  $i$  y comienza a descargar la ventana  $i + 1$ , abandona la población  $X_i$  a tasa  $\tau_i$  y se agrega a la población en la ventana  $i + 1$  ( $X_{i+1}$ ).

El suceso de transferencia a la ventana superior inmediata provoca una transición en el estado de la cadena de Markov como se describe a continuación:

$$\{X_c, X_{c-1}, \dots, X_{i+1}, X_i, \dots, X_0\} \rightarrow \{X_c, X_{c-1}, \dots, X_{i+1} + 1, X_i - 1, \dots, X_0\}$$

El suceso descrito anteriormente está plasmado en la parte inferior de la Figura 20.

$\tau_i$  : Representa la tasa promedio de transferencia de la ventana  $i$ . Y se define con la siguiente expresión:

$$\tau_i = \min\{C_\omega X_i, r_i\} \text{ para } i \in [0: c]$$

Donde:

$C_\omega$ : Es la tasa de descarga general de un usuario dentro del sistema.

$r_i$ : Representa a los recursos de descarga efectivos en penuria para la ventana  $i$ , es decir, cuando la tasa de descarga es mayor que la tasa de subida el sistema entra en penuria y debe obtener recursos de la red *CDN*. Esta expresión depende del esquema de asignación de recursos, por ejemplo, el esquema de distribución uniforme que se retomará con más detalle en la siguiente sección.

Transferencia de un *peer* a la ventana inmediata inferior: Cualquier población de *peers* correspondiente a una ventana contenida  $i, 0 \leq i \leq c$ , contenida en la hiperventana, se modifica al generarse la transferencia de un *peer* a la ventana

inferior inmediata a tasa  $P_\omega X_i$ , es decir, cuando un usuario por diversas razones (fallas en los servicios, fallas en sus dispositivos, etc.) deja de descargar el video en vivo a la misma tasa que se está capturando el evento en tiempo real y produciendo una nueva ventana del video, el *peer* se atrasa en la descarga del video y pasa de la ventana  $i$  (abandona la población  $X_i$ ) a la ventana  $i - 1$  (se adiere a la población  $X_{i-1}$ ).  $P_\omega X_i$  representa la tasa promedio de producción de la población que se encuentra descargando la ventana  $i$ . Es el resultado de multiplicar la población de la ventana  $i$  ( $X_i$ ) por  $P_\omega$  (tasa de producción del archivo de video).

El suceso de transferencia a la ventana inferior inmediata provoca una transición en el estado de la cadena de Markov como se describe a continuación:

$$\{X_c, X_{c-1}, \dots, X_i, \dots, X_0\} \rightarrow \{X_c, X_{c-1}, \dots, X_i - 1, X_{i-1} + 1, \dots, X_0\}$$

El suceso descrito anteriormente está plasmado en la parte izquierda de la Figura 20.

Desconexión de un *peer* antes de terminar la transmisión: Por causas diversas (fallas de conexión, fallas en sus servicios, fallas en sus dispositivos, desinterés en el contenido, etc.) un *peer* que está visualizando la transmisión de video en vivo puede desconectarse.

Entonces, cualquier población de *peers* de la cadena cambia cuando se genera la desconexión de un *peer* a tasa  $X_i \theta$ , antes de que finalice la transmisión en vivo.  $X_i \theta$  representa la tasa promedio de desconexión de la población que se encuentra descargando ventana  $i$ . Es el resultado de multiplicar la población de la ventana  $i$  ( $X_i$ ) por  $\theta$  (tasa de desconexión de un *peer* conectado al sistema en general).

Este suceso provoca una transición en el estado de la cadena de Markov como se describe a continuación:

$$\{X_c, X_{c-1}, \dots, X_i, \dots, X_0\} \rightarrow \{X_c, X_{c-1}, \dots, X_i - 1, \dots, X_0\}$$

Lo anterior quiere decir que un *peer* que formaba parte de la población  $X_i$  por alguna razón abandono el sistema antes de finalizar la transmisión en vivo, por lo tanto, la población de la ventana  $i$  se decrementa en 1. Este suceso es representado en la sección derecha de la Figura 20.

Los *peers* conectados a la ventana 0 son más susceptibles a desconectarse del sistema, en caso de interrumpir su proceso de descarga, se generó una nueva ventana del video, por una mala conexión a internet, problemas con el hardware, no tener óptimas *QoE* ó *QoS*. Por lo tanto, se considera a  $\theta_0$  como la tasa de desconexión de los *peers* pertenecientes a la población  $X_0$ . Cabe mencionar que en el presente proyecto se modela únicamente la desconexión de un *peer* y no se modelan los casos reportados en la Tabla 4 para una desconexión.

Una vez diseñado el modelo para servicios de video en vivo y definidos los sucesos que generan un cambio en su estado, se diagramo la solución matemática a implementar para simular la ocurrencia de los sucesos y la transición en el estado en la cadena.

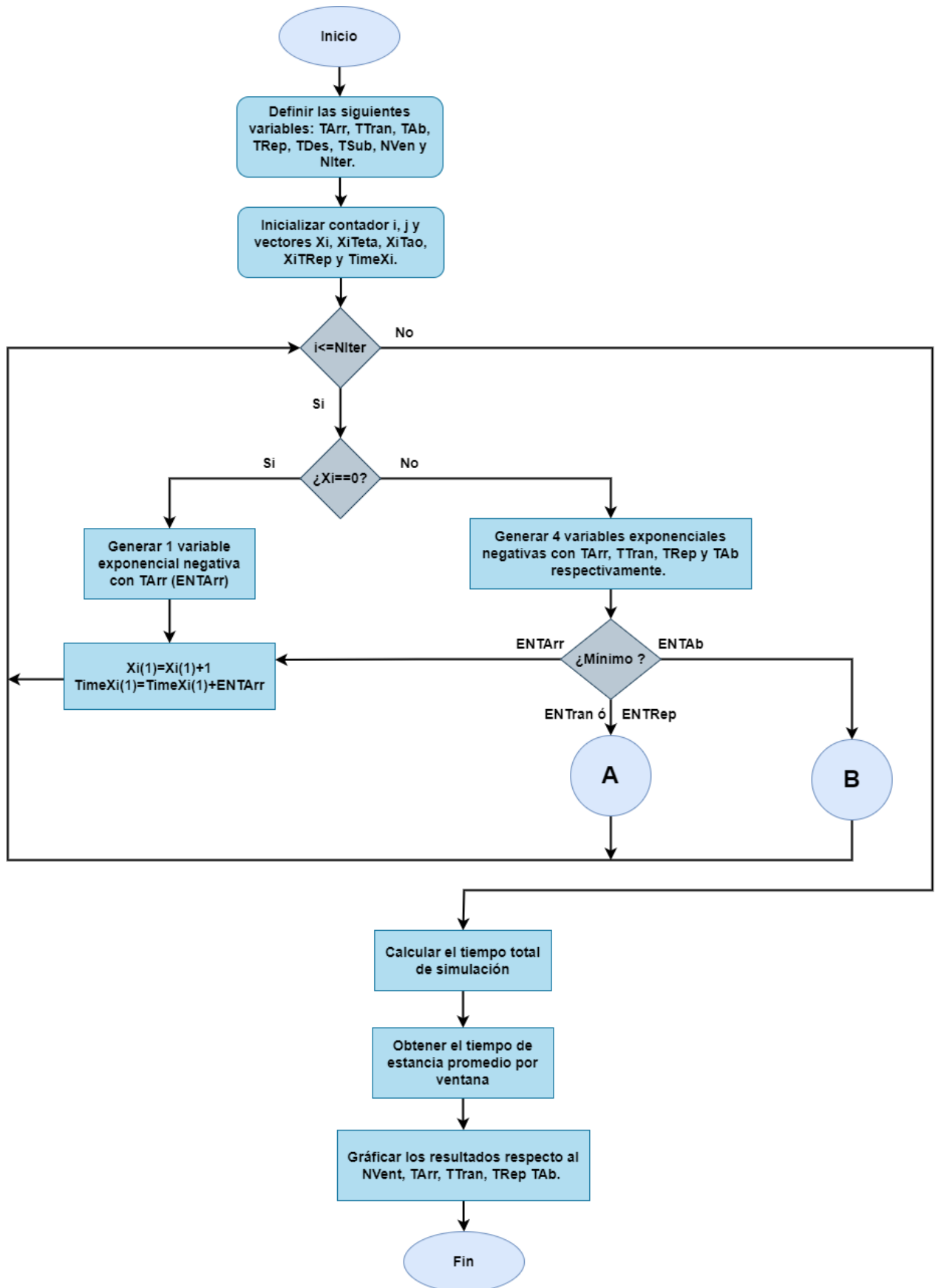


Figura 21. Diagrama de flujo de solución matemática para la cadena de Markov de un sistema de video en vivo

En el diagrama de la Figura 21 se encuentra el algoritmo que se debe implementar para dar solución por implementación a la cadena de Markov que representa un sistema de servicio de video en vivo. En primer lugar, se deben definir los parámetros de entrada para simular el comportamiento de los usuarios dentro del sistema y el cambio que resulta en el estado de la cadena.

Los parámetros que deben ser definidos al inicio son tasa de conexión, tasa de desconexión, tasa de descarga y tasa de subida. La tasa de transferencia a la ventana superior inmediata depende del esquema de asignación de recursos y se definirá con más detalle en las secciones próximas. Del mismo modo la tasa de producción del video depende de la tasa de descarga promedio de cada usuario.

Otro parámetro importante para la implementación es el número de ventanas contenidas en la hiperventana, que representa la longitud de esta. Finalmente se debe definir el número de iteraciones que se desea ejecutar la solución por implementación.

Una vez definidos los parámetros de entrada, se inicializan algunos contadores que servirán para recorrer las iteraciones en distintos procesos (recorrer vectores y ejecutar una sentencia  $n$  veces). En este paso igual se inicializan vectores que almacenan valores de poblaciones promedio por ventana, tiempos de estadía por ventana, variables aleatorias exponenciales negativas que se utilizan para la ocurrencia de los sucesos, anchos de banda consumidos por ventana, etc. Así mismo se calculan algunos parámetros como la tasa de producción del video, tasa de descarga promedio por usuario, tasa de subida promedio por usuario, ancho de banda proveniente de servidores, etc.

Posteriormente, comienzan las iteraciones de la solución por implementación del modelo. Se comienza por verificar si las poblaciones en todas las ventanas de la hiperventana son cero, en caso de que sea cierto, el único suceso que se puede generar es la conexión de un usuario al sistema. Por lo tanto, se genera una variable aleatoria con distribución exponencial negativa y la tasa de conexión, se incrementa en una unidad la población de la ventana 0 y se almacena el valor de la variable aleatoria en tiempo de estadía y se retorna a la condición de iteración.

En caso de que el vector de poblaciones sea diferente de cero en alguna de sus posiciones, se deben generar cuatro variables aleatorias con distribución exponencial negativa y tasas de conexión, desconexión, producción y de transferencia. Esto con la finalidad de conocer que suceso ocurrió (conexión, desconexión, transferencia a la ventana inferior inmediata o transferencia a la ventana superior inmediata). La variable con el valor mínimo es la que indicara que suceso ocurrió.

Si la de menor valor fue la generada a partir de la tasa de conexión, se incrementa en una unidad la población de la ventana 0 y se almacena este valor

en tiempo de estadía y se retorna a la condición de iteración. En caso contrario se prosigue a los procesos A (cuando la de menor valor es la variable generada con la tasa de transferencia o la tasa de producción) o bien el proceso B (cuando la de menor valor es generada con la tasa de desconexión).

En cada una de las iteraciones obtienen las poblaciones promedio por ventana y los anchos de banda consumidos por la población de cada ventana (ancho de banda total consumido, ancho de banda consumido de la red *P2P* y ancho de banda consumido de la red *CDN*).

El cálculo de las poblaciones por promedio por ventana se realiza de la siguiente forma:

1.- Se multiplica el vector de poblaciones de la hiperventana (HV) por el suceso ganador (conexión, desconexión, transferencia a la ventana superior inmediata, transferencia a la ventana inferior inmediata) denotada por  $E_{vfinal}$ .

2.- Se acumula este valor en la variable  $xi\_prom$

3.- Se divide  $xi\_prom$  por el tiempo de simulación total (la suma de los tiempos de estadía por ventana) denotado por  $tp$  y se almacena este valor en  $x\_prom$ .

Al finalizar las iteraciones totales de la solución por implementación se gráfica el valor final almacenado en  $x\_prom$ .

Para el cálculo de los anchos de banda por ventana se sigue el siguiente proceso:

1.- Se calcula el ancho de banda ( $B_d^i$ ) total consumido por ventana en esa iteración de acuerdo con la siguiente expresión:

$$B_d^i = \min \{ C_\omega * X_i, \sum_{k=i+1}^c \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j} + \frac{\mu_s * X_i}{\sum_{j=0}^{c-1} X_j} \}$$

2.- En caso de que la operación  $\min$  sea  $B_d^i = C_\omega * X_i$  se dice que el sistema está en abundancia, ya que la tasa de descarga de los *peers* se limita a la tasa de descarga máxima. Es decir, los *peers* descargan a tasa máxima el archivo de video debido a que dentro del sistema la cantidad de recursos ofrecidos por la red *P2P* y la red *CDN*  $\left( \sum_{k=i+1}^c \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j} + \frac{\mu_s * X_i}{\sum_{j=0}^{c-1} X_j} \right)$  es mayor.

Cuando el sistema está en abundancia se sabe que la tasa máxima de descarga fue atendida por recursos proveniente de ambas redes, sin embargo, se necesita conocer la porción que fue proporcionada cada una.

En el presente proyecto se asignan en primer lugar los recursos provenientes de la red *P2P* en caso de que estos recursos sean insuficientes para cubrir la tasa

máxima se asignan recursos de la red *CDN*. Por lo tanto, una vez calculado el ancho de banda total consumido ( $B_d^i = C_\omega * X_i$ ) se realiza una operación min entre este valor y ancho de banda consumido de la red *P2P* ( $B_{u,p}^i = \sum_{k=i+1}^c \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j}$ ).

- Si el valor mínimo fue  $B_{u,p}^i$ , quiere decir que estos recursos no fueron suficientes para cubrir la tasa máxima, por lo tanto, se recurrió a recursos de la red *CDN*. Entonces, el ancho de banda consumido de la red *CDN* es la diferencia entre el ancho de banda total consumido y el ancho de banda consumido de la red *P2P*. Lo cual se expresa como:

$$B_{u,s}^i = C_\omega * X_i - B_{u,p}^i$$

- Si el valor mínimo fue  $C_\omega * X_i$ , quiere decir que el ancho de banda total fue consumido de la red *P2P*. Entonces, el ancho de banda consumido de la red *CDN*,  $B_{u,s}^i = 0$ .

3.- En caso de que la operación min sea  $B_d^i = \sum_{k=i+1}^c \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j} + \frac{\mu_s * X_i}{\sum_{j=0}^{c-1} X_j}$ , se entiende que el sistema opera en condición de penuria, debido a que los *peers* deben descargar a tasa menor que la máxima tasa de descarga, es decir, los *peers* consumen todo el ancho de banda ofrecido tanto por la red *P2P* como por la red *CDN*. En este caso, al calcularse ambos anchos de banda por separado, se obtiene:

$$B_{u,p}^i = \sum_{k=i+1}^c \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j}$$

$$B_{u,s}^i = \frac{\mu_s * X_i}{\sum_{j=0}^{c-1} X_j}$$

En el presente proyecto se tiene como principal objeto analizar el sistema en condición de abundancia, sin embargo, se establecen las expresiones propias a penuria dado que en alguna de las iteraciones, el sistema puede operar bajo esta condición y se debe registrar este ancho de banda consumido para obtener así el ancho de banda consumido en todas las iteraciones de la solución por implementación.



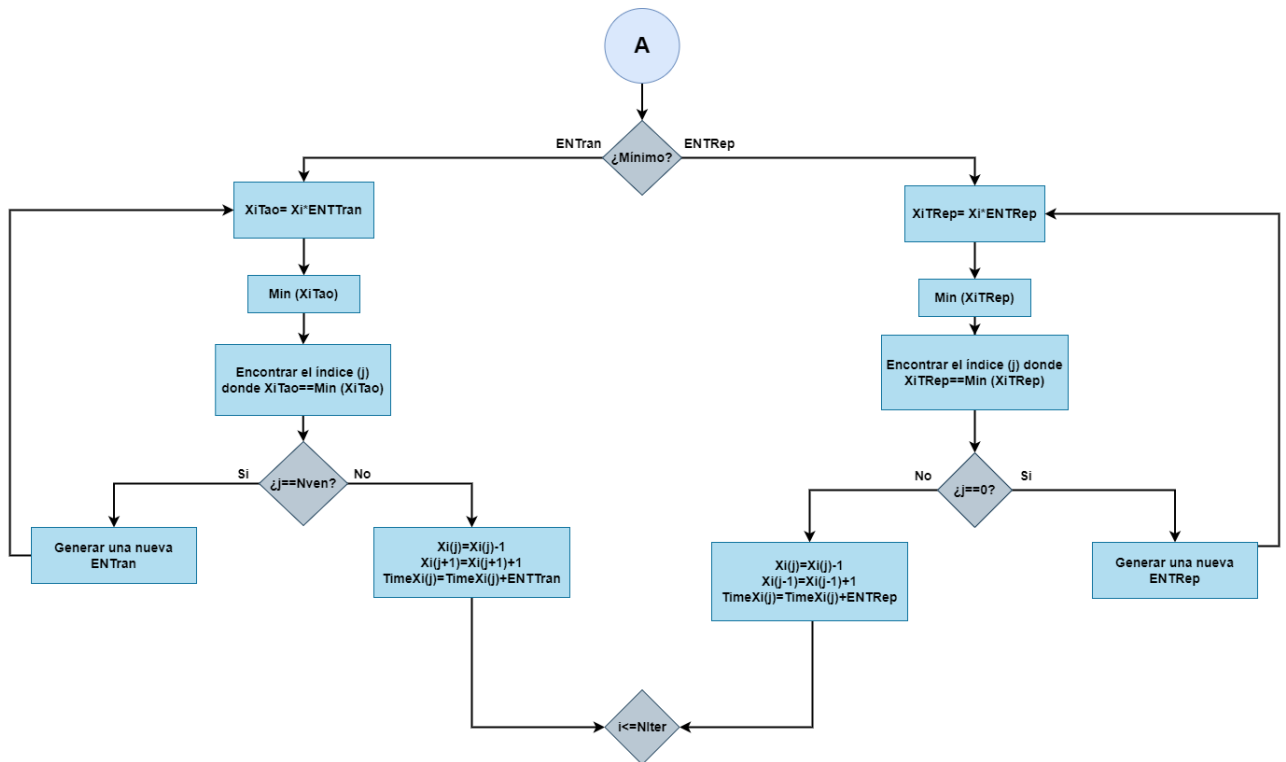


Figura 22. Diagrama para  $\text{mín} = \text{ENTran}$  ó  $\text{ENTRep}$

En el diagrama de la Figura 22 se representa el algoritmo del proceso A, que es cuando la variable aleatoria de menor valor es la variable generada con la tasa de transferencia a la ventana superior inmediata o la tasa de producción del video.

Se realiza la comparación para saber cuál fue la de valor mínimo.

- Variable aleatoria generada a partir de la tasa de transferencia a la ventana superior inmediata: se escalan las poblaciones de cada una de las ventanas pertenecientes a la hiperventana por la variable aleatoria generada. Posteriormente, se obtiene el mínimo de ese vector escalado con la finalidad de conocer el índice de la ventana donde ocurrió la transferencia a la ventana superior inmediata.

Una vez que se conoce el índice, se verifica si es igual a la longitud de la hiperventana. En caso de ser verdadero, se genera una nueva variable aleatoria y se escala nuevamente el vector de poblaciones por esta nueva variable. Debido a que un usuario en la ventana  $c$ , no puede avanzar hacia otra ventana puesto que se encuentra en sincronía con la captura del evento en vivo.

En caso de ser falso se decrementa en una unidad la población de la ventana  $j$ , se incrementa en una unidad la población de la ventana  $j + 1$  y se almacena en tiempo de estadía por ventana el valor de la variable aleatoria. Finalmente se retorna a la condición de iteración.

- Variable aleatoria generada a partir de la tasa de producción: se escalan las poblaciones de cada una de las ventanas pertenecientes a la hiperventana por la variable aleatoria generada. Posteriormente, se obtiene el mínimo de ese vector escalado con la finalidad de conocer el índice de la ventana donde ocurrió la transferencia a la ventana inferior.

Una vez que se conoce el índice, se verifica si es igual a 0. En caso de ser verdadero, se genera una nueva variable aleatoria y se escala nuevamente el vector de poblaciones por esta nueva variable. Debido a que un usuario en la ventana 0, no puede retroceder hacia una ventana inferior puesto que saldría la hiperventana (visualización en tiempo real), el resultado es decrementar esa población y tomarlo como un abandono de la ventana 0.

En caso de ser falso se decrementa en una unidad la población de la ventana  $j$ , se incrementa en una unidad la población de la ventana  $j - 1$  y se almacena en tiempo de estadía por ventana el valor de la variable aleatoria. Finalmente se retorna a la condición de iteración.

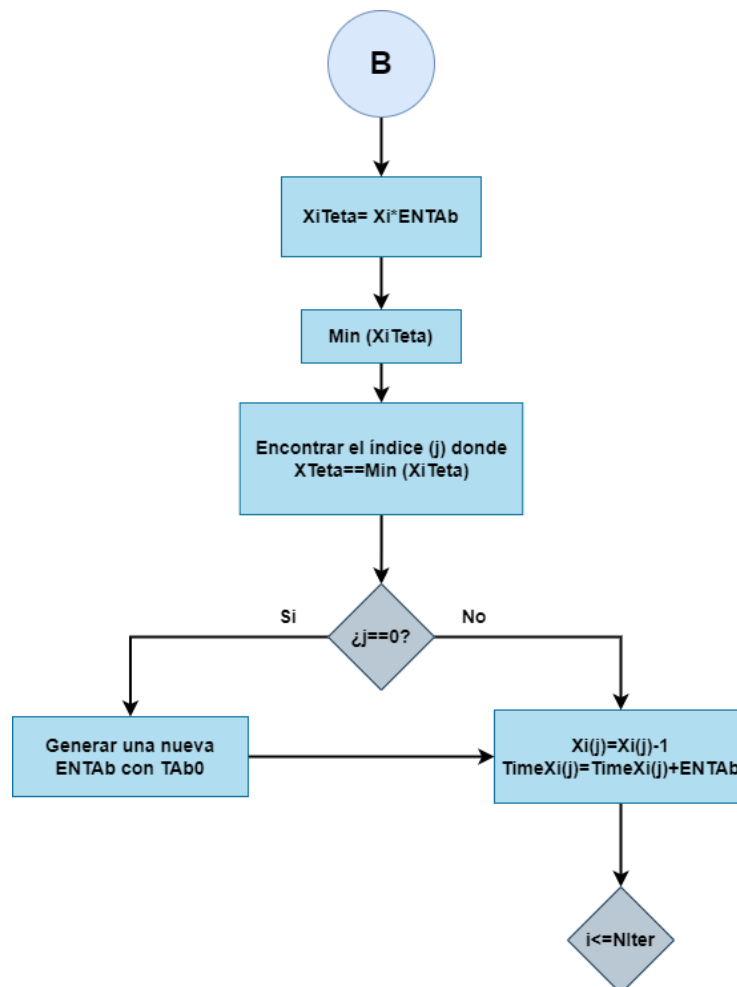


Figura 23. Diagrama para  $mín=ENTAb$

En el diagrama de la Figura 23 se representa el algoritmo de proceso B, que es cuando la variable aleatoria de menor valor es la variable generada con la tasa de desconexión.

Se escalan las poblaciones de cada una de las ventanas pertenecientes a la hiperventana por la variable aleatoria generada. Posteriormente, se obtiene el mínimo de ese vector escalado con la finalidad de conocer el índice de la ventana donde ocurrió la desconexión.

Una vez que se conoce el índice, se verifica si es igual a 0. En caso de ser verdadero, se genera una nueva variable aleatoria a partir de la tasa de desconexión de la ventana 0. Debido a que en el presente trabajo se suponen que un usuario en la ventana 0, se desconecta del sistema a una tasa de abandono promedio diferente a la tasa de abandono en las demás ventanas.

Posteriormente, se decrementa en una unidad la población de la ventana 0 y se almacena en tiempo de estadía por ventana el valor de la variable aleatoria.

En caso de que el índice sea diferente de cero se decrementa en una unidad la población de la ventana  $j$ , y se almacena en tiempo de estadía por ventana el valor de la variable aleatoria. Finalmente se retorna a la condición de iteración.

