ANÁLISIS DEL DESEMPEÑO DEL RWA EN REDES OBSWDM CON CONTROL COGNITIVO BASADO EN ALGORITMOS GENÉTICOS.

Documento de Trabajo de grado como requisito para optar al título de Ingeniera en Electrónica y Telecomunicaciones

Adriana María Hincapié Moncayo

Tatiana Peña Valencia

Director: PhD(c). Ing. José Giovanny López Perafán

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telecomunicaciones
Grupo de I+D Nuevas Tecnologías en Telecomunicaciones GNTT
Popayán
2013

TABLA DE CONTENIDO

INDICE DE FIGURAS

Pág.

INDICE DE TABLAS

ANEXOS.

Pág.

LISTA DE ACRÓNIMOS

GA	*Genetic Algorithms,* Algoritmos Genéticos.
AW	*Available Wavelengths*, Longitudes de Onda Disponibles.
BCP	*Burst Control Packet*, Paquete de Control de Ráfaga.
CHRON	*Cognitive Heterogeneous Reconfigurable Optical Network,* Redes Ópticas Reconfigurables Heterogéneas Cognitivas.
DRWA	*Dynamic Routing and Wavelength Assignment,* Enrutamiento y Asignación de Longitudes de Onda Dinámico.
DW	*Distance Weight*, Menor Distancia.
EAs	*Evolutive Algorithms*, Algoritmos Evolutivos.
FDL	Fiber Delay Lines, Líneas de retardo de Fibra.
FA	*Fixed Alternate,* Fijo Alternado.
FF	*First Fit,* Primer Ajuste.
HAW	*Hop and Available Wavelengths*, Saltos y Longitudes de Onda Disponibles.
HPDL	*Heaviest Path Load Deviation*, Desviación de Carga de la Peor Ruta
HTAW	*Hop Count and Total Wavelengths and Available Wavelengths*, Número de saltos y Longitudes de Onda Totales y Disponibles
HW	*Hop Weight*, Algoritmo Peso del Salto.
IBT	*In Band Terminator,* Terminación en Banda.
IP	*Internet Protocol,* Protocolo de Internet.
JET	*Just Enough Time,* Tiempo Justo.
JIT	*Just In Time,* Justo a Tiempo.
LAUC	*Latest Available Unused Channel*, Último Canal Libre Disponible.
LBRWA	*Load Balance RWA*, Balance de Carga de Enrutamiento y Asignaciones de Onda.
LCPR	*Least Congested Path Routing*, Enrutamiento de menor congestión.
LL	*Least Loaded*, Menos Cargado
LU	*Least Used*, Menos Usado.

MAC	*Media Access Control,* Control de Acceso al Medio.
MFC	*Modified Future Cost*, Costo Futuro Modificado.
MNH	Minimum Number of Hops, Menor Número de Saltos.
MP	*Min-Product*, Producto Mínimo.
MS	*Min-Sum*, Suma Mínima.
MU	*Most Used*, Más Usado.
M∑	*Max-Sum*, Máxima Suma.
NSFNET	*National Science Foundation Network,* Red de la Fundación Nacional de la Ciencia.
NP-Completo	*Nondeterministic Polynomial Time,* tiempo polinomial no determinista.
OBS	*Optical Burst Switching,* Conmutación Óptica de Ráfagas.
OCS	*Optical Circuit Switching,* Conmutación Óptica de Circuitos.
OPS	*Optical Packet Switching,* Conmutación Óptica de Paquetes.
OSI	*Open System Interconnection,* Interconexión de Sistemas Abiertos.
RFD	*Reserve a Fixed Duration*, Reserva de Duración Fija.
R	*Random*, Aleatorio.
RWA	*Routing and Wavelength Assignment*, Enrutamiento y Asignación de Longitudes de Onda,
SP	*Shortest Path*, Ruta más Corta.
TAG	*Tell And Go*, Recomienda y Sigue.
TAW	*Total and Available Wavelengths*, Longitudes de Onda Totales y Disponibles.
TIC	Tecnologías de la Información y la Comunicación.
TP	*Threshold Protection*, Protección Umbral.
WAN	*Wide Area Network, Red de Área Amplia.*
WDM	*Wavelength Division Multiplexing*, Multiplexación por División de Longitud de Onda.
WR	*Wavelength Reservation*, Reserva de Longitudes de Onda.
WRON	*Wavelength Routed Optical Networks,* Redes Ópticas con Enrutamiento por Longitudes de Onda.

1 INTRODUCCIÓN.

1.1 PLANTEAMIENTO DEL PROBLEMA.

La creciente demanda de servicios referentes a las Tecnologías de la Información y la Comunicación (TIC) [1] ha impulsado la implementación de técnicas [2] y tecnologías [3] [4] cada vez más eficientes, que van de la mano con el aumento en el ancho de banda para satisfacer el flujo de información requerido por los usuarios.

En el entorno de la redes de comunicación, la fibra óptica es el medio de transmisión por excelencia para los enlaces, brindando flexibilidad en el transporte, la instalación, un amplio rango de distancias entre redes, calidad y seguridad de la señal, entre otras [5]. La implementación del Protocolo de Internet (IP, Internet Protocol) [6] sobre dicho medio, permite el flujo de datos multimedia y aprovecha el ancho de banda ofrecido por la fibra.

Para satisfacer el crecimiento exponencial de la demanda de tráfico logrando, al mismo tiempo, altas velocidades de transmisión, se introduce la Multiplexación por División de Longitud de Onda (WDM, Wavelength Division Multiplexing) [7] [8], la cual permite soportar sobre una misma fibra, diferentes portadoras ópticas (longitudes de onda), cada una de ellas llevando un flujo particular de información.

En este contexto, es necesario contar con técnicas que permitan transmitir de manera eficiente la información, por esta razón se utiliza la Conmutación Óptica de Ráfagas (OBS, Optical Burst Switching) como método que integra las ventajas de la conmutación de paquetes y de circuitos [9], ofreciendo mayor eficiencia al agrupar conjuntos de paquetes IP que tengan un mismo destino y generando menor cantidad de encabezados, lo que podría garantizar una disminución en el procesamiento[1] y, por lo tanto, menor retardo en la transmisión de la información [10].

A nivel físico, las redes ópticas han evolucionado con dispositivos que facilitan la inserción de longitudes de onda sin necesidad de modificar la infraestructura de la red [3] [11]. Es importante, complementando lo anterior, contar con técnicas que transporten la información y asignen una longitud de onda adecuada en un entorno de comunicación óptica OBS/WDM [12]. Este proceso se conoce como Enrutamiento y Asignación de Longitud de Onda (RWA, Routing and Wavelength Assignment) [13], y está catalogado como un problema NP – COMPLETO (Nondeterministic Polynomial Time) [14] [15] lo cual impide encontrar un tipo de solución trivial por medio de algoritmos deterministas, pero permite obtener soluciones aproximadas a la óptima. RWA, a su vez, presenta variantes de tipo estático y dinámico [16] según la demanda de tráfico, siendo el RWA dinámico el que más se acerca a un entorno real de la red, ya que las peticiones de conexión para generar un camino óptico se presentan de manera aleatoria y la demanda de tráfico ocurre en tiempo real, orientando la evaluación del desempeño hacia la probabilidad de bloqueo de conexión[2][17]. Luego, como el tráfico total no se conoce de antemano (pero debe estimarse para realizar la simulación), numerosas técnicas para el RWA carecen de eficiencia teniendo como limitante el tiempo de reconocimiento, búsqueda y procesamiento para generar la solución adecuada [18].

Por otra parte, muchos de los procedimientos encaminados a la gestión de redes suelen ser complejos y costosos, más aún si se implementan manualmente para cada necesidad. Diversas investigaciones se están desarrollando para analizar el desempeño de las redes cognitivas [19] como mecanismo que

[1]Proceso en donde se extrae y se analiza la información proveniente de la cabecera de los paquetes o ráfagas de datos.
[2]Indicador de rendimiento de la red, en el caso del proyecto actual se relaciona con el número de ráfagas perdidas dentro de la red.

permita gestionar la complejidad y la utilización eficiente de los recursos disponibles para ofrecer aplicaciones y servicios al menor precio posible [20]. Dentro de las soluciones se encuentran los procesos inspirados biológicamente.

Específicamente, las redes biológicas se guían por mecanismos de adaptación, y en este sentido, el rendimiento es un factor que depende del número de iteraciones para obtener una solución adecuada. Numerosas áreas de investigación se encuentran explorando los métodos inspirados biológicamente, debido a las notables ventajas que ofrecen en cuanto a escalabilidad, estabilidad y flexibilidad ante cambios en la red [18]. Los algoritmos genéticos (GA, Genetic Algorithms) hacen parte de estos procesos adaptativos, en donde la evolución es una táctica de supervivencia en entornos cambiantes usando métodos heurísticos [21] como estrategias de optimización [22].

A nivel internacional, se está llevando a cabo un proyecto fundado por la comisión europea, denominado Redes Ópticas Heterogéneas Reconfigurables Cognitivas (CHRON, Cognitive Heterogeneous Reconfigurable Optical Networks) [18], donde las redes son capaces de observar, actuar, aprender y optimizar su comportamiento. Aunque a nivel nacional se presentan investigaciones relacionadas con algoritmos genéticos y su rol computacional [22], así como el uso de métodos de aprendizaje maquinal orientados al desarrollo educativo [23], no se han encontrado aún investigaciones que traten de forma directa el problema RWA en redes ópticas OBS/WDM usando Algoritmos Genéticos y control cognitivo para analizar el desempeño de una red.

Aunque los AG presentan implícitamente cognición, debido a su naturaleza adaptativa y evolutiva, en las investigaciones realizadas no hay indicios de mecanismos de control cognitivo que permitan a la red retroalimentarse usando las soluciones anteriores, esperando obtener posteriormente un mejor desempeño. Teniendo en cuenta lo anterior, este trabajo busca aplicar un control de cognición a una red OBS/WDM que implementa algoritmos genéticos para los procesos de enrutamiento y asignación de longitudes de onda y analizar el desempeño, comparándolo con la misma red que solo esté haciendo uso de Algoritmos Genéticos para el RWA. En este sentido surge la siguiente pregunta de investigación.

¿Cuál es la incidencia en el desempeño del mecanismo de RWA al introducir un método de control cognitivo basado en algoritmos genéticos sobre redes ópticas OBS/WDM?

1.2 ESCENARIO DE MOTIVACIÓN.

Las redes ópticas se han visto sometidas a grandes cambios debido a la necesidad de soportar el aumento exponencial de usuarios y un incremento considerable del tráfico que circula por la red. Prueba de esto es la transparencia[3]de los datos, y el hito tecnológico que permitió implementar la multiplexación de canales, encaminando información a través de distintas longitudes de onda, todo por una misma fibra [21]. No obstante, estos datos deben viajar hacía el nodo de destino cumpliendo con criterios de calidad de servicio[4], desempeño[5] y eficiencia[6] dentro de un rango establecido [24]. Los

[3] Habilidad para transmitir datos a través de la red de manera transparente a los usuarios. http://www.itei.org.mx/v3/micrositios/congresotrans2012/

[4] Capacidad que tiene un elemento de red para asegurar que el tráfico y los requisitos del servicio previamente establecidos puedan satisfacerse.

[5] Calidad de funcionamiento de una red. http://www.itu.int/online/terminology/index.html

[6] Capacidad que tiene una red para ofrecer un objetivo determinado con el mínimo de recursos posibles viable. http://ceres.ugr.es/~gmacia/papers/JITEL10_MDTSN.pdf

métodos para que la red cumpla con estos requerimientos son numerosos y variados, pasando por técnicas de encaminamiento, asignación de longitud de Onda, Ingeniería de Tráfico (inmersa dentro de la Conmutación de Etiquetas), entre otros. Muchos de estos algoritmos se encuentran en desventaja al interactuar con dispositivos de las redes ópticas actuales, es decir, se vuelven obsoletos ante características de escalabilidad o convergencia. Por otra parte, aunque las redes ópticas presentan ventajas debido a la velocidad de transmisión y aumento del ancho de banda [25], no disponen de recursos ilimitados, lo cual afecta su capacidad al manejar demandas de tráfico aleatorio, originado por las peticiones de conexión en tiempo dinámico. Por esta razón, se descartan las soluciones que traten tareas de asignación y encaminamiento de forma estática, puesto que el problema actual está relacionado con demanda de tráfico en tiempo real [26].

Además, es preciso recordar que el proceso RWA hace referencia a un problema NP-Completo, por esta razón, el nivel de aptitud[7] de cada algoritmo, se mide de acuerdo con la cercanía a la respuesta óptima [27]. En este sentido, es necesario contar con aplicaciones sensibles capaces de controlar la demanda de tráfico, gestionando caminos y longitudes de onda a través de la red.

Estrategias de búsqueda y optimización basadas en métodos inspirados biológicamente se encuentran en implementación debido a las ventajas que ofrecen en cuanto a robustez, escalabilidad, convergencia hacía el óptimo global y capacidad de abstracción dentro de las redes ópticas [28], [29]; los algoritmos genéticos hacen parte de este enfoque. Así mismo, surgen los métodos de control cognitivo como técnicas que permiten observar, razonar y aprender de la red, tomando decisiones que podrían brindar beneficios globales [19], [30].

En conclusión, la propuesta basada en Algoritmos Genéticos y métodos de control cognitivo como técnicas de retroalimentación dentro de los procesos de RWA, son el principal objeto de investigación de la presente monografía, ya que dichos métodos lidian con la creciente demanda de Internet centrando sus esfuerzos en la adecuada gestión de los recursos.

1.3 OBJETIVOS.

1.3.1 Objetivo General.

- Analizar el desempeño del RWA en una red óptica OBS/WDM incluyendo un método de control cognitivo desde la perspectiva de algoritmos genéticos.

1.3.2 Objetivos específicos.

- Caracterización del problema RWA en redes OBS/WDM.
- Implementar un método de control cognitivo basado en algoritmos genéticos que se ajuste a los requerimientos del RWA en redes OBS/WDM.
- Evaluar mediante simulación (en la herramienta OMNeT++) el desempeño del RWA en redes ópticas OBS/WDM al aplicar algoritmos genéticos con y sin métodos de control cognitivo.

[7]Mide la capacidad que tiene los individuos evaluados en un algoritmo genético mediante la función de aptitud, mientras más grande sea, ellos tienen más posibilidades de vivir y de reproducirse para aportar sus características genéticas a nuevas generaciones.

1.4 ENFOQUE.

El desarrollo del estado del arte permitió observar numerosos algoritmos de asignación de longitud de onda y enrutamiento, los cuales se clasifican según su interacción con el estado en tiempo real de la red. En este sentido, se encuentran los algoritmos de encaminamiento fijo y fijo-alternado, los cuales, aunque brindan alternativas para enrutar la información de forma simple, presentan altas probabilidades de bloqueo de conexión puesto que escogen el camino por medio de una función ejecutada sin tener en cuenta el estado de rutas disponibles en tiempo real [21]. Ahora bien, los algoritmos de encaminamiento adaptativo también hacen parte de esta clasificación, exhibiendo, no obstante, mayor probabilidad de establecer una conexión exitosa, ya que analizan los caminos y longitudes de onda disponibles teniendo en cuenta los recursos actuales de la red [21]. Por esta razón, se propone un algoritmo RWA dinámico (DRWA, Dynamic Routing and Wavelength Assignment) basado en algoritmos genéticos, valiéndose de estrategias de búsqueda y optimización ofrecidas por los AG.

Por lo tanto, el enfoque del presente trabajo de grado comienza caracterizando el enrutamiento y asignación de longitud de onda, los Algoritmos Genéticos y los Métodos de Control Cognitivo, dentro de las redes ópticas OBS/WDM. Posterior a este análisis se realiza la definición de parámetros y requerimientos de cada objeto de investigación; lo cual hace referencia a las especificaciones de la red óptica (número de nodos, longitudes de onda por fibra, topología, enlaces, etc.), diseño del algoritmo genético y del método de control cognitivo para dicha red; este último, hace referencia a una base de información cognitiva en la etapa inicial del proceso evolutivo, el cual, abstrae las soluciones anteriores del algoritmo genético, escogiendo algunas que puedan ser aptas para la siguiente etapa del proceso RWA.

Habiendo identificado los parámetros más adecuados de cada objeto de investigación, se efectúa la implementación de los procesos del RWA usando AG y un método de control cognitivo dentro de la red óptica OBS/WDM; en esta instancia, se realiza la simulación de dicho procedimiento en el entorno de simulación de eventos discretos OMNeT++ [31], [32]. La Figura 1 permite visualizar el enfoque de manera global, encaminado hacía el cumplimiento de los objetivos planteados en el trabajo de grado. Los resultados que muestra el entorno de programación, serán la prueba que permita identificar la pertinencia de la inclusión de métodos de control cognitivo en redes ópticas OBS/WDM.

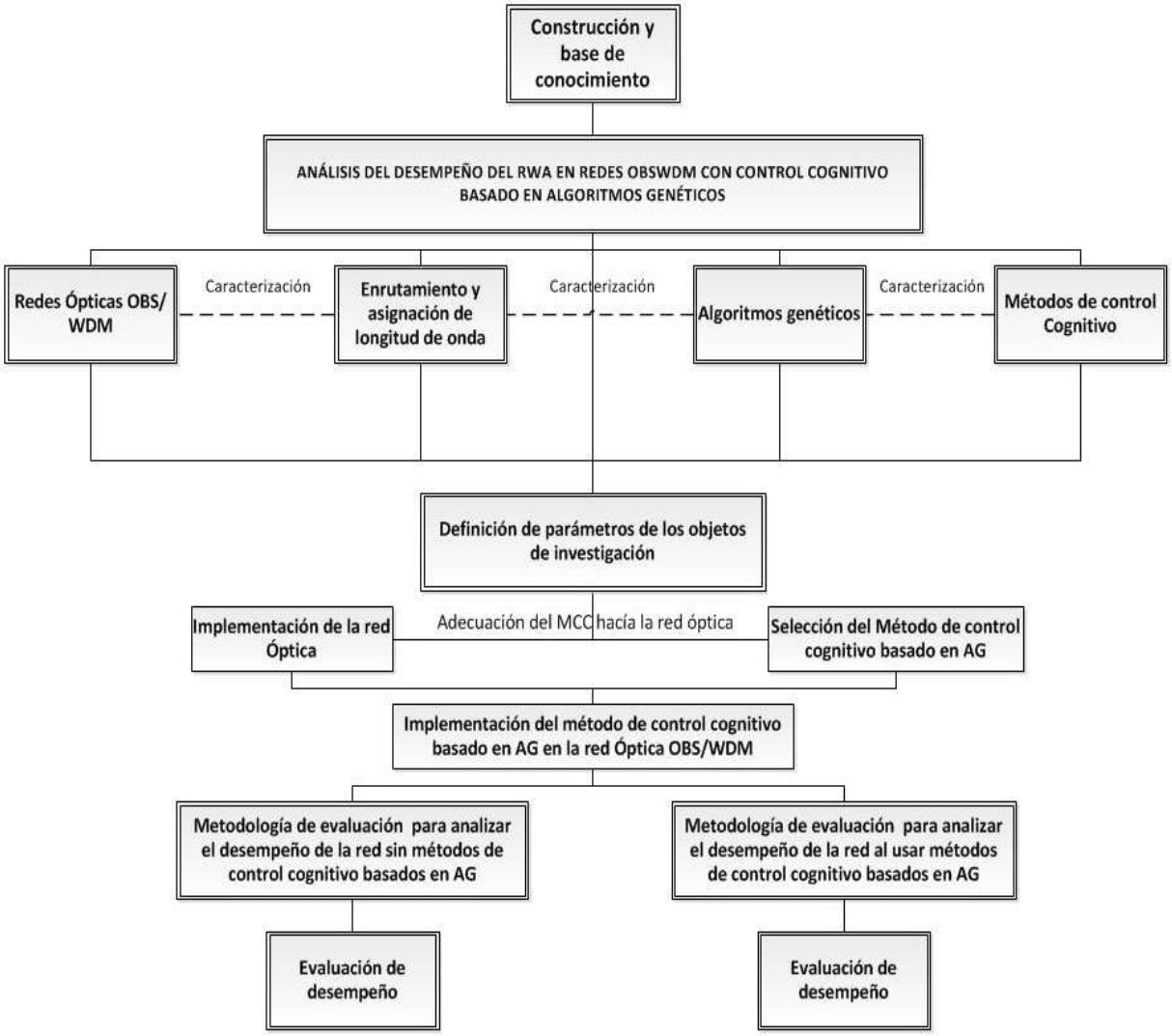


Figura 1. Enfoque del trabajo de grado.