En el diagrama de la Figura 21 se presenta a la variable  $\tau(TTran)$ , cuyos valores se definen a partir de un esquema de asignación de recursos que se retoma de forma detallada en la sección próxima. Las estadísticas que se obtendrán de la implementación del diagrama (poblaciones promedio por ventana y anchos de banda consumidos en el sistema) se graficarán en función de distintos valores para los parámetros de entrada ( $c, P_\omega, \mu_\omega, \theta, etc.$ )

## 5.2 Diseño del esquema de asignación de recursos para servicios de video en vivo

En esta sección se desarrolla el bosquejo del esquema de asignación de recursos para servicios de video en vivo. En este diseño se explican las condiciones que debe cumplir el sistema para realizar la asignación de recursos. En esta etapa se ha desarrollado un esquema de asignación uniforme y considerando condición de abundancia dentro del sistema.

Tomando como base el análisis del comportamiento de los *peers* se plantea desarrollar un esquema de asignación de recursos conveniente para los servicios de video en vivo. Los esquemas presentados en [7] (Q ventanas hacia atrás) y [13] (GDPV) y muestran un desempeño eficiente para la asignación de

recursos para servicios de video bajo demanda (VoD). Sin embargo, tomando en cuenta los objetivos de este proyecto con un enfoque dirigido a los servicios de video en vivo, se plantea realizar una adecuación en primera instancia al esquema de asignación de recursos de distribución uniforme, sin dejar de lado la posibilidad de retomar Q ventanas hacia atrás o GDPV para ser aplicados a este tipo de servicios.

En primer lugar, se analizó la cantidad de recursos necesarios para que la población correspondiente a una ventana sea atendida y pueda descargar de manera satisfactoria el video en vivo. Con el propósito de establecer una expresión que permita cuantificar esta cantidad de recursos al cual se le denomina ancho de banda consumido.

De manera general dentro del sistema se tiene la tasa de descarga global que regula el ancho de banda consumido en el proceso de descarga del contenido en vivo, dicha tasa es representada con  $C_\omega$ . Esta tasa permite conocer la máxima velocidad a la que un *peer* en general conectado al sistema realiza la descarga del contenido. Sin embargo, esta tasa no especifica la cantidad de recursos necesarios para que la población de una ventana descargue de forma continua el contenido de la ventana superior inmediata.

Por lo tanto, se establece a  $C_\omega X_i$  como la cantidad de recursos (ancho de banda de descarga consumida) requerida por la población en la ventana  $i$  para descargar el video en vivo. La cantidad  $C_\omega X_i$  es el resultado de multiplicar la tasa global  $C_\omega$ , que es la tasa individual máxima de descarga, por  $X_i$  (la población de *peers* en la ventana  $i$ ). De igual forma dentro del sistema existe la tasa de producción, la cual indica la velocidad a la cual se captura el evento en vivo y se genera el archivo de video, es decir, esta tasa indica la velocidad con la que los usuarios deben descargar el video para no perder sincronía respecto a la ventana actual ( $c$ ). Esta tasa es denotada por  $P_\omega$ .

Para el presente proyecto se supone que la tasa de producción debe ser menor que la tasa de descarga ( $C_\omega > P_\omega$ ) para evitar que un usuario pierda sincronía con la producción del video, sea transferido a la ventana inferior inmediata y/o eliminado del sistema.

Una vez que se analizó la cantidad de recursos necesarios para que la población correspondiente a una ventana en específico pudiera descargar el video en vivo se analizó el tamaño de la población de las ventanas superiores para conocer si la cantidad de recursos requeridos puede ser cubierta únicamente por los recursos procedentes de los *peers* o se deben solicitar recursos a los servidores de la red CDN.

En el sistema de servicios de video en vivo se busca que los recursos provenientes de la red P2P (*peers*) sean asignados en primer lugar y en caso de requerir más recursos, asignar la cantidad faltante de la red CDN (servidores).

Dentro del sistema, para poder realizar la asignación de recursos se deben considerar dos casos principales:

El primero de ellos es cuando la cantidad de recursos requeridos por una población ( $C_{\omega}X_i$ ) para ser atendidos es menor a la cantidad de recursos disponibles provenientes de los *peers* y de los servidores *CDN*. Por lo tanto, el existe condición de abundancia y los *downloaders* descargan el video a la velocidad requerida.

Bajo condición de abundancia, se necesita calcular la cantidad de recursos que proporcione la red *P2P* y la cantidad de recursos que proporcione la red *CDN*, con la finalidad de conocer si el sistema fue autónomo, es decir, únicamente basto con los recursos provenientes de los *peers* para atender a los *downloaders*. O bien, si fue necesario solicitar recursos de la red *CDN*.

El segundo caso, es cuando la cantidad de recursos requeridos por una población ( $C_{\omega}X_i$ ) es mayor a la cantidad de recursos ofrecidos por la red *P2P* y la red *CDN*, en este caso, dentro del sistema existe condición de penuria. Y, por lo tanto, los recursos provenientes tanto de la red *CDN* como de la red *P2P* son insuficientes para que los *downloaders* descarguen el video a la velocidad requerida.

Si el sistema, se encuentra en penuria, los *downloaders* pierden sincronía con la producción del video, se atrasan y finalmente son eliminados del sistema, y como resultado el sistema colapsa.

A razón del segundo caso, es que se decide trabajar con condición de abundancia para modelar de manera más cercana a la realidad los eventos realizados por los usuarios finales al momento de descargar y visualizar un archivo de video.

Dentro del sistema que se desarrolla en el proyecto, los *peers* conectados a la transmisión en directo reciben también el nombre de *downloaders*. Debido a la naturaleza de la red *P2P* los *downloaders*, son parte medular del ancho de banda de subida que requiere el sistema, ya que los *peers* tienen la capacidad de compartir el contenido almacenado en su *buffer* a otros *peers* que se encuentren descargando el video en ventanas inferiores dentro del sistema. Este hecho permite que el ancho de banda requerido por una población no sea consumido en su totalidad de la red *CDN* y así se evita que la red se sature y el sistema colapse, por lo tanto, los servidores atienden de manera equitativa a los *downloaders* situados en las diferentes ventanas que componen a la hiperventana, de acuerdo con la distribución uniforme.

Sin embargo, el sistema puede distribuir los recursos provenientes de la red *CDN* de manera tal, que algunos *peers* reciban mayor cantidad de recursos de estos servidores. Y así evitar que algunas poblaciones tengan mayor disponibilidad de recursos para ser atendidas que otras. Como se ha mencionado anteriormente, en primer lugar, se asignan los recursos provenientes de la red *P2P* y en caso

de ser necesario se debe proporcionar un espacio del ancho de banda proveniente de la red *CDN*.

El sistema tiene una tasa de subida global que establece el ancho de banda de subida promedio que puede proporcionar un *peer* en el proceso de la transmisión de contenido en vivo, dicha tasa es representada con  $\mu_\omega$ . Esta tasa permite conocer la velocidad promedio a la que un *peer* conectado al sistema sube la porción del archivo de video que tiene almacenada en su *buffer*.

Sin embargo,  $\mu_\omega$  no especifica la cantidad de recursos que puede proporcionar la población de la ventana  $i$  a los *peers* en ventanas  $j$ ,  $0 \leq j < i - 1$ , para que puedan comenzar el proceso de descarga del archivo de video.

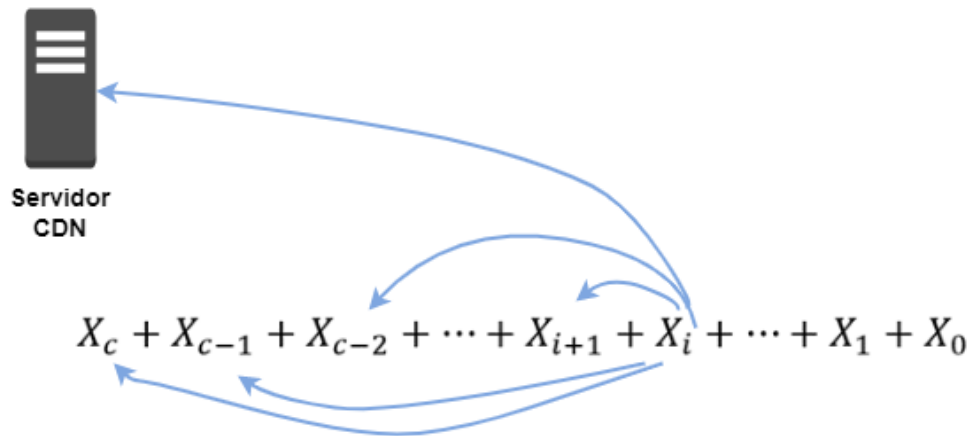
Por lo tanto, se establece a  $\mu_\omega X_i$  como la cantidad de recursos (ancho de banda de subida) proporcionada por la población en la ventana  $i$ . Es decir, los *chunks* contenidos en su *buffer* para compartir con otros *peers* en ventanas inferiores. La cantidad  $\mu_\omega X_i$  es el resultado de multiplicar la tasa global  $\mu_\omega$ , que es la tasa individual promedio de subida, por  $X_i$  la población de *peers* en la ventana  $i$ .

De igual manera, dentro del sistema existe una tasa de subida que establece el ancho de banda de subida que, en caso de ser requerido, es proporcionado por los servidores de la red *CDN*. Dicha tasa se representa como  $\mu_s$ , y se asigna en función del esquema de asignación de recursos.

El objetivo principal de un esquema de asignación de recursos es procurar condiciones de abundancia dentro del sistema, es decir, que los recursos disponibles en la red (ancho de banda de subida) sea mayor o igual al demandado por el sistema (ancho de banda de descarga) y que este a su vez sea proporcionado equitativamente a todos los *peers* conectados al sistema.

Con base en lo descrito hasta este punto de la investigación se decide desarrollar un bosquejo del esquema de asignación de recursos para servicios de video en vivo basado en el esquema de distribución uniforme.

En la Figura 24, se observa que la población de la ventana  $i$  puede ser atendida (obtener recursos) de poblaciones en ventanas superiores, es decir, la población en la ventana  $i$  puede ser atendida por poblaciones en ventanas  $k$ ,  $i + 1 \leq k \leq c$ .



*Figura 24. Obtención de recursos para la población en la ventana  $i$*

En la Figura 24 se observa que la población de una ventana cual esta sea puede obtener recursos únicamente de poblaciones situadas en ventanas superiores. Esto debido a que los usuarios en ventanas superiores tienen un mayor progreso en la descarga del contenido en vivo.

### Esquema de Asignación de Recursos Uniforme para servicios de video en vivo

#### Inicio

- 1.- Arriba o es transferido un usuario a la ventana  $i, i = 0, 1, 2, \dots, c - 1$  de la hiperventana
- 2.- Se define  $k = i + 1$
- 3.- **Repetición de Recolección**  
Se almacenan las poblaciones de *peers* situadas en las ventanas  $k, i + 1 \leq k \leq c$ , que pueden atender a los usuarios en la ventana  $i$ .  
**Fin de Repetición**
- 4.- Conjunto de poblaciones de *peers* con recursos disponibles para atender a la población en la ventana  $i$ .
- 5.- ¿La tasa real de descarga es igual a la tasa máxima de descarga?  
**Si:** Asignar los recursos  
**No:** Se asignan recursos, ¿aún hay *peers* en la lista?  
**Si:** Retornar a paso 4  
**No:** Buscar recursos en el *CDN* y repetir condición **Si** del paso 5
- 6.- Generar estadísticas

#### Fin

Debido a que el esquema de asignación de recursos en esta etapa del proyecto es un esquema uniforme, el número de ventanas de donde provienen los recursos no está limitado, es decir, la población en la ventana  $i$  pueden obtener recursos de ventanas  $k$  tal que  $k, i + 1 \leq k \leq c$ , además de obtener recursos provenientes de los servidores *CDN*, como se muestra en la siguiente expresión:

$$\begin{aligned}
R_i = & \left( (\mu_\omega * X_{i+1}) \left( \frac{X_i}{X_0 + X_1 + \dots + X_i} \right) + \mu_s \left( \frac{X_i}{X_0 + X_1 + \dots + X_c} \right) \right) + \\
& \left( (\mu_\omega * X_{i+2}) \left( \frac{X_i}{X_0 + X_1 + \dots + X_{i+1}} \right) + \mu_s \left( \frac{X_i}{X_0 + X_1 + \dots + X_c} \right) \right) + \dots + \\
& \left( (\mu_\omega * X_{c-1}) \left( \frac{X_i}{X_0 + X_1 + \dots + X_{c-2}} \right) + \mu_s \left( \frac{X_i}{X_0 + X_1 + \dots + X_c} \right) \right) + \\
& \left( (\mu_\omega * X_c) \left( \frac{X_i}{X_0 + X_1 + \dots + X_{c-1}} \right) + \mu_s \left( \frac{X_i}{X_0 + X_1 + \dots + X_c} \right) \right)
\end{aligned}$$

Realizando análisis matemático se observa que  $X_i$ ,  $\mu_\omega$  y  $\mu_s$  son factores en común en las expresiones sumadas, por lo tanto, esa suma se puede resumir y generalizar para cualquier población como se muestra en la siguiente expresión:

$$R_i = X_i \left( \sum_{k=i+1}^c \frac{\mu_\omega X_k(t)}{\sum_{j=0}^{k-1} X_j(t)} + \frac{\mu_s}{\sum_{j=0}^c X_j(t)} \right)$$

En la etapa de implementación se detectaron algunas inconsistencias en el modelo para servicios de video en vivo que se había desarrollado en la etapa de diseño del presente proyecto. Con la finalidad de representar de una forma más próxima el comportamiento y funcionamiento de un sistema de transmisión de video en vivo se decidió realizar dichos cambios y reportar en la sección X, donde se especifican las inconsistencias encontradas y el cambio realizado que se consideró pertinente para solucionar estos puntos.



## Capítulo 6

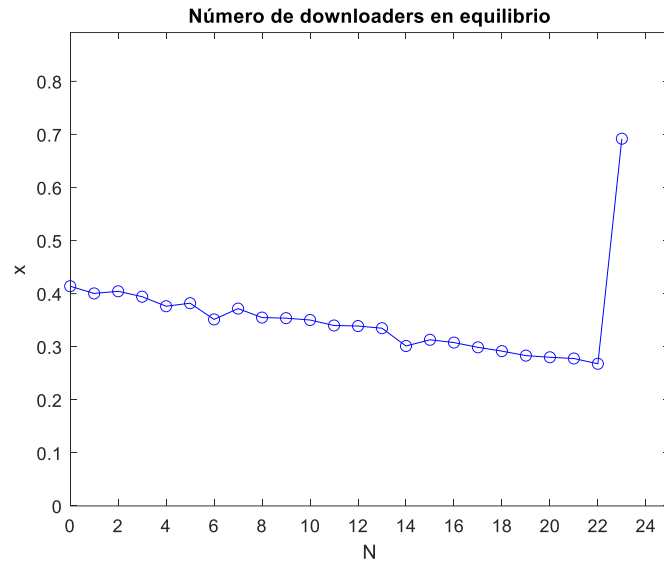
### Pruebas e Implementación

Para iniciar el desarrollo e implementación del código fuente, que permite realizar la evaluación por implementación de un sistema de transmisión de video en vivo, se siguió el algoritmo mostrado en la Figura 21. En primera instancia, se aplicó este algoritmo para servicios de video bajo demanda (*VoD*) abordados en [5]. Esto con la finalidad de obtener resultados aproximados a los aprobados, validados y publicados acerca del funcionamiento de un sistema de transmisión de video bajo demanda y así poder tomar estos resultados como marco de referencia para conocer los resultados esperados que se deben mostrar al ejecutar y probar la evaluación por implementación de una cadena de Markov que representa un sistema *livestreaming*.

Es importante señalar que el valor de los parámetros para implementar el diagrama de la Figura 21 fueron tomados de [5]. Esto debido a que en ese artículo se menciona que los parámetros ahí manejados fueron extraídos y basados en el funcionamiento de sistemas reales de distribución de video, por lo tanto, y aunado a que este trabajo es un modelado de un sistema se decide tomar como válidos estos parámetros para realizar pruebas de funcionamiento.

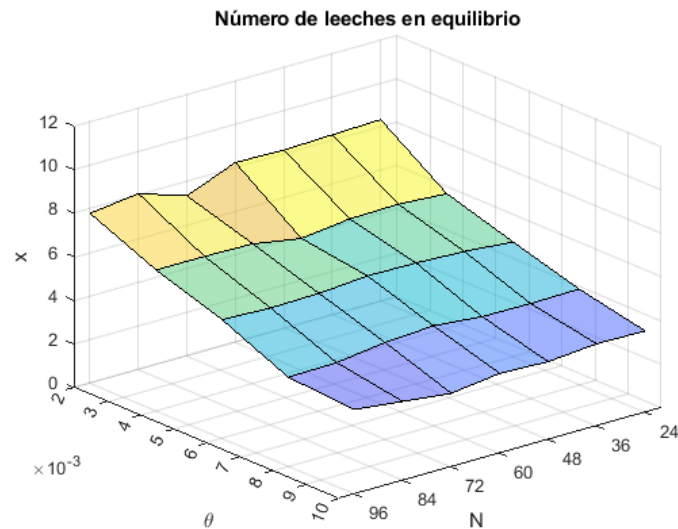
La primer tarea a realizar es obtener una gráficas similares a algunas reportadas en [5] para conocer si la evaluación por implementación de una cadena de Markov para *VoD* realizada en este trabajo es correcta y genera el comportamiento adecuado bajo los parámetros utilizados.

Para la primera gráfica presentada a continuación se trabajó con el esquema de asignación de recursos de distribución uniforme (DU), en la cual se observa el número de *downloaders* (*leeches*) en el sistema de video bajo demanda. En esta primera gráfica se observa el número promedio de *downloaders* en un archivo de video dividido en 24 ventanas ( $N=24$ ).



*Figura 25. Gráfica bidimensional de downloaders promedio en VoD bajo DU*

Una vez obtenida la gráfica de poblaciones promedio para un archivo compuesto por N ventanas y una tasa de desconexión fija, se procedió a obtener las poblaciones de *downloaders* promedio por ventana dentro de un archivo de video. En esta prueba se obtuvieron las poblaciones promedio de *downloaders* variando el número de ventanas (N) y la tasa de desconexión general ( $\theta$ ) de los *peers* dentro del sistema.

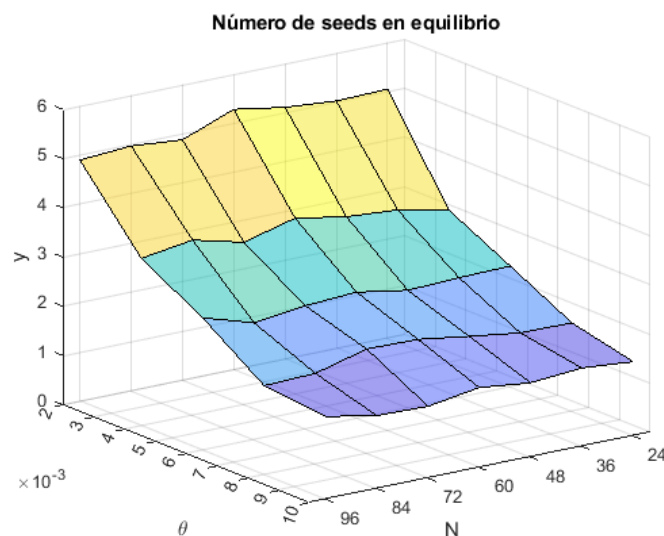


*Figura 26. Gráfica de numero de downloaders en VoD bajo DU*

Una vez obtenida la gráfica mostrada en la Figura 26, se procedió a comparar la curva obtenida con la curva de la Figura 12 reportada en [5]. Al realizar dicha comparación se observó que la forma de la curva era similar y, por lo tanto, se tiene una aproximación correcta de la evaluación por implementación de una cadena de Markov para servicios de video bajo demanda. La gráfica reportada en la Figura 26 se obtuvo empleando algunos de los parámetros ( $\lambda$ ,  $c$  y  $\mu$ ) con

los mismos valores utilizados en [5] para obtener una curva aproximada a la curva de la Figura 12.

De manera paralela se generó una curva que representa el número de *seeds* promedio dentro de un sistema de video bajo demanda, esta curva al igual que la curva de *downloaders* promedio se generó empleando los mismos parámetros y usando el mismo esquema de asignación de recursos (DU) utilizados para obtener la curva de la Figura 26. La curva de la Figura 27 se comparó con la curva de la Figura 13 publicada en [5] con la finalidad de conocer si la lógica del algoritmo representa aproximadamente el comportamiento del sistema de video bajo demanda analizado en dicho artículo.

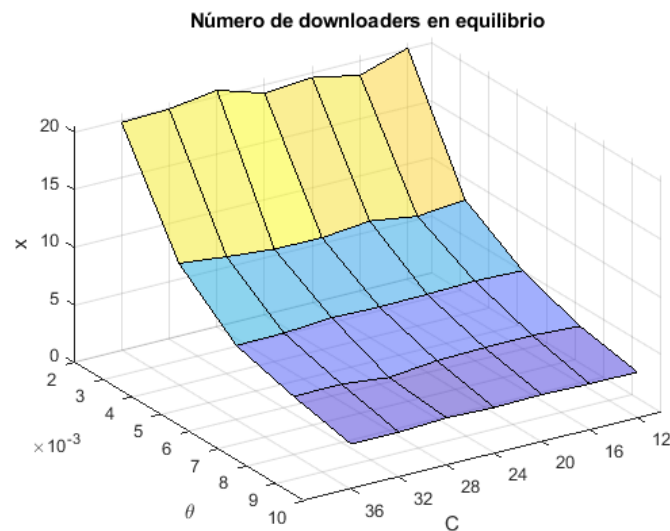


*Figura 27. Gráfica de numero de seeds en VoD bajo DU*

Después de obtener las gráficas arriba mencionadas, compararlas con las que se describen en [5] para video bajo demanda y observar que se tienen resultados similares en cuanto a la forma que tiene la curva. Se puede establecer que la lógica empleada en el presente proyecto para evaluar por implementación una cadena de Markov que representa servicios de video, es la adecuada para evaluar la cadena propia para servicios de video en vivo. Por otro lado, se puede establecer que el algoritmo del programa para servicios de video bajo demanda funciona adecuadamente, es decir, genera curvas de poblaciones de *downloaders* y *seeds* promedio que muestran la forma esperada.

Ahora bien, se procedió a realizar cambios y adecuaciones al programa de evaluación por implementación de servicios de video bajo demanda utilizando el mismo esquema de asignación de recursos, con la diferencia de que ahora el programa generado a partir de estas adecuaciones representa los servicios de video en vivo (*live streaming*) con el objetivo de observar el comportamiento de los *peers* en un sistema de video en vivo y poder comparar las gráficas obtenidas para video en vivo con las gráficas obtenidas para *VoD*.

Continuando con la siguiente gráfica se observan las poblaciones promedio de *downloaders* dentro de un sistema *live streaming* con el esquema de asignación de recursos DU.



*Figura 28. Gráfica poblaciones promedio de downloaders live stream DU*

En esta primera evaluación por implementación de una cadena de Markov que representa un sistema de video en vivo, la curva se obtuvo al variar el número de ventanas ( $C$ ) que componen a la hiperventana. De igual forma se varió el valor de la tasa de desconexión general ( $\theta$ ) de los *peers* conectados a sistemas de servicios de video en vivo. En la gráfica de la Figura 28 se observa en el eje  $x$  la variación de la tasa de desconexión, en el eje  $y$  la variación del número de ventanas que componen a la hiperventana (cabe mencionar que esta es una de las principales diferencias respecto a los servicios de video bajo demanda, en este proyecto se denota al número de ventanas con la letra  $C$ , la cantidad de ventanas es menor respecto a *VoD*, y las ventanas están contenidas en la hiperventana). Y finalmente en el eje  $z$  se muestran las poblaciones promedio de *downloaders* en un sistema *livestreaming*.

Posteriormente se compararon las gráficas de las Figuras 26 y 28, haciendo énfasis en el eje  $z$ , correspondiente a las poblaciones promedio de *downloaders* ( $\bar{X}$ ), se observa un aumento notable del número de *downloaders* en servicios de video en vivo respecto al número de *downloaders* en video bajo demanda, sin embargo, se puede observar que la forma que tiene la curva es similar, lo cual indica que el algoritmo representa de manera adecuada el comportamiento que tienen los usuarios una vez conectados a un sistema de servicios de video.

Una vez que se realizó la comparación de las gráficas *VoD* y *live streaming* se continuó con la programación de la evaluación por implementación de la cadena de Markov que representa a los servicios de video en vivo. Anteriormente se había trabajado considerando únicamente tres sucesos que pueden modificar el

estado de la cadena de Markov para servicios de video en vivo (conexión de un usuario, desconexión de un *peer* y transferencia de un *peer* a la ventana superior inmediata).

En esta fase del proyecto se ha añadido a la evaluación por implementación un cuarto suceso. Como se ha mencionado anteriormente, un archivo de video en vivo no es un archivo estático sino dinámico debido a que no se conoce la duración total del evento en tiempo real y por lo tanto no se puede establecer un número definido de ventanas, por lo cual se optó por definir un contenedor llamado hiperventana que contiene C ventanas, consideradas como video en tiempo real.

Al añadir esta hiperventana, surge el cuarto suceso, el cual establece que un usuario es transferido a la ventana inferior inmediata en caso de que se produzca una nueva ventana de video y por lo tanto las ventanas contenidas en la hiperventana serán corridas y reenumeradas, es decir, la ventana 1 ahora es la ventana 0 y así sucesivamente hasta llegar a la ventana C.

Este cuarto suceso que se incorpora y puede modificar el estado de la cadena de Markov para servicios de video en vivo se denomina transferencia a la ventana inferior inmediata (TVI), a partir de añadir este suceso se obtiene nuevamente una gráfica de las poblaciones promedio de *downloaders* para *live streaming* empleando el esquema de asignación de recursos DU y los parámetros utilizados anteriormente para generar las curvas.

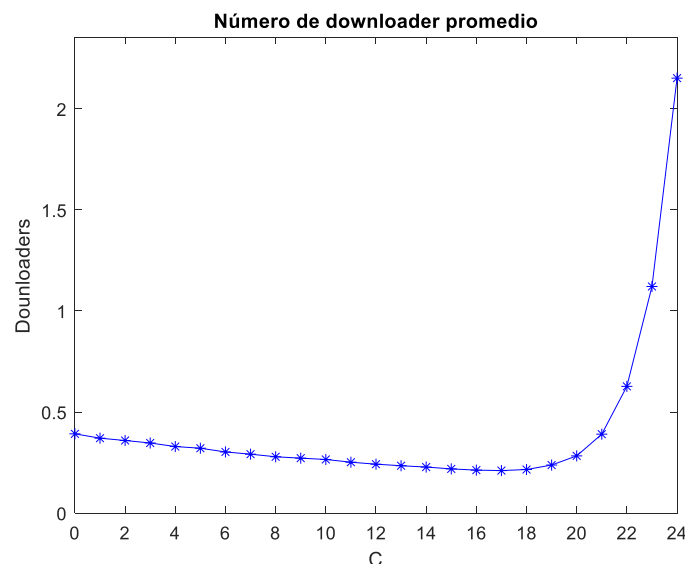


Figura 29. Gráfica bidimensional de downloaders promedio en livestreaming utilizando el esquema DU

Al igual que en los casos anteriores se generó una curva 3D con el objetivo de observar la forma de la curva generada cuando se varían el número de ventanas que componen a la hiperventana y el valor de la tasa general de desconexión. Y como se observa en la gráfica de la Figura 30, la forma de la curva es similar a las obtenidas y reportadas anteriormente.

Es importante mencionar que a partir de la adecuación del programa que hace la evaluación por implementación de un sistema *live streaming*, no se han generado curvas que representen el número de *seeds* para este tipo de sistemas, y la razón es que, en los sistemas de video en vivo no existe como tal el concepto *seed* debido a que se considera que ningún usuario tiene el video completo almacenado en su *buffer* y por lo tanto todos son catalogados como *downloaders*.

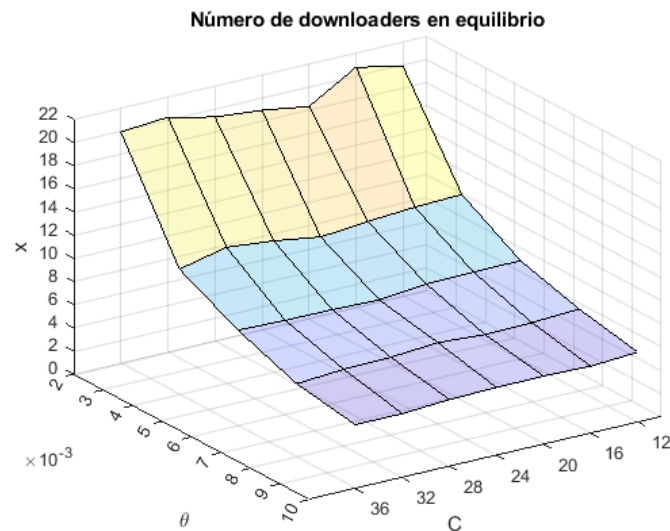


Figura 30. Gráfica downloaders promedio live streaming DU con TVI

Después de tener programado el suceso transferencia a la ventana inferior (TVI), que puede modificar el estado de la cadena, se realizaron pruebas variando el parámetro  $\lambda$ . Las siguientes tres curvas en la Figura 31 representan las poblaciones de *downloaders* promedio en el sistema de *live streaming* considerando ya el evento TVI.

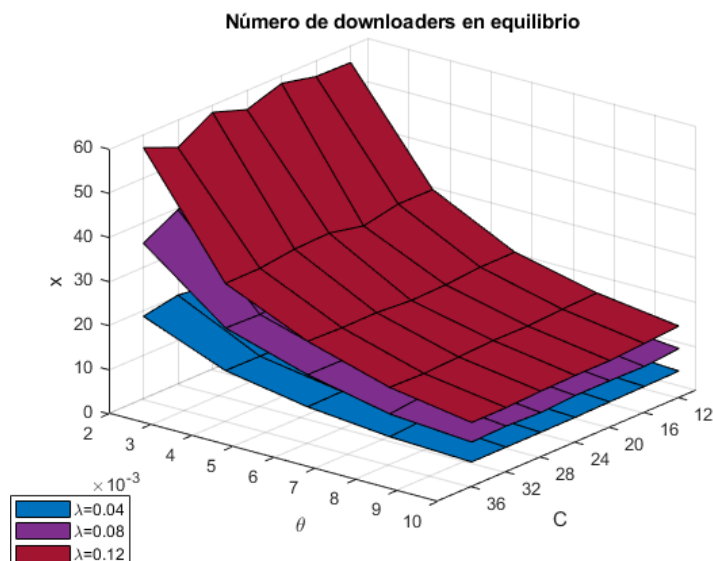


Figura 31. Gráfica de downloaders promedio variando  $\lambda$

En la Figura 31 se utilizaron los valores  $\lambda = 0.04, 0.08, 0.12$ , se puede observar cómo es que las poblaciones de *downloaders* incrementan de forma reciproca al aumento de la tasa de conexión  $\lambda$ , es decir, cuando escalamos el valor de esta tasa de conexión las poblaciones muestran un incremento en igual magnitud.

Recordando que  $\lambda$  representa el número de conexiones de los usuarios al sistema, se especula que al variar esta tasa exista un incremento proporcional en el tamaño de las poblaciones promedio de *downloaders*. En la Figura 31, se aprecia que cuando el valor de  $\lambda$  es duplicado y triplicado, respectivamente, el tamaño de las poblaciones promedio de *downloaders* sufre un incremento de la misma magnitud.

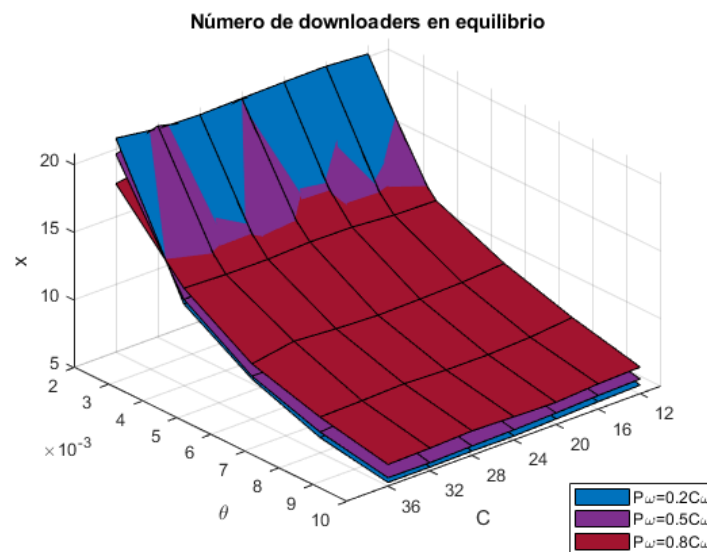


Figura 32. Gráfica de *downloaders* promedio variando  $P_\omega$ (tasa de producción)

En la gráfica de la Figura 32 se modificó el valor de la tasa de producción del video ( $P_\omega = 0.2C_\omega, 0.5C_\omega, 0.8C_\omega$ ), es decir, se modifica la velocidad a la que se produce el video y por ende la velocidad a la que los usuarios deben descargar el video para evitar un retraso respecto a la producción del video. En este sentido, los usuarios que no descarguen el video a tasa  $P_\omega$  presentan un atraso en su proceso de descarga y por lo tanto son transferidos a la ventana inferior inmediata.

En la Figura 32 se representa este evento y se puede observar que cuando  $P_\omega \ll C_\omega$  y la tasa de desconexión  $\theta$  es pequeña el número de *downloaders* es alto, pero cae precipitadamente cuando  $\theta$  aumenta. Por otro lado cuando  $P_\omega$  es la mitad que  $C_\omega$  los *downloaders* descargan el video de manera estable y no ocurre una diferencia abrupta entre los valores de poblaciones de *downloaders* promedio.

Finalmente, cuando  $P_\omega \cong C_\omega$  el abandono de usuarios va en función de la tasa de abandono  $\theta$  y no muestran una gran diferencia entre las poblaciones promedio de *downloaders*.

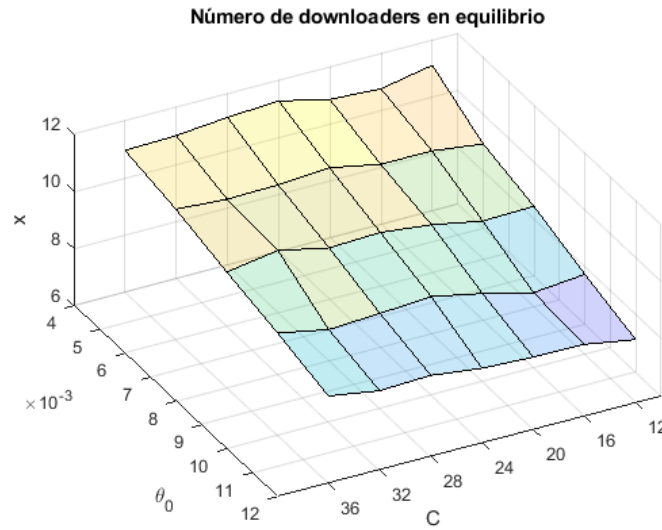
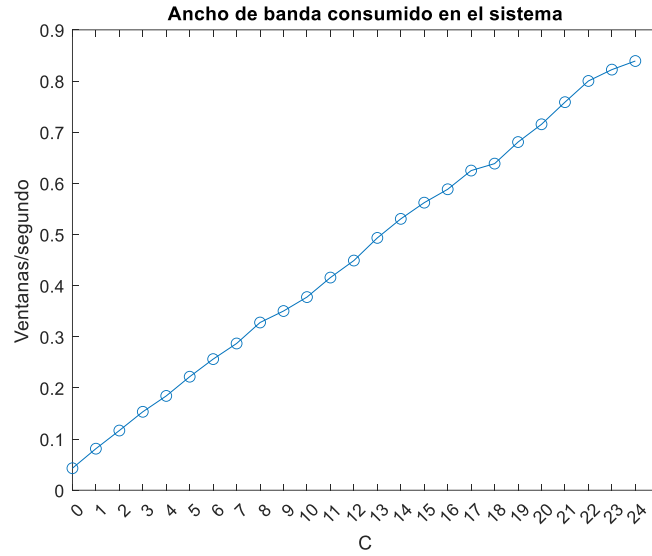


Figura 33. Gráfica de *downloaders* promedio variando  $\theta_0$

En la Figura 33 se observan las poblaciones promedio de peers dentro del sistema de distribución de video en vivo variando la tasa de desconexión de la población que descarga la ventana 0. Como se mencionó en el Capítulo 5, los *downloaders* que se encuentran descargando esta ventana se considera que tienen una tasa mayor de desconexión, lo cual provoca que se desconecten del sistema con mayor probabilidad. El resultado que esta tasa provoca sobre las poblaciones promedio de *peers* es que estas disminuyan considerablemente, es decir, como los *peers* que descargan la ventana 0 se desconectan del sistema con mayor frecuencia esto provoca que no exista una población con mayor cantidad de *peers* que tengan la oportunidad de avanzar hacia adelante y finalmente esto produce una disminución considerable en las poblaciones promedio finales como se observa en el eje z de la gráfica en la Figura 33.

Una vez obtenidas las poblaciones promedio de *peers*(*downloaders*), el siguiente paso fue obtener los anchos de banda (ancho de banda total consumido, ancho de banda consumido de la red *P2P* y ancho de banda consumido de la red *CDN*) consumidos dentro del sistema *live streaming*. La siguiente gráfica de la Figura 34 muestra el primer acercamiento realizado para obtener el ancho de banda total consumido dentro del sistema. Este ancho de banda es el resultado de multiplicar población promedio de cada ventana ( $X_i$ ) por la tasa de descarga general ( $C_\omega$ ) para un *peer* dentro del sistema *livestreaming*.





*Figura 34. Ancho de banda que demanda el sistema*

Hasta este punto de la implementación se trabajó con la adecuación al esquema de asignación de recursos de distribución uniforme para sistemas de distribución de video en vivo donde los recursos de los servidores ( $\mu_s$ ) se distribuían de igual manera que los recursos de los *peers* (uniformemente).

Con la finalidad de distribuir los recursos de los servidores *CDN* de manera más eficiente a partir de ahora se hacen pruebas para establecer una curva distinta para distribuir estos recursos. Las gráficas de las Figuras 35 y 36 muestran el tiempo de descarga promedio de los *peers* dentro del sistema *livestreaming*. La Figura 35 es el tiempo promedio de descarga para los *peers* en ventanas 1 – C y la Figura 36 representa los tiempos promedio para los *peers* en la ventana 0.

Los tiempos promedio de descarga para los *peers* en ventanas 1 – C se obtuvo mediante:

$$T = \frac{1}{C_\omega + \theta}$$

Mientras que los tiempos de descarga para los *peers* de la ventana 0 se calculan mediante:

$$T_0 = \frac{1}{C_\omega + \theta_0}$$

Recordando que la principal diferencia está en la tasa de desconexión, la cual es diferente para los *peers* de la ventana 0. Ambas ecuaciones se basan en el análisis de la ecuación 33 de [5].

Los recursos de los servidores ( $\mu_s$ ) fueron distribuidos con la siguiente ecuación:

$$\mu_{si} = \left( \frac{1}{1 + 2^i} \right) * \mu_s$$

Donde:

$\mu_{si}$ : Representa los recursos de servidores *CDN* con que son atendidos los *peers* de la ventana *i*

$\mu_s$ : Representa los recursos generales de los servidores *CDN*

*i*: Representa el índice de la ventana

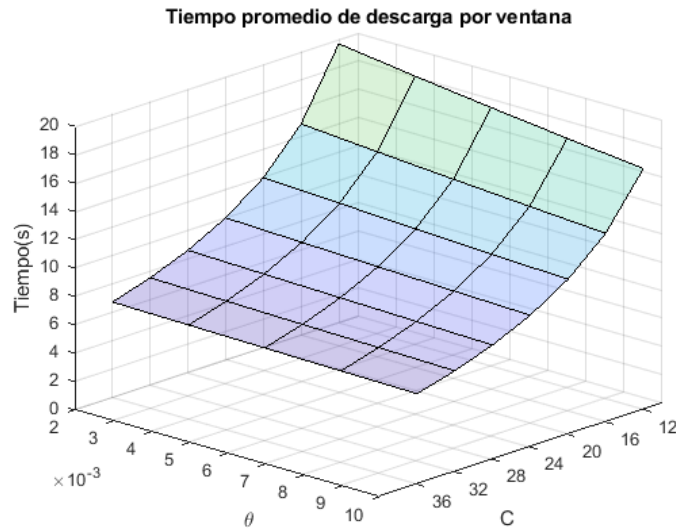


Figura 35. Tiempo promedio de descarga por ventana

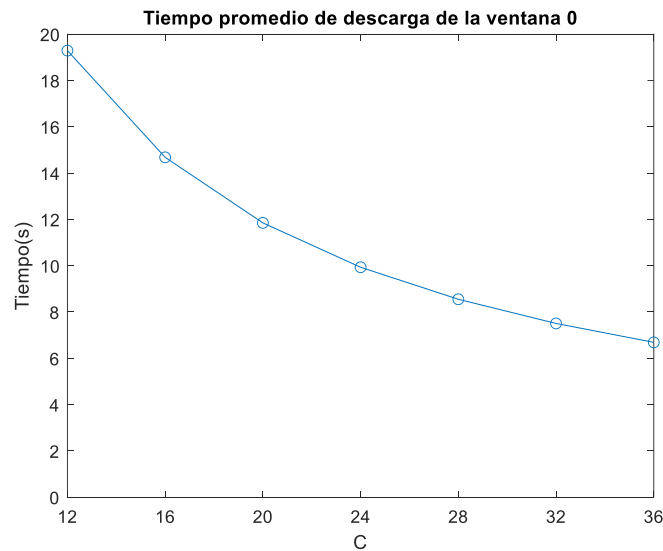
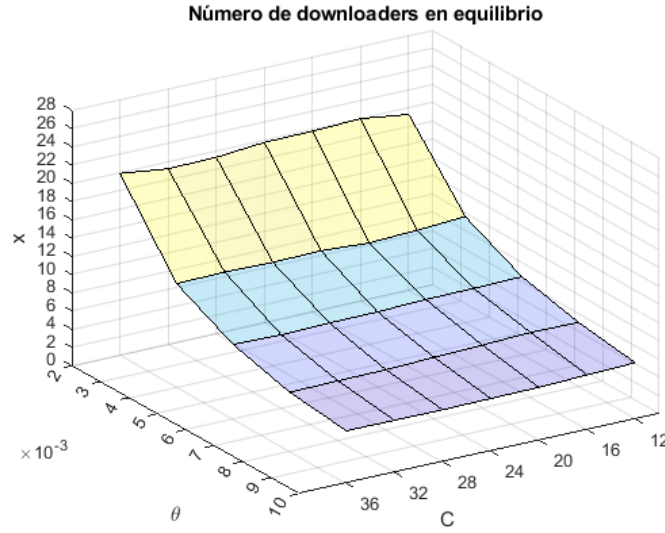


Figura 36. Tiempo promedio de descarga ventana 0

Con base en la nueva ecuación para la distribución de recursos de los servidores *CDN* se obtuvo nuevamente la gráfica de las poblaciones promedio de *downloaders* en un sistema *livestreaming*.

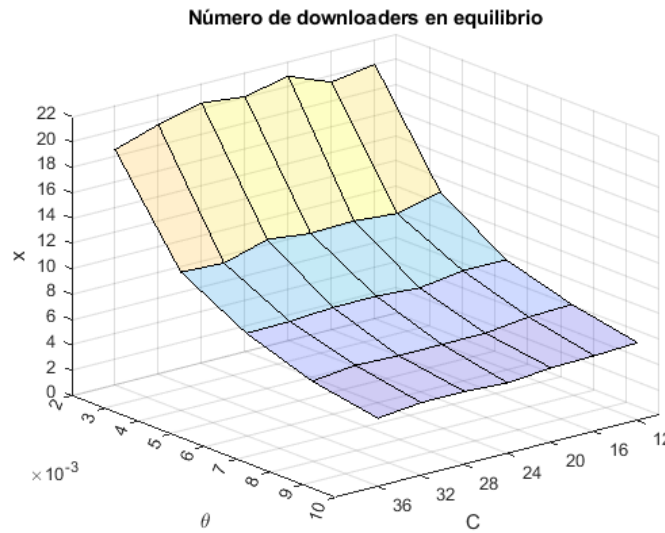


*Figura 37. Poblaciones promedio de downloaders con nueva expresión*

La Figura 37 representa las poblaciones promedio de *downloaders* en el sistema *live streaming*, considerando que los recursos de servidores  $\mu_s$  se distribuyen de acuerdo con la nueva ecuación.

El esquema de asignación de recursos, que ahora se emplea es:

$$\tau_i = \min \left\{ C_\omega X_i, X_i \left( \sum_{k=i+1}^C \frac{X_k \mu_\omega}{\sum_{j=0}^{k-1} X_j} + \left( \frac{1}{1 + 2^i} \right) * \mu_s \right) \right\}$$



*Figura 38. Downloaders promedio considerando el avance los usuarios en la ventana C*

En esta etapa de la implementación se observó que a los usuarios ubicados en la ventana C tenían la posibilidad de ser transferidos a una ventana superior inmediata, lo cual es erróneo debido a que si aún no se produce una nueva

ventana que entre a la hiperventana estos usuarios no pueden avanzar a una ventana más allá de los límites de la hiperventana.

Para contrarrestar este suceso, se optó por definir una variable  $\tau_c$ , que indica la transferencia de los *peers* a la ventana inferior inmediata en vez de que avancen hacia la superior inmediata (inexistente).

$$\tau_c = P_\omega X_i - \tau_c$$

Donde:

$\tau_c$ : Representa la transición de los *peers* en la ventana C

$\tau_c$ : Representa la tasa de transferencia de los *peers* en la ventana c a la ventana inmediata superior

$P_\omega X_i$ : Representa la tasa de entrada de una ventana a la hiperventana

Si  $\tau_c$  es positiva los usuarios en la ventana C son transferidos a la ventana inferior inmediata, en caso contrario, cuando  $\tau_c$  es negativa no existe transición y los *peers* en la ventana C permanecen ahí.

En la Figura 38 se representa a las poblaciones promedio de *downloaders* en el sistema *livestreaming*, considerando la nueva distribución de  $\mu_s$  y considerando la transición en la ventana C. La cual establece que, en vez de moverse a una ventana superior, las poblaciones quedan igual y únicamente aumenta el tiempo de simulación o bien son retrocedidos a la ventana inferior inmediata e igualmente aumenta el tiempo de simulación.

Anteriormente, en la Figura 34 se obtuvo el ancho de banda total consumido en el sistema a continuación en la Figura 39 se muestra el ancho de banda consumido por las poblaciones de *peers* servidores *CDN*.

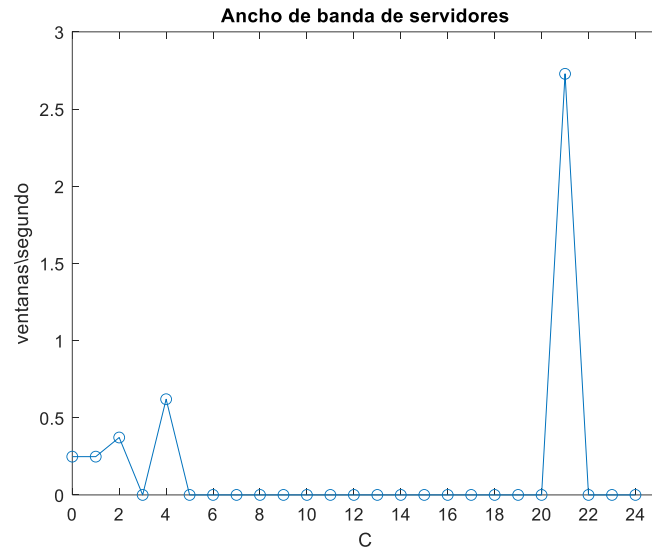


Figura 39. Ancho de banda por iteración proveniente de los servidores.

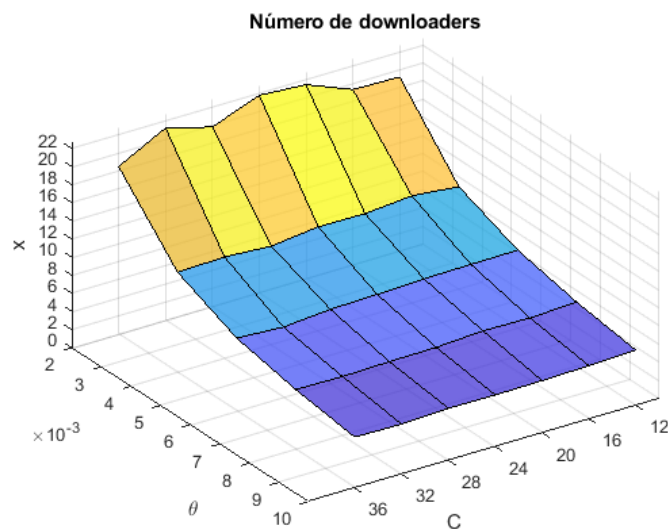


Figura 40.. Downloaders promedio en el sistema livestreaming

En la Figura 40 se muestran nuevamente las poblaciones promedio de *peers* dentro del sistema *livestreaming* pero ahora considerando la transferencia de los *peers* en la ventana C.