1.5 ALCANCE.

El presente trabajo de grado se realizó simulando la red óptica OBS/WDM NSFNet (explicada con detalle en el capítulo 3), una red de área amplia (WAN, Wide Area Network) que cubre Estados Unidos y consta de catorce nodos, veintiún conexiones bidireccionales con distancias que varían entre los trescientos y dos mil ochocientos kilómetros y con diferentes números de longitudes de onda por enlace para realizar el análisis.

Es importante considerar que la red escogida presenta restricción de la continuidad de longitud de onda, es decir, no se implementan conversores que permitan reemplazar una determinada longitud de onda durante el camino establecido entre origen y destino.

El tráfico o carga de la red llega de manera dinámica y se realizan procesos de enrutamiento y asignación de longitudes de onda para dicho tráfico.

Con el fin de analizar el desempeño de los procesos del RWA dentro de las redes ópticas, se implementó un algoritmo genético simple, el cual escoge un determinado número de rutas de acuerdo a la probabilidad de selección de cada una de ellas, les aplica los operadores de selección y cruce (explicados con detalle en el capítulo 2) con el fin de obtener individuos más aptos, y analiza la pertinencia de las rutas resultantes de acuerdo a la función de aptitud implementada, la cual define el número de iteraciones que debe realizar el algoritmo antes de establecer el criterio de parada.

El desarrollo del trabajo de grado incluyó un método de control cognitivo basado en memoria, cuyo enfoque consistió en extraer rutas almacenadas con anterioridad, producto del comportamiento previo de la red, con el objetivo de analizar el desempeño al implementar dicho método.

1.6 APORTES.

- Informe referente al estudio sobre la aplicación de los procesos inspirados biológicamente y su implementación en redes ópticas, específicamente en el RWA.
- Desarrollo de un método de control cognitivo basado en algoritmos genéticos para redes ópticas OBS/WDM.
- Simulación del impacto que generan los algoritmos genéticos al incluir o no métodos de control cognitivo dentro de redes ópticas OBS/WDM.
- Evaluación del desempeño de los procesos RWA en redes ópticas OBS/WDM al aplicar algoritmos genéticos con y sin métodos de control cognitivo, mediante la estimación de parámetros como probabilidad de bloqueo y tiempo de procesamiento.

1.7 CONTENIDO DE LA MONOGRAFÍA.

- **Capítulo 2:** Estado del arte.

  Este capítulo aborda los principales conceptos tratados en la monografía, los cuales facilitarán la compresión y el entendimiento del proyecto. Definiciones formales sobre los Algoritmos Genéticos, redes ópticas cognitivas, técnicas de enrutamiento y redes OBS, serán algunos de los temas que se explicarán. Además, se realiza una presentación del estado actual de conocimiento, basándose en las áreas que se abarcan en el presente trabajo de grado.

- **Capítulo 3:** Implementación.

  Este capítulo contiene la descripción del procedimiento y los módulos usados para diseñar la red óptica OBS/WDM; así mismo, se describen los procesos de diseño del algoritmo genético y el método de control cognitivo.

- **Capítulo 4:** Análisis del desempeño de la red OBS/WDM basada en algoritmos genéticos con y sin métodos de control cognitivo.

  Este capítulo contiene los resultados de simulación de una red óptica OBS/WDM al aplicar algoritmos genéticos con y sin métodos de control cognitivo. La evaluación se realiza en términos

de la probabilidad de bloqueo y el tiempo de procesamiento. De igual forma se explican los parámetros de diseño utilizados para poner en funcionamiento el prototipo de simulación.

- **Capítulo 5: Conclusiones y trabajos futuros.**

  Este capítulo contiene el análisis de los resultados obtenidos, identificando los aportes representativos de la ejecución del proyecto. Además, lo anterior permite visualizar los factores relevantes y las recomendaciones que indicen en el desarrollo de trabajos futuros.

2 ESTADO DEL ARTE.

2.1 REDES ÓPTICAS.

Se habla de una red óptica cuando se tiene un sistema de telecomunicaciones en el cual los enlaces de transmisión están hechos de fibra óptica, con una arquitectura que permite la explotación de cada una de las características del medio. Su diseño resulta complejo, ya que se requiere generalmente de una combinación de elementos electrónicos y ópticos, además del software apropiado para el correcto funcionamiento de la red; sin embargo, para el transporte de información, la fibra óptica resulta ser una excelente opción debido a su inmunidad ante las interferencias, las bajas atenuaciones y la gran capacidad que ofrecen en cuanto a ancho de banda [29], [33].

Ciertamente, el crecimiento de la demanda de ancho de banda presentado en los últimos años, debido a la gran cantidad de aplicaciones multimedia y servicios ofrecidos a los usuarios, genera la necesidad de usar redes de transporte que soporten grandes velocidades y con gran capacidad, las redes ópticas presentan una excelente opción para enfrentar el incremento de esta demanda.

2.1.1 Evolución de las redes ópticas.

Primera generación de redes ópticas.

Esta generación se caracteriza por el uso de fibra óptica únicamente para procesos de transmisión de la información de alta capacidad, es decir, los procesos de conmutación y enrutamiento se llevan a cabo en el dominio eléctrico. Las redes ópticas de primera generación son utilizadas comúnmente en redes públicas de telecomunicaciones y no difieren en gran medida de las redes que no hacen uso de fibra óptica, excepto por el empleo de altas velocidades en la transmisión de los datos.

Un problema que afecta estas redes son los cuellos de botella, esto ocurre cuando el rendimiento de la red óptica se ve afectado por un número de elementos o recursos limitados.

Segunda generación de redes ópticas.

En esta generación se realiza la adición de procesos de gestión, control y seguridad. El enrutamiento y conmutación que se efectuaba en el dominio eléctrico en la primera generación, ahora se realiza en el dominio óptico, con esto se genera una reducción de los dispositivos electrónicos en los nodos de las redes [25].

Se presenta, además, una capa adicional dentro del modelo de capas de la red, con el fin de realizar las funciones de seguridad, conmutación, enrutamiento y gestión, es decir, es la encargada de establecer caminos ópticos o conexiones origen-destino, empleando longitudes de onda para la comunicación entre nodos; esta capa se denomina, capa óptica.

Estas redes pueden presentar conversión de longitudes de onda para algunos nodos intermedios de la red, los cuales poseen dispositivos que tienen la capacidad de cambiar la longitud de onda de una señal que lo requiera, para ser enviada por un camino óptico. Lo anterior hace posible el envío de datos por varios caminos, cada uno con una longitud de onda característica, por un enlace de fibra óptica [33].

Tercera Generación de redes ópticas: Redes todo ópticas (All Optical Networks).

Son redes capaces de transportar datos desde el origen hasta el destino en el dominio óptico sin necesidad de realizar conversiones electro-ópticas u opto-electrónicas en los nodos intermedios.

2.1.2 Técnicas de multiplexación.

Un sistema de multiplexación es aquel que permite el envío de múltiples señales que proceden de canales diferentes, por un único canal físico sin interferirse entre sí. Entre las técnicas de multiplexación en el contexto de las redes ópticas se tienen: Multiplexación por División de Tiempo (TDM, Time Division Multiplexing), Multiplexación espacial (SDM, Space Division Multiplexing), y la Multiplexación por División de Longitud de onda (WDM, Wavelength Division Multiplexing), esta última la más implementada actualmente.

2.1.2.1 Multiplexación por División de Longitud de onda.

La tecnología de Multiplexación por División de Longitud de onda, aumenta la capacidad de transporte de información en redes ópticas. Esta técnica permite anexar nuevos canales ópticos a la red de acuerdo con las solicitudes de los usuarios, además, actualmente no sólo es usada para aumentar el ancho de banda, sino que también cumple con tareas de encaminamiento [34]. Esta gestión se realiza a través de los caminos ópticos (unidades básicas de las redes WDM) que permiten establecer un circuito óptico entre un par de nodos no necesariamente adyacentes, evitando así la gestión eléctrica en los nodos intermedios.

De esta forma, los procesos de gestión y enrutamiento pueden realizarse en el dominio óptico, lo que implica una ventaja para los proveedores de servicio debido a la reducción de costos, flexibilidad y grandes velocidades [35], [36].

2.1.3 Sistemas de conmutación ópticos.

Con el fin de aprovechar las ventajas de la Multiplexación por División de Longitud de Onda en cuanto a encaminamiento y reducción de procesamiento eléctrico, en las redes ópticas se presentan tres tipos de conmutación: Conmutación Óptica de Circuitos (OCS, Optical Circuit Switching), Conmutación Óptica de Paquetes (OPS, Optical Packet Switching) y Conmutación Óptica de Ráfagas (OBS, Optical Burst Switching) [38]. El funcionamiento de la conmutación óptica de circuitos se encuentra orientado a conexión, es decir, la transmisión entre nodos se realiza sobre canales WDM pre-establecidos sobre una ruta física. La conmutación de circuitos no hace uso de conversores opto-electrónicos o electro-ópticos en los nodos intermedios y no permite el uso eficiente del ancho de banda, ya que no se presenta tráfico constante en el enlace, es decir, un camino utiliza exclusivamente una longitud de onda dentro de un enlace haciendo que éste sólo sea usado en determinados momentos. Otro de los inconvenientes que presenta esta técnica es que no hay suficientes longitudes de onda para que todos los nodos estén conectados entre sí y la red no tendría una carga distribuida simétrica entre los caminos, ya que el tráfico por los caminos ópticos va variando con respecto al tiempo.

En la técnica de Conmutación Óptica de Paquetes, los datos son enviados junto con el paquete de control sin necesidad de establecer un camino óptico, por lo tanto se requieren sistemas de control, sincronización y almacenamiento de los paquetes en los nodos intermedios. La falta de dispositivos que permitan realizar estas tareas como los buffers y las memorias ópticas, hacen que OPS aún no sea totalmente viable, planteándose cómo opción a largo plazo.

La Conmutación Óptica de Ráfagas (OBS, Optical Burst Switching), reúne lo mejor de las técnicas de conmutación ya nombradas y se ha convertido en una de las técnicas más prometedoras para la transmisión de información sobre redes sólo ópticas, gracias a que genera esquemas de señalización de un solo camino, para disminuir retardos y permite el transporte de gran cantidad de carga útil [10], [39].

2.1.3.1 Conmutación Óptica de Ráfagas.

La conmutación óptica de ráfagas agrupa paquetes según parámetros específicos (como la dirección y/o clase de destino), formando ráfagas. Éstas se componen habitualmente de un paquete de control o cabecera y un conjunto de paquetes de datos, evitando de esta manera que el conmutador lea el encabezado de cada paquete y se encargue solamente de analizar el paquete de control de la ráfaga, obteniendo una disminución considerable en el retardo de los paquetes [40].

En este tipo de conmutación, la cabecera de la ráfaga es enviada por un canal de señalización fuera de banda, asignándole una longitud de onda diferente a la del canal de datos; en los conmutadores se convierte el canal de control al dominio eléctrico para realizar el proceso de reconocimiento del paquete y configurar el ancho de banda para el envío de la ráfaga de datos.

En el proceso de reconocimiento del paquete de control, es empleado un tiempo determinado [41], para poder compensar dicho tiempo se hace uso de un periodo de espera entre el envío del paquete de control y la ráfaga de datos, llamado tiempo offset, el cual también podría ser utilizado en caso de que se presente la necesidad de retardar los datos en la red.

Entre las ventajas que presenta la conmutación óptica de ráfagas sobre las otras tecnologías de conmutación anteriormente mencionadas, se tienen: la eficiencia en términos de ancho de banda; en OBS los recursos de la red son reservados para períodos cortos y específicos de tiempo, algo que es inalcanzable para OCS. Además, OPS y OBS son más adecuados para el tráfico y generan menor latencia[8] que OCS [42].

2.1.3.1.1 Arquitectura de una red por Conmutación Óptica de Ráfagas.

Una red OBS está compuesta por dos tipos de nodos: Los nodos Frontera y los nodos Centrales como se muestra en la Figura 2. Los primeros son los encargados de la multiplexación de paquetes provenientes de los clientes para ser clasificados, además se encargan de ensamblar las ráfagas, transmitir la cabecera de éstas por el canal de control y después de un tiempo offset, enviar la ráfaga de los datos; todos estos procesos son llevados a cabo en el dominio electrónico. Los nodos centrales generan una conmutación óptica rápida haciendo uso de longitudes de onda y trabajando en el dominio óptico [43].

[8]Suma de retardos temporales producidos por la demora en la propagación, procesamiento y transmisión de paquetes dentro de la red. http://ingenieria.udea.edu.co/grupos/revista/revistas/nro045/148_156.pdf

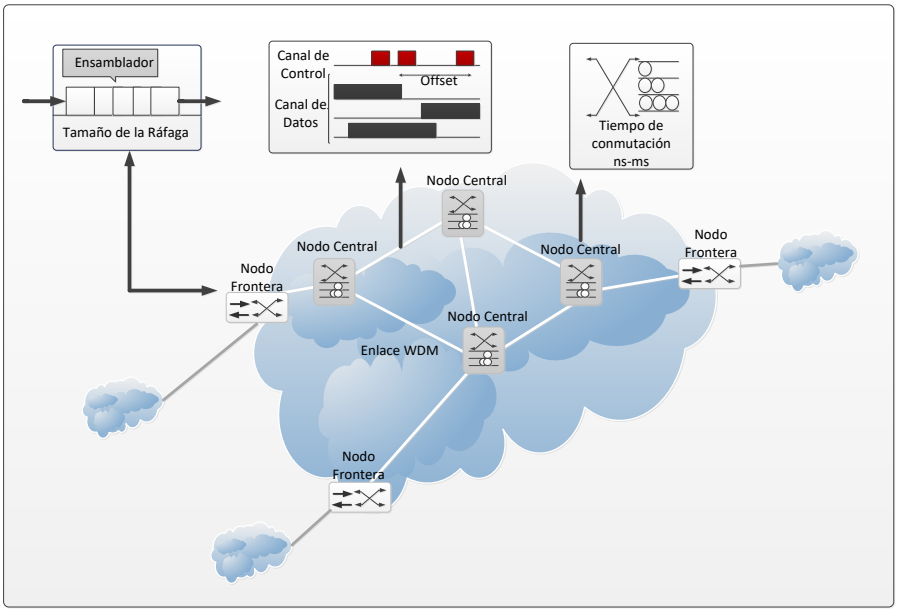


Figura 2. Arquitectura de una red OBS.

Las funciones llevadas a cabo por los nodos dentro de la red se observan en la Figura 3.

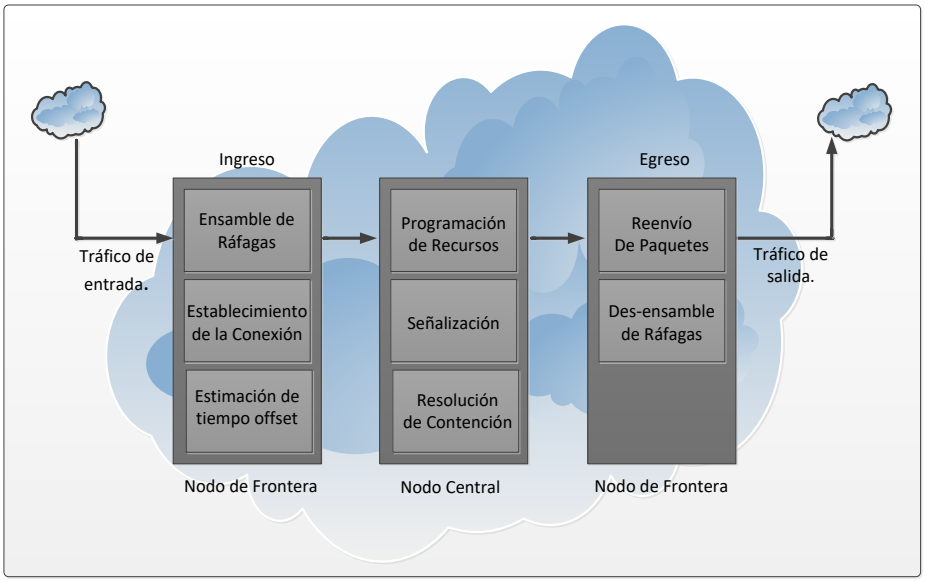


Figura 3. Funciones principales de los nodos OBS.

2.1.3.1.2 Nodo Frontera.

Los nodos frontera operan como una interfaz de entrada y/o salida de datos, se encargan de las siguientes funciones: ensamble y des-ensamble de ráfagas, establecimiento de la conexión y estimación de tiempo Offset, su arquitectura se muestra en la Figura 4.

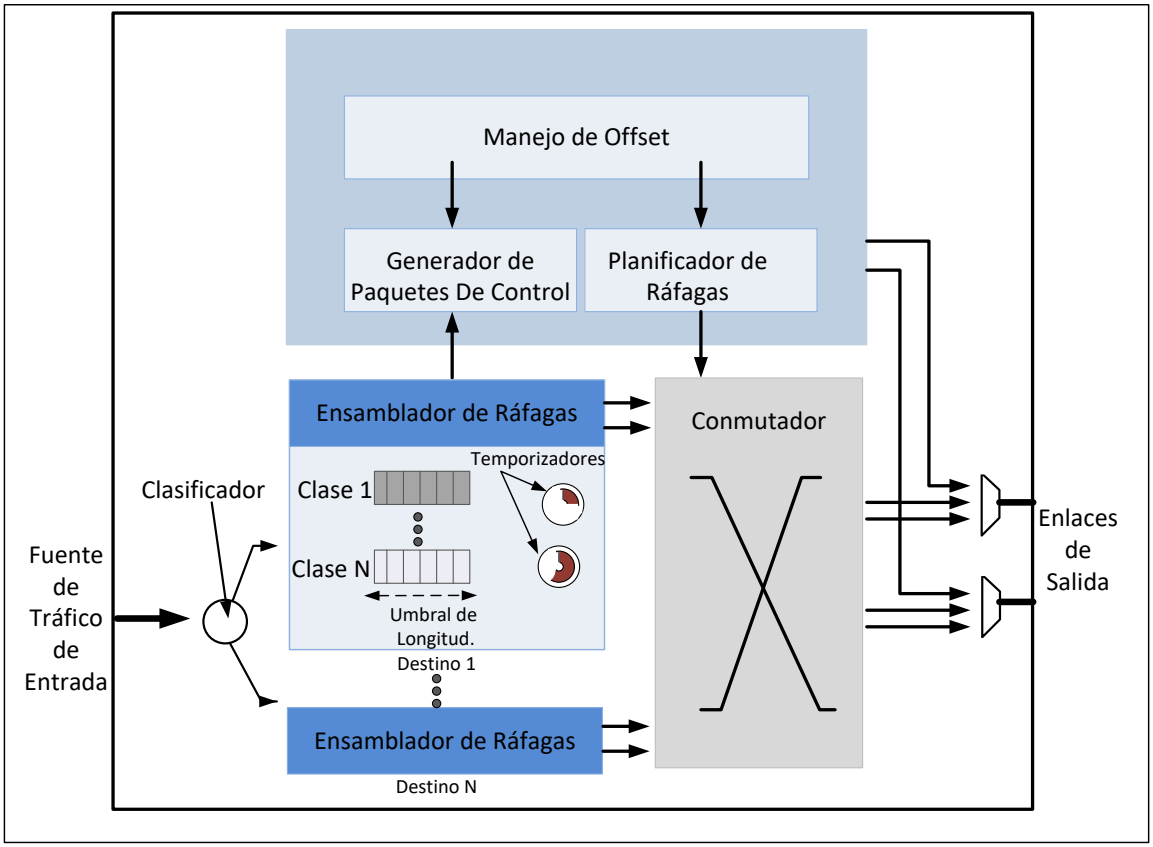


Figura 4. Nodo Frontera.

En el proceso de ensamble de ráfagas, el nodo frontera recibe múltiples paquetes procedentes de redes de usuarios, que son clasificados y agrupados en colas de acuerdo a criterios de selección como el tipo de tráfico, direcciones de destino, entre otros [44]. Una ráfaga se ensambla a partir de las colas de paquetes con ayuda de un algoritmo encargado de indicar el momento en el que se detiene el ensamble y puede enviarse la ráfaga; los algoritmos encargados de detener el proceso de agregación de paquetes hacen uso de parámetros como: el tiempo de ensamble, que es utilizado para definir cuándo se debe montar una nueva ráfaga y la longitud mínima y máxima a la que dicha ráfaga puede ser enviada [45].