Hasta este punto de la implementación se siguió el modelo presentado en el capítulo 5 sección 5.1, sin embargo, se discutieron algunas inconsistencias que presentaba el modelo de servicios de video en vivo, fueron analizadas y se estableció una solución que de acuerdo con la experiencia adquirida a lo largo de la implementación indican un acercamiento más próximo al comportamiento que muestra un sistema de distribución de video.

Estos criterios, fueron discutidos y analizados con base en resultados obtenidos en trabajos reportados en la literatura referentes a la distribución de video bajo demanda.

## 6.1 Rediseño de la cadena de Markov para servicios de video en vivo

En esta sección se describen las modificaciones realizadas al modelo descrito en el capítulo de diseño. En la Figura 20 se muestra el modelo para servicios de video en vivo, sin embargo, dicho modelo presenta inconsistencias que fueron descubiertas una vez que se realizó la solución por implementación de la cadena de Markov que representa servicios de video en vivo.

Las inconsistencias que se encontraron fueron:

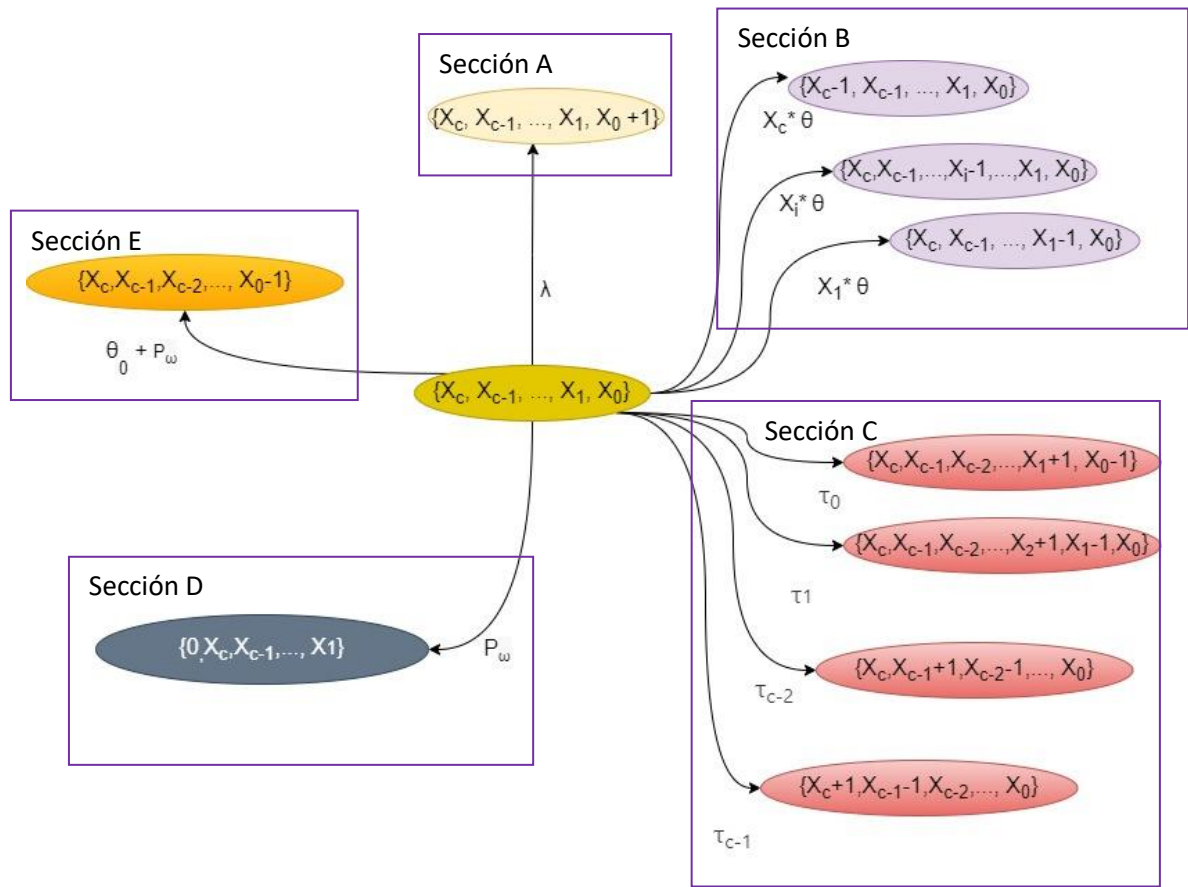
- La desconexión de los *peers* que descargan la ventana cero.
- La transferencia de *peers* a la ventana inmediata inferior
- La transferencia de *peers* que descargan la ventana *c* a la ventana superior

Tanto la desconexión de *peers* que descargan la ventana cero como la transferencia de *peers* que descargan la ventana *c* a la ventana superior inmediata son tomados como casos bordes. Esto debido a que se encuentran en los límites de contención de la hiperventana.

Por la naturaleza de la hiperventana que se ha definido en este proyecto, ningún *peer* conectado al sistema de transmisión de video en vivo puede salir de los límites de la hiperventana, debido a que se considera como una salida del sistema.

En cuanto a la transferencia de *peers* a la ventana inmediata inferior, el desplazamiento (disminución en una unidad de la ventana que descarga el incremento en una unidad de la ventana inferior) es un concepto que se había manejado erróneamente. Esto debido a que, en los sistemas de distribución de contenido en tiempo real, al producirse una nueva ventana del video, las poblaciones que se encuentran descargando cualquier ventana de video son desplazadas simultáneamente, en vez de desplazar la de una ventana en específico únicamente. Además, las poblaciones son desplazadas en su totalidad en vez de desplazar a un solo *peer* de esa ventana como se había realizado en el modelo de la Figura 20.

A continuación, se describen a detalle los cambios realizados y en la Figura 42 se establece el modelo modificado que representa a los servicios de video en vivo.



*Figura 41. Replanteamiento de la cadena de Markov de un sistema de transmisión de video en vivo*

En la Figura 42, se observan los sucesos que generan una transición en el estado de la cadena y el resultado de esta.

En el centro de la Figura 42, se observa al vector  $\{X_c, X_{c-1}, \dots, X_1, X_0\}$ , que representa al estado de la cadena de Markov para servicios de video en vivo. Este vector contiene las poblaciones de *peers* en cada una de las ventanas contenidas en la hiperventana de un archivo de video (desde la ventana 0 hasta la ventana  $c$ ).

Este vector ilustra un estado general de la cadena de Markov, es decir, la cadena se encuentra en este estado en cualquier instante de la transmisión de video en vivo. Cada una de las poblaciones  $X_i$ , contenidas en este vector estado, tiene un valor aleatorio.

En el sistema a modelar para el proyecto se considera que los sucesos son discretos, es decir, ocurre un único evento a la vez (conexión de un usuario, transferencia de un *peer* a la ventana superior inmediata, transferencia de un *peer* a la ventana inferior inmediata o desconexión de un *peer* antes de terminar la transmisión).

A continuación, se describen los sucesos que producen un cambio en alguna población de *peers* perteneciente a una ventana del archivo de video y por ende el estado de la cadena de Markov transita.

Conexión de un nuevo usuario al sistema (sección A): Una vez que inicia la transmisión de video en vivo, diversos usuarios se pueden conectar a la transmisión para visualizar la distribución de contenido en tiempo real. Dicha conexión se realiza a tasa  $\lambda$ , que representa la tasa de conexión de un usuario en general del sistema.

En este proyecto se supone que el usuario se debe conectar a la ventana 0 de la hiperventana para comenzar a descargar el video, crear *buffer* y así evitar congelamientos en la descarga del contenido.

Al conectarse un usuario al sistema, el estado general de la cadena de Markov transita del estado  $\{X_c, X_{c-1}, \dots, X_1, X_0\}$  al estado  $\{X_c, X_{c-1}, \dots, X_1, X_0 + 1\}$ .

Transferencia de un *peer* a la ventana inmediata superior (sección C): Cualquier población de *peers* correspondiente a una ventana contenida en  $[0, c - 1]$  se modifica al generarse la transferencia de un *peer* a la ventana superior inmediata, esta transferencia ocurre a tasa  $\tau_i$ , es decir, una vez que un usuario termina de descargar la ventana  $i$  y comienza a descargar la ventana  $i + 1$ , abandona la población  $X_i$  a tasa  $\tau_i$  y se agrega a la población en la ventana  $i + 1$  ( $X_{i+1}$ ).

Debido a la naturaleza de los videos en vivo y la definición de hiperventana que se introdujo para el desarrollo de este proyecto, un *peer* que está descargando la ventana  $c$ , al finalizar la descarga de esta, no puede ser transferido a una ventana superior inmediata; porque no hay disponible otra ventana para descargar. Es decir, la población de *peers* que se encuentra descargando la ventana actual deben esperar a que se produzca otra ventana del video y esta entre a la hiperventana para poder descargarla.

Una vez que la nueva ventana es producida e ingresa a la hiperventana, de manera automática la población  $X_c$  ahora es la población  $X_{c-1}$ . Y, por lo tanto, pueden ser trasferidos a la ventana superior inmediata y comenzar la descarga de la ventana actual nuevamente.

El suceso de transferencia a la ventana superior inmediata provoca una transición en el estado de la cadena de Markov como se describe a continuación:

$$\{X_c, X_{c-1}, \dots, X_{i+1}, X_i, \dots, X_0\} \rightarrow \{X_c, X_{c-1}, \dots, X_{i+1} + 1, X_i - 1, \dots, X_0\}$$

$\tau_i$  : Representa la tasa promedio de transferencia de la ventana  $i$ . Y se define con la siguiente expresión:

$$\tau_i = \min\{C_\omega X_i, r_i\} \text{ para } i \in [0: c - 1]$$



Donde:

$C_\omega$ : Es la tasa de descarga general de un usuario dentro del sistema.

$r_i$ : Representa a los recursos de descarga efectivos en penuria para la ventana  $i$ , es decir, cuando la tasa de descarga es mayor que la tasa de subida el sistema entra en penuria y debe obtener recursos de la red *CDN*. Esta expresión depende del esquema de asignación de recursos, por ejemplo, el esquema de distribución uniforme que se retomará con más detalle en la siguiente sección.

Transferencia de un *peer* a la ventana inmediata inferior (sección D): Cualquier población de *peers* correspondiente a una ventana contenida en  $[1, c]$  se modifica al generarse la transferencia de un *peer* a la ventana inferior inmediata a tasa  $P_\omega X_i$ , es decir, cuando un usuario por diversas razones (fallas en sus servicios, fallas en sus dispositivos, etc.) deja de descargar el video en vivo a la misma tasa que se está capturando el evento en tiempo real y produciendo una nueva ventana del video, por lo tanto el *peer* se atrasa en la descarga del video y pasa de la ventana  $i$  (abandona la población  $X_i$ ) a la ventana  $i - 1$  (se adiere a la población  $X_{i-1}$ ).

Por la naturaleza de los videos en vivo y la definición de hiperventana, un *peer* que está descargando la ventana 0 si interrumpe su proceso de descarga, no puede ser transferido a una ventana inferior inmediata; porque esto provoca que salga de la hiperventana y por lo tanto abandone el sistema. Este caso en particular representa un factor para generar la desconexión de un usuario en la ventana, por ello, su efecto en el estado de la cadena de Markov se hace en conjunto a la desconexión de un usuario en la ventana 0, que se describirá más adelante.

El suceso de transferencia a la ventana inferior inmediata provoca una transición en el estado de la cadena de Markov como se describe a continuación:

$$\{X_c, X_{c-1}, \dots, X_i, X_{i-1}, \dots, X_0\} \rightarrow \{0, X_c, X_{c-1}, \dots, X_i, X_1\}$$

Desconexión de un *peer* antes de terminar la transmisión (sección B): Por causas diversas (fallas de conexión, fallas en sus servicios, fallas en sus dispositivos, desinterés en el contenido, etc.) un *peer* que está visualizando la transmisión de video en vivo puede desconectarse.

Entonces, cualquier población de *peers* de la cadena cambia cuando se genera la desconexión de un *peer* a tasa  $X_i \theta$ , antes de que finalice la transmisión en vivo.  $X_i \theta$  representa la tasa promedio de desconexión de la población que se encuentra descargando ventana  $i$ . Es el resultado de multiplicar la población de la ventana  $i$  ( $X_i$ ) por  $\theta$  (tasa de desconexión de un *peer* conectado al sistema en general).

Este suceso provoca una transición en el estado de la cadena de Markov como se describe a continuación:

$$\{X_c, X_{c-1}, \dots, X_i, \dots, X_0\} \rightarrow \{X_c, X_{c-1}, \dots, X_i - 1, \dots, X_0\}$$

Lo anterior quiere decir que un *peer* que formaba parte de la población  $X_i$  por alguna razón abandono el sistema antes de finalizar la transmisión en vivo, por lo tanto, la población de la ventana  $i$  se decrementa en 1.

Los *peers* conectados a la ventana 0 son más susceptibles a desconectarse del sistema, en caso de interrumpir su proceso de descarga, se generó una nueva ventana del video, por una mala conexión a internet, problemas con el hardware, no tener óptimas *QoE* ó *QoS*. Por lo tanto, se considera a  $\theta_0$  como la tasa de desconexión de los *peers* pertenecientes a la población  $X_0$  y se define como:

$$\theta_0 = \theta + P_\omega$$

Donde:

$\theta$ : Representa la tasa de desconexión general de un usuario dentro del sistema

$P_\omega$ : Representa la tasa a la cuál es producido el archivo de video.

Una vez diseñada la cadena de Markov y definidos los sucesos que generan un cambio en su estado, se diagramo la solución matemática a implementar para simular la ocurrencia de los sucesos y la transición de estado en la cadena.

Una vez diseñada la cadena de Markov y definidos los sucesos que generan un cambio en su estado, se diagramo la solución matemática a implementar para simular la ocurrencia de los sucesos y la transición de estado en la cadena.

Desconexión de un *peer* de la población  $X_0$  (sección E): este caso se encarga únicamente de las desconexiones de los *peers* que se encuentran en la población  $X_0$  y ocurren a tasa  $\theta_0 + P_\omega$ .

**Nota:** Agregar gráficas de poblaciones para el cambio propuesto por Mario

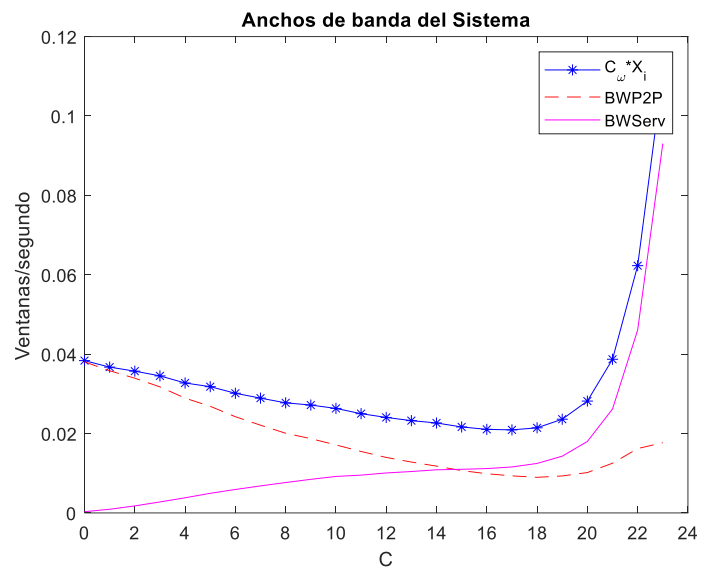
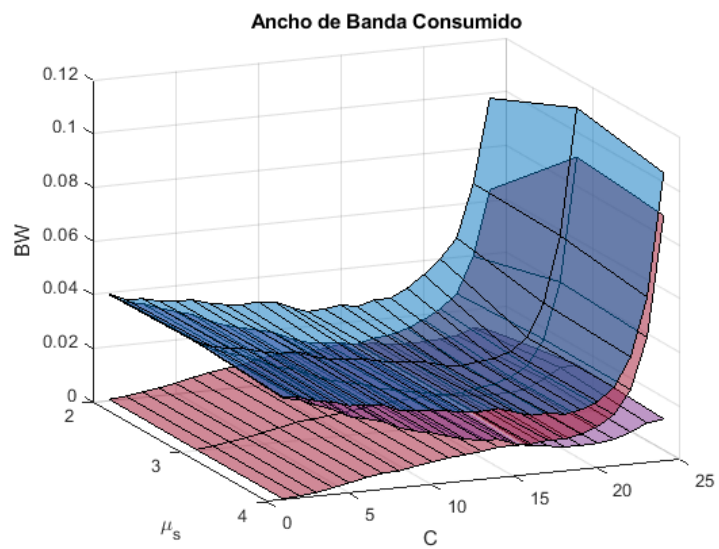
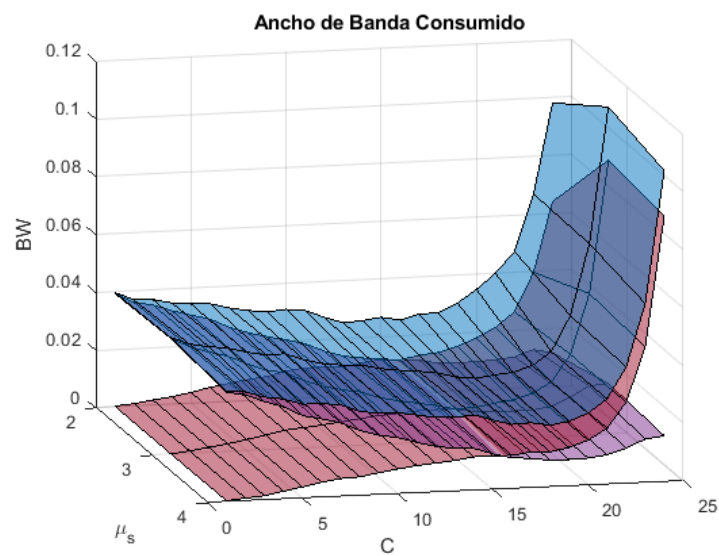
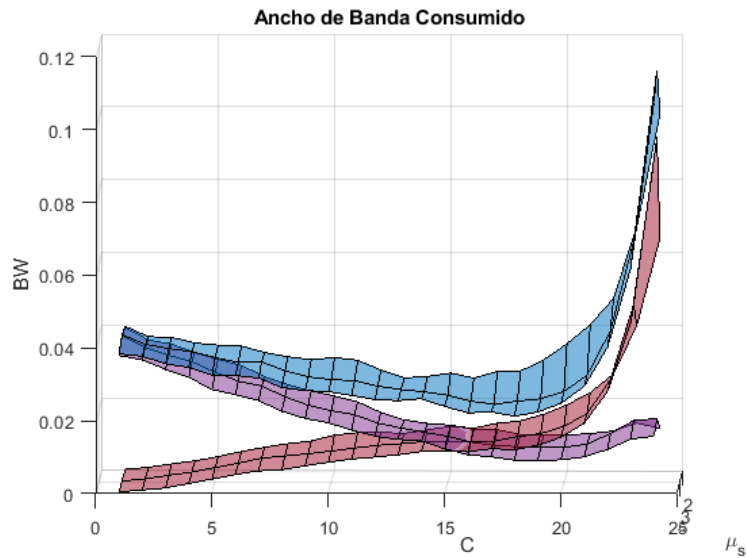


Figura 42. Ancho de Banda consumido en la descarga de video





Como se había mencionado anteriormente este trabajo tiene como objetivo realizar adecuaciones a esquemas de asignación de recursos implementados en sistemas de distribución de video bajo demanda con la finalidad de proponer un esquema de asignación de recursos para los servicios de video en vivo.

Inicialmente se consideró al esquema de asignación de recursos de distribución uniforme para observar el comportamiento de un sistema *livestreamig*. Una vez observados, analizados e interpretados los resultados de poblaciones promedio de *peers* por ventana dentro de la hiperventana y anchos de banda consumidos total, así como de las redes *CDN* y *P2P* se realizó el cambio de esquema de asignación de recursos.

En la sección de diseño se habló de adecuar el esquema de asignación de recursos Q ventanas hacia atrás o el esquema de asignación de recursos DVPG. El esquema más relacionado a distribución uniforme y que por tanto proporciona un panorama más cercano para poder comparar resultados es el esquema Q

ventanas hacia atrás. Por lo tanto, se decidió realizar adecuaciones a este esquema de asignación de recursos para obtener nuevamente poblaciones promedio de *peers* y anchos de banda consumidos. Una vez obtenidos dichos recursos, se comparan y contrastan contra los obtenidos mediante el esquema de distribución uniforme.

El principal objetivo de esta actividad es observar cuál de los dos esquemas de asignación de recursos adecuados para servicios de video en vivo, muestra un menor consumo de recursos de la red *CDN*.

En el esquema Q ventanas hacia atrás se introduce un nuevo parámetro llamado Q, el cual indica la cantidad de ventanas superiores que atenderán a una ventana en específico, es decir, este parámetro acota el número de ventanas a las cuáles una población de *peers* que descarga el video podrá acceder para obtener recursos.

Las expresiones que muestran el esquema de asignación de recursos Q ventanas hacia atrás para servicios de video en vivo, son las siguientes.

$$\tau_i = \min \left\{ C_\omega * X_i, \sum_{k=i+1}^{\min(i+Q,c)} \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j} + \frac{\mu_s * X_i}{\sum_{j=0}^{c-1} X_j} \right\}$$

La expresión anterior establece que el la tasa a la cuál un *peer* será transferido a la ventana superior inmediata es el mínimo entre la tasa máxima de descarga que puede tener esa población  $i$  ( $C_\omega * X_i$ ) y la cantidad de recursos que pueden proporcionar las redes *CDN* y *P2P*  $\left( \sum_{k=i+1}^{\min(i+Q,c)} \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j} + \frac{\mu_s * X_i}{\sum_{j=0}^{c-1} X_j}, 0 \leq i \leq c-1 \right)$ .

Al igual que con el esquema de asignación de recursos de distribución uniforme, con Q ventanas hacia atrás el sistema puede operar bajo dos condiciones (abundancia y penuria). Para conocer el ancho de banda total consumido, el ancho de banda consumido de la red *CDN* y el ancho de banda consumido de la red *P2P*, se deben analizar ambos casos:

#### Abundancia:

Es cuando la operación min para conocer la cantidad de recursos necesarios para descargar tiene como resultado:

$$B_d^i = C_\omega * X_i$$

En este caso los recursos requeridos por una población para realizar la descarga son menores a los recursos con los que atienden las redes *P2P* y *CDN* en conjunto. Por lo tanto, los *peers* deben limitarse a descargar a la tasa máxima de descarga. En este caso, se conoce que ese fue el total de ancho de banda consumido en el sistema por una población  $i$ , sin embargo, someramente no se

conoce que fracción de este ancho de banda proviene de la red *P2P* y que fracción proviene de la red *CDN*.

Por lo tanto, para obtener el ancho de banda comido de la red *P2P* y el ancho de banda consumido de la red *CDN* se debe realizar una operación min nuevamente, pero ahora, para obtener la fracción obtenida desde la red *P2P*. Se realiza conforme a la expresión siguiente:

$$B_{u,p}^i = \min \left\{ C_\omega * X_i, \sum_{k=i+1}^{\min(i+Q,c)} \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j} \right\}$$

La expresión anterior establece, que el ancho de banda obtenido de la red *P2P* ( $B_{u,p}^i$ ) va a ser el mínimo entre la tasa máxima de descarga y la cantidad de recursos proporcionada por la red *P2P*.

Caso 1:

$$B_{u,p}^i = C_\omega * X_i$$

En este caso el ancho de banda consumido (tasa máxima de descarga) fue atendido en su totalidad por la red *P2P*, es decir, el resultado de la operación mínima fue  $C_\omega * X_i$ , por lo tanto, se establece que la cantidad de recursos ofertada por los *peers*  $\left( \sum_{k=i+1}^{\min(i+Q,c)} \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j} \right)$  es mayor y por ende los *peers* se limitan a descargar el video con la tasa máxima de descarga.

El ancho de banda consumido de la red *CDN*, se calcula conforme a la siguiente expresión:

$$B_{u,s}^i = C_\omega * X_i - B_{u,p}^i$$

En el caso  $B_{u,p}^i = C_\omega * X_i$  se dice que el sistema es autosuficiente, ya que  $B_{u,s}^i$  tendría un valor de 0 y por lo tanto la red *P2P* es la que satisface toda la demanda, en caso contrario, el ancho de banda consumido de la red *CDN* es la resta de la expresión anterior.

### Penuria:

Es cuando la operación min para conocer la cantidad de recursos necesarios para descargar tiene como resultado:

$$B_d^i = \sum_{k=i+1}^{\min(i+Q,c)} \frac{\mu_\omega * X_k * X_i}{\sum_{j=0}^{k-1} X_j} + \frac{\mu_s * X_i}{\sum_{j=0}^{c-1} X_j}$$

En este caso los recursos disponibles para atender a las poblaciones que están descargando distintas ventanas de la hiperventana son inferiores a la tasa

máxima de descarga a la cual puedes descargar los *peers* dentro del sistema y por lo tanto la red *CDN* y la red *P2P* proporcionan todos los recursos que tienen a su alcance.

Entonces el ancho de banda proporcionado por la red *P2P* y *CDN* se obtiene conforme a las siguientes expresiones:

$$B_{u,p}^i = \sum_{k=i+1}^{\min(i+Q,c)} \frac{\mu_{\omega} * X_k * X_i}{\sum_{j=0}^{k-1} X_j}, \quad 0 \leq i \leq c-1$$

$$B_{u,s}^i = \frac{\mu_s * X_i}{\sum_{j=0}^{c-1} X_j}, \quad 0 \leq i \leq c-1$$

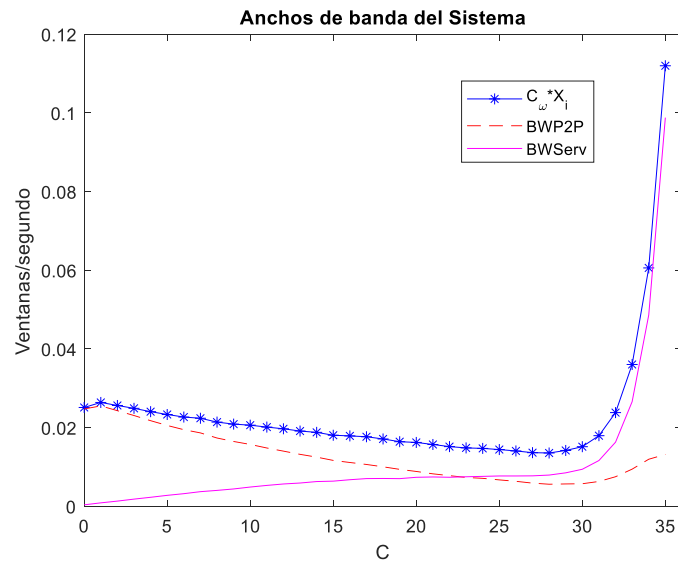
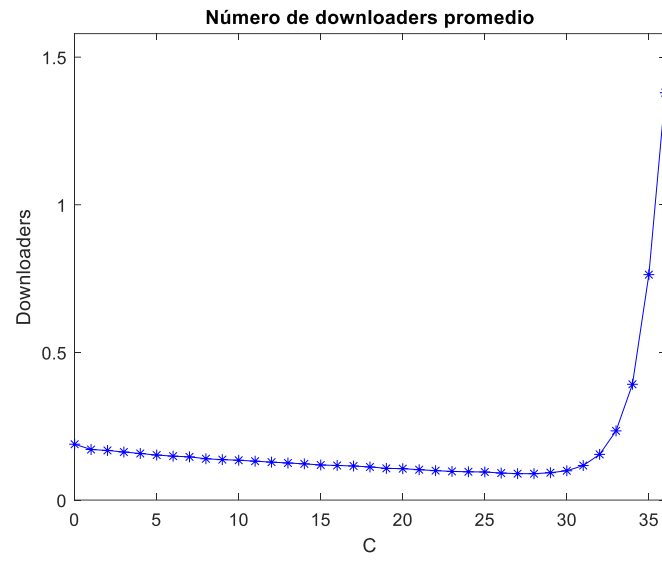
Hasta este momento de las pruebas, el cálculo del ancho de banda consumido se había realizado en cada iteración donde ocurría el suceso de transferencia a la ventana superior inmediata, debido a que este suceso representa el proceso de descarga del video y, por lo tanto, el ancho de banda se consume únicamente cuando los *peers* terminan de descargar una ventana y deben continuar su proceso de descarga con la ventana subsecuente.

El algoritmo desarrollado en el presente proyecto que permite evaluar el modelo por implementación también permite conocer el ancho de banda consumido dentro del sistema, si bien, el objetivo del presente proyecto es que el sistema opere en condición de abundancia. La herramienta desarrollada para evaluar por implementación permite analizar la condición de penuria.

La condición de penuria no es analizada de forma detallada, debido a que se espera la mayor parte del tiempo de simulación, el sistema opere bajo condición de abundancia. Sin embargo, también se reconoce que en alguna de las iteraciones el sistema puede operar en condición de penuria y, por lo tanto, es que se ha mantenido el caso, donde esta condición tiene ocurrencia.

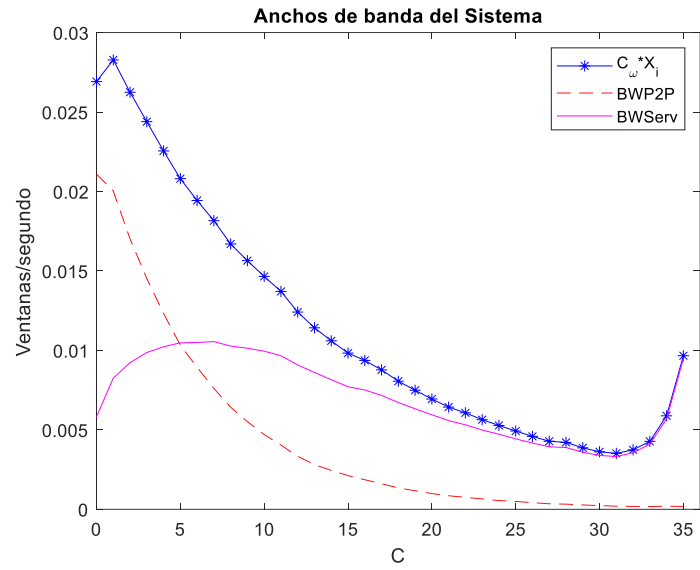
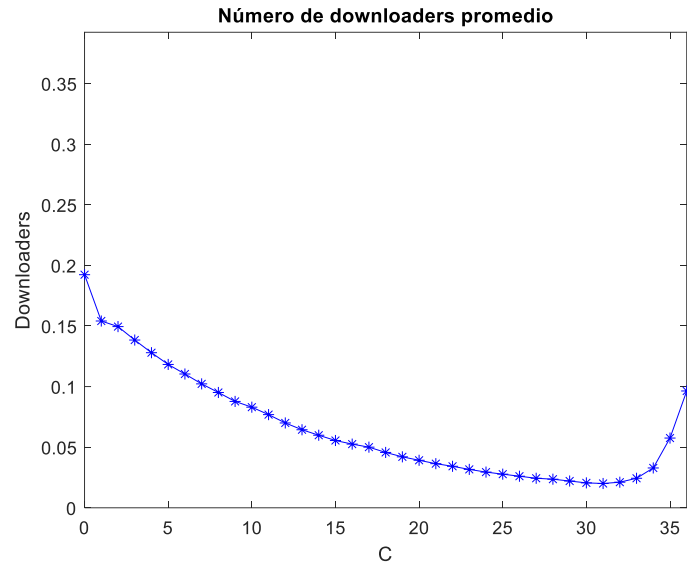
### Resultados Parciales

$$C = 36, \lambda = 0.04, \mu = 0.00255, c = 0.00407, \theta = 2 \times 10^{-3}, \mu_s = 40\mu_\omega$$

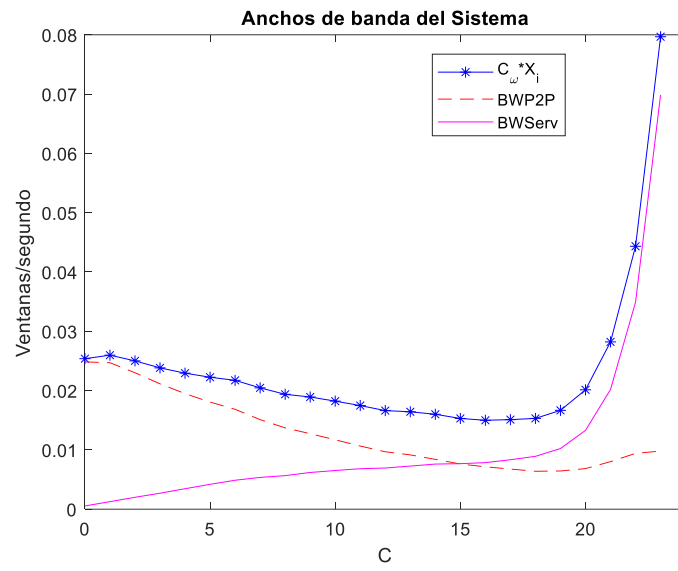
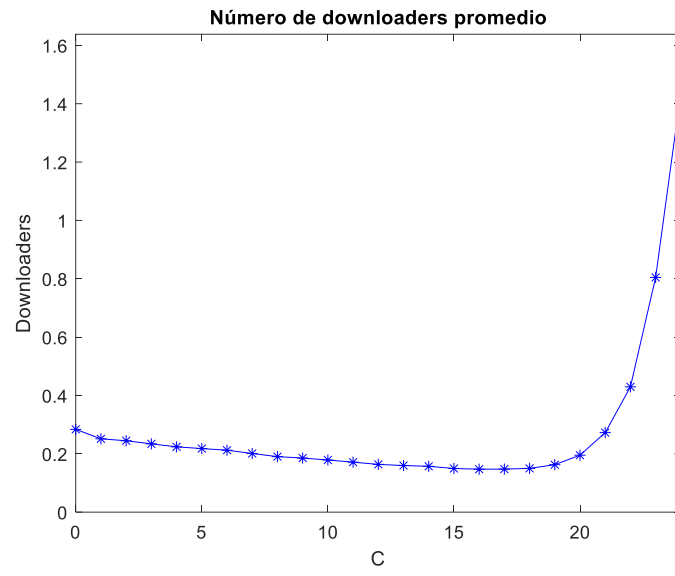


$$C = 36, \lambda = 0.04, \mu = 0.00255, c = 0.00407, \theta = 6 \times 10^{-3}, \mu_s = 40\mu_\omega$$

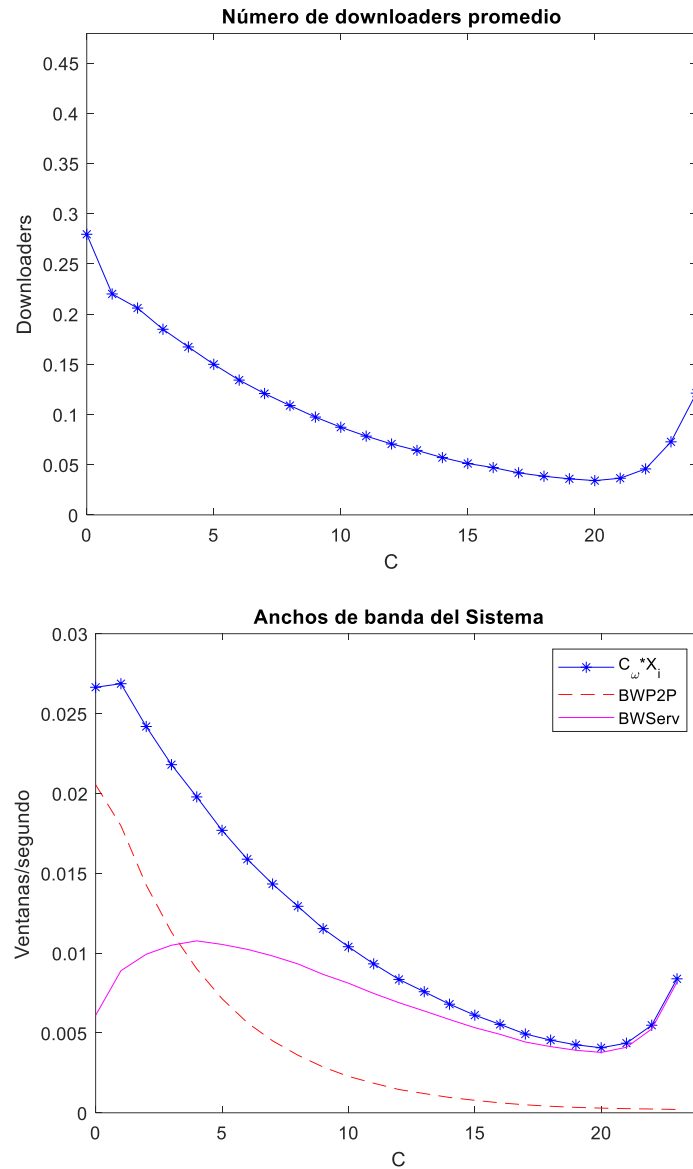




$$C = 24, \lambda = 0.04, \mu = 0.00255, c = 0.00407, \theta = 2 \times 10^{-3}, \mu_s = 40\mu_{\omega}$$



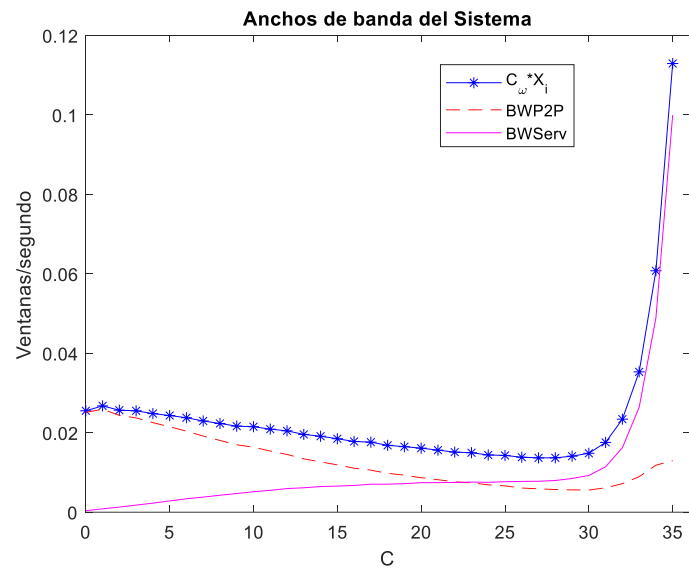
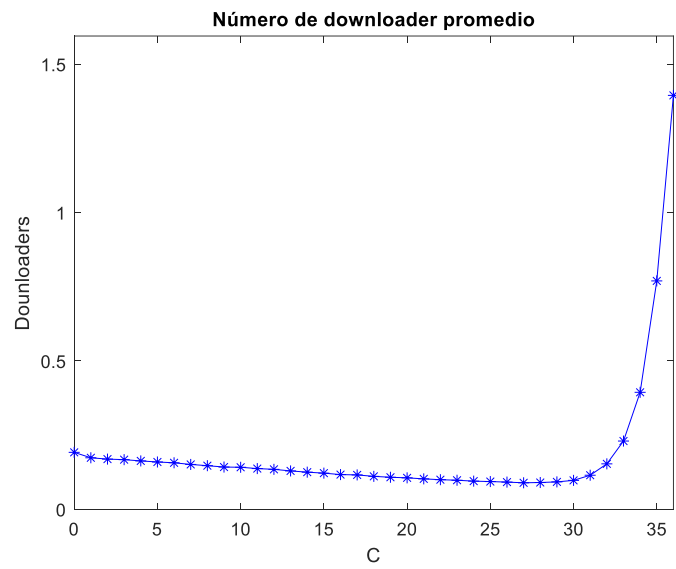
$$C = 24, \lambda = 0.04, \mu = 0.00255, c = 0.00407, \theta = 6 \times 10^{-3}, \mu_s = 40\mu_{\omega}$$



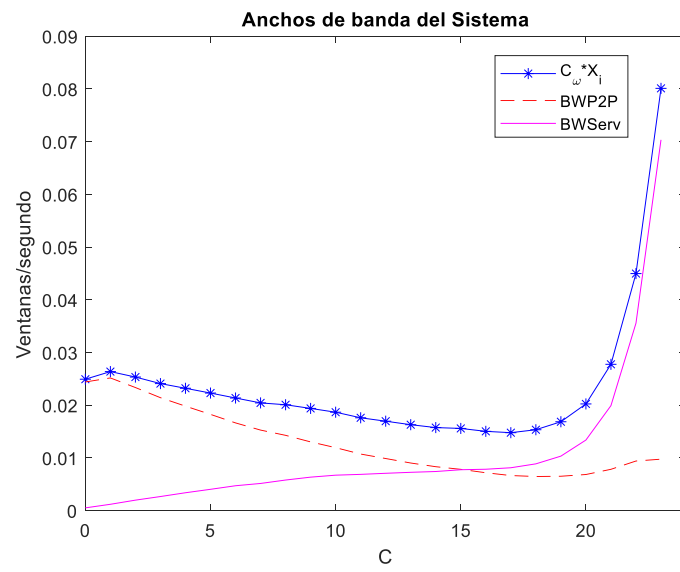
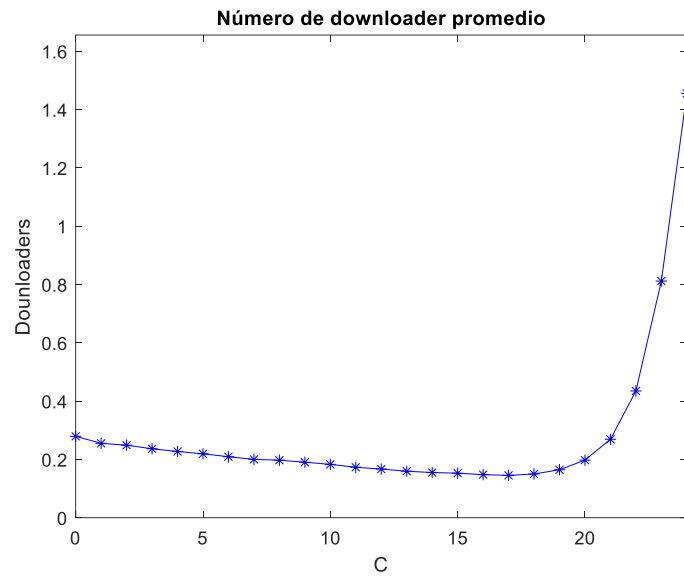
A continuación, se presentan resultados de poblaciones promedio de *peers* por ventanas y anchos de banda consumidos en el sistema al descargar las ventanas de video utilizando el modelo presentado en la Figura 42. En este caso, como se había mencionado anteriormente, una tarea de este proyecto es evaluar el mismo modelo para el esquema de asignación de recursos y una adicional.

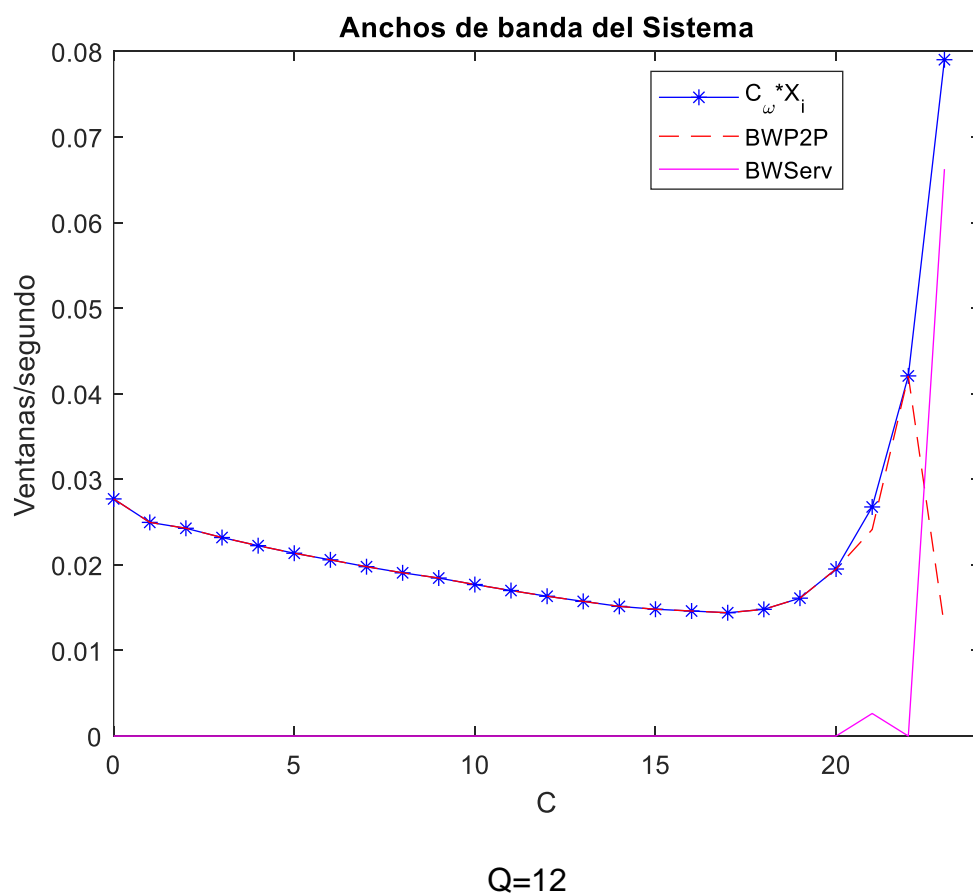
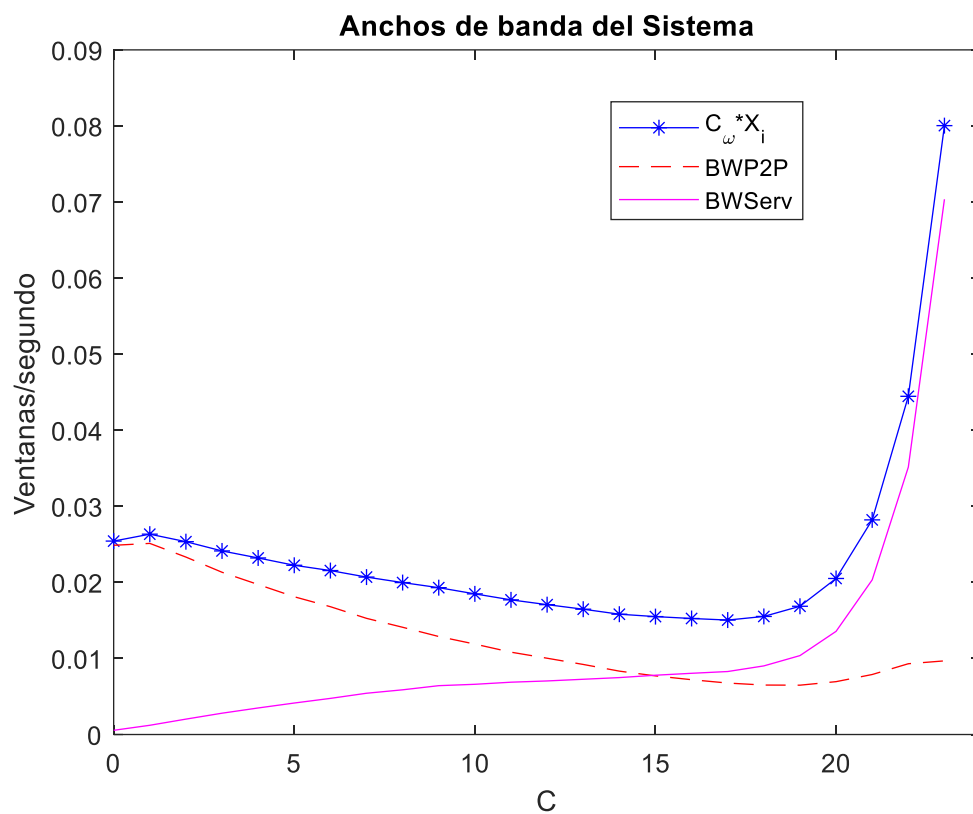
Como parte de esa tarea a partir de ahora, se presentan gráficas obtenidas mediante el esquema de asignación de recursos Q ventanas hacia atrás tomado de [7] y se adecuo para los sistemas de distribución de video en vivo.