Si el enfoque con el que trabaja el algoritmo para el ensamble de la ráfaga es el temporizador, se determinan períodos de tiempo constantes para el proceso de agregación de paquetes, por lo tanto las ráfagas que son transmitidas por los caminos ópticos tienen longitudes variables. Sin embargo, si una ráfaga completa el periodo de tiempo establecido para ser enviada, pero no supera la longitud mínima de ensamble, es enviada con bits de relleno.

Cuando el parámetro que tiene en cuenta el algoritmo es la longitud, se fija un número máximo de paquetes que puede contener la ráfaga antes de ser enviada.

Algunos algoritmos de montaje se basan en tiempo y longitud. Éstos establecen valores umbral para dichos parámetros, y la ráfaga es enviada cuando se cumpla uno de los dos. Así, cuando el tráfico de la red es pesado, el parámetro que se cumple es el de la longitud máxima de las ráfagas, por el contrario, si la red está manejando bajo tráfico, el parámetro límite para el ensamble es el tiempo. La elección del valor umbral para la longitud de la ráfaga y el temporizador es un factor muy importante en

el ensamble de la ráfaga para minimizar la probabilidad de la pérdida de paquetes en una red OBS [46].

Otra de las funciones del nodo frontera es la configuración de mecanismos de conexión, es decir, el establecimiento de la comunicación entre dos nodos sobre una red óptica, proceso en el cual se asignan los recursos de red requeridos para la transmisión de la ráfaga mediante la señalización, el enrutamiento y la asignación de longitudes de onda [47].

Finalmente, el nodo frontera se encuentra encargado de la estimación del tiempo offset, dicho periodo de desplazamiento permite al paquete de control de la ráfaga procesar y configurar los recursos necesarios para la transmisión de la ráfaga por un camino óptico. La estimación apropiada de este periodo de tiempo es crucial para el funcionamiento de la red y preferiblemente se encuentra basado en el número de nodos OBS, tiempos de configuración y procesamiento en cada uno de ellos, ésta es una característica clave en la configuración de una red OBS, para la asignación apropiada de recursos y bajas pérdidas de ráfagas.

2.1.3.1.3 Nodo Central.

Los nodos centrales se encuentran en el núcleo de la red OBS, permitiendo la conmutación de ráfagas de datos transparente sobre fibra óptica mediante la configuración de la matriz de conmutación del equipo para que la ráfaga salga por el puerto adecuado, su arquitectura se encuentra descrita en la Figura 5.

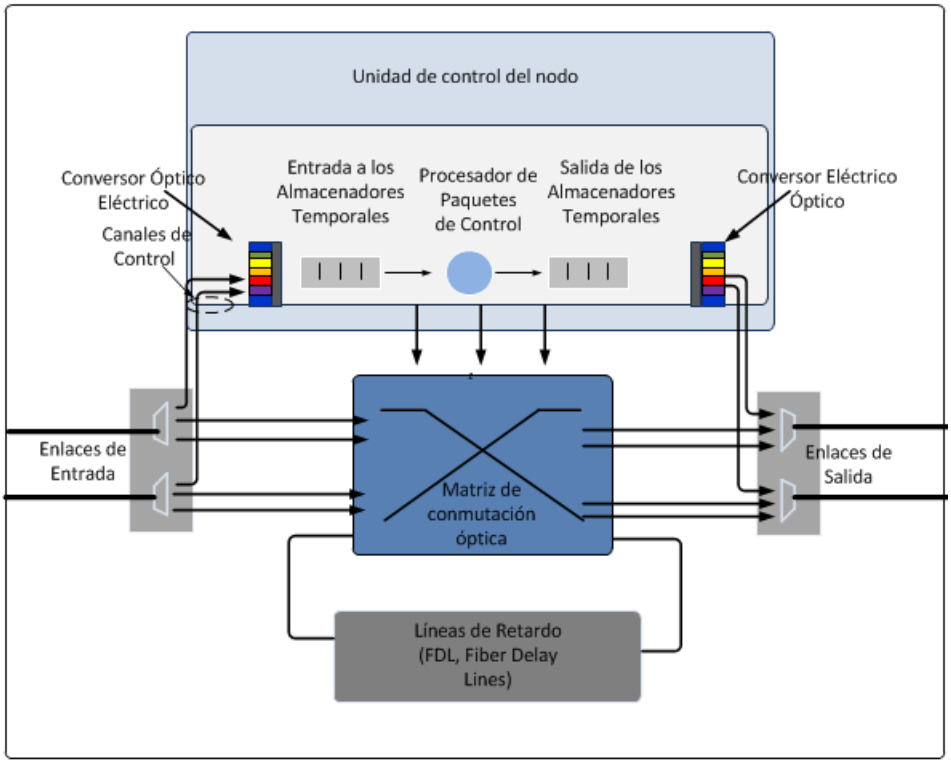


Figura 5. Nodo Central.

Un elemento importante en el nodo central, es la matriz de conmutación óptica, que permite la conmutación de las ráfagas de una fibra de entrada a una de salida. Dicha matriz se modifica automáticamente desde la unidad de control al igual que su velocidad, lo cual es clave para lograr una

buena utilización del canal; otros elementos importantes en el nodo central son los conversores óptico-electrónicos ya que el reconocimiento y manejo de los paquetes de control se realiza en el dominio electrónico y la información contenida en dichos paquetes es necesaria para procesos de señalización, como planificación de ráfagas de datos y solución a problemas de contención, llevados a cabo en la unidad de control del nodo [6], [36].

2.1.3.1.4 Señalización.

La señalización permite reservar el camino óptico por el que se va a transmitir una ráfaga, además especifica cuáles son los protocolos que permiten la comunicación entre nodos OBS, determina el funcionamiento de la red y si los recursos están siendo utilizados eficientemente. Este proceso se lleva a cabo mediante un canal de longitud de onda dedicada [48].

La señalización puede trabajar de manera distribuida o centralizada. En la primera, la reserva se realiza nodo a nodo; en la segunda, será un elemento central con conocimiento de toda la red el que se encarga de enviar las solicitudes de reserva de recursos a los nodos involucrados. Cabe señalar que este trabajo de grado usa señalización distribuida.

Se describen tres tipos de protocolos para la señalización en conmutación óptica de ráfagas que se clasifican según la liberación de recursos: Terminador en Banda (IBT, In Band Terminator), Recomienda y sigue (TAG, Tell And Go) y Reserva de duración fija (RDF, Reserve a Fixed Duration) [45].

2.1.3.1.5 Protocolos de reserva en conmutación óptica de ráfagas.

En el protocolo IBT, cada ráfaga contiene una cabecera y un parámetro que indica su final, este protocolo deja que un nodo intermedio envíe el principio de una ráfaga sin que haya llegado el final de la anterior, permitiendo así menos retardos [45], [21].

Para TAG, basado en OBS, una fuente envía un paquete de control por el canal que lleva el mismo nombre con el fin de establecer los parámetros necesarios como el ancho de banda para enviar los datos y liberar recursos [10].

RFD (Reserve a Fixed Duration), a diferencia de los dos protocolos anteriores, sólo ha sido estudiado para redes ópticas. Al igual que TAG, este protocolo envía inicialmente un paquete de control que lleva información del ancho de banda necesario para la transmisión de la ráfaga y, adicionalmente, contiene el tiempo de duración de la misma, así que permite la reserva del ancho de banda por un periodo determinado.

Los dos protocolos más atractivos para la señalización en redes OBS son JIT (Just In Time) basado en TAG y JET (Just Enough Time) basado en RFD. Cuando se está trabajando con el protocolo JIT, un nodo OBS configura los conmutadores ópticos para la ráfaga; tan pronto como es recibido y procesado el paquete de control correspondiente, los recursos del nodo se ponen a disposición antes de la llegada de la ráfaga sin tener en cuenta tiempos offset, lo que permite la simplicidad del sistema de señalización, pero también el uso ineficiente de los recursos de la red.

JET basado en RFD, es el protocolo más utilizado en redes OBS, se presenta como el más atractivo frente a los otros protocolos, debido a que muestra una mejor utilización del ancho de banda al realizar la reserva de recursos únicamente desde la llegada de la ráfaga hasta que es conmutada y no desde la llegada del paquete de control al nodo. Este protocolo es el escogido para implementar las redes OBS/WDM basadas en algoritmos genéticos con y sin métodos de control cognitivo.

2.1.3.1.5.1 Protocolo JET.

Este protocolo envía la cabecera de una ráfaga por el canal de control, reserva el ancho de banda necesario para transmitir una ráfaga en un tiempo definido que sea proporcional al tamaño de esta y permite la liberación de recursos. El valor del periodo offset que se encuentra en la cabecera, más el tiempo de proceso de la misma definido por el nodo anterior para el nodo actual, indica el tiempo que tarda la ráfaga en llegar. Cuando el proceso de reserva de ancho de banda realizado por el Paquete de Control (BCP, Burst Control Packet) es exitoso, dicho paquete define el tiempo offset para el nodo siguiente y es enviado; si dicho proceso no fue exitoso, entonces se bloquea y se descarta la ráfaga o se hace uso de una fibra de retardo para que la ráfaga no sea bloqueada hasta que sea posible realizar la reserva de ancho de banda. Dicho proceso puede observarse detalladamente en la Figura 6 [10].

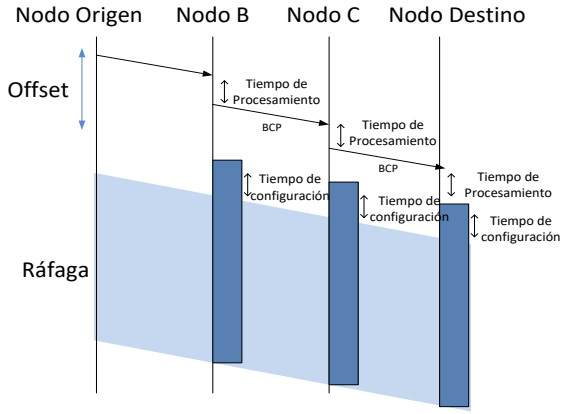


Figura 6. Funcionamiento del protocolo JET.

Gracias al protocolo de enrutamiento JET, se optimiza la reserva de recursos para las ráfagas que atraviesan la red, no obstante, al no existir mecanismos centralizados que coordinen los envíos, o técnicas que permitan conocer sobre la reserva de recursos, se usan comúnmente métodos en el nodo central que contrarrestan la acción hecha por varias ráfagas de solicitar los mismos recursos al mismo tiempo [47] [36], este problema es conocido como contención y las técnicas más usadas para evitarla son:

Deflexión: la ráfaga se envía por otro camino si el que tenía se encuentra ocupado. Este método puede realizarse en tres dominios distintos [10]:

- Dominio de longitud de onda: la ráfaga se envía por otro camino óptico, usando conversores de longitud de onda.
- Dominio espacial: la ráfaga es enviada por otro puerto y su ruta cambia por una alternativa.
- Dominio temporal: se evita la pérdida de una ráfaga que no puede ser transmitida, usando fibras de retardo (FDL, Fiber Delay Lines) para retrasar su salida por un tiempo determinado.

Descarte: si una ráfaga no puede ser enviada a su destino porque en su camino no encuentra algún recurso disponible (longitud de onda, puerto, fibra), la ráfaga es descartada. Para el diseño de las redes analizadas se incluyó este método cómo mecanismo de contención.

Segmentación: para evitar la pérdida de ráfagas, es posible fraccionarlas en segmentos, enviándolos por otros caminos o desechándolos.

2.1.4 Enrutamiento y asignación de longitud de onda en redes OBS/WDM.

2.1.4.1 Enrutamiento y Asignación de Longitud de Onda.

El RWA es el proceso de encaminar la información y asignarle una longitud de onda adecuada en un entorno de comunicación óptica, en dicho proceso se establecen caminos de luz entre un par origen-destino, es decir, se calcula una ruta para la información a través de la topología física y se asigna una longitud de onda. Lo anterior se realiza intentando minimizar al máximo los recursos, como el número de fibras o el número de longitudes de onda, basándose en la información de estado de la red [10].

2.1.4.2 Metodologías de enrutamiento y asignación de Longitud de Onda.

A lo largo de la historia se han presentado tres estrategias heurísticas de encaminamiento aplicadas a redes ópticas OBS, dichas tendencias difieren en el nivel de manejo de la información de estado de la red y son encaminamiento fijo, encaminamiento fijo alternado y encaminamiento adaptativo. Mientras algunos algoritmos representativos de dichas estrategias pueden resolver únicamente el enrutamiento o la asignación de longitudes de onda, otros pueden darle solución de manera conjunta.

En la técnica de encaminamiento fijo, la ruta es calculada una sola vez. Si dicha ruta llegase a fallar o no es posible establecer el camino óptico se rechaza la petición. Esta técnica busca disminuir el tiempo de establecimiento de conexión, pero se arriesga a tener una probabilidad de bloqueo mayor. Dentro de las técnicas de encaminamiento clásicas fijas se tienen:

Algoritmos del Camino más Corto (SP, Shortest Path), busca el mínimo consumo de recursos; Algoritmo Peso del Salto (HW, Hop Weight), su objetivo es minimizar la probabilidad de bloqueo, eligiendo el camino con el menor número de saltos; algoritmo de la Menor Distancia (DW, Distance Weight), con esta técnica se obtiene una reducción en los retardos ya que se selecciona el camino de menor distancia.

La estrategia de encaminamiento fijo-alternado, establece un número determinado de caminos ópticos, calculados a través de una función de costo fija, si el primer camino no se logra establecer, se sigue intentando con los otros caminos hasta agotarlos todos. Con esta técnica se obtiene una mejora en la capacidad de red y en la probabilidad de bloqueo, aunque con un tiempo de conexión más largo debido al número de rutas que se calculan, un ejemplo de esta estrategia es el algoritmo Mínimo Número de Saltos (MNH, Minimum Number of Hops), que permite disminuir la probabilidad de bloqueo asignando inicialmente la longitud de onda a las rutas más cortas.

Cuando se trabajan técnicas de encaminamiento adaptativo, el establecimiento de caminos ópticos se lleva a cabo en tiempo real, es decir, las conexiones se actualizan al establecerse o terminar una conexión. Esta técnica trabaja creando enlaces dinámicamente, debido a esto se requiere un tiempo mayor para generar el camino óptico, pero permite minimizar la probabilidad de bloqueo. Algunos algoritmos representativos de éste tipo de técnica son: Longitudes de Onda Disponibles (AW, Available Wavelengths), Saltos y Longitudes de Onda Disponibles (HAW, Hop and Available Wavelengths), Longitudes de Onda Totales y Disponibles (TAW, Total and Available Wavelengths), Número de saltos, Longitudes de Onda Totales y Disponibles (HTAW, Hop Count and Total Wavelengths and Available Wavelengths), Costo Futuro Modificado (MFC, Modified Future Cost), Enrutamiento de menor congestión (LCPR, Least Congested Path Routing), Balance de Carga de Enrutamiento y Asignaciones

de Onda (LBRWA, Load Balance RWA), Desviación de Carga de la Peor Ruta (HPLD, Heaviest Path Load Deviation) [21].

El algoritmo de Encaminamiento de Menor Congestión, busca disminuir la sobrecarga en los enlaces. Otro de los algoritmos que trabajan de manera adaptativa es el de Costo Futuro Modificado, cuyo objetivo es mejorar la probabilidad de bloqueo de la red.

Por otra parte, la asignación de longitudes de onda puede realizarse de manera paralela al encaminamiento; las técnicas para la asignación de longitudes de onda se efectúan de tres maneras: organizando las longitudes de acuerdo al orden en el que fueron usadas en la última ocasión, realizando una elección aleatoria, o haciendo uso de expansión o contracción del espectro de las longitudes de ondas que han sido usadas en los enlaces.

La elección del algoritmo que se utiliza depende del objetivo deseado, del número de longitudes de onda usadas sobre las disponibles (también llamado carga de la red) y de la topología.

Algunos algoritmos clásicos usados para la asignación de longitud de onda son: la Técnica Aleatoria (R, Random), Algoritmo de primer ajuste (FF, First-Fit), Ultimo Canal Libre Disponible (LAUC, Latest Available Unused Channel), Menos Usado (LU, Least-Used), Más Usado (MU, Most-Used), Producto Mínimo (MP, Min-Product), Menor Carga (LL, Least-Loaded), Mínima Suma (MS, Min-Sum), Máxima Suma (MΣ, Max-Sum), Reserva de Longtudes de Onda (WR, Wavelength Reservation), Protecció Umbral (TP, Threshold Protection), entre otros [48], [49].

Entre los algoritmos mencionados anteriormente se describen a continuación algunos que son aplicados en redes que usan enlaces monofibra, y cuyo objetivo es reducir la probabilidad de bloqueo de la red.

En la técnica Aleatoria (R, Random), se buscan longitudes de onda disponibles para el enlace y se elige una aleatoriamente para el establecimiento del camino óptico, con el Algoritmo de Primer Ajuste (FF, First-Fit), se realiza una numeración a las longitudes de onda existentes, cuando se desea realizar una conexión se elige entre dichas longitudes la primera que se encuentre disponible empezando desde la de menor numeración hasta la de mayor numeración, el algoritmo basado en horizonte[9] llamado: Último canal libre disponible (LAUC, Latest Available Unused Channel), donde el planificador guarda la información del horizonte de cada longitud de onda para disminuir al máximo las brechas de ancho de banda generadas por los tiempos de reserva, la técnica para la asignación de longitudes de onda de máxima suma, (MΣ, Max-Sum), permite la optimización de la capacidad de los enlaces haciendo una reserva de recursos, eligiendo una longitud de onda que haga posible disminuir la perdida de dicha capacidad en los enlaces. Ahora bien el algoritmo de mínima suma, (MS, Min-Sum), permite disminuir el número de fibras sobre la red [50], [48].

2.2 ALGORITMOS EVOLUTIVOS.

2.2.1 Introducción.

Los seres humanos a lo largo de su historia han optado por plasmar principios y características de la naturaleza dentro sus invenciones, tal es el caso del avión, como su acrónimo en francés lo

[9]Se define como el instante temporal hasta el cual hay alguna ráfaga planificada en el canal [48].

indica: **A**ppareil **V**olant **I**mitantl'**O**iseau **N**aturel ("aparato volador que imita el ave natural"); el submarino, cuya aerodinámica es similar a la de los peces, o el radar, aparato que presenta un sistema de funcionamiento muy similar al mecanismo usado por los murciélagos para esquivar obstáculos y/o conseguir alimento; como estos, se presentan muchos otros casos [51].

Por otra parte, muchas de las disciplinas científicas presentan problemas, en su intento por generar sistemas novedosos o explorar métodos y procedimientos en búsqueda de la solución a la cual quieren llegar [51]. Tal es el caso del RWA en redes de comunicaciones, no obstante, entre los enfoques para hallar una solución adecuada a este problema, se encuentran los procesos de búsqueda y optimización, los cuales evolucionan después de un número de iteraciones. Dichos mecanismos se denominan Algoritmos Evolutivos [51].

Dentro de las características más destacadas de los Algoritmos Evolutivos se encuentra la base poblacional, es decir, estos algoritmos trabajan con individuos pertenecientes a una población específica. Ahora bien, es posible abstraer las características de cada uno de estos individuos, representándolas mediante un código. Al mismo tiempo, poseen un indicador que mide la bondad (aptitud) de cada individuo, este valor es generado gracias a la Función de Aptitud [52], [53], la cual se presenta como otra característica relevante debido a la importancia de contar con soluciones idóneas, garantizando los procesos de optimización al igual que la convergencia de los algoritmos. Como última característica a resaltar, estos algoritmos deben asegurar que los individuos experimenten etapas e iteraciones imitando el proceso evolutivo con el fin de delimitar el espacio de la solución [51].

La Figura 7 presenta las características que todo algoritmo evolutivo debe tener.

Figura 7. Características de los Algoritmos Evolutivos [51].

2.2.2 Métodos Heurísticos y Metaheurísticos.

Debido a la naturaleza de los problemas de enrutamiento y asignación de longitud de onda [14] [15], algoritmos como los evolutivos, no permiten establecer una solución exacta encontrando el valor óptimo en el menor tiempo posible, sin embargo, permiten encontrar una respuesta muy aproximada a la óptima, que asegure buen rendimiento y calidad de acuerdo a los recursos usados. Los algoritmos heurísticos son los responsables de generar este tipo de soluciones en un tiempo relativamente corto. Además, atendiendo a su clasificación, pertenecen a métodos de búsqueda y optimización, es decir, comienzan a partir de una solución e intentan mejorarla. No obstante, una de las desventajas que presentan, es su imposibilidad para salir de óptimos locales, en otras palabras, hallan una solución dentro de una vecindad y no en el entorno global. La Figura 8 muestra este problema.