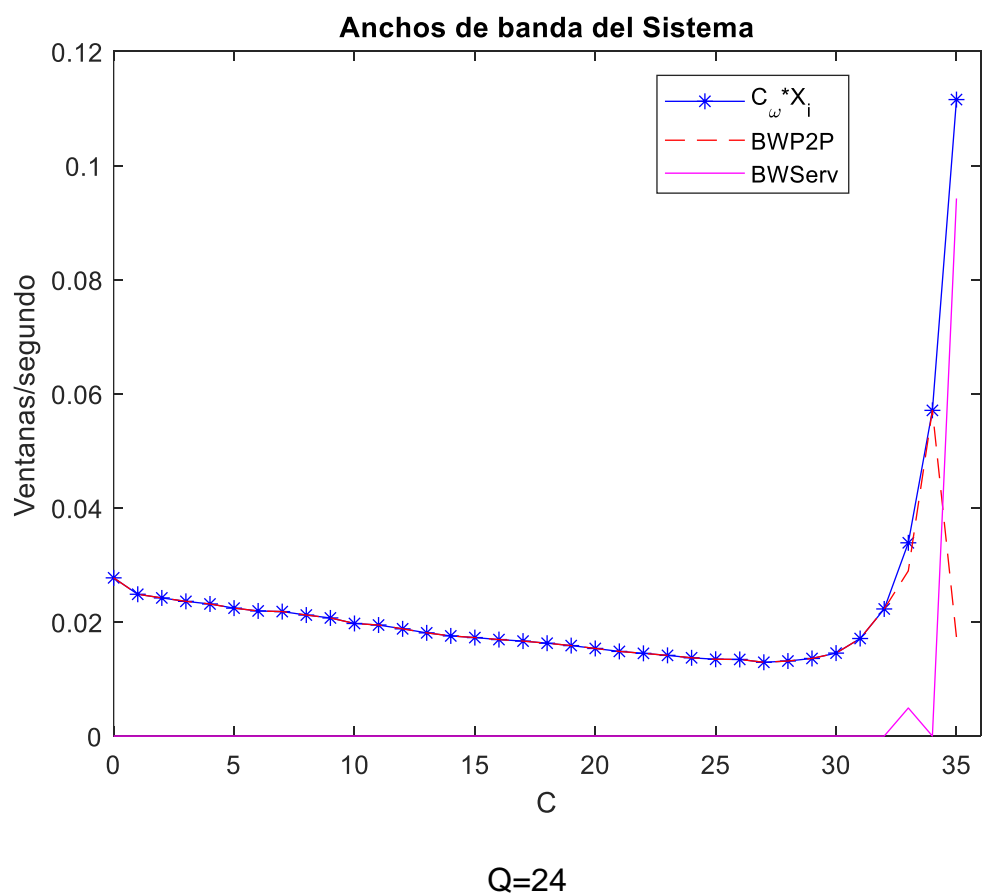
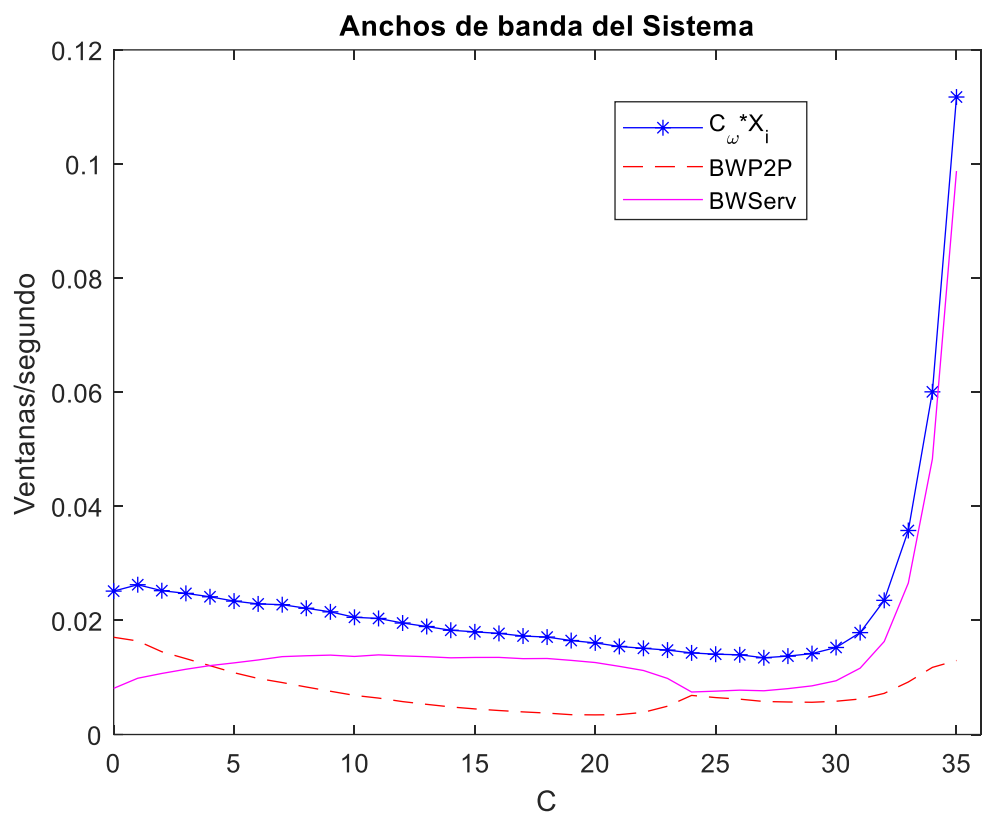
$$C = 36, \lambda = 0.04, \mu = 0.00255, c = 0.00407, \theta = 2 \times 10^{-3}, \mu_s = 40\mu_\omega, Q = 36$$

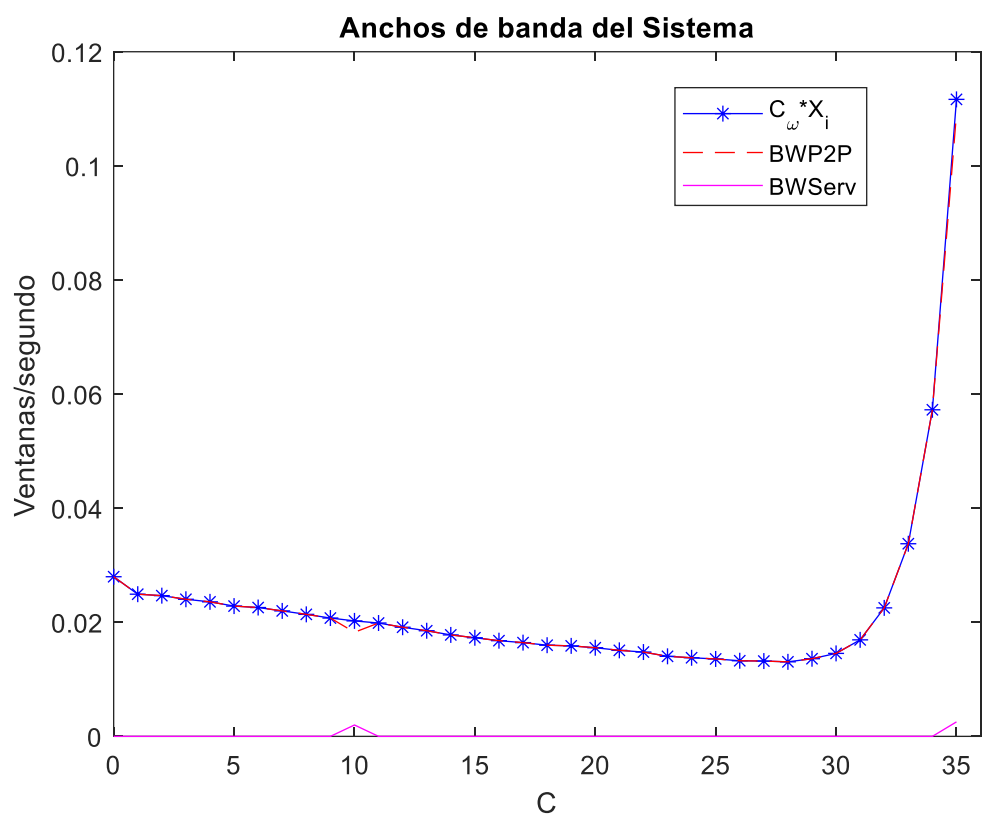
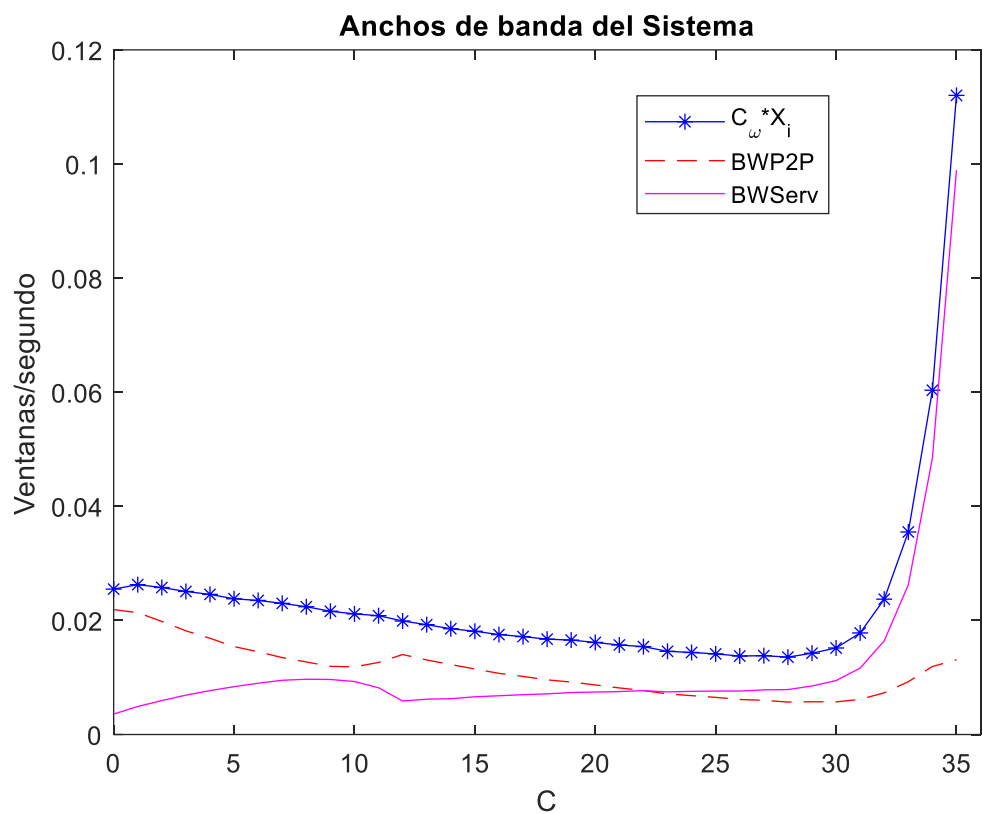


$$C = 24, \lambda = 0.04, \mu = 0.00255, c = 0.00407, \theta = 2 \times 10^{-3}, \mu_s = 40\mu_\omega, Q = 24$$



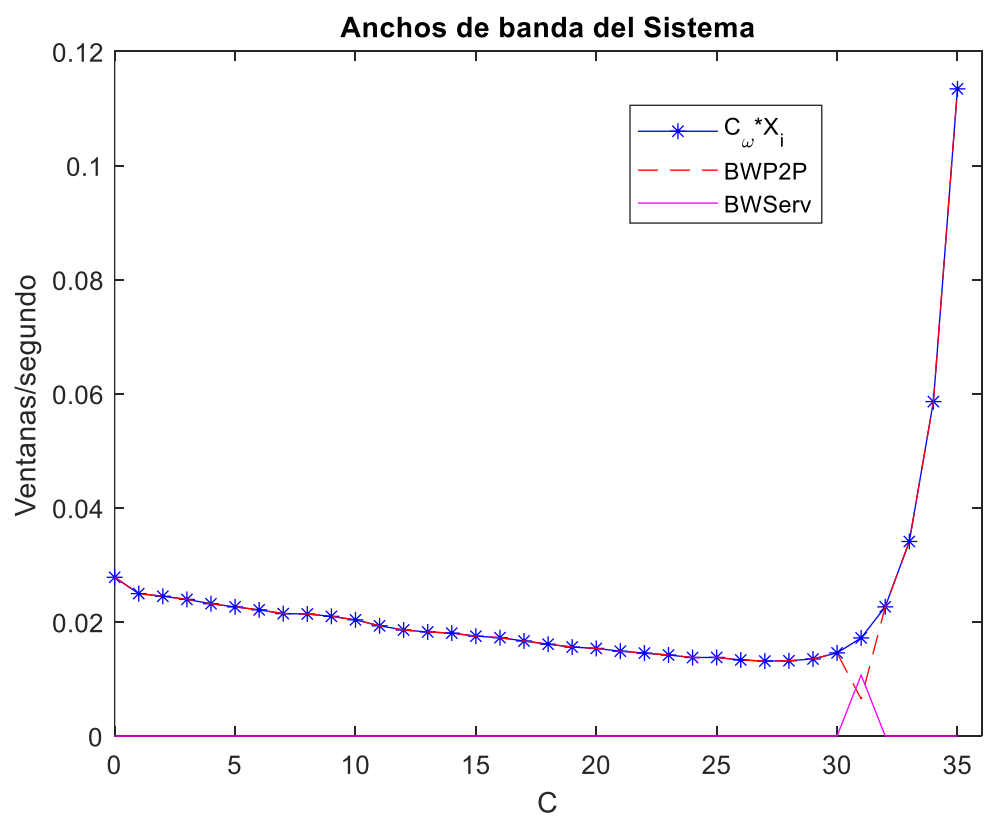
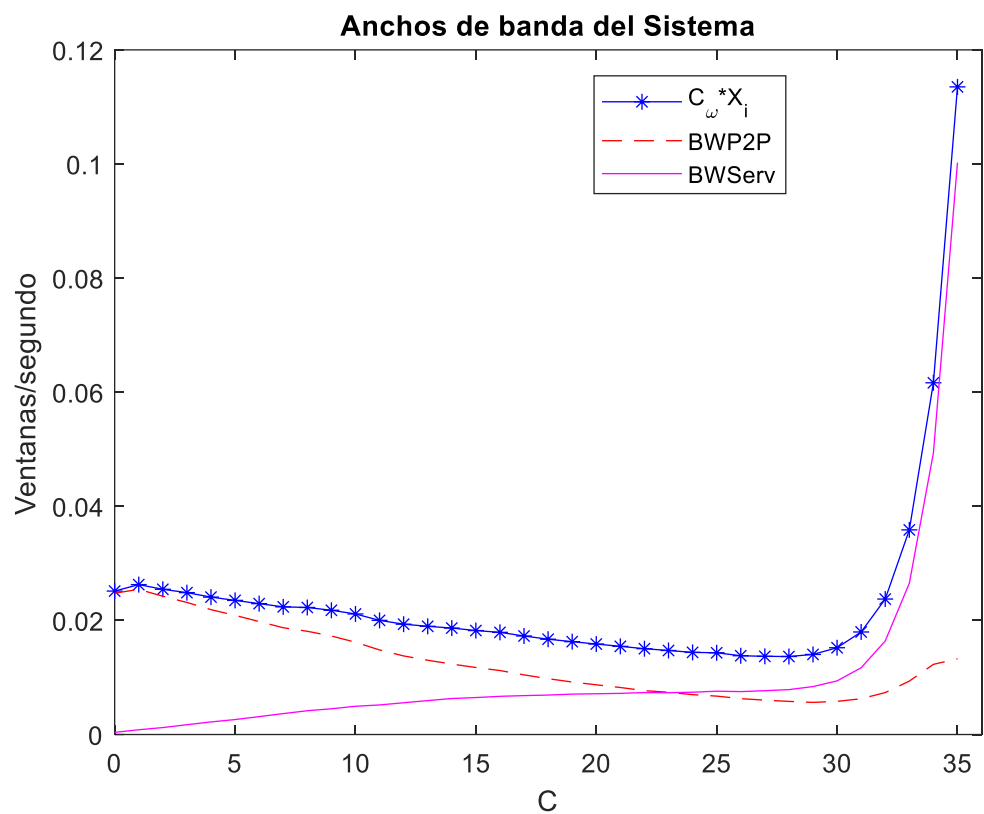




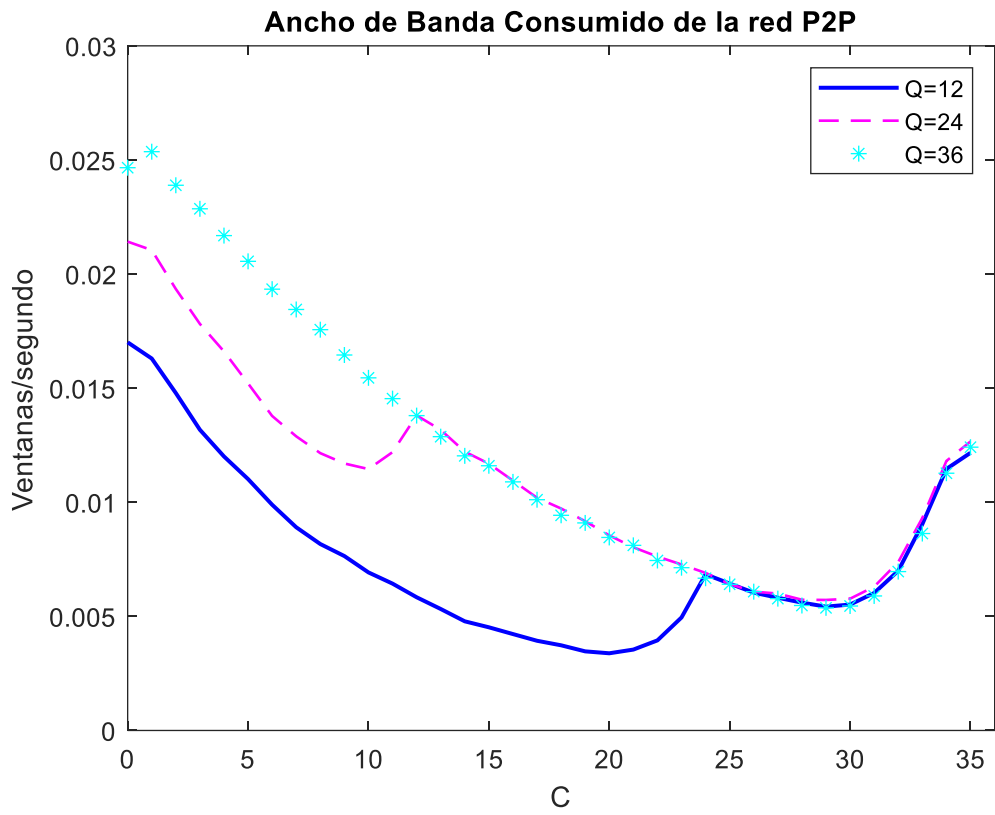
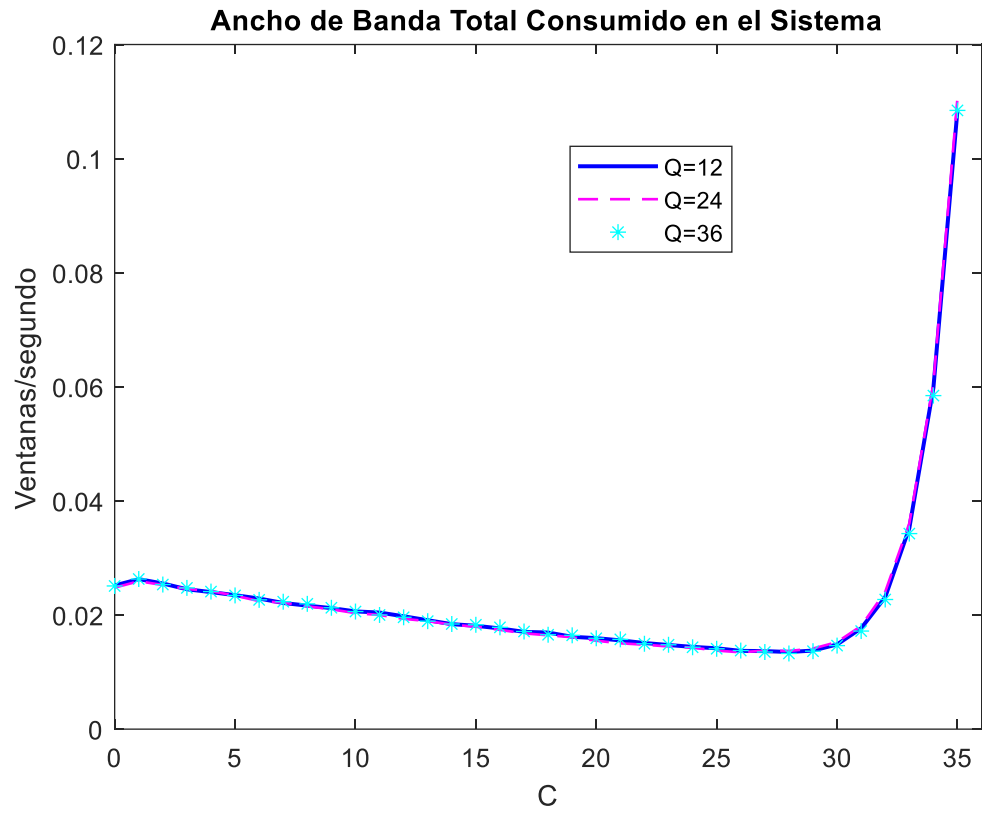


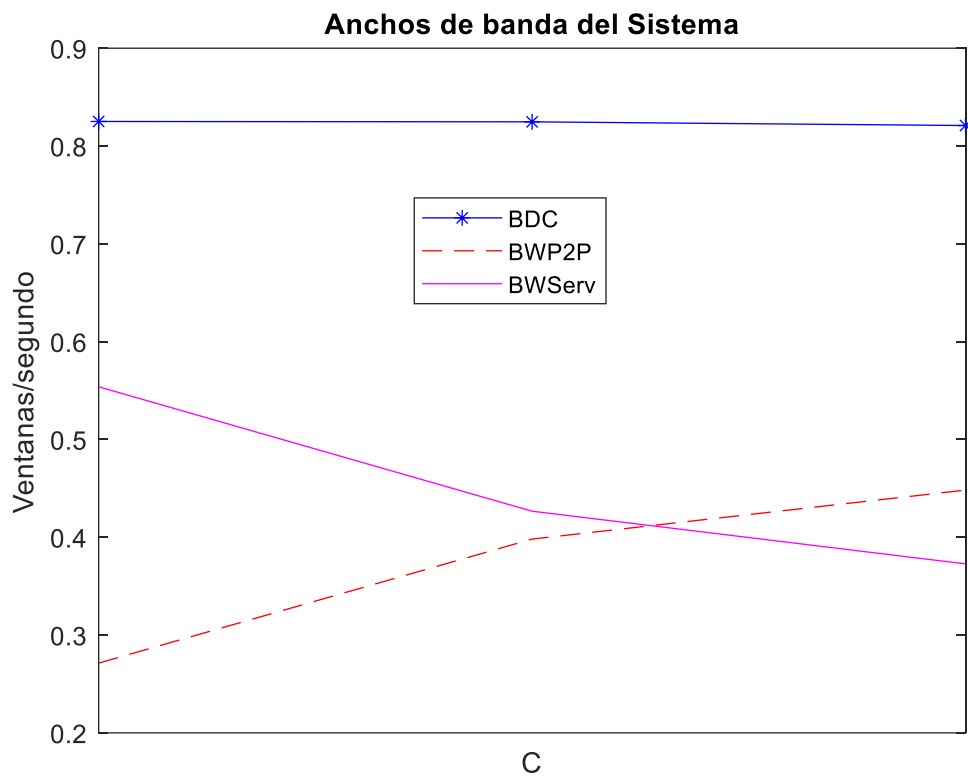
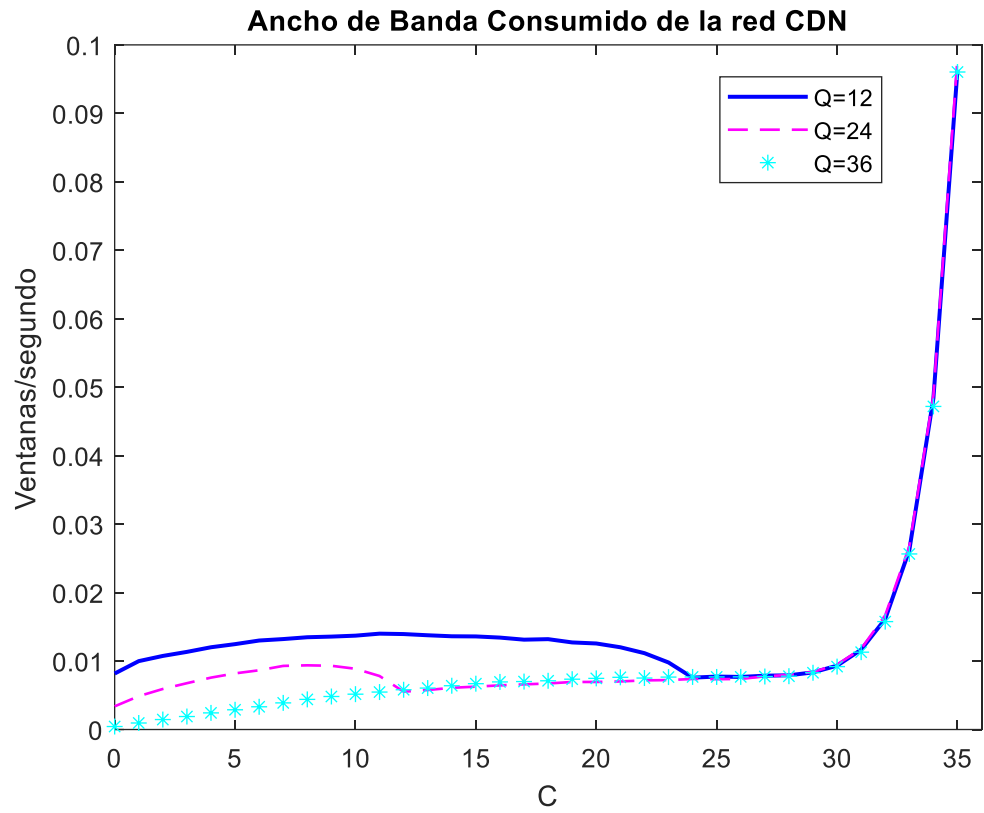
Q=36





## Múltiples Q





# Capítulo 7

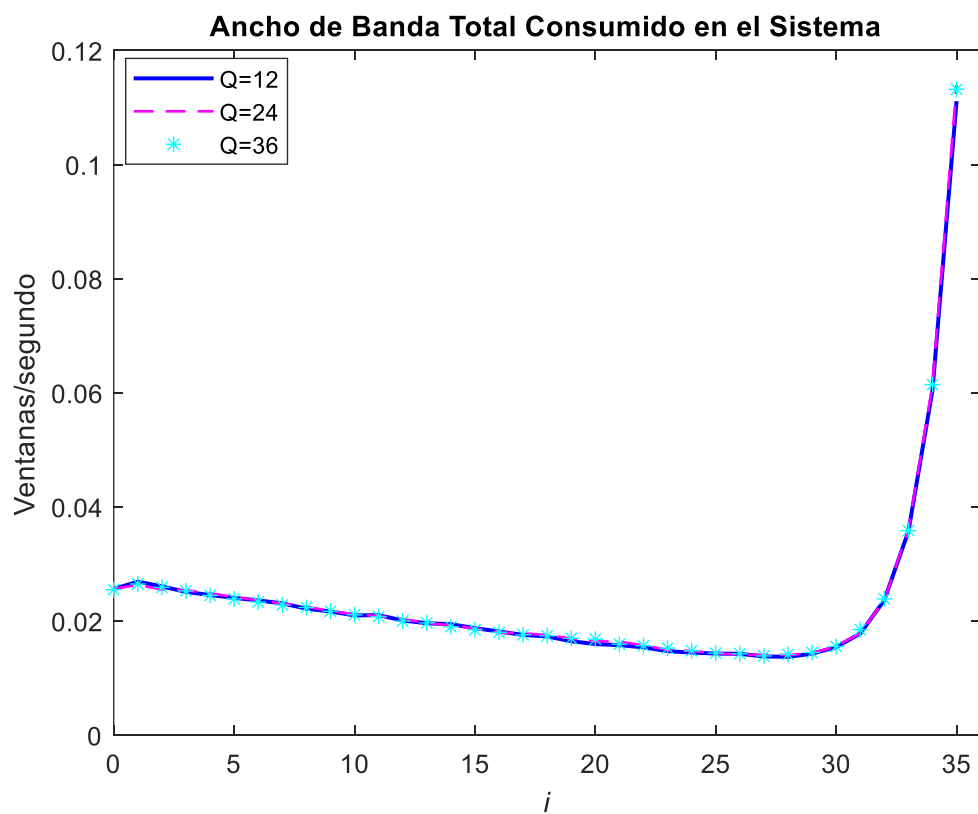
## Resultados

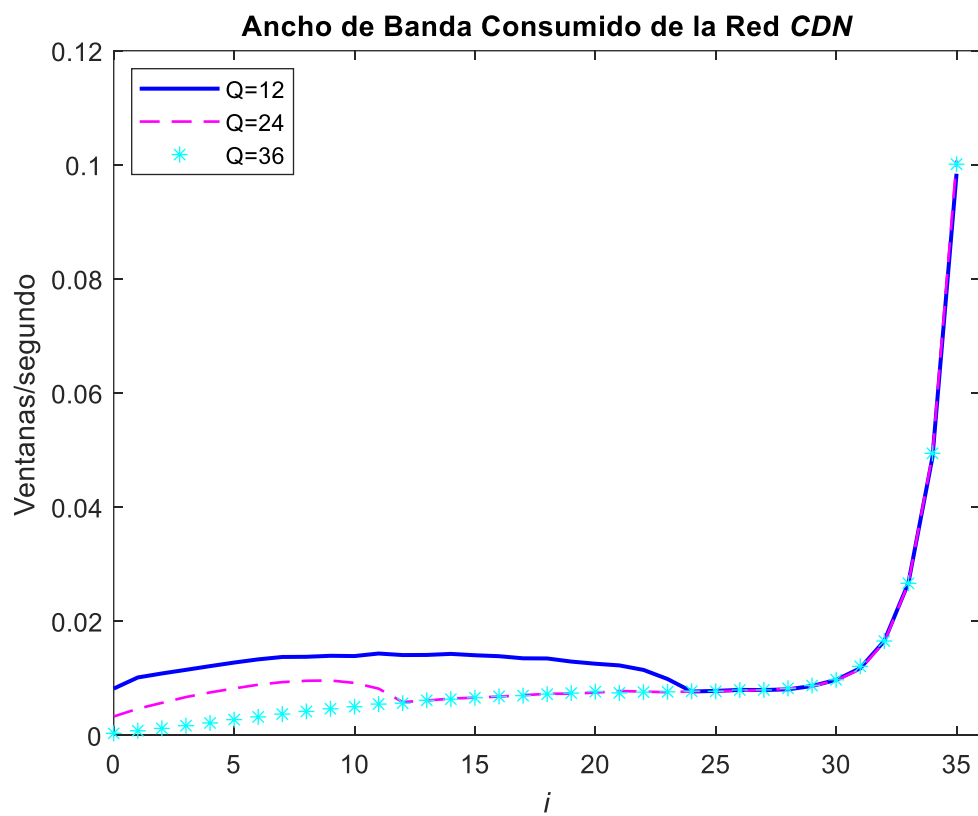
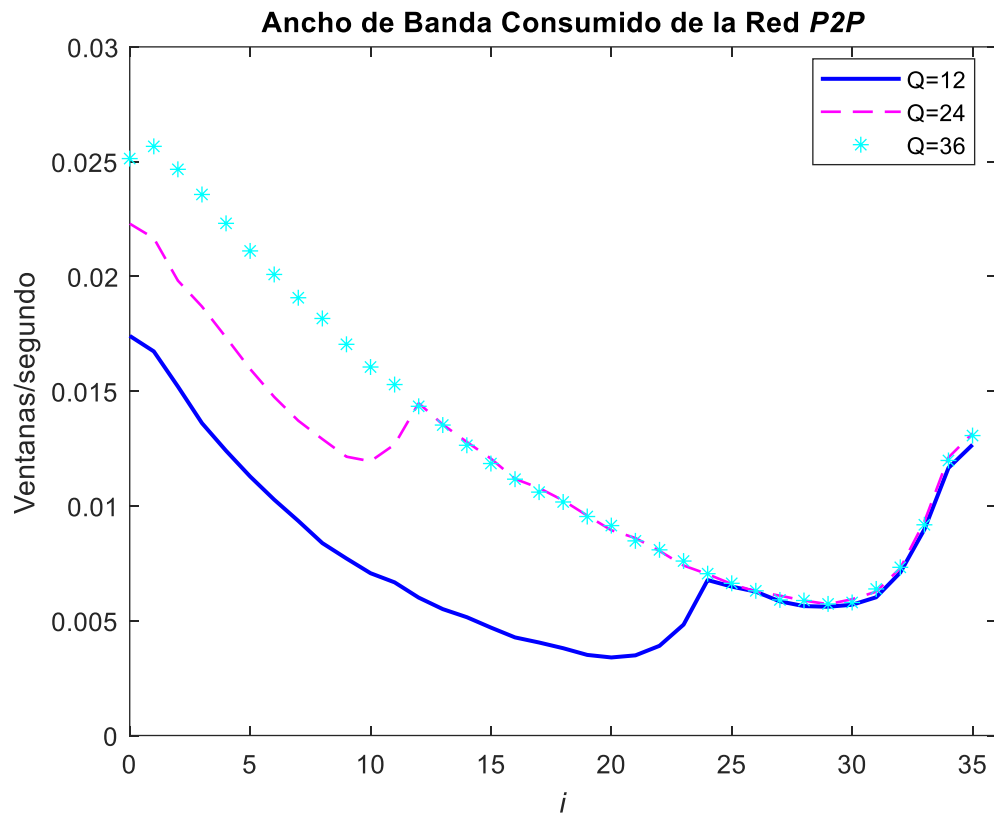
En este capítulo se reportan los resultados finales obtenidos después de realizar cambios y adecuaciones a la implementación. Los resultados mostrados e interpretados en este capítulo tienen una aproximación a resultados reportados en [5].

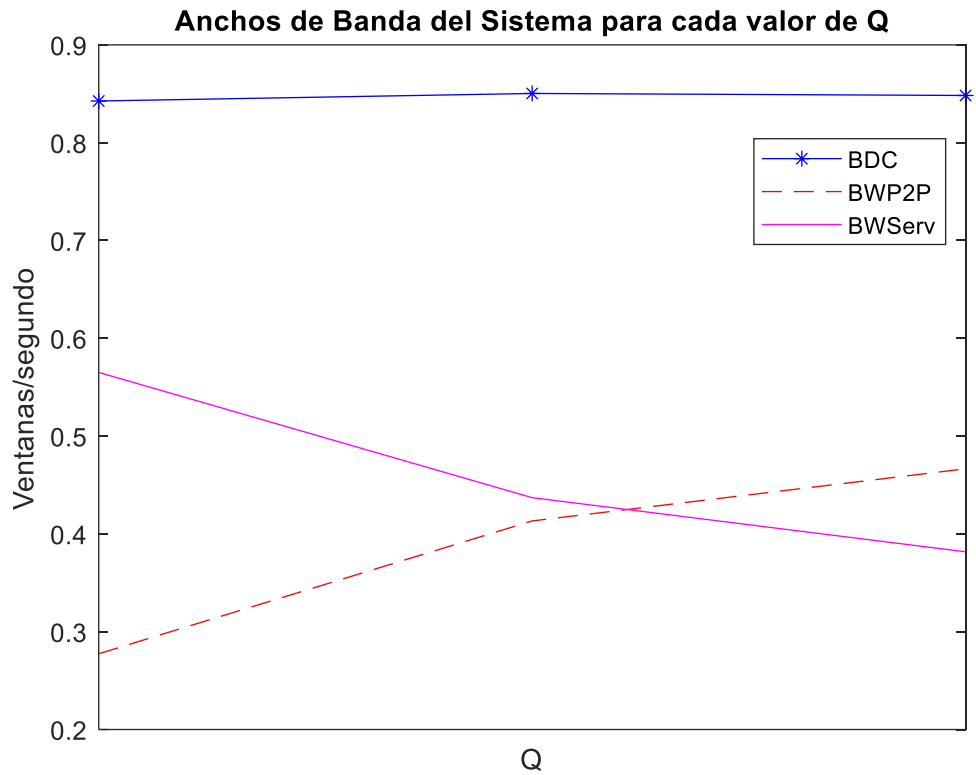
Las Figuras mostradas e interpretadas a continuación fueron obtenidas al solucionar por implementación el modelo establecido en la Figura 42 y empleando un esquema de asignación de recursos de distribución uniforme.

### Resultados Finales Finales

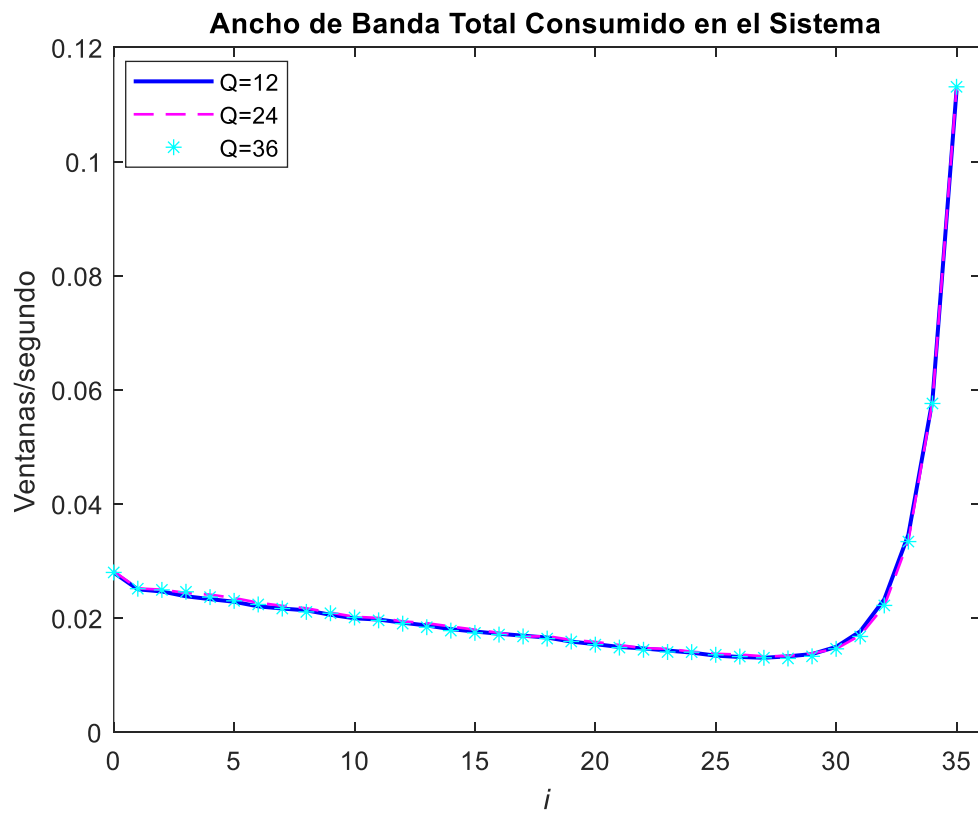
Resultados Obtenidos por iteración para Q ventanas

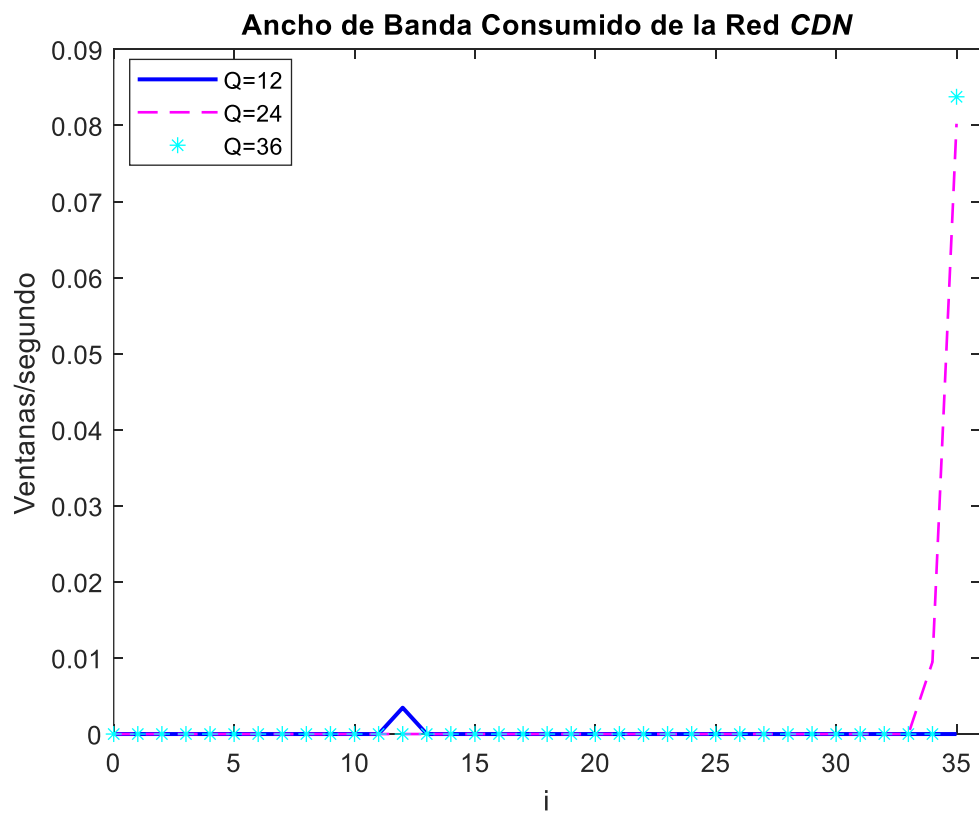
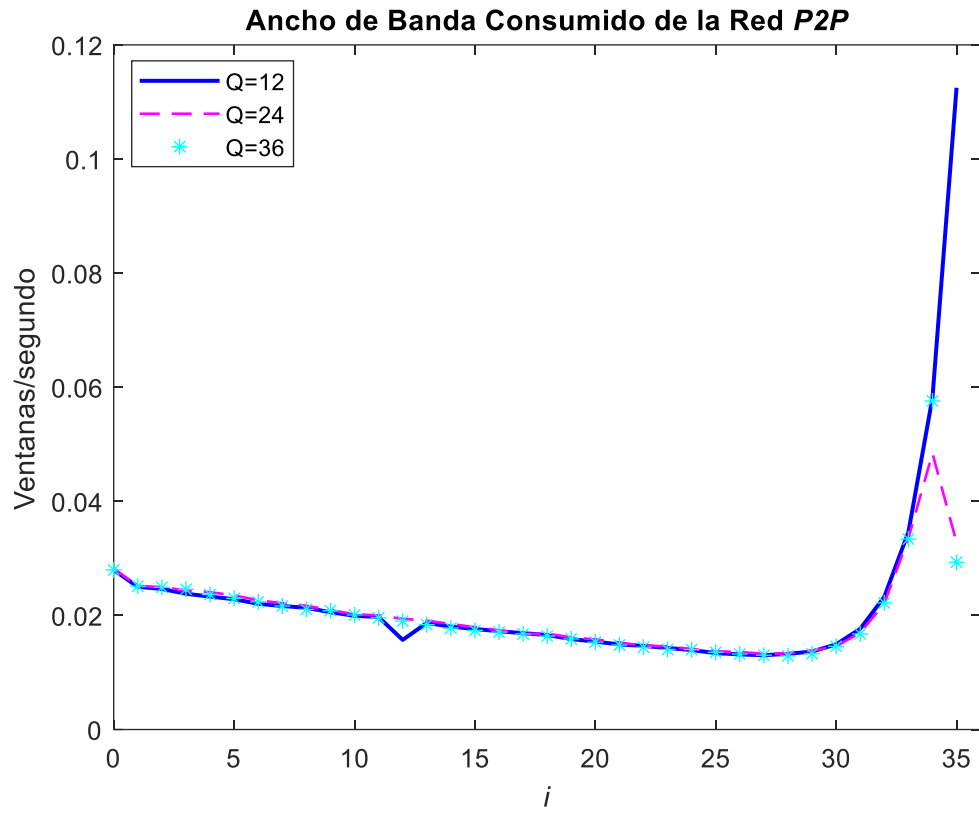


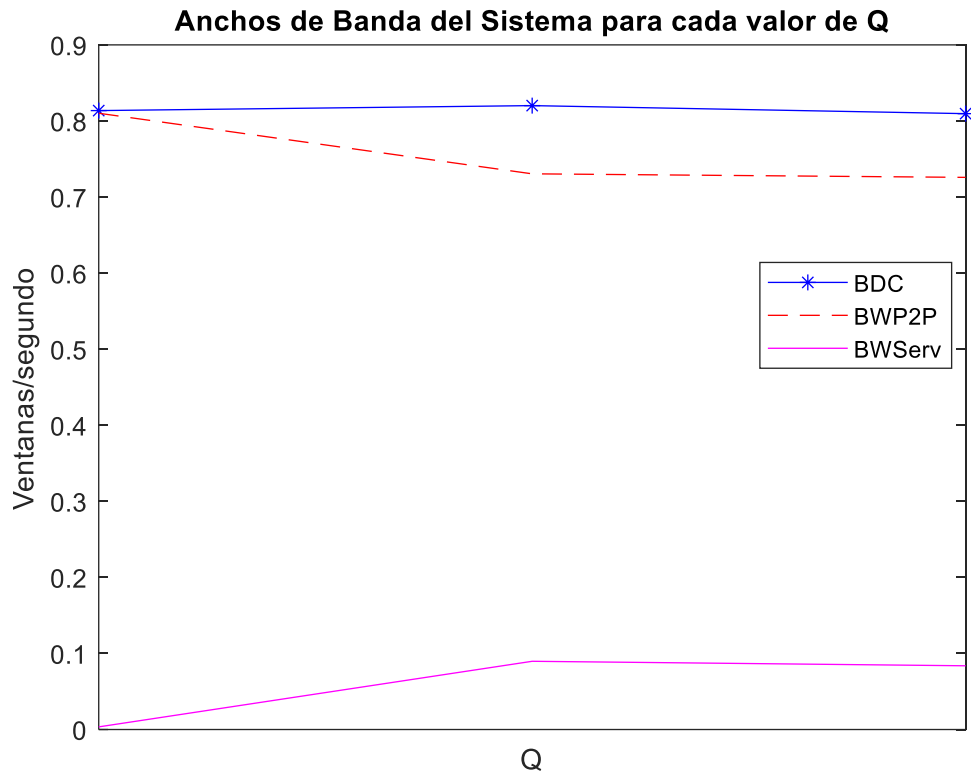




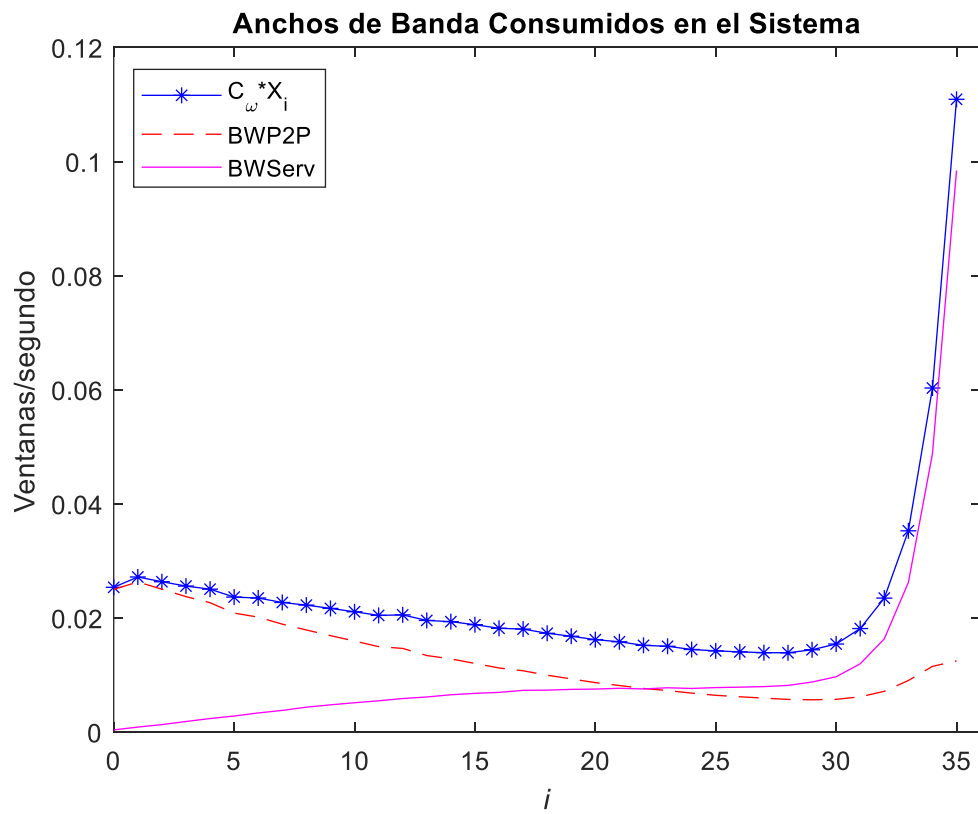
Resultados obtenidos en estado estable para Q ventanas