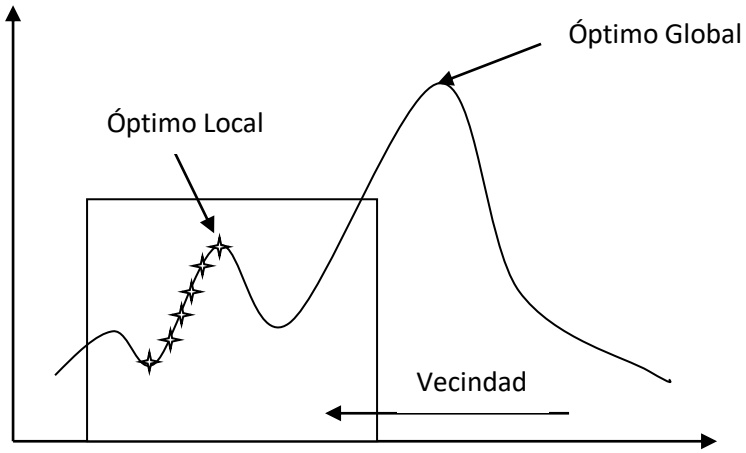


Figura 8. Algoritmo Heurístico encasillado dentro de un óptimo local.

No existe forma de evitar que los algoritmos heurísticos opten por óptimos locales como solución, es por esto que se proponen algoritmos metaheurísticos como mecanismos de alto nivel que evitan este inconveniente y además pueden ser utilizados en diversos problemas con tan sólo cambiar algunas de sus propiedades [43]. Los algoritmos evolutivos, específicamente los Algoritmos Genéticos (se tratará más adelante) pertenecen a este último grupo.

Estas técnicas de alto nivel para la resolución de problemas resultan adecuadas al implementarlas en situaciones reales cuyo modelo no es posible representar mediante una función lineal. De los inconvenientes presentados al realizar el enrutamiento y asignación de longitud de onda dentro de la planificación en redes ópticas, surge una oportunidad para enfocar este problema por medio de técnicas heurísticas [55].

Además, características como la robustez, la adaptabilidad y la generalidad hacen de estos métodos, opciones aplicables a diversos entornos de la vida real, con la ventaja de no tener que incurrir en modificaciones excesivas para su implementación [55].

2.2.3 Problemas P, NP Y NP-Completo.

Para hablar de este tema es necesario profundizar sobre los conceptos relacionados con la solución de problemas y su caracterización dependiendo del grado de dificultad y del tiempo que le toma a un computador resolverlo (Teoría de la complejidad computacional) [56].

Los problemas que pueden ser resueltos mediante un algoritmo en tiempo polinómico y cuya solución es proporcional a los datos de entrada, se denominan problemas determinísticamente polinómicos o problemas tipo P [57].

Ahora bien, existen problemas no solucionables en tiempo polinomial, pero dada una instancia que represente una solución particular, es posible comprobar en tiempo polinómico que efectivamente dicho valor representa una solución. Estos problemas se denominan problemas no determinísticamente polinómicos, o problemas tipo NP. Supuesto esto, dado un problema de decisión X, el cual sea NP, si existe algún algoritmo polinomial que transforme todos los problemas no determinísticamente polinómicos en X, se dice que X es un problema NP-completo [57], [51]. El enrutamiento y la asignación de longitud de onda en redes es un problema NP-completo.

2.2.4 Algoritmos Genéticos.

Los AG (Algoritmos Genéticos) son procedimientos metaheurísticos derivados de los algoritmos evolutivos y formulados por John Holland en 1975 [58], capaces de generar soluciones de optimización basándose en los procesos de selección natural y evolución postulados por Charles Darwin [59]. Las etapas que debe seguir todo AG están relacionadas con los procesos genéticos de los seres vivos. El espacio inicial contiene individuos pertenecientes a una población, los cuales representan potenciales soluciones al problema planteado, posteriormente, con ayuda de una función que permite evaluar la aptitud de cada solución y usando operadores genéticos como la selección, el cruce y la mutación, la población evoluciona para formar así un nuevo conjunto de soluciones más apto que el anterior, este proceso se repite varias veces hasta que se cumpla con un criterio de parada [60].

Por otra parte, los algoritmos genéticos expresan a cada individuo mediante un código, usando para esto representaciones binarias, reales, matrices, en árbol, con permutaciones, entre otras [55]. Con el fin de facilitar la explicación de este apartado, se usarán codificaciones binarias, atendiendo además a las ventajas computacionales asociadas con esta clase de código [55]. En este orden de ideas los AG se caracterizan de la siguiente manera, Figura 9.

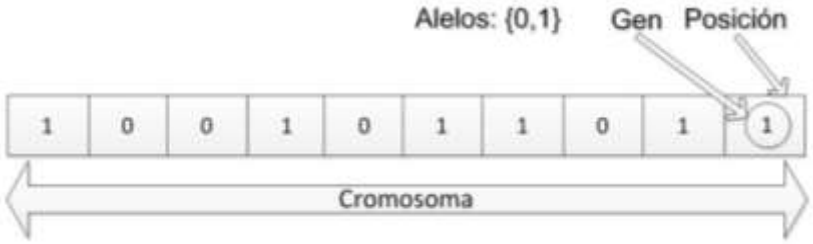


Figura 9. Estructura de un algoritmo genético clásico.

- El conjunto total de números binarios que forman la cadena de valores se conoce como cromosoma.
- Cada uno de los valores dentro del cromosoma se denomina gen.
- El valor que tomen los genes (ya sea 1, 0 u otro valor real) se denomina Alelo.

La anterior descripción sobre los algoritmos genéticos permite diferenciarlos de otros métodos de búsqueda y optimización en los siguientes aspectos.

- Los parámetros con los cuales trabajan son codificados.
- La búsqueda de la solución comienza con una población generada de manera aleatoria.
- Usan una función de aptitud orientada a la optimización.
- Usan métodos probabilísticos como reglas de transición.

2.2.5 Generación de individuos aleatorios.

Inicialmente, antes de usar los operadores genéticos, es necesario contar con una población a la cual aplicárselos. La población habitualmente se escoge generando individuos de forma aleatoria, los cuales representan soluciones aptas, más no las mejores; sin embargo, esto es lo más recomendable ya que no se aconseja obtener la población inicial mediante procesos de optimización u otros métodos heurísticos, porque puede ocurrir que el algoritmo genético converja prematuramente, obteniendo posiblemente soluciones dentro de óptimos locales.

2.2.6 Función de aptitud.

Dentro del comportamiento de los algoritmos genéticos, uno de los aspectos trascendentales es la creación de una función que permita sintetizar, en un factor de calidad, la aproximación de una posible solución a un objetivo conjunto.

La función de aptitud permite evaluar qué tan bueno es determinado individuo para alcanzar un punto óptimo, esto con el fin de realizar la selección de los padres que formarán la próxima generación de soluciones al problema. De esta forma, como sucede en la vida real, los individuos con mejores características, serán capaces de generar una mayor cantidad de descendientes, permitiendo que estas propiedades se transmitan a generaciones futuras [61], [28], [52].

Lo más importante de una función de aptitud es que debe reflejar de manera "real" valores de individuos que se acerquen al óptimo global, aunque cabe la posibilidad de obtener individuos no válidos para el espacio de búsqueda.

2.2.7 Operadores Selección, Cruce y Mutación.

Los algoritmos genéticos se llevan a cabo mediante operadores que garantizan la evolución de los individuos con el paso de las generaciones. Por lo tanto, si un individuo sobresale entre los demás por su capacidad de adaptación, es muy probable que sea escogido para reproducirse con otro individuo seleccionado de forma similar.

El primer operador que actúa en el algoritmo genético se denomina Selección y es el encargado de elegir un número de individuos (generados como se explica en el apartado 2.2.5.), los cuales harán parte de la población inicial para su posterior reproducción. Algunas formas de seleccionar los individuos incluyen: estrategias que involucran la generación de soluciones aleatorias mediante técnicas de distribución uniforme, la técnica de la Ruleta de Rueda, ventanas, competencias o procedimientos adaptados al problema que se pretende resolver [62], [63], [64]. En el siguiente capítulo se profundizará sobre el método escogido para el desarrollo del algoritmo genético en redes ópticas OBS/WDM.

Luego de seleccionar los individuos, comienza la tarea del siguiente Operador. El cruce se realiza de acuerdo a una probabilidad denominada Pc, que indica la relación entre el número de hijos, y el tamaño total de la población; del valor que tome Pc, dependerá la búsqueda adecuada dentro del espacio de soluciones y el tiempo de procesamiento computacional [58].

Una de las técnicas más usadas para intercambiar la información genética entre dos individuos se denomina operador cruce basado en un punto [58] y se efectúa dividiendo aleatoriamente las ristras de dos cromosomas padres para obtener dos cadenas iniciales y finales, éstas últimas se intercambian entre sí obteniendo dos cromosomas nuevos correspondientes a los descendientes o hijos, los cuales entrarán a formar parte de la población y de una nueva generación, como muestra la Figura 10 [65].

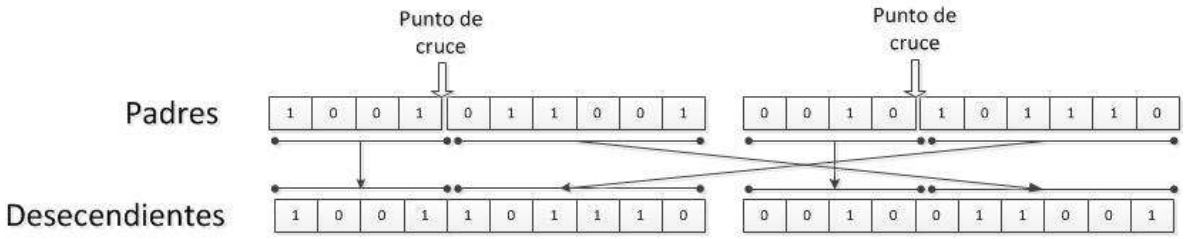


Figura 10. Operador cruce basado en un punto en cadenas de longitud l = 10.

Al generar descendientes, tal y como sucede en la vida real, algunos individuos tienden a mutar sus genes variando así su material genético, en el caso de los AG, este proceso se realiza para ampliar el espacio de búsqueda, obteniendo diversidad en la población, lo cual permite aumentar el conjunto de soluciones [52].

La mutación se encarga de variar un gen de uno o más cromosomas en lapsos separados a lo largo de la población como muestra la Figura 11.

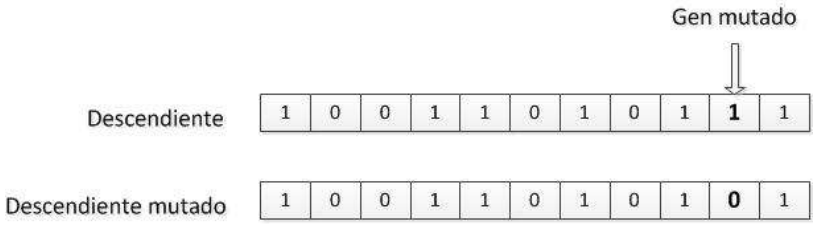


Figura 11. Operador mutación.

Habiendo aplicado el Algoritmo Genético de forma correcta, se espera que la población converja hacía un óptimo Global, es decir, que después de las iteraciones realizadas se encuentre la solución más adecuada y cercana a la correcta para el problema tratado**.**

2.3 CONTROL COGNITIVO EN REDES ÓPTICAS.

2.3.1 Introducción.

Para comprender el significado de los métodos de control cognitivo, es necesario hablar sobre las redes cognitivas y las funcionalidades que aportan a los entornos de comunicación. En este sentido, las redes cognitivas son sistemas autoconscientes de su estado y entorno, capaces de observar, planear, decidir, aprender, adaptarse a los cambios y modificar sus funciones dependiendo de los requerimientos del usuario o de las condiciones actuales de la red, todo lo anterior teniendo en cuenta objetivos extremo a extremo. Ahora bien, los métodos de control cognitivo hacen referencia a los procesos implementados en las redes para que éstas adquieran características propias de una red cognitiva [30].

Los avances en cuanto a la implementación de estas redes incluyen modelos con transformación a nivel físico, es decir, dispositivos o tecnología cognitiva capaz de administrar la red; diseño de una capa cognitiva que actúa teniendo en cuenta objetivos globales, brindando además seguridad, calidad de servicio y manejo adecuado de recursos; comunicación entre capas no adyacentes del Modelo de Interconexión de Sistemas Abiertos (OSI, Open System Interconnection) con el fin de aumentar el

desempeño; retroalimentación a través de un módulo de memoria, realizado para adaptar y optimizar la red de acuerdo a los cambios efectuados en el entorno; tecnología radio-cognitiva, entre otros [30].

2.3.2 Redes inspiradas biológicamente.

En términos evolutivos, los enfoques inspirados biológicamente no sólo se usan como estrategias de búsqueda y optimización en el área de RWA. Actualmente, redes ópticas globales pueden involucrar características provenientes de inspiración biológica, valiéndose de entornos dinámicos, en donde los nodos puedan movilizarse y no haya un control centralizado de la red. Adicionalmente, esta evolución debe cumplir con criterios de escalabilidad, robustez y diversidad [60], [71], que garanticen el crecimiento y el adecuado flujo de tráfico dentro de la red. En este sentido, los enfoques inspirados biológicamente presentan estrategias de auto-organización y auto-adaptación que se adecuan a las exigencias de este tipo de redes. Así mismo, son capaces de reaccionar positivamente ante errores críticos dada su naturaleza fluctuante y aleatoria, exhiben características de retroalimentación positiva y negativa, permitiendo a la red evolucionar, mantener una estructura sólida, y controlando la incidencia de previas adaptaciones erróneas, respectivamente.
Cabe aclarar que el enfoque de la presente monografía se encuentra orientado a optimizar los procesos de enrutamiento y asignación de longitud de onda en redes ópticas OBS/WDM con ayuda de métodos inspirados biológicamente, específicamente, usando algoritmos genéticos. Sin embargo, es pertinente nombrar que estos métodos inspirados biológicamente pueden comprender un uso más amplio, formando parte de la estructura de la redes cognitivas y abarcando aspectos que involucren la mayoría o en los mejores casos, todas las capas de la red.

2.3.3 Arquitectura en redes ópticas cognitivas.

Se dice que una red óptica es consciente de su entorno cuando es capaz de recopilar información proveniente de alguna(s) de sus capas y usarla para establecer tareas de gestión o mantener la información actualizada del estado de la red. Al hablar de capas, se hace implícito afirmar que las redes ópticas cognitivas presentan una arquitectura definida; ahora bien, es necesario realizar una descripción detallada de esta arquitectura para establecer a qué área pertenece cada función que se realice dentro de la red óptica.

La inmersión en la arquitectura comienza cuando un usuario o servicio desea establecer una conexión con el fin de gestionar los datos hacía un destino específico, usando para esto determinados servicios. Antes de mandar esta petición, la Capa de Requerimientos (RL, Requeriments Layer) ya debe conocer los objetivos “extremo a extremo” de la red. Esta capa es la responsable de abstraer dichos objetivos y expresarlos en un sentido lógico con ayuda del conocimiento que posee del entorno, sus dispositivos y las aplicaciones.

La capa de Aplicación (AL, Aplication Layer) se enfoca en realizar tareas de codificación/decodificación, distribución de datos, compresión de aplicaciones multimedia, y gestión en aplicaciones distribuidas como es el caso de la computación en la nube (referencia ICTON). Todo lo anterior es realizado en un contexto cognitivo, es decir, los aspectos mencionados varían o se adaptan de acuerdo a los requerimientos de la red.

Posteriormente se encuentra la capa de Servicio (SL, Service Layer), la cual permite implementar tareas de auto-optimización y auto-configuración en aspectos como la composición de servicios, la virtualización[10], o abstracción de recursos provenientes de los dispositivos de la red.

[10]Creación a través de software una versión de algún recurso tecnológico, como puede ser una plataforma de hardware, un sistema operativo, entre otros recursos de red. http://docs.oracle.com/cd/E26921_01/html/E25833/gfkbw.html

El plano de control (CP, Control Plane) involucra protocolos y algoritmos de enrutamiento y señalización que pueden adecuar su estructura, condición o estado según los requerimientos de las aplicaciones. En esta capa también se brindan mecanismos para adaptar la topología de la red.

Ahora bien, la capa de enlace de datos o MAC, por sus siglas en inglés, brinda la posibilidad de reconfigurar los protocolos y algoritmos de enlace de datos, al mismo tiempo, implementa un canal cognitivo y ofrece un formato de transporte de información. Las memorias hardware son un ejemplo de elementos reconfigurables añadidos a la red con el fin de implementar nuevos aspectos operacionales.

Por último se encuentra la capa Física (PL, Physical Layer), compuesta por módulos tanto hardware como Software, capaces de auto-implementar medidas de adaptación, configuración y optimización teniendo en cuenta las necesidades planteadas por los proveedores de servicio, las aplicaciones, el usuario y/o la infraestructura de red. El formato de modulación, la tasa de bits, la ganancia de amplificación y el número de longitudes de onda usadas, son algunos de los parámetros que pueden verse modificados en esta capa cognitiva. PL se encarga al mismo tiempo de analizar dispositivo por dispositivo, con el fin de reparar posibles fallas, o adecuar la configuración para lograr la calidad de servicio (QoS, Quality of service) requerida por la red.

Este tipo de redes implementa, además del despliegue de servicios por cada capa, un módulo cognitivo de optimización multicapa (Cognitiv Cross-layer Optimization) que interactúa con el Ciclo Cognitivo (Observar, Orientar, Planear, Decidir, Aprender, Actuar) de una o más capas, para proveer beneficios de desempeño y calidad de servicio.

La Figura 12 muestra la información descrita anteriormente.

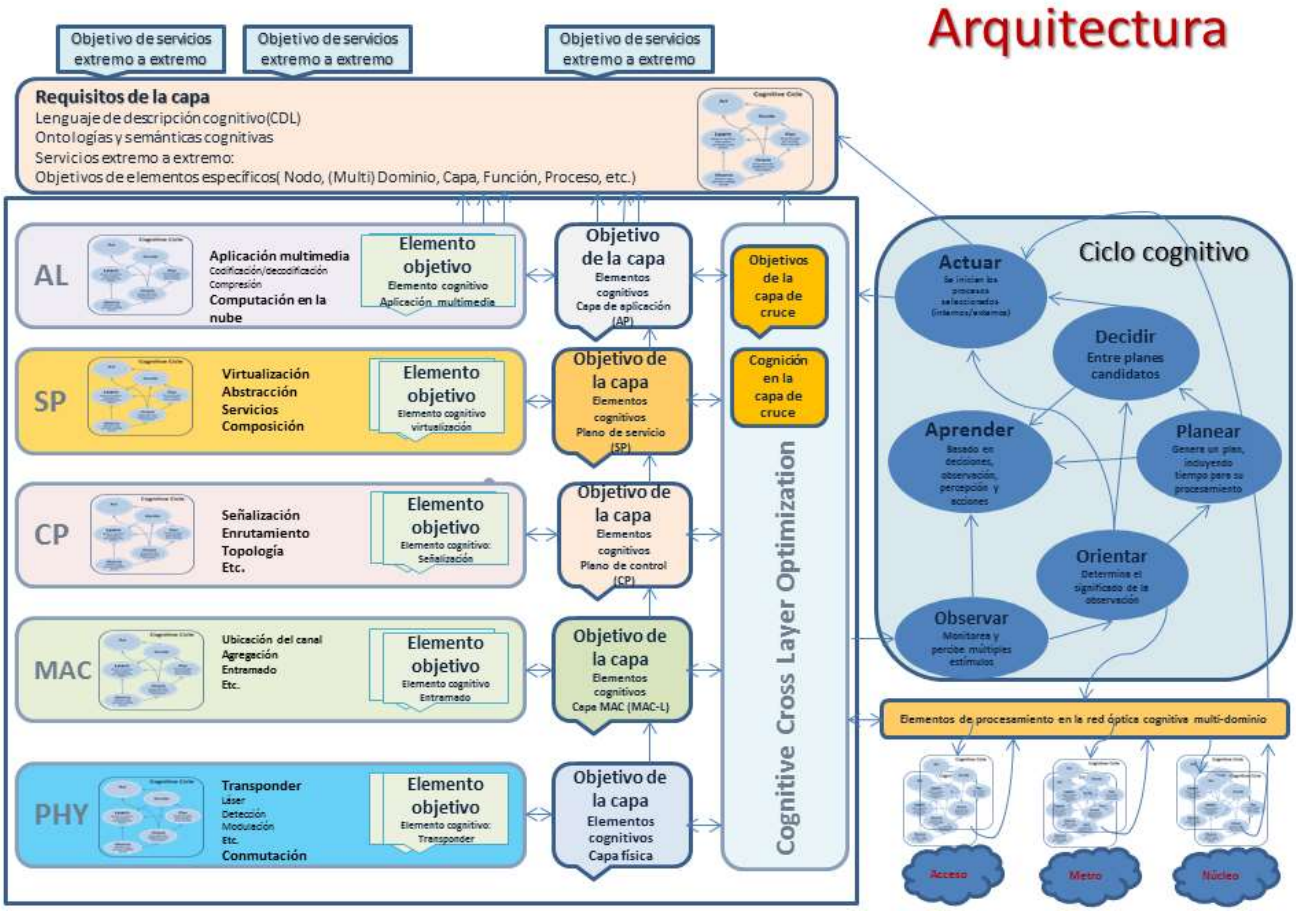


Figura 12. Arquitectura de una red óptica Cognitiva [73].