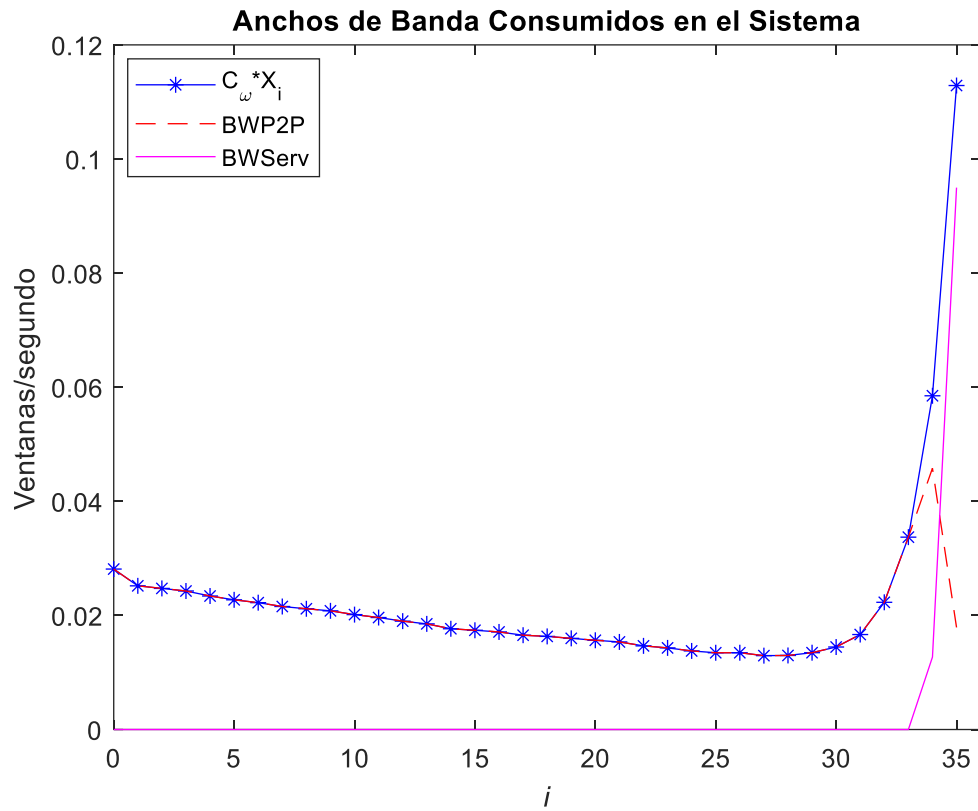


Resultados obtenidos para DU por iteración

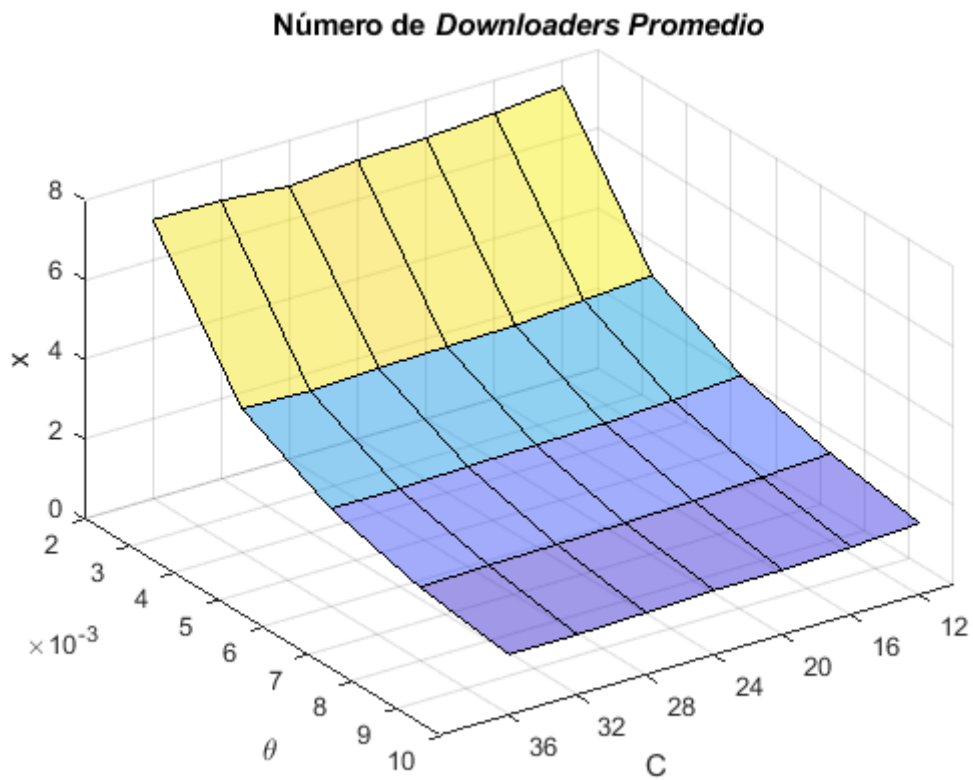


Resultados obtenidos para DU en estable

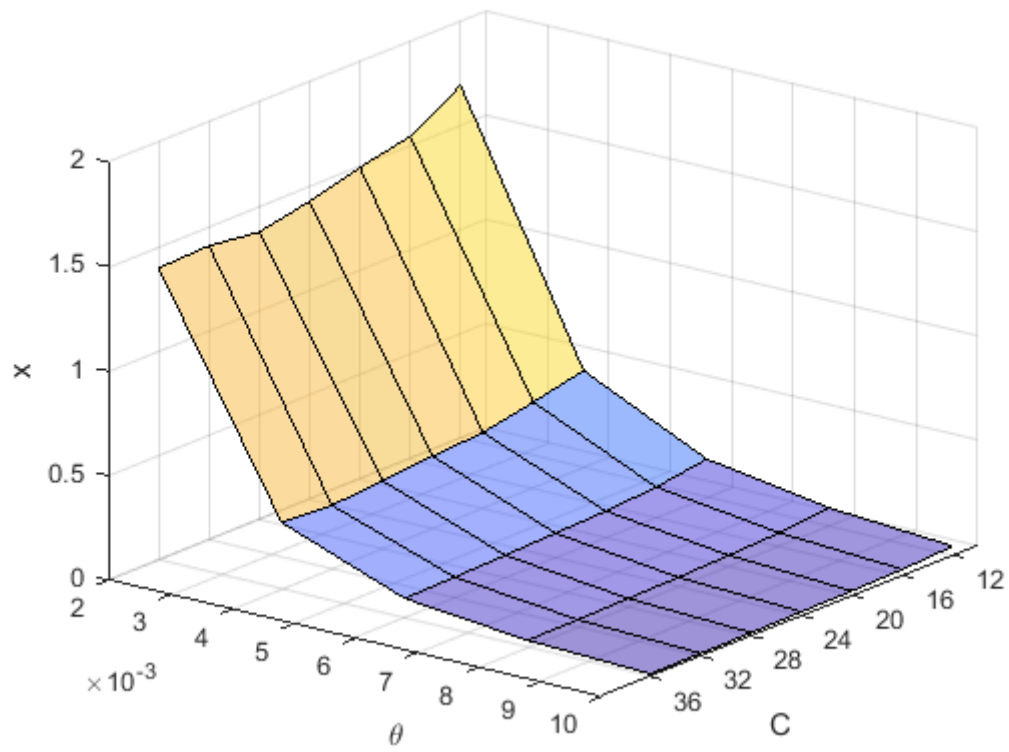




Resultados 3D para DU

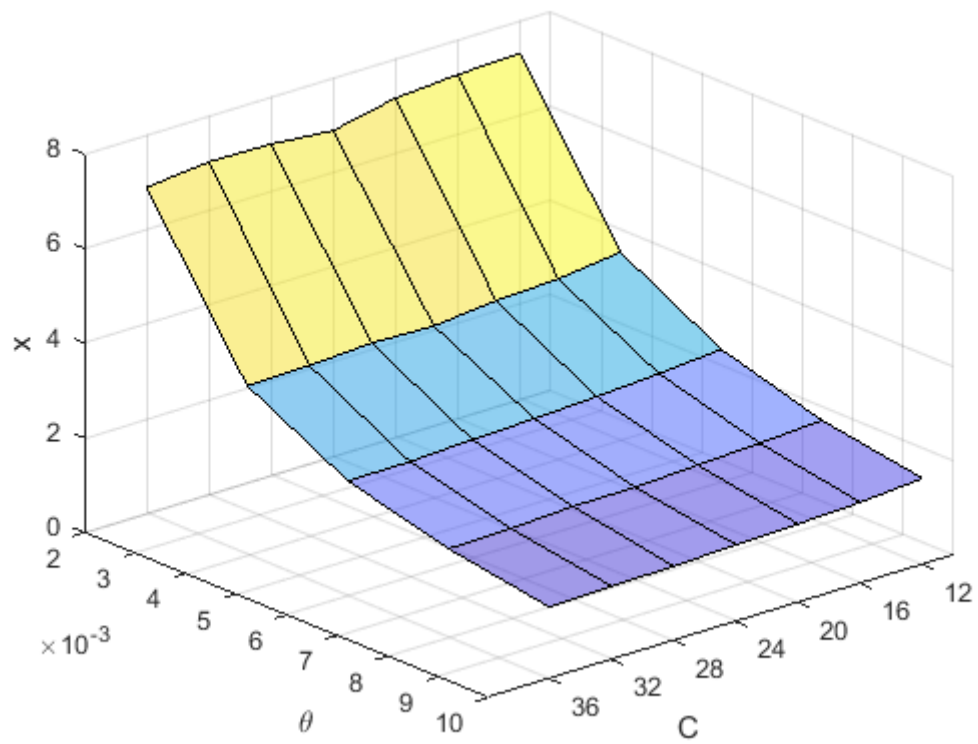


**Número de *Downloaders* Promedio en la Ventana C**

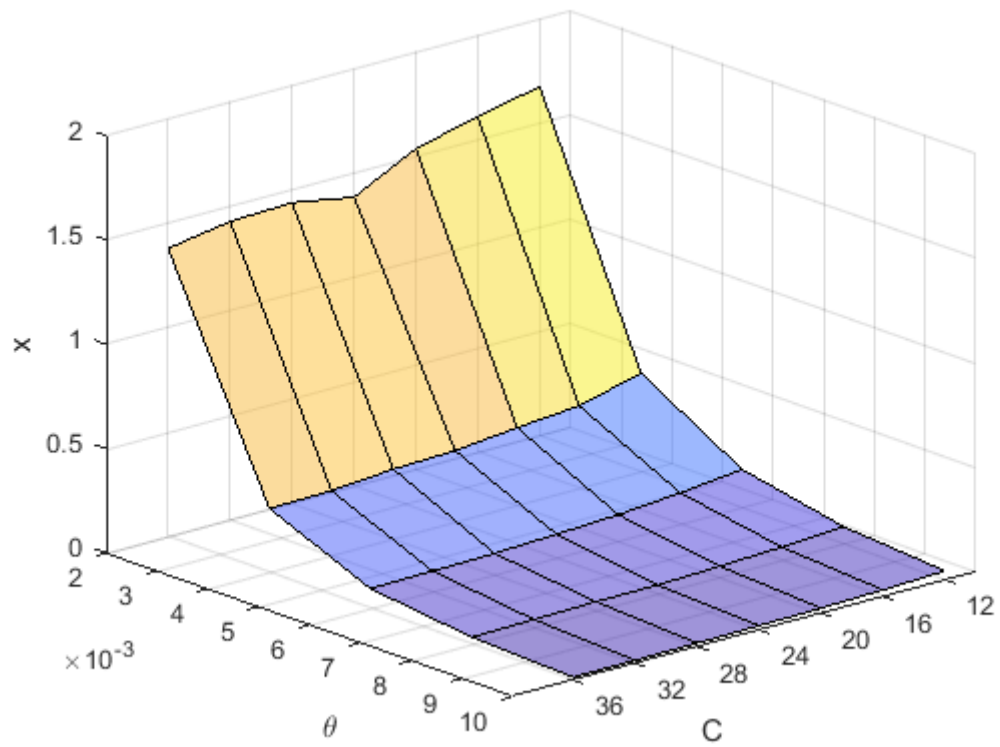


Resultados 3D para Q ventanas

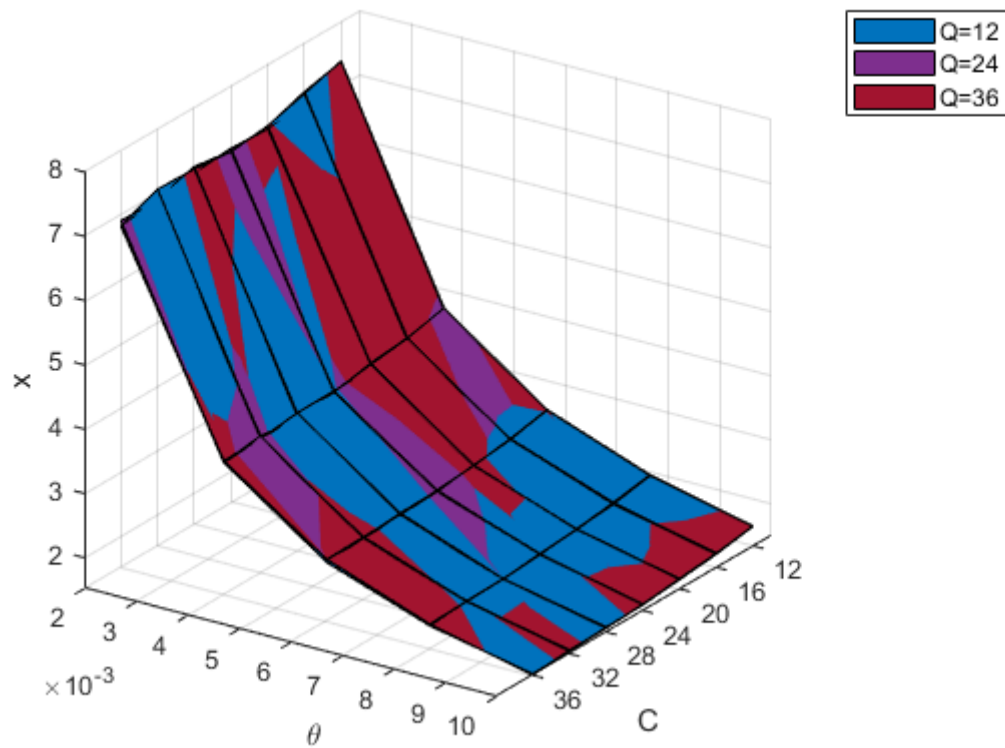
**Número de *Downloaders* Promedio**

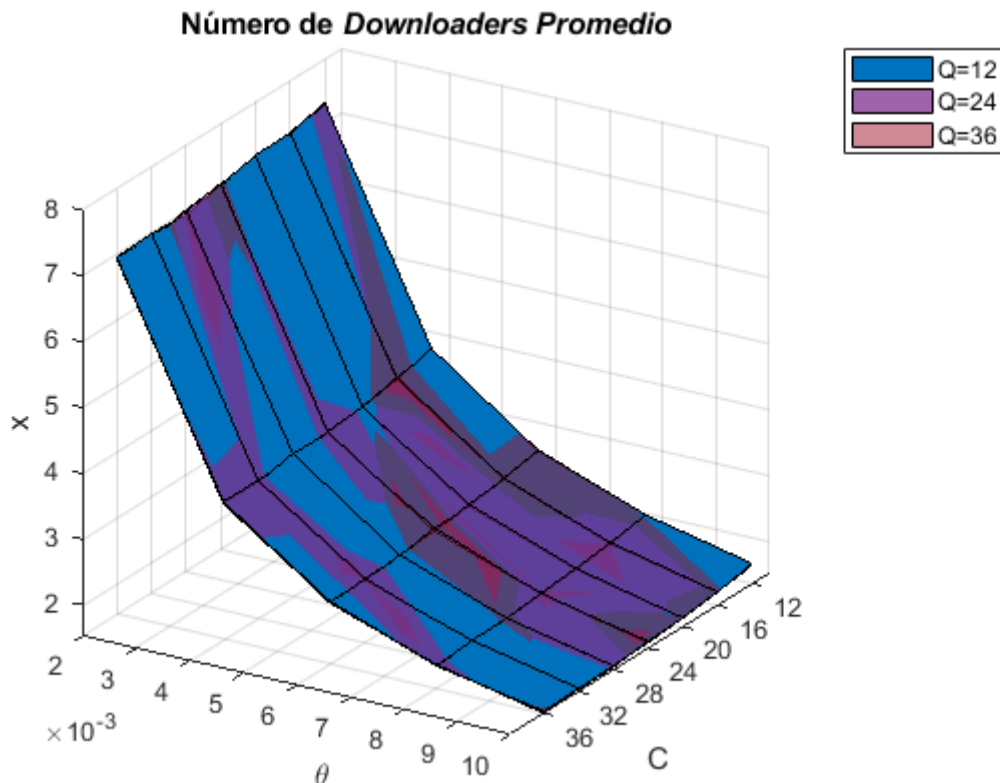


**Número de *Downloaders* Promedio en la Ventana C**



**Número de *Downloaders* Promedio**





## Conclusiones

Con base en la problemática que se desarrolla en el presente proyecto y siguiendo los objetivos planteados se han realizado diversas actividades que permiten plantear una propuesta de solución asertiva. Se realizó el análisis de esquemas de asignación de recursos reportados en trabajos relacionados enfocados a servicios de video bajo demanda (*Video on Demand-VoD*) con el propósito de sentar bases para el desarrollo y evaluación de un esquema de asignación de recursos ad hoc para servicios de video en vivo.

Otra actividad fundamental fue el análisis del modelado de servicios de video bajo demanda mediante cadenas de Markov que permitió desarrollar una cadena de Markov que representa la dinámica de un sistema de transmisión de video en vivo sobre una red *P2P-CDN*.

La implementación de la solución por simulación de una cadena de Markov unidimensional permitió conocer en un ambiente simulado la transición en el estado de una cadena de Markov una vez que ocurre un evento del tipo discreto dentro del sistema modelado a través de la cadena. Por otro lado, esta implementación permitió conocer el *IDE* Matlab, su sintaxis y las variables resultantes que entrega una vez que se simulan eventos discretos.

Debido a que la implementación de la solución por simulación de la cadena unidimensional se realizó sin grandes problemas referentes a la lógica de programación, la sintaxis y las herramientas ofrecidas por el *IDE* Matlab se ha elegido a este para realizar la implementación de la solución por simulación de la cadena desarrollada en Proyecto Terminal I que representa a un sistema de servicio de video en vivo.

Una vez realizadas las actividades descritas anteriormente se espera que la implementación de la solución por simulación correspondiente a la cadena de Markov para servicios de video en vivo a lo largo de Proyecto Terminal II no represente complicaciones salvo por la lógica y sintaxis de programación dentro del *IDE* seleccionado.

Finalmente, se espera que durante el desarrollo de Proyecto Terminal II el comportamiento del sistema de servicios de video en vivo analizado y obtenido en Proyecto Terminal I sea el mismo o muy aproximado al que se obtendrá con la implementación de la solución por simulación de la cadena de Markov para servicios de video en vivo.

Por otro lado, se espera que el esquema de asignación de recursos a proponer en el presente proyecto muestre un desempeño eficiente frente a los servicios de video en vivo, es decir, el esquema de asignación de recursos debe ser capaz de producir una cooperación efectiva entre los *peers* de la red *P2P* y por lo tanto disminuir las peticiones de atención a la red *CDN*.

# Referencias

- [1] T. Cooper, «BROADBANDNOW,» 28 Octubre 2020. [En línea]. Available: <https://broadbandnow.com/report/cable-vs-satellite-vs-iptv-vs-ott-streaming/#:~:text=Essentially%2C%20IPTV%20is%20a%20formally,available%20to%20consumers%20from%20ISPs.&text=Rather%20than%20streaming%20content%20directly,to%20display%20on%20your%20TV..> [Último acceso: 21 Mayo 2021].
- [2] S. M. Y. Seyyedi y B. Akbari, «Hybrid CDN-P2P Architectures for Live Video,» de *International Symposium on Computer Networks and Distributed Systems (CNDs)*, Tehran, 2011.
- [3] A. Mansy y M. Ammar, «Analysis of Adaptive Streaming for HybridCDN/P2P Live Video Systems,» IEEE, Atlanta, 2011.
- [4] R. Trestian, I.-S. Comsa y M. Fatih, «Seamless Multimedia Delivery Within aHeterogeneous Wireless NetworksEnvironment: Are We There Yet?,» *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, vol. 20, nº 2, pp. 945-977, 2018.
- [5] N. Torres Cruz, M. E. Rivero Angeles, G. Rubino, R. Menchaca Mendez y R. Menchaca Mendez, «A Window-Based, Server-Assisted P2P Network forVoD Services with QoE Guarantees,» *Hindawi*, vol. 2017, nº 2084684, pp. 1-18, 2017.
- [6] N. Torres Cruz, M. E. Rivero-Angeles, G. Rubino, R. Menchaca Mendez, R. Mechaca Mendez y D. Ramirez, «A comprehensive analytical framework VoD Services in hybrid CDN-P2P,» *Elsevier*, vol. 102643, nº 161, pp. 1-17, 2020.
- [7] E. D. Terrones Celis, Artist, *Modelado y evaluación de servicios de video sobre redes P2P en ambientes móviles 5G*. [Art]. Centro de Investigación en Computación, 2020.
- [8] R. Naylor, J. Todd, M. Moretto y R. Traverso, «Unesco,» 2021. [En línea]. Available: [https://unesdoc.unesco.org/ark:/48223/pf0000377863\\_spa](https://unesdoc.unesco.org/ark:/48223/pf0000377863_spa). [Último acceso: 15 Mayo 2023].
- [9] T. T. Thu Ha, J. Kim y J. Nam, «Design and Deployment of Low-Delay Hybrid CDN–P2PArchitecture for Live Video Streaming Over the Web,» *Springer Science+Business Media*, vol. 10, nº 94, p. 13, 2015.
- [10] M. Hassan, C. K. Neng y L. C. Suan, «Performance Analysis of Video Streaming on different HybridCDN & P2P Infrastructure,» MIMOS Berhad, Malaysia, 2021.
- [11] S. Nacakli y A. Tekalp, «Controlling P2P-CDN Live Streaming Services at SDN-enabled Multi-Access Edge Datacenters,» *Fellow,IEEE*, vol. 10, nº Y, pp. 1-12, 2020.

- [12 J. Sun, Y. Zhou, Y. Duan y Z. Guo, «A Low-latency Peer-to-Peer Live and VODStreaming System Based on Scalable Video Coding,» Beijing, IEEE, 2014, p. 319.
- [13 N. Torres Cruz, *Esquemas de asignación de recursos para servicios de video en redes heterogéneas 5G*, México: IPN, 2019.
- [14 V. Pichardo Herrera, *Evaluación de esquemas de asignación de recursos*, México: IPN, 2021.
- [15 C. Zhao, J. Zhao, X. Lin y C. Wu, «Capacity of P2P on-demand streaming with simple, robust and decentralized control,» IEEE, Turin, 2013.
- [16 BlackBox, «BlackBox,» 20 Julio 2018. [En línea]. Available:  
] <https://www.blackbox.com.mx/mx-mx/page/40830/Recursos/Technical/black-box-explica/Multimedia/Compresion-de-video-H264>. [Último acceso: 22 Mayo 2021].
- [17 R. Catro, «WIKIVERSUS,» 25 Julio 2020. [En línea]. Available:  
] <https://www.wikiversus.com/fotografia-y-video/codecs-h-265-vs-vp9/>. [Último acceso: 22 Mayo 2021].
- [18 C. Cano y M. Raffo, «Arquitectura de alta frecuencia de un filtro de escalabilidad para sobremuestreo de imágenes en factor 2 sobre una FPGA,» 2013.
- [19 J. Alvaro, «Formatos Multimedia: Codificación y Empaquetado,» 2014. [En línea].  
] Available: <https://edu.fauno.org/R.multimedia.formatos.html>. [Último acceso: 22 Mayo 2021].
- [20 UMA, «Herramientas web para la enseñanza de protocolos de comunicación,» evirtual,  
] [En línea]. Available:  
<https://neo.lcc.uma.es/evirtual/cdd/tutorial/presentacion/mpeg.html>. [Último acceso: 5 Junio 2021].
- [21 B. Jedari, G. Premsankar, G. Illahi, M. DiFrancesco, A. Mehrabi y A. Ylä-Jääski, «Video Caching, Analytics, and Delivery at the,» *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, vol. 23, nº 1, pp. 431-471, 2021.
- [22 V. Rodríguez, «Scenikus blog,» 21 Abril 2021. [En línea]. Available:  
] <https://blog.scenikus.com/diferencia-entre-live-streaming-y-vod/>. [Último acceso: 23 Mayo 2022].
- [23 M. Willbert, «Live Streaming vs. VOD: Comparing the 10 Best Video Platforms That Do Both,» datacast, 20 Enero 2022. [En línea]. Available:  
] <https://www.dacast.com/es/blog/live-streaming-vs-vod/>. [Último acceso: 23 Mayo 2022].

- [24 D. Matus, «digitaltrends,» 18 Noviembre 2020. [En línea]. Available:  
] <https://es.digitaltrends.com/entretenimiento/tv-streaming-comparativa-servicios/>.  
[Último acceso: 22 Mayo 2021].
- [25 M. Wilbert, «HLS vs. MPEG-DASH: A Live Streaming Protocol Comparison for 2021,»  
] datacast, 3 Mayo 2021. [En línea]. Available: <https://www.dacast.com/blog/mpeg-dash-vs-hls-what-you-should-know/>. [Último acceso: 5 Mayo 2021].
- [26 W. J. Stewart, Probability, Markov Chains, Queues, and Simulation, New Jersey: Princeton  
] University Press, 2009.
- [27 V. Rogriguez. [En línea].  
]