2.3.4 Ciclo Cognitivo.

En este apartado se explican con mayor detalle las funciones del Ciclo Cognitivo dentro de las redes ópticas, teniendo en cuenta que este ciclo se puede aplicar ya sea en uno, o varios elementos por capa.

El primer estado es la observación. Esta etapa Descubre y supervisa estímulos provenientes del entorno de la red, lo cual puede hacer referencia a enlaces, nodos, capas, caminos ópticos, longitudes de onda, dispositivos, entre otros. Posteriormente, en la etapa Orientar, se establece el significado de la información proveniente de la observación. En seguida, y por medio de la etapa de Planeación (Planear), dicha información adquiere un objetivo particular, el cual se puede ejecutar a largo plazo si primero es pertinente escoger entre un conjunto de planes candidatos, el que más se adecue a los requerimientos de la red; esta escogencia se realiza a través de la etapa de decisión (Decidir). Finalmente, gracias a la información previa, el ciclo se dispone a Actuar.

La figura 13 muestra cómo interactúa el Sistema de Control Cognitivo (Ciclo Cognitivo). Tal como se aprecia, este sistema se retroalimenta de las demandas de servicio y de tráfico generadas, así como del estado actual de la red a través del sistema de monitoreo [20].

Por otra parte, el subsistema denominado Base de Conocimiento (Knowledge Base) permite almacenar información correspondiente a estados anteriores en la red, las decisiones tomadas y las respuestas arrojadas por dichas decisiones. Esto resalta la importancia de este módulo, ya que brinda la capacidad de analizar los acontecimientos pasados de la red con el fin de tomar futuras decisiones que puedan influir sobre el mismo escenario de manera positiva [20]. La Figura 13 muestra los subsistemas para la realización de tareas del ciclo cognitivo.

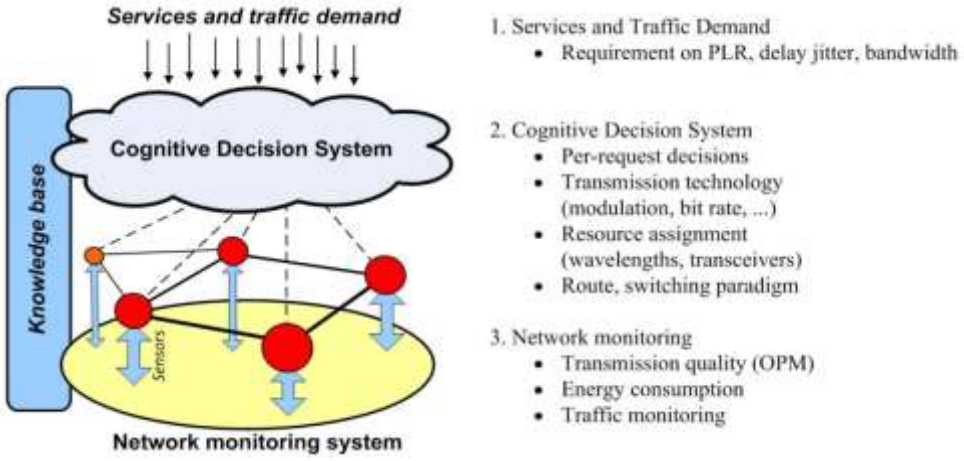


Figura 13. Principales subsistemas para las tareas del ciclo cognitivo [20].

Además de lo anterior, es importante nombrar dos conceptos relacionados directamente con el ciclo cognitivo: Razonamiento, el cual se define como la facultad que permite tomar una decisión o actuar basándose en la observación y/o en el conocimiento obtenido previamente. Por otra parte, el aprendizaje se centra en la adquisición y acumulación de conocimiento, producto de las acciones acontecidas dentro de la red. Dichas acciones son almacenadas con el objetivo de aumentar la eficiencia y el desempeño ante razonamientos futuros.

2.4 TRABAJOS RELACIONADOS.

Como es posible observar, el enfoque de la monografía encierra un contexto que unifica varios temas de estudio. Trabajos relacionados sobre el problema de RWA en redes OBS/WDM usando métodos de control cognitivo y AG, aún no han sido ampliamente desarrollados. Hasta el momento, los enfoques más cercanos incluyen investigaciones donde el tratamiento del enrutamiento y asignación de longitud de onda en redes ópticas se realiza por medio de Algoritmos Genéticos, sin embargo, dichas redes en su mayoría presentan conmutación de circuitos y además, no incluyen técnicas de control cognitivo que incidan en el desempeño de la red.

Debido a lo anterior, los trabajos relacionados se agrupan según los enfoques más relevantes en la presente investigación.

2.4.1 Esquema comparativo entre los trabajos relacionados.

Dada la cantidad de trabajos encontrados, en este apartado se propone un esquema que permite analizar las brechas al igual que los temas de interés inmersos en algunas de las investigaciones más destacadas. En este sentido, se definen los siguientes parámetros que constatan el grado de pertinencia de cada investigación con el presente trabajo de grado: el primero de ellos hace referencia al enrutamiento y asignación de longitud de onda en redes ópticas (RWA) [74], asegurando que, independientemente de la técnica usada, este problema es el que pretende ser resuelto; el segundo parámetro corresponde a los algoritmos genéticos (AG), métodos de búsqueda y optimización basados en los mecanismos de selección natural postulados por Darwin; la conmutación ópticas de ráfagas (OBS) [75] que agrupa conjuntos de paquetes con características similares para enviarlos a través de la red; la cognitividad en redes ópticas (CRO) [76], que otorga a la red capacidad de adaptación y autoconciencia; la Probabilidad de Bloqueo (PB), como principal medida de desempeño y por último, la inclusión del simulador modular de eventos discretos orientado a objetos OMNeT++ (OMN).

Para la evaluación de cada parámetro se tendrá en cuenta el nivel de inclusión de los temas en las investigaciones, por esta razón, las calificaciones se agrupan de acuerdo a 4 niveles: Fuerte (F), Medio (M), Débil (D) y Ausente (A).

Tabla 1. Esquema comparativo entre los trabajos relacionados.

ENFOQUES	RWA	AG	OBS	CRO	PB	OMN
William P. et al. (2008) [6]	A	A	F	A	D	A
Tzvetelina B. et al. (2003) [9]	D	A	F	A	A	A
Arturo B.R. et al. (2009) [14]	F	F	A	A	F	A
Sara L. (2010). [21]	F	D	A	A	F	A
Urmila B. et al. (2010). [27]	F	M	A	A	F	A
Alvaro B. (2009) [45]	F	A	F	A	D	F
Yousef K. et al. (2009) [59]	F	F	A	A	A	A
Georgioss S.Z. et al. (2010) [73]	M	A	A	F	A	A
J. Gond. et al. (2010). [77]	M	A	A	A	F	A
L. N. Binh et al. (2009). [78]	F	A	A	A	F	F
Le Vinh T. (2005). [79]	F	F	A	A	F	A
Y. S. Kavian (2010) [80]	F	F	A	A	D	A
R. Nakkeeran, et al. (2004) [81]	F	F	A	A	F	A
David B. et al. (2004) [82]	F	F	A	A	F	A
Vinh T. et al. (2005). [83]	F	F	A	A	F	A
Ignacio M. et al. (2004) [84]	F	F	A	A	M	A
E. Kozlovski (2003) [85]	M	A	F	A	F	A
Alvaro B. et al. (2009) [86]	F	A	F	A	M	F
Felix E. et al. (2010) [87]	A	A	F	A	A	F
R.J. Duran. et al (2011) [88]	M	F	A	F	F	A

De la tabla anterior es posible realizar las siguientes inferencias de acuerdo al estudio minucioso de cada tema de investigación (Cabe nombrar que los temas relacionados con la probabilidad de bloqueo y la herramienta de simulación utilizada, se analizan incluyéndolos en los subtemas propuestos a continuación).

2.4.1.1 Enrutamiento y Asignación de Longitud de Onda.

Los mecanismos para solucionar el enrutamiento y asignación de longitud de onda son numerosos y abarcan técnicas fijas, dinámicas y mixtas, según el tipo de tráfico cursado por la red. Los algoritmos de este apartado, no necesariamente se encuentran relacionados con métodos evolutivos, además, hacen referencia a redes que pueden o no presentar conversores de longitud de onda y brindan solución al RWA por medio de un establecimiento dinámico de la conexión.

Trabajos de investigación como [77], se enfocan en el problema de Asignación de Longitud de Onda (WA, Wavelength Assingment) utilizando para esto conversores ópticos que si bien permiten la reutilización de longitudes de onda, aumentan considerablemente el costo de la red y (en el plano físico) complican tecnológicamente los dispositivos OXC [17].

[78] por ejemplo implementa el RWA en redes DWDM, teniendo en cuenta restricciones físicas e incluyendo también conversores ópticos. En este caso, el algoritmo implementado usa el ancho de banda y los nodos del camino óptico para su funcionamiento, el cual, al ser desarrollado, facilita y agiliza los procesos de enrutamiento.

Investigaciones orientadas al análisis cualitativo de los métodos heurísticos enfocados al problema DRWA también hacen parte de este grupo. El estudio presentado en [21] es útil para caracterizar los algoritmos dentro del contexto de enrutamiento y asignación de longitud de onda dinámica, no obstante, el objetivo de este proyecto es formular un algoritmo para el DRWA que considere efectos físicos como el alcance y la regeneración de la señal óptica.

Según lo anterior, se observan trabajos enfocados al enrutamiento y asignación de longitud de onda que utilizan conversores dentro de las redes, incluyen la capa física dentro de su análisis y no hacen uso de métodos de control cognitivo dentro de la red.

2.4.1.2 Algoritmos Genéticos para el RWA.

En [27] los autores desarrollan un algoritmo evolutivo usando algoritmos genéticos de manera parcial, para este proceso establecen una única ruta inicial y usan solamente el operador mutación para así generar una notable disminución en el tiempo de ejecución. Al momento de desarrollar la función de aptitud, tienen en cuenta factores como la longitud de onda y el camino óptico, alcanzado menores probabilidades de bloqueo para la red. La longitud de onda se escoge de acuerdo a algoritmos predeterminados (R, FF) teniendo en cuenta el que sea de mayor beneficio según la función de aptitud.

Existen, por otra parte, proyectos que involucran el desarrollo de algoritmos genéticos en redes ópticas WDM, analizando los casos donde se presentan conversores de longitud de onda y también aquellos donde existe restricción de la continuidad de longitud de onda [59].

Los trabajos de esta categoría, realizan en su mayoría una propuesta sobre algoritmos genéticos para resolver el problema de RWA en redes WDM [59], [79], [80] algunos incluso proponen estrategias de equidad a través de las conexiones y tolerancia a fallos [81]. Otras investigaciones combinan algoritmos evolutivos como la optimización por colonia de hormigas y los algoritmos genéticos [82], [83], [84], sin embargo, la primera técnica no se trata directamente en esta investigación y, además, debido a su estructura, puede generar sobrecarga en algunos caminos ópticos a los largo de la red.

La principal desventaja que presentan estas investigaciones se debe a la ausencia de un enfoque basado en conmutación óptica de ráfagas; si bien, en algunos trabajos se habla de conmutación óptica de circuitos, otros no nombran ni siquiera un método de conmutación específico. Además, no hay presencia de un método de control cognitivo que retroalimente a la red.

2.4.1.3 Conmutación Óptica de Ráfagas.

Los trabajos de este grupo, aunque reflejan estrategias de enrutamiento y asignación de longitud de onda en redes OBS/WDM, no incluyen el uso de los algoritmos genéticos dentro de sus estrategias. Específicamente, emplean otros métodos para el RWA y realizan un análisis orientado hacia la obtención de resultados en términos de probabilidad de bloqueo y calidad de servicio [85].

Son de destacar también, investigaciones puntuales sobre la conmutación óptica de ráfagas, ya que permiten orientar la monografía gracias a los contextos ampliamente desarrollados en este entorno de investigación, incluso aunque no presenten información sobre el uso de los AG y los métodos de control cognitivo en la red [6], [9]. Por ejemplo, algunas propuestas ofrecen información relevante sobre el modelado de una red OBS en el entorno OMNeT++, el cual será usado para simular los resultados obtenidos al desarrollar la monografía [86], [87].

2.4.1.4 Control cognitivo en Redes Ópticas.

Dentro de este enfoque se encuentran dos investigaciones relevantes que involucran conceptos fundamentales sobre las redes ópticas cognitivas y su aplicación al campo de los algoritmos genéticos, respectivamente.

En primer lugar, se encuentra [73] el cual lleva las redes ópticas cognitivas hacía un perspectiva que proporciona autonomía y contrarresta la creciente complejidad en el uso y desempeño de la red. La investigación muestra, además, el proceso que debe atravesar la red para ser cognitiva, generando, en un comienzo consciencia de su entorno, adaptándose posteriormente a la condiciones de la red y por último aprendiendo de estas condiciones, para ofrecer soluciones y prevenciones oportunas en una situación dada.

Se plantea, así mismo, la posibilidad de introducir memoria a corto, mediano y largo plazo para manejar sistemas de asignación de ancho de banda y métodos de control cognitivo a la red, supliendo los métodos comúnmente utilizados para este proceso.

El trabajo de grado, sin embargo, presenta una notable diferencia respecto a la investigación analizada, ya que esta última sólo hace alusión a la arquitectura de las redes ópticas cognitivas atendiendo a un modelo de capas, y aunque presenta una innovadora propuesta sobre el diseño y requerimientos de dichas redes, no habla de la implementación de los algoritmos genéticos para el RWA en redes ópticas, lo cual constituye un eje primordial del presente trabajo.

Por otra parte, se encuentra la investigación [88] la cual brinda una estrategia de solución basada en cognición ante los problemas de diseño de topologías virtuales en redes ópticas. Este trabajo, desarrolla un algoritmo genético encargado de generar la topología adecuada para la red y paralelamente genera una extensión del mismo, incluyendo un método de control cognitivo basado en memoria para optimizar dicha topología.

En conclusión, esta última investigación es la más similar al tema global de la monografía, aunque en [88] no sea posible obtener información detallada que permita indagar sobre la clase de conmutación utilizada y el desarrollo de un método RWA para las topologías virtuales.

3 IMPLEMENTACIÓN DE LOS MÓDULOS DE LA RED OBS/WDM.

Para el desarrollo de esta investigación, es necesario usar una herramienta que permita simular el comportamiento de una red OBS/WDM, implementar un método de control cognitivo basado en algoritmos genéticos para el RWA, y analizar el desempeño de esta red en términos de la probabilidad de bloqueo y el tiempo de procesamiento.

3.1 PLATAFORMAS PARA LA SIMULACIÓN Y ANÁLISIS DE ALGORITMOS GENÉTICOS EN REDES ÓPTICAS OBS/WDM.

Se analizan las plataformas más reconocidas en el área de las redes de comunicaciones, que ofrecen la documentación, el soporte y la precisión en cuanto a rendimiento y escalabilidad necesarias. Se elige la herramienta más adecuada para cumplir con el objetivo de este trabajo de grado [89].

NS-2 (Network simulator 2):

Esta herramienta de simulación es utilizada para investigación y educación. NS-2 se encuentra basada en eventos discretos, cuyas simulaciones están compuestas por código C++, usando lenguaje de comandos orientado a objetos para la definición de los escenarios de simulación. Las principales desventajas de NS-2 hacen referencia a que el modelado es una tarea bastante complicada, ya que no presenta una interfaz gráfica, además, el lenguaje de scripting y las técnicas de modelado deben ser aprendidos con anterioridad, lo que genera una gran consumo de tiempo. Así mismo, los desarrolladores han presentado inconformidades debido a la inconsistencia de los resultados y los errores que presentan muchos de los protocolos [90].

NS-3 (Network simulator 3):

El objetivo de esta herramienta es recrear un entorno de simulación que permita la investigación sobre redes. NS-3 presenta algunas características de su antecesor NS-2, pero con muchas mejoras entre ellas la eliminación de problemas que resultan de la combinación de C++ con OTCL, ya que no hace uso de lenguaje de comandos orientado a objetos para el control de la simulación. NS-3 está compuesto por 24 módulos que contienen uno o más modelos de elementos de red reales y protocolos, las implementaciones pueden ser realizadas en lenguaje C++ y Python scripting. No obstante, presenta problemas como la falta de compatibilidad con su predecesor NS-2 y los protocolos que éste contiene.

NetSim:

Simulador de redes muy usado para investigación sobre varias tecnologías como WLAN, WiMax, TCP, IP, ofreciendo para la simulación más de quince protocolos a través de redes LAN y WAN. Netsim fue desarrollada por Tetcos, en conjunto con el Instituto Indio de Ciencia, impulsada por CISCO. Es una herramienta de eventos discretos estocásticos que presenta un entorno de desarrollo el cual puede emplearse como una interfaz entre el código, las bibliotecas y el núcleo del sistema de simulación. Esta herramienta permite trabajar con algoritmos, pseudo código y diagramas de flujo. Los usuarios pueden escribir sus propios códigos fuente en C o C + + y vincularlos a NetSim [91], [92].

OPNET:

Es una herramienta de simulación orientada a las comunicaciones. Presenta una interfaz amigable para los usuarios, incluyendo librerías de modelos que permiten la familiarización entre el programador y la jerarquía interna del simulador. OPNET trabaja internamente con programación de código C++. El desarrollo de los modelos se da a través de la conexión de nodos, compuestos internamente por diferentes módulos y conexiones.
Cada módulo se encuentra definido mediante modelos de proceso representados a través de máquinas de estados finitos [93].

OMNeT++IDE:

OMNeT++ permite la simulación de redes en cualquier ambiente, es una herramienta de simulación modular de eventos discretos de redes orientada a objetos, hace uso de protocolos, sistemas multiprocesadores y distribuidos, validación de arquitectura hardware, evaluación del rendimiento y permite el modelaje de cualquier sistema que posibilite su representación mediante eventos discretos. Este simulador es usado principalmente en redes, ya que presenta un paquete denominado INET, (Integrated Network Enhanced Telemetry) que ofrece una gran colección de modelos de protocolos de Internet. OMNeT se encuentra basada en C++, implementando los módulos en este lenguaje, los cuales a su vez pueden convertirse en módulos compuestos mediante el lenguaje de la simulación y descripción de la red de OMNeT++ [94]. Jerárquicamente, en el nivel más bajo se encuentran los módulos simples, programados en C++, éstos se pueden personalizar mediante la asignación de parámetros para obtener la red con las características deseadas y agruparse en módulos compuestos sin tener una limitación en el número de niveles jerárquicos.

Los parámetros pueden ser asignados a cualquier archivo en la red o en el archivo de configuración omnetpp.ini y se utilizan para personalizar el comportamiento del módulo simple y la topología del modelo, esta información se encuentra en el ANEXO 2.

Los módulos se pueden conectar entre sí a través de puertos y se comunican con el paso de mensajes a lo largo de rutas preestablecidas por las conexiones o directamente a su destino. Las conexiones se pueden utilizar para modelar enlaces físicos con parámetros como la velocidad de datos, el tiempo de propagación, tasa de error de bit y tasa de paquetes perdidos [31].

OMNeT++ presenta una interfaz gráfica de usuario que permite controlar la ejecución de la simulación, y, posiblemente, que el usuario pueda intervenir en el modelo mediante el cambio de variables, además, presenta un ambiente amigable para la visualización del funcionamiento de una red.

3.1.1 Plataforma seleccionada para la simulación y análisis de algoritmos genéticos con control cognitivo en redes ópticas OBS/WDM.

Para la elección de la herramienta de simulación se tuvo en cuenta su capacidad para modificar las características, protocolos y algoritmos, que permiten alcanzar los objetivos del trabajo de grado, esto hace referencia a la flexibilidad. De igual forma, se estudia si es posible incluir métodos de control cognitivo basados en algoritmos genéticos, y preferiblemente, se escoje una herramienta que presente licencia gratuita, con el fin de disminuir costos y facilitar el acceso.

En la tabla 2 se presenta la síntesis de los requerimientos correspondientes a las herramientas de simulación anteriormente descritas.

Tabla 2. Requerimientos analizados sobre las herramientas de simulación.

Herramientas de Simulación	Flexibilidad	Implementación de Algoritmos Genéticos	Implementación de Cognición	Licencia
NS-2	Débil	No Permite	No Permite	No Requiere
NS-3	Débil	No Permite	No Permite	No Requiere
OPNET	Media	No Permite	No Permite	Requiere
NetSim	Media	No Permite	No Permite	Requiere
OMNeT++	Alta	Permite	Permite	No Requiere

Con el fin de simular una red óptica OBS/WDM que permita la implementación de Algoritmos Genéticos, la herramienta que se muestra como la más apropiada es OMNeT++ ya que se caracteriza por ser un software flexible, organizado y apto para la simulación de redes de interconexión de altas prestaciones. Proporciona librerías y herramientas básicas para el desarrollo de las simulaciones, aunque no posee componentes específicos presentes en redes, ofrece elementos de simulación como módulos simples y una interfaz en la cual se puede modificar dichos módulos [31], [32], [95], [96].

3.2 CREACIÓN DE LOS MÓDULOS DE LA RED OBS/WDM.

Para obtener una red que cumpla con las características representativas de OBS/WDM, se parte de la creación de módulos simples y de su agrupación, con el fin de formar los nodos frontera y central de dichas redes.

Inicialmente se crean los módulos “Fuente” (Source), los cuales generan paquetes TCP/IP que se desean transmitir por la red. Se encargan de asignar aleatoriamente la dirección de destino y el tipo de información que va a ser enviada, estas características son asignadas con funciones uniformes como se describe en el ANEXO A.

Los parámetros del nodo “Fuente” son: la dirección de origen, la dirección de destino, el tamaño de los paquetes y el tiempo entre la generación de paquetes, los dos últimos parámetros se modifican y programan desde el archivo de configuración mencionado anteriormente.

El módulo “Fuente” se agrupa con el módulo de “Destino” que recibe los paquetes provenientes de la red óptica, como muestra la Figura 14.

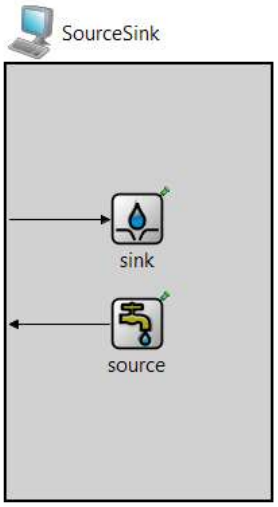


Figura 14. Módulo Fuente-Destino.

3.2.1 Nodo Frontera.

Inicialmente, se estudiaron las funciones de este nodo que opera como una interfaz de entrada y salida de datos para la creación de los módulos simples con las características adecuadas, la Figura 15 muestra las módulos que forman un nodo frontera.

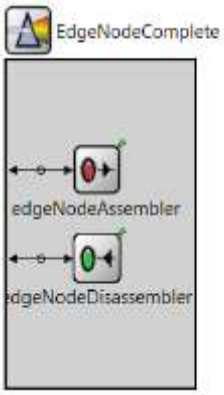


Figura 15. Módulos que componen el nodo Frontera.

La primera función analizada fue la del ensamblaje de datos, en donde los paquetes TCP/IP provenientes de la fuente se clasifican según la dirección de destino y el tipo de información (voz, vídeo o datos). Para esto se crea un módulo compuesto llamado “Ensamblaje” (Edge Node Assembler) que se muestra en la Figura 16, cuya entrada se encuentra conectada directamente con los equipos o fuentes de tráfico TCP y su salida con los nodos centrales mediante fibra óptica; contiene tres módulos simples: “Clasificador” (classifier), “Generador de ráfagas” (burstifier) y “Salida” (sender).

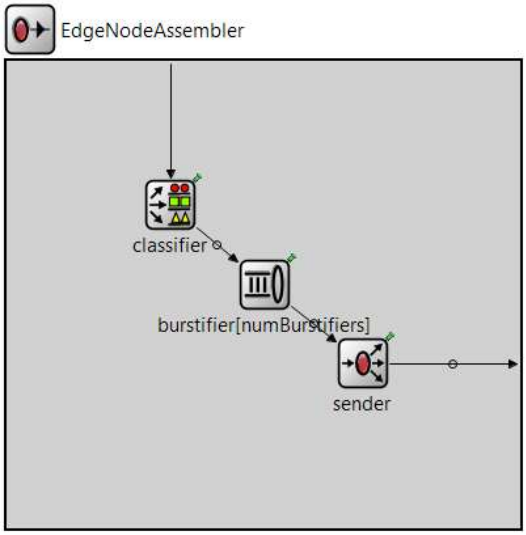


Figura 16. Módulo compuesto Ensamblador.

El “Clasificador” es un módulo simple que almacena un archivo de texto llamado reglas para realizar la clasificación de los paquetes; si un paquete coincide con dichas reglas es enviado hacia la compuerta de salida correspondiente, si no es así, el paquete es descartado. Este módulo simple se encuentra conectado con la entrada del módulo ensamblaje mencionado anteriormente y el generador de ráfagas.

El módulo “Generador de ráfagas” contiene parámetros como: el tiempo y tamaño máximo para el establecimiento de la ráfaga, el límite superior de paquetes, los valores máximo y mínimo del tamaño del BCP, la longitud mínima de la ráfaga y la etiqueta de destino, estos parámetros permiten el ensamble de los paquetes provenientes del “Clasificador” en ráfagas.

El algoritmo usado para el montaje de las ráfagas se basa en tiempo y longitud, esto se realiza mediante un ciclo descrito en el Anexo A, que permite analizar cuál de los dos parámetros se cumple primero y con respecto a éste se realiza el montaje.

Finalmente para cumplir con las funciones de estimación de tiempo offset y el enrutamiento de las ráfagas a través de la red óptica se tiene el módulo de “Salida”, conectado al módulo “Generador de ráfagas” y la salida del módulo “Ensamblador” (EdgeNodeAssembler).

El módulo “Salida” contiene parámetros como: el color (hace referencia al valor de las longitudes de onda), el número de longitudes de onda, número de fibras ópticas por nodo, la tasa de transmisión, el valor de offset, el tamaño de la cola de ráfagas que esperan para ser enviadas y la cola de paquetes de control que encuentran su camino dedicado (longitud de onda asignada al paquete de control) ocupado.

Si este módulo recibe una ráfaga desde el nodo “Generador”, el primer proceso que se realiza es la búsqueda de una longitud de onda u horizonte apropiado para transmitirla hacia el destino, haciendo uso del algoritmo LAUC mencionado anteriormente.

Para generar en la simulación una buena visualización del proceso OBS, tanto al paquete de control como a la ráfaga se le asignan dos mensajes que indican el inicio y el final de la transmisión,

seguidamente, se crea el paquete de control que contiene la información del tamaño de la ráfaga y el tiempo offset.

Después de analizar mediante el protocolo LAUC si existe una longitud de onda disponible para enviar la ráfaga, el nodo "Salida" transmite el mensaje que indica el inicio del paquete de control por la longitud de onda dedicada (cualquiera de los primeros caminos ópticos disponibles). Si el tiempo offset establecido no es suficiente para el envío de la ráfaga, ésta se descarta; si dicho tiempo es suficiente, se envía el mensaje final del BCP por el canal de control y la ráfaga por la longitud de onda asignada con el algoritmo, que se representa por un color almacenado en un mapa. El mapa es un vector bidimensional que contiene dos valores, el primero hace referencia a la clave que debe ser única para poder encontrar el arreglo que le correspondiente y un valor asociado que representa el canal óptico (lambda).

Los valores para los tiempos de offset máximo y mínimo (descritos en el cuarto capítulo) se programan desde el configurador omnet.ini y no son los únicos valores que generan la perdida de ráfagas, otra razón por la cual se descartan es porque el número de éstas en la fila de espera es muy alto.

Por otro lado, al nodo Frontera se le programa un módulo simple llamado "Desensamblador" (EdgeNodeDisassembler), encargado del desensamblaje de la ráfaga, envío de los paquetes hacia el destino y manejo estadístico del número de ráfagas que llegan al nodo.

El nodo Frontera se agrupa para procesos de diseño de la red con el equipo generador de tráfico (host) como se muestra en la Figura 17.

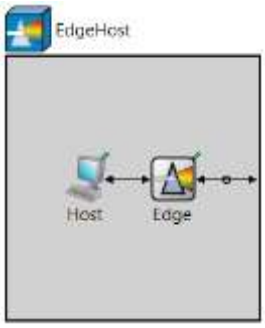


Figura 17. Nodo Frontera agrupado con el equipo generador de tráfico TCP.

3.2.2 Nodo Central.

Los nodos centrales se encuentran conectados con los nodos frontera mediante enlaces de fibra óptica y permiten la conmutación transparente de ráfagas de datos sobre ésta, como muestra la Figura 18.

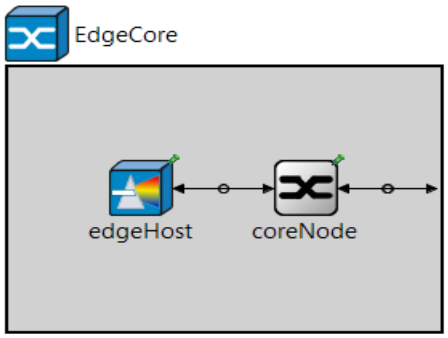


Figura 18. Conexión entre el nodo central y frontera mediante fibra óptica.

La primera característica analizada de este nodo corresponde a la información para el encaminamiento de la ráfaga sobre la red. Esta información se encuentra en su respectivo encabezado al que se le debe realizar un tratamiento en el dominio electrónico para capturar datos como la dirección de destino y el tamaño de la ráfaga, con el fin de realizar un enrutamiento correcto sobre la red. Por otro lado la ráfaga de datos es transmitida en el dominio óptico por un módulo que permite la separación de las longitudes de onda de datos y de control, llamado "Entrada del nodo central" con parámetros como: puertos de entrada y longitudes de onda por puerto (incluyendo la longitud de onda dedicada para el canal de control). En este módulo se le asigna un índice a cada puerto y longitud de onda (camino óptico).

Los paquetes de control y las ráfagas que llegan al módulo "Entrada" del nodo central son clasificadas según la longitud de onda asignada desde el modulo Salida del nodo frontera; si se trata de un BCP, es enviando hacia la compuerta que lo comunica con el módulo "Unidad de Control" y si es una ráfaga, es enviada hacia la compuerta que corresponde al módulo "Matriz de conmutación" (OXC), como se puede observar en la Figura 19.

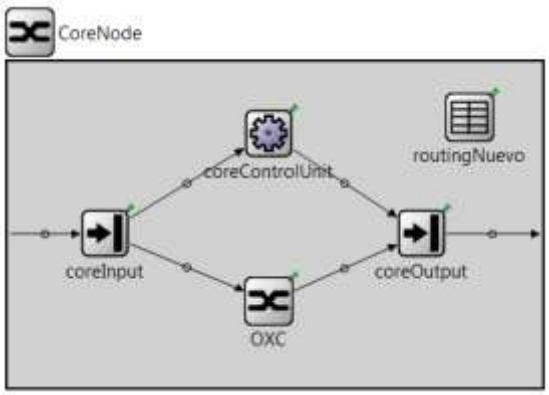


Figura 19. Módulos que componen el nodo central.

Otro elemento presente en el nodo central es el módulo "Matriz de conmutación óptica" (OXC), que permite la conmutación de las ráfagas de una fibra de entrada a una de salida mediante un algoritmo sencillo explicado en el ANEXO A.

Para procesos de señalización y planificación de ráfagas se crea el módulo "Unidad de control" compuesto por módulos simples como muestra la Figura 20.

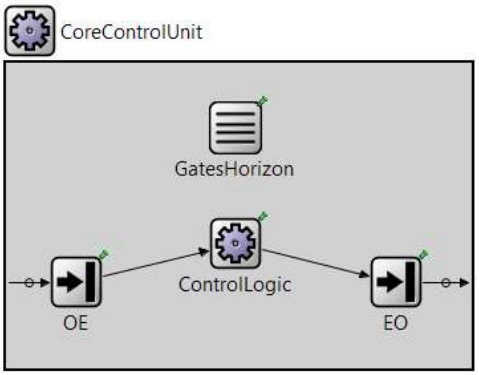


Figura 20. Módulo Unidad de Control.

El primer módulo presente en la “Unidad de Control” es el “Conversor óptico-electrónico” que realiza una copia de los paquetes de control generando un retardo por conversión dentro de la red. Dichos paquetes se envían por una compuerta hacia el módulo de “Control lógico” en el dominio electrónico.

Seguidamente, el módulo de “Control lógico” que recibe únicamente encabezados, almacena datos estadísticos de la red como: ráfagas recibidas, perdidas y programadas. Al recibir el BCP proveniente del “Conversor” se realiza la captura de datos (como el destino, y la longitud de onda), contenidos en dicho paquete para el encaminamiento. Mediante la clase cTopology presente en OMNeT++ y descrita en el ANEXO A, se realiza el descubrimiento de la red óptica que permite obtener los puertos a los que se encuentra conectado un nodo determinado y abstraer datos de todos los nodos interconectados en la red. La elección de la longitud de onda se realiza de nuevo haciendo uso del algoritmo LAUC; si ninguna lambda se encuentra disponible se borra el BCP y se descarta la ráfaga, pero si el horizonte está libre para el BCP, se busca el camino óptico apropiado para la transmisión de la ráfaga. Este proceso se realiza generando mensajes hacia los módulos “coreOutput” y “OXC” permitiendo que éstos habiliten sus compuertas y la longitud de onda disponible para la llegada de la ráfaga a los módulos. Finalmente, el módulo envía el paquete de control hacia la compuerta conectada con el “Conversor Eléctrico-Óptico”, y programa la salida de la ráfaga del módulo “Matriz de conmutación”.

En el módulo “Conversor Eléctrico-óptico” se almacenan en colas los BCPs que esperan la actualización y encuentran el canal ocupado. Después de pasar el paquete de control al dominio óptico se envía hacia la compuerta “Salida del nodo Central”.

“Salida del nodo Central” es el último módulo del nodo central, envía el paquete de control para establecer el camino de la ráfaga programada por el módulo de “Control Lógico” hacia la red óptica. Reúne los canales de control y de datos y los reordena con el fin de conectar el nodo Central con otro nodo OBS.

3.3 IMPLEMENTACIÓN DEL ALGORITMO GENÉTICO EN LOS MÓDULOS DE LA RED OBS/WDM.

En el contexto de las redes de telecomunicaciones, específicamente en el entorno de enrutamiento y asignación de longitud de onda para redes ópticas WDM, el proceso del algoritmo genético comienza con un número determinado de individuos, cada uno de los cuales se presenta como una solución tentativa al problema de RWA [97], [98], [21], [22]. En este caso, los individuos hacen referencia a posibles caminos entre el origen y destino especificados, los cuales se establecen con ayuda de una función que genera rutas de forma aleatoria [28].

Por otra parte, la analogía entre las redes y los procesos genéticos viene dada por el camino que seguirá la información, el cual es representado por un cromosoma [98], [99], que a su vez está compuesto por genes, alusivos a los nodos de la ruta óptica. Para este proceso, el camino óptico se codifica usando una cadena de enteros, donde cada número hace referencia a un nodo perteneciente a la ruta.

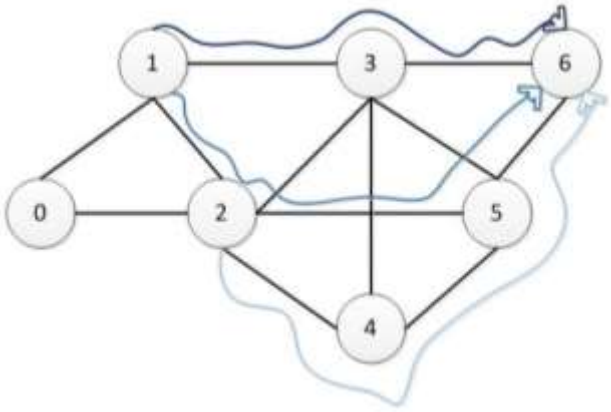


Figura 21. Representación codificada tres rutas. En orden de arriba abajo: {1, 3, 6}, {1, 2, 5, 6}, {2, 4, 5, 6}.

3.3.1 Generación de la población en la red OBS/WDM.

El primer paso para la aplicación de un algoritmo genético sobre la red óptica OBS/WDM es la generación aleatoria de la población inicial de individuos; en el proyecto actual ésta hace referencia al número de rutas que se desean obtener, estableciendo de antemano que todas las rutas deben tener en común el nodo de origen y destino.

Su función comienza ubicándose en el lugar desde dónde se transmitirá la información; desde este punto escoge aleatoriamente un nodo i cuya posición sea adyacente al nodo de origen; una vez se determina, éste es marcado como visitado. Estando en la posición i, la función realiza de nuevo la búsqueda y selección para establecer el siguiente nodo. El proceso se repite varias veces y finaliza una vez se llegue al nodo de destino, o bien, cuando todos los nodos adyacentes a la posición actual ya hayan sido visitados; en este último caso la función comienza de nuevo. El número máximo de rutas creadas no debe exceder al establecido durante el diseño del algoritmo. Una vez obtenida la ruta, se debe validar que no pertenezca al conjunto de rutas obtenidas anteriormente o debe ser descartada.

Se puede presentar que una ruta tenga como dirección de destino la misma dirección de origen, para éste tipo de caminos no se implementa el AG ya que la ráfaga va dirigida hacia el mismo equipo. Para

otros casos se realiza el análisis empezando por el descubrimiento de la red, empleando métodos presentes en la herramienta de simulación descritos en el ANEXO A.

La población inicial es evaluada mediante una función de aptitud que muestra numéricamente la bondad de los individuos del espacio de búsqueda de la primera población.

3.3.2 Función de aptitud para la evaluación de las rutas en la red OBS/WDM.

Para el desarrollo de la función de aptitud de este proyecto se integra información que permite disminuir la probabilidad de bloqueo en la red como: el número saltos, la distancia ellos y el número de fibras conectada a cada nodo de la red [99], [100], [101].

Se hace necesaria la introducción de un término importante dentro de la realización de la función de aptitud y es el costo, que permite analizar en qué proporción se ve afectada una red con respecto a un factor específico.

Para evaluar los individuos con las características mencionadas anteriormente se tienen en cuenta dos funciones de costo:

$$C1 = s + \alpha \sum_{i=1}^{s} (Li - Lf\,i) \quad (1)$$

La función de costo C1, permite analizar el impacto creado por el número de saltos en una ruta y las longitudes de onda libres de ésta, donde s es el número de saltos, L el número de longitudes de onda en la ruta, Lf el número de longitudes libres y α un parámetro de diseño que varía entre 0 y 1. Dicha función se trabajó en el proyecto de Vinh Tron Le [76], en el cual se arroja un valor máximo para α que dependa del número de longitudes de onda del enlace.

$$\alpha < \frac{1}{W-1} \quad (2)$$

La segunda función, representa la congestión de cada nodo generada por el número de enlaces conectados al mismo, entre más conexiones presente con otros, su congestión aumentará, el costo C2 representa la sumatoria de los enlaces de cada uno de los nodos sobre una ruta y se muestra a continuación.

$$C2 = \sum_{i=1}^{n} Ei \quad (3)$$

En donde Ei es el número de conexiones de cada nodo en la ruta.

La Función de Aptitud es inversamente proporcional a la función de Costo, por lo tanto las Funciones de Aptitud resultantes son:

$$F1 = \frac{1}{s + \alpha \sum_{i=1}^{s} (Li - Lf\,i)} \quad (4)$$

$$F2 = \frac{1}{\sum_{i=1}^{n} (Ei)} \quad (5)$$

En la segunda función de aptitud F2, se debe realizar una restricción debido a que si no existen nodos intermedios en la ruta, la sumatoria de los enlaces es cero, lo que genera un valor infinito para dicha función, es decir, un individuo no válido en el espacio de búsqueda; como el objetivo es encontrar un

óptimo global real, se recurre a un método de restricción y reconstrucción de dicho valor. En el proyecto actual, si dicha función genera un resultado infinito, éste es reemplazado por 1, que representa el máximo valor que toma la función y se da cuando no existen nodos intermedios en la ruta.

Se trabajan las dos funciones ya que en algunos casos la elaboración de n funciones de aptitud aproximadas puede resultar mejor que la evaluación de una única función [100].

El valor de aptitud máximo en la red es 2, siendo éste el resultado de la suma de las dos funciones anteriores, sin embargo, para el establecimiento del criterio de aptitud máximo, se realizan unos cálculos con los parámetros de diseño de la red mostrados a continuación:

- Para la primera función de aptitud, una ruta óptima se obtendría cuando existen la menor cantidad de saltos en la red y todas las longitudes de onda se encuentren disponibles, lo que generaría como resultado el punto máximo de la función (1), seguido de 0.5 para una ruta con todas las longitudes de onda libres y dos saltos únicamente.

- Para la segunda función de aptitud se debe analizar el valor de enlaces posibles que presenta cualquier nodo en la red, y cuál es el más frecuente dentro de ésta. Se obtienen como resultados 0.5, 0.33 y 0.25 para 2, 3 y 4 enlaces por nodo respectivamente, con 0.33 el valor más común en dicha red (para 17 de 21 enlaces); cabe resaltar que si una ruta presenta solo un salto su valor máximo sería 1 (valor de la restricción para enlaces entre nodos adyacentes).

Finalmente los valores de aptitud máximos los obtendría una ruta con las siguientes características:

Una ruta con un solo salto y con todas las longitudes de onda disponibles con un valor de aptitud de 2, una ruta con dos saltos, todas las longitudes de onda disponibles y con un nodo intermedio que presente dos enlaces con un valor de aptitud de 1, y una ruta con dos saltos que presente un nodo intermedio con 3 enlaces, con todas las longitudes de onda disponibles y un valor de aptitud de 0.88 (el más frecuente).

Por lo tanto se elige como criterio de aptitud máximo 1, que abarca las dos mejores rutas y se fija el siguiente rango de comparación en 0.88.

Seguidamente, las funciones de aptitud se aplican en las rutas obtenidas para realizar el proceso de selección.

3.3.3 Implementación de la función de aptitud para la evaluación de las rutas en la red OBS/WDM.

El primer paso para la ejecución de la función de aptitud en la herramienta de simulación es el descubrimiento de todos los parámetros que influyen en la probabilidad de bloqueo de la red como: saltos, número de enlaces por nodo y longitudes de onda libres en el camino.

Para abstraer las longitudes de onda, se analiza el vector de tiempos presente en la clase “Routing Nuevo”, dicho valor depende del número de “puertos” y el número de “lambdas Core” (longitudes de onda por enlace) de ese nodo. El Algoritmo LAUC muestra las longitudes de onda libres u horizontes en el puerto y los enlaces de cada nodo son descubiertos mediante la función cTopology descrita en el ANEXO A.

Teniendo los valores necesarios para resolver las funciones, se implementan las operaciones para obtener la aptitud de los individuos. Dado que la segunda función presenta una restricción, se programa un reparador que le asigna el valor “1” si su resultado inicial es infinito.

3.3.4 Operadores de evolución.

La selección de los individuos para la creación de la nueva generación, debe realizarse de la mejor manera con el fin de obtener la solución óptima en un tiempo apropiado, en el proyecto actual se emplea el método de la ruleta sesgada [100].

Todos los individuos con su respectivo valor de aptitud se almacenan en un vector, el método implementado para la selección asigna a cada individuo del arreglo un sector circular proporcional a su función objetivo, como muestra la Figura 22.

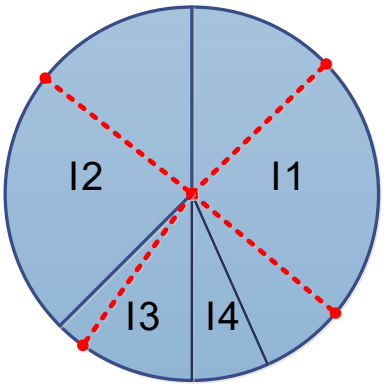


Figura 22. Método de selección de Ruleta, el individuo I1 es elegido dos veces, los I2 e I3 una vez y el I4 ninguna.

Para la asignación de los sectores a las rutas, se divide el valor de la aptitud de cada ruta entre la suma de todos los valores de aptitud.

Los individuos son seleccionados al girar una única vez la ruleta, para esto se crea un valor aleatorio que se encarga de elegir un sector.

Claramente, las rutas que poseen el valor de aptitud más alto tendrán una mayor probabilidad de ser seleccionadas para generar una nueva población.

Una vez terminado el proceso de selección, las rutas son almacenadas en vectores que se analizan para descartar aquellas que no son aptas (por ejemplo: las que poseen solo dos nodos o son destinos adyacentes), para aplicar el operador de cruce. Seguidamente se busca en las rutas aptas un elemento en común (números iguales que no se encuentren en la primera o en la última posición) para combinar las rutas y obtener descendencia. La ruta obtenida no puede contener dos nodos iguales, ni debe ser igual a una ruta almacenada (rutas padres), si es así se descarta, de lo contrario, se guarda y es aplicada la función de aptitud a cada uno de los hijos.

El operador de mutación no fue aplicado a esta red OBS/WDM, básicamente porque el uso de este podría afectar el rendimiento del sistema [101]. Su utilización ocasiona la perdida de individuos aptos, lo que genera retardos en la convergencia; razón por la cual los proyectos realizados con dicho operador trabajan con probabilidades de mutación muy bajas [102].

Con el fin de facilitar el entendimiento de las etapas mencionadas con anterioridad, se presenta el diagrama de flujo del algoritmo genético establecido en la Figura 23.

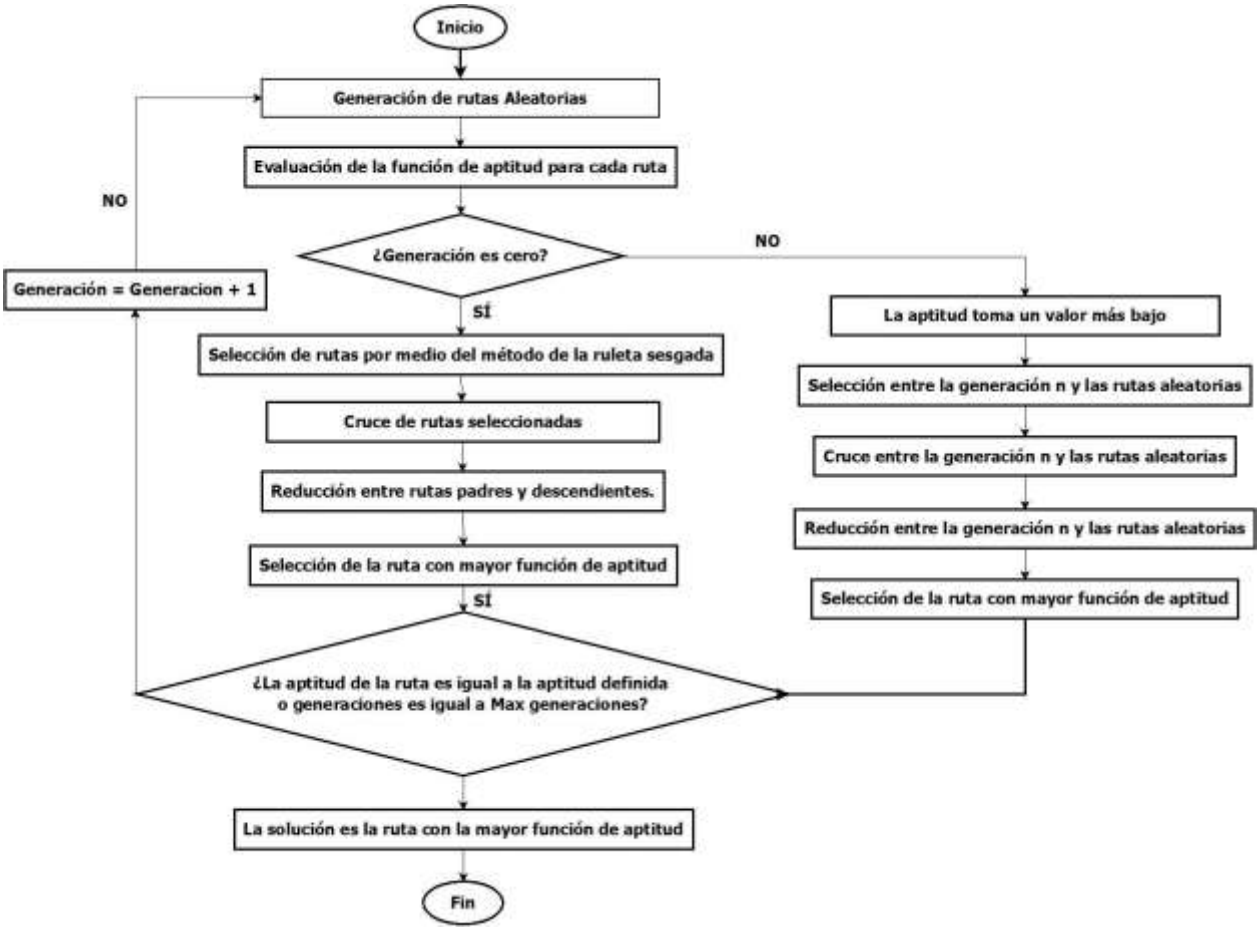


Figura 23. Diagrama de flujo del algoritmo genético.

Finalmente, el algoritmo genético fue implementado en el módulo denominado “RoutingNuevo”, incluido en el nodo central y encargado de las tareas de conmutación y enrutamiento. Además, el AG se ejecuta siempre que una ráfaga presente una solicitud de un camino para llegar a su destino, razón por la cual, éste es llamado desde el nodo frontera, específicamente en el módulo “sender”, el cual realiza las tareas de señalización y del envío de la ráfaga a la primera longitud de onda disponible.

Teniendo en cuenta que el algoritmo genético se realiza para resolver el RWA de tipo dinámico, la ruta resultante de este método debe viajar dentro del paquete de control, eliminando en cada salto el nodo correpondiente a la posición actual, hasta que el camino solo contenga el nodo de destino y una vez la ráfaga llegue, también sea eliminado.

3.4 IMPLEMENTACIÓN DEL MÉTODO DE CONTROL COGNITIVO BASADO EN ALGORITMOS GENÉTICOS EN LOS MÓDULOS DE LA RED OBS/WDM.

Como se explicó en el capítulo 2, las redes ópticas cognitivas presentan una arquitectura que describe las funciones realizadas en cada capa de la red; atendiendo a esto, el método de control cognitivo implementado se aplica al plano de control (CP, Control Plane), ya que éste trata directamente los

aspectos de enrutamiento, señalización y topología de la red. Por lo tanto, en esta capa es posible actuar sobre el RWA en redes OBS/WDM; no obstante, el proceso cognitivo es realizado una vez se tenga implementado el algoritmo genético dentro la red óptica.

Con el fin de explicar el método cognitivo implementado, se retoma el ciclo cognitivo, cuyo primer estado corresponde a la observación; esto hace alusión al descubrimiento de la red con ayuda de cTopology. La ejecución de esta clase permite al algoritmo genético implementar los métodos y etapas de su proceso, ofreciendo información importante para la base de información cognitiva. Posteriormente, surge la orientación, en donde la información proveniente del algoritmo genético adquiere significado. En este caso, el factor de interés para el proceso cognitivo es la ruta resultante del AG cada vez que una ráfaga se forma.

La Figura 24 muestra el método cognitivo inmerso en el proceso genético.

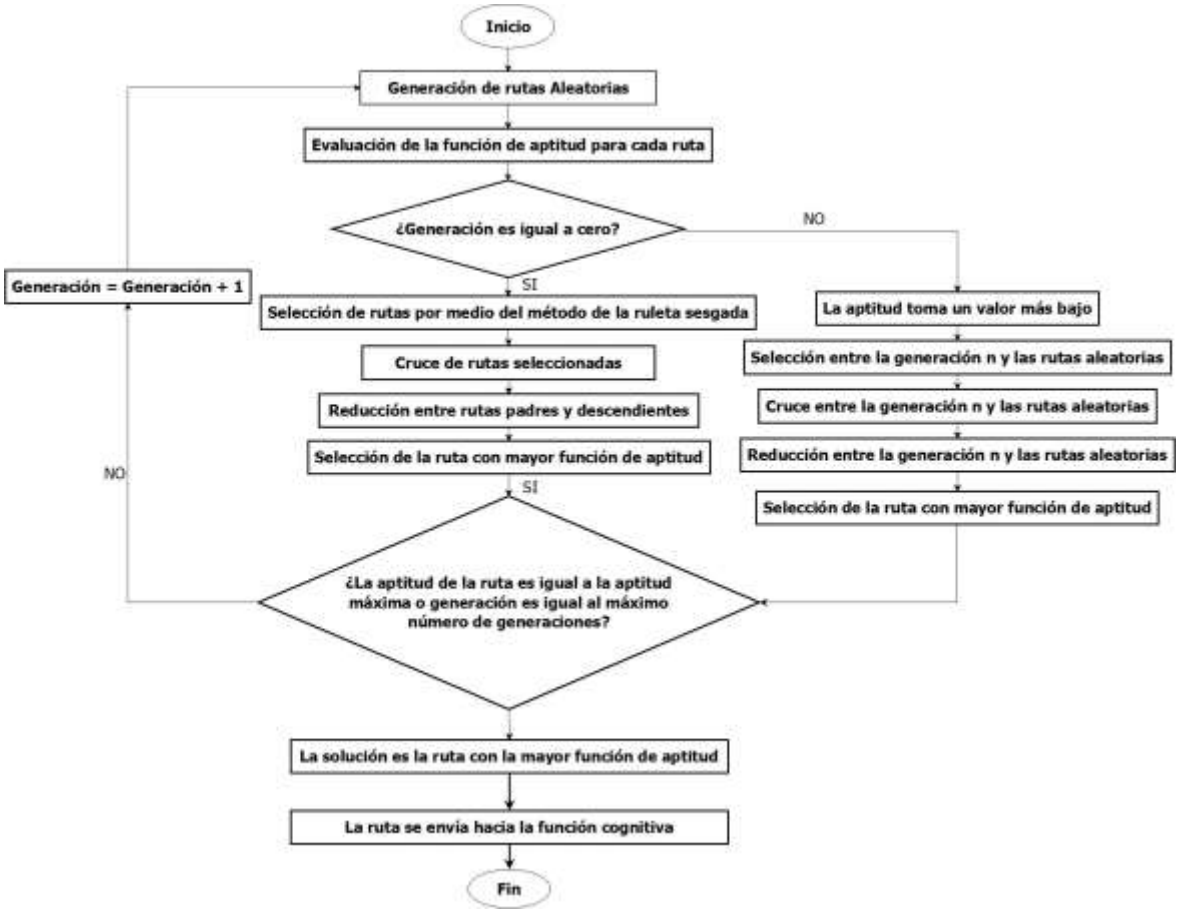


Figura 24. Diagrama de flujo del Algoritmo genético incluyendo el método de control cognitivo.

La tercera etapa del ciclo corresponde a la planeación; es aquí donde la información sobre la mejor ruta adquiere un objetivo para el método de control cognitivo, en otras palabras, una vez obtenido el vector solución (ruta) en un nodo particular de la red, éste es almacenado en un vector de vectores que contendrá un número máximo de rutas con el mismo origen pero destino variable, las cuales formarán parte de la estrategia cognitiva implementada dentro de la red, este ciclo corresponde al aprendizaje. Es importante aclarar que el tamaño de dicho vector de vectores es limitado y se establece como parámetro de diseño. Seguidamente, y gracias a la etapa de decisión, el vector de

rutas finales correspondiente a un nodo, interactúa con la primera etapa del algoritmo genético, llamada “Generación de rutas aleatorias”.

La Figura 25 muestra el cambio que presenta la generación de rutas aleatorias al introducir el método de control cognitivo.

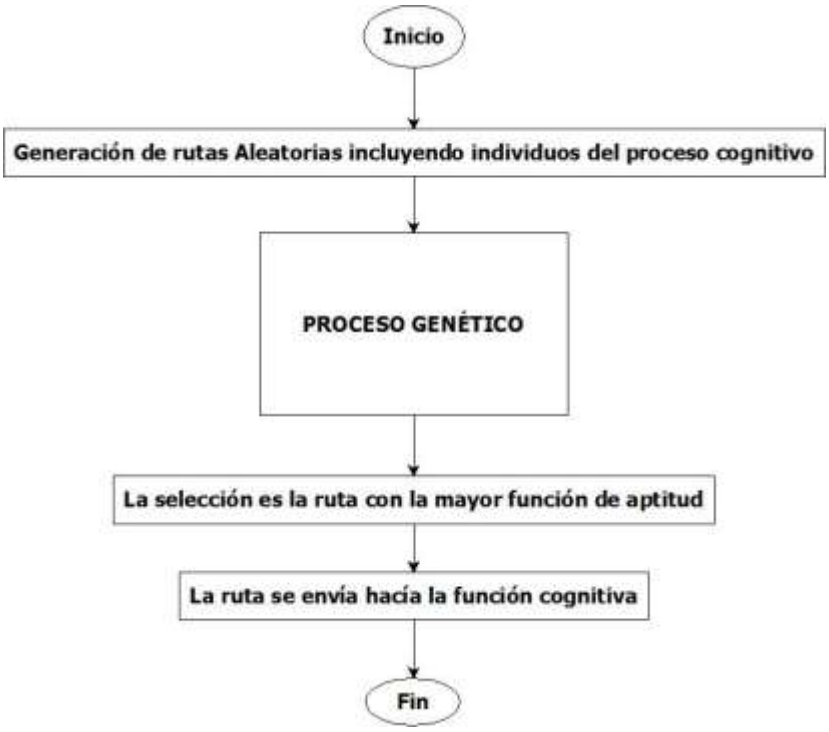


Figura 25. Diagrama de flujo del algoritmo y el método de control cognitivo generalizado.

Una vez dentro de este método, se debe validar si alguna o algunas de las rutas contienen el mismo destino para el cual se realiza la petición de la generación del AG, si el resultado es positivo, y uno o más vectores son escogidos, éstos formarán parte del conjunto de rutas aleatorias generado (entrando en el ciclo “actuar” del ciclo cognitivo), siempre y cuando no excedan el número máximo de población establecido como parámetro de diseño.

En la Figura 26, se explica la generación de rutas aleatorias al incluir el método de control cognitivo.

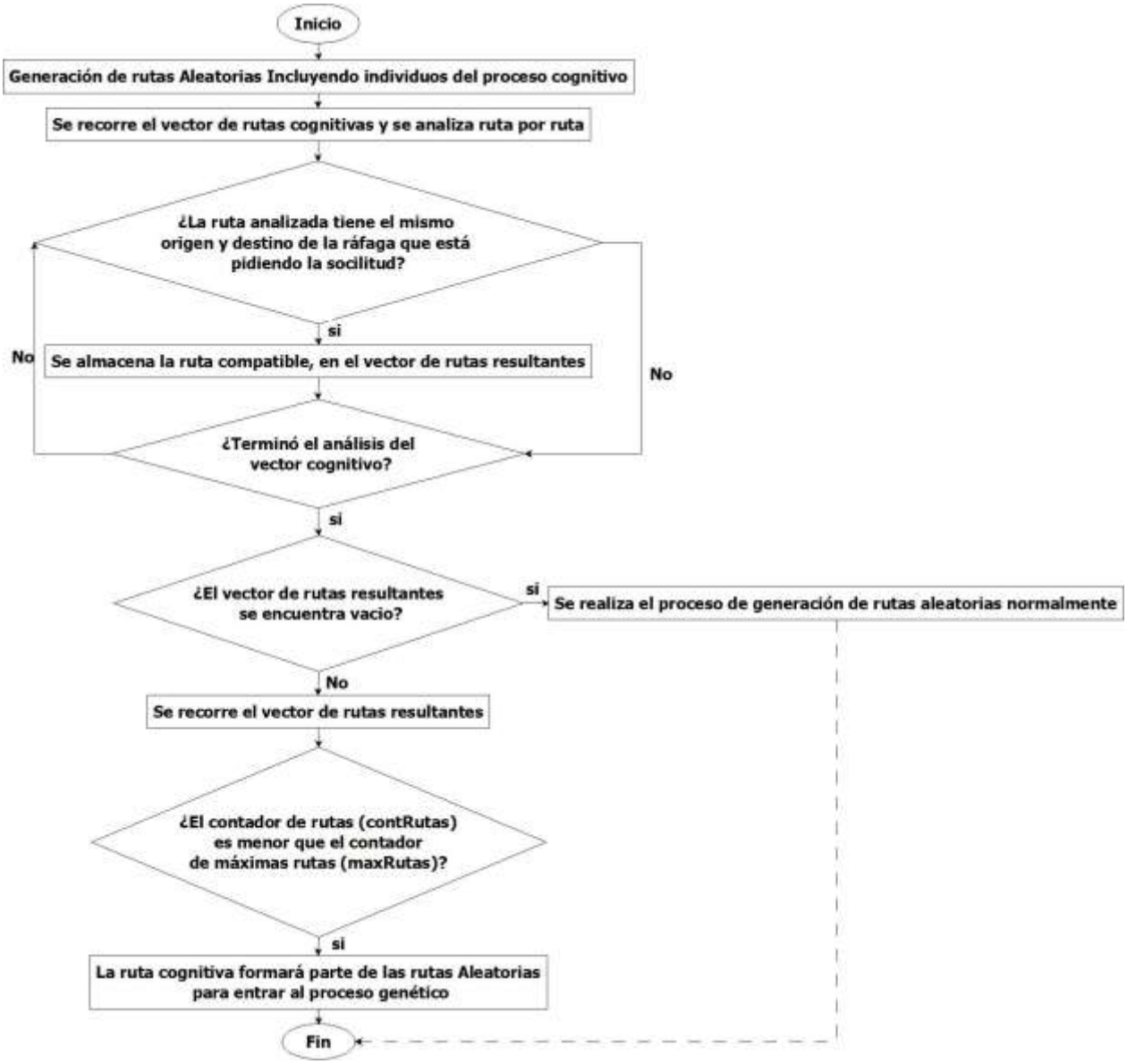


Figura 26. Generación de rutas aleatorias incluyendo el método de control cognitivo.

Este método presenta un factor adicional que añade dinamismo y amplia el espacio de soluciones: si una nueva ruta es generada y el vector de vectores se encuentra lleno, éste descarta la ruta correspondiente a su última posición y permite el ingreso de la nueva solución en la primera posición del vector de vectores, corriendo a las demás respuestas, una casilla. De esta forma las rutas analizadas no serán siempre las mismas.

La Figura 27, muestra el proceso de llenado del vector de vectores usando las rutas generadas por el algoritmo genético.

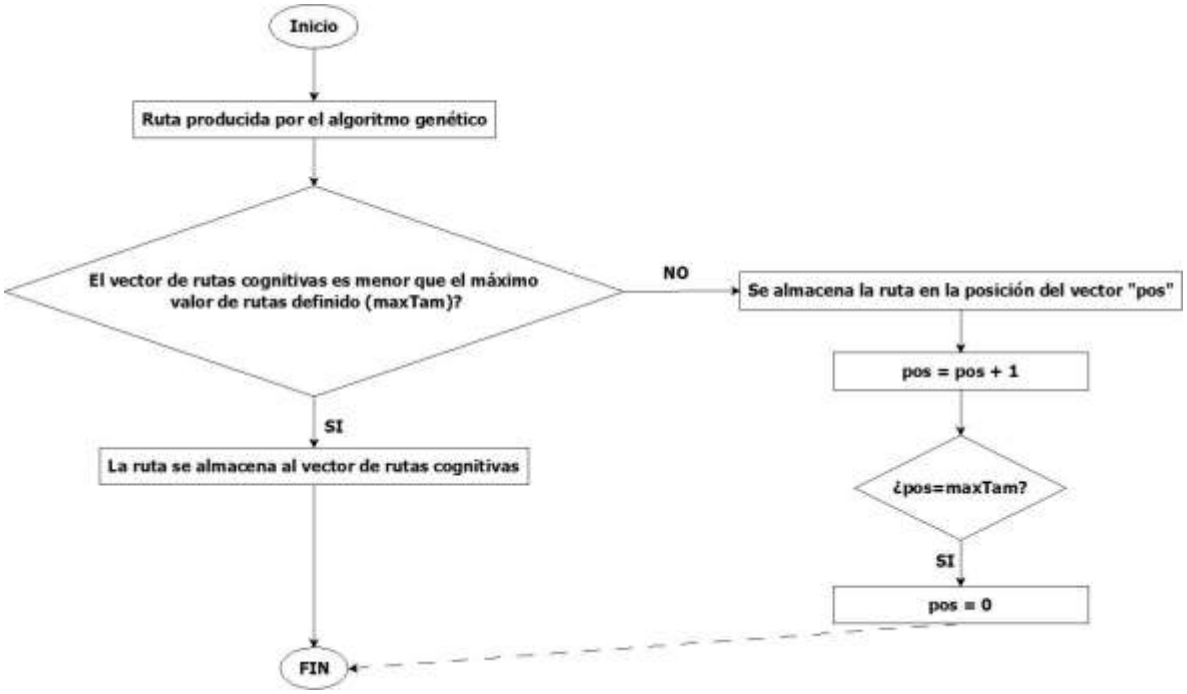


Figura 27. Diagrama de flujo del proceso de llenado del vector de vectores usando rutas provenientes de la respuesta del algoritmo genético.

Finalmente, el método de control cognitivo se implementa en el módulo “RoutingNuevo”, ubicado en el nodo central y usando para esto los métodos añadidos al proceso genético, como se mencionó anteriormente. De igual manera, el método de control cognitivo será llamado desde el módulo “Sender” ubicado en el nodo frontera y se ejecutará paralelamente al algoritmo genético, siempre que una ráfaga realice la petición de una ruta para llegar a su destino.

4 ANÁLISIS DEL DESEMPEÑO DE LA RED OBS/WDM BASADA EN ALGORITMOS GENÉTICOS CON Y SIN MÉTODOS DE CONTROL COGNITIVO.

4.1 TOPOLOGÍA ESCOGIDA PARA LA SIMULACIÓN Y ANÁLISIS DEL RWA EN REDES OBS/WDM.

Para la aplicación de las características propias de OBS/WDM se trabaja sobre la red NSFNet, esta red está compuesta por catorce nodos, veintiún conexiones bidireccionales con distancias que varían entre los 300 y 2800 kilómetros, como muestra la Figura 28 [103].

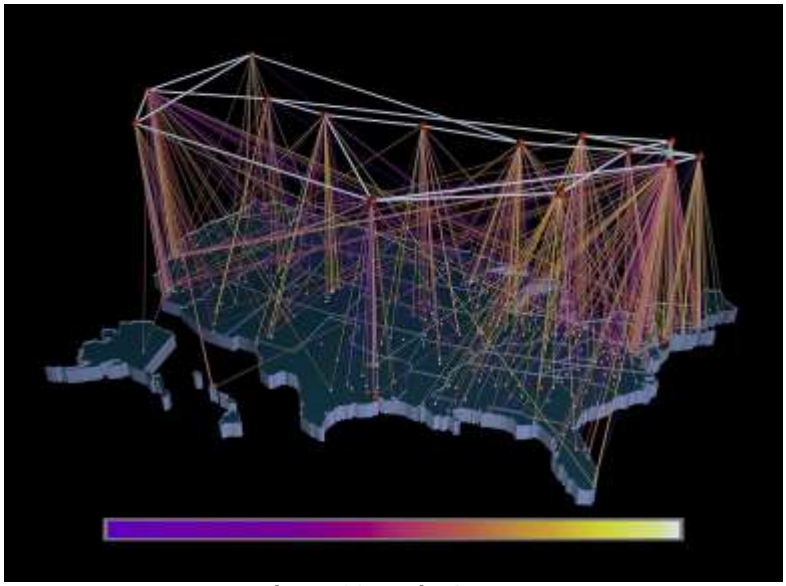


Figura 28. Red NSFNet.

NSFNet es una red de área amplia que cubre el territorio estadounidense y presenta documentación necesaria para la ejecución en la herramienta de simulación, además se debe agregar que OMNeT++ contiene imágenes de mapas de diferentes regiones lo que genera una implementación de los nodos de dicha red muy agradable visualmente.

Previo a la elección de NSFNeT se realizó un estudio de redes ópticas como Japanese backbone, NTTbackbone Network y CyLnet, pero fueron descartadas porque no se obtuvo la información necesaria para su implementación. Además diferentes investigaciones acerca de las redes ópticas se han basado en dicha topología, lo que se presenta como una guía para la elaboración del actual proyecto [28], [75].

Posterior a la elección de la topología, los módulos compuestos creados en la en la sesión 2.2., correspondientes a los nodos frontera y central, se agrupan para formar el módulo “edgeCore”. Dichos nodos se ubican en ciudades predefinidas del mapa de Estados Unidos para formar la topología de la red. OMNeT++, proporciona mapas de diferentes regiones del mundo; se hace uso del mapa de Estados Unidos que se muestra en la Figura 29, para la implementación de la red NSFNet.

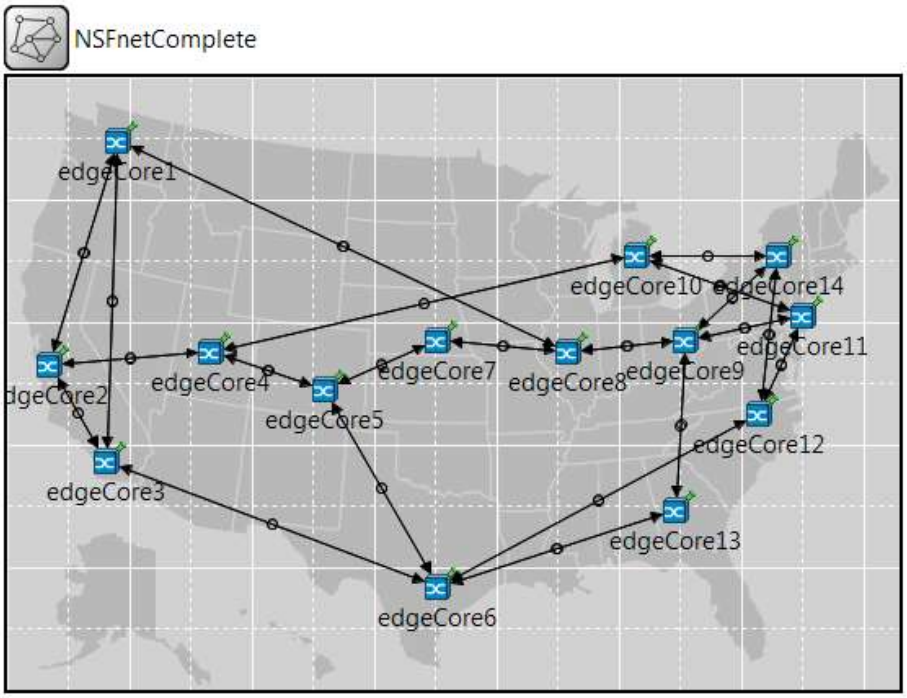


Figura 29. Red NSFNeT.

4.2 PARÁMETROS DE DISEÑO DE LA RED OBS/WDM.

Una vez realizado el montaje de la topología en la herramienta de simulación, se estudian los parámetros que inciden sobre el desempeño de la red OBS/WDM, evaluando la probabilidad de bloqueo y tiempo de procesamiento. Se examina la influencia de diferentes aspectos, como el tráfico, el tiempo offset y el número de longitudes de onda en la red, teniendo en cuenta los siguientes parámetros de diseño:

- Se emplean funciones de distribución exponenciales para el tiempo medio entre envío de paquetes, dichas funciones se pueden modificar desde el archivo de inicialización de la herramienta de simulación, como se muestra en la Figura 30.

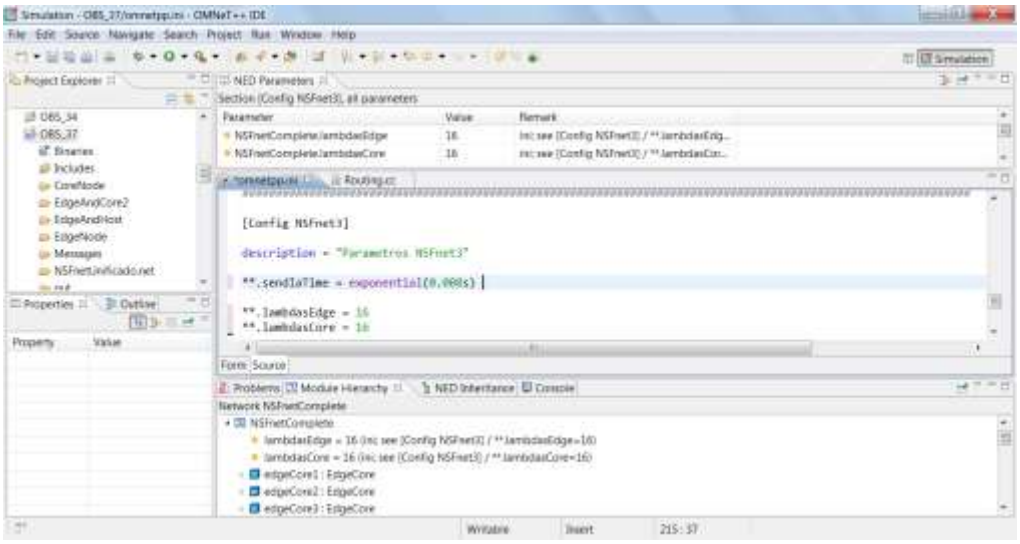


Figura 30. Archivo de inicialización.

La respuesta de la red a estos tipos de tráfico se analiza en los módulos que guardan las estadísticas propias de los nodos, como el número de ráfagas programadas, perdidas y recibidas. En la Figura 31 se muestra la ventana que contiene los valores de los datos estadísticos para cada nodo dentro de la red.

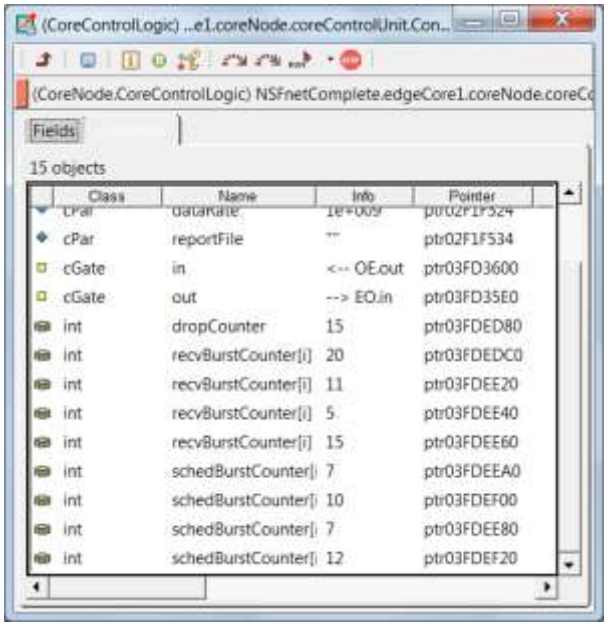


Figura 31. Ventana de visualización del "CoreControlLogic".

- Longitud del paquete: los paquetes que fluyen a través de la red son encapsulados usando el Protocolo de Control de Transmisión (TCP, Transmission Control Protocol). La longitud máxima permitida para estos paquetes es de 1500 bytes (IPv4).

- Archivo de reglas (rules.dat): archivo de texto plano que contiene las direcciones de destino y la clase de servicio que podría tomar un paquete dentro de la red. Éste posee 14 posibles direcciones de destino, teniendo en cuenta que son 14 los nodos que forman la red evaluada; además, por cada dirección pueden existir tres clases de prioridad, por lo tanto son en total 42 reglas diferentes que un paquete podría tomar para formar parte de la red y ser ensamblado con otro conjunto de paquetes que cumplan las mismas características de destino y clase de servicio. En la Figura 32 se muestra el archivo “rules.dat”.

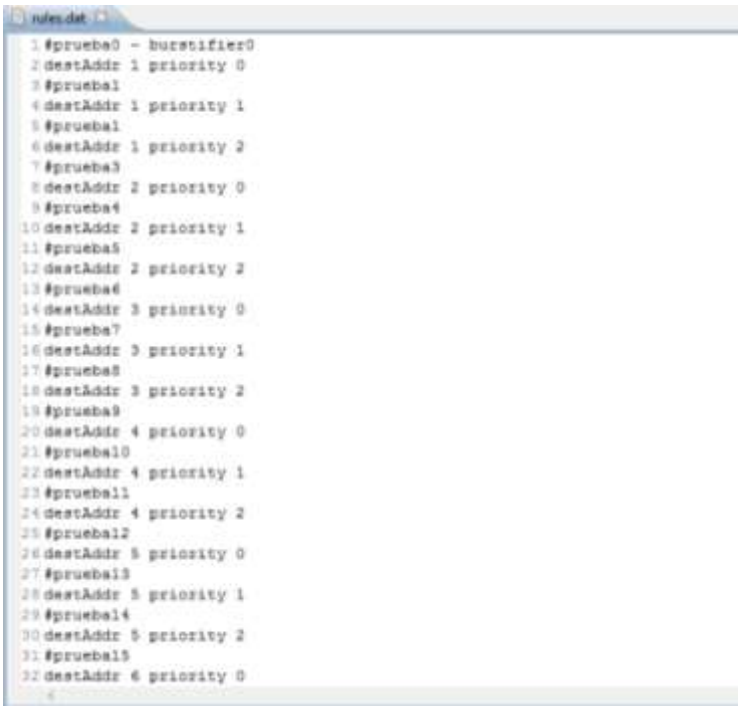


Figura 32. Archivo de texto “rules.dat”.

- Los colores de entrada y salida para los nodos frontera y central se establecen como una cadena vacía, de esta forma el programa asigna los colores por código en los módulos “sender” y “coreInput”. La Figura 33, detalla lo explicado anteriormente.

```
14
15 #Se definen los colores del módulo edge para cada lambda.
16 #============================================================================
17
18 **.edge*.Edge.outColours = ""
19
20 #Colores para la conexión del móudlo core
21 #============================================================================
22
23 **.core*.inputColours = ""
24 **.core*.outputColours = ""
25
```

Figura 33. Cadenas de colores para los módulos frontera y central.

- Tiempo de procesamiento del conversor óptico-eléctrico: es el tiempo que la cabecera tarda en realizar el procesamiento O-E dentro del módulo de control lógico [39].

- Tiempo de procesamiento del conversor eléctrico: hace referencia al tiempo que la cabecera tarda en realizar el procesamiento E-O dentro del módulo de control lógico [39].

- Tiempo de procesamiento del BCP: es el tiempo que tarda la cabecera de la ráfaga en procesar la información que transporta por cada nodo recorrido.

- Offset: definido como el retardo entre el BCP y la ráfaga. Éste debe ser calculado teniendo en cuenta los retardos de procesamiento correspondientes a los conversores óptico-eléctrico y eléctrico-óptico, el máximo número de nodos que una ráfaga puede atravesar a lo largo de la red y el retardo de procesamiento del BCP. En este sentido, el tiempo de offset crecerá a medida que la ráfaga tenga que atravesar una mayor cantidad de nodos, ya que los retardos mencionados anteriormente se acumulan por cada nodo intermedio correspondiente al camino de la ráfaga [104].

  Teniendo en cuenta lo mencionado anteriormente, se sabe que la red analizada posee 14 nodos; ahora bien, se ha encontrado que el mayor número de saltos que podría recorrer una ráfaga entre origen y destino (buscando las distancias más cortas) es seis. Por lo tanto, el offset máximo se obtiene sumando los tiempos de procesamiento y multiplicándolos por el número de nodos que en el peor de los casos, debe recorrer la ráfaga (en este caso, este número corresponde a 6). De esta forma, los resultados de la operación se muestran a continuación:

$$\text{maxOffset} = (((1\mu s)\text{x}2) + 10\mu s)\text{x}6$$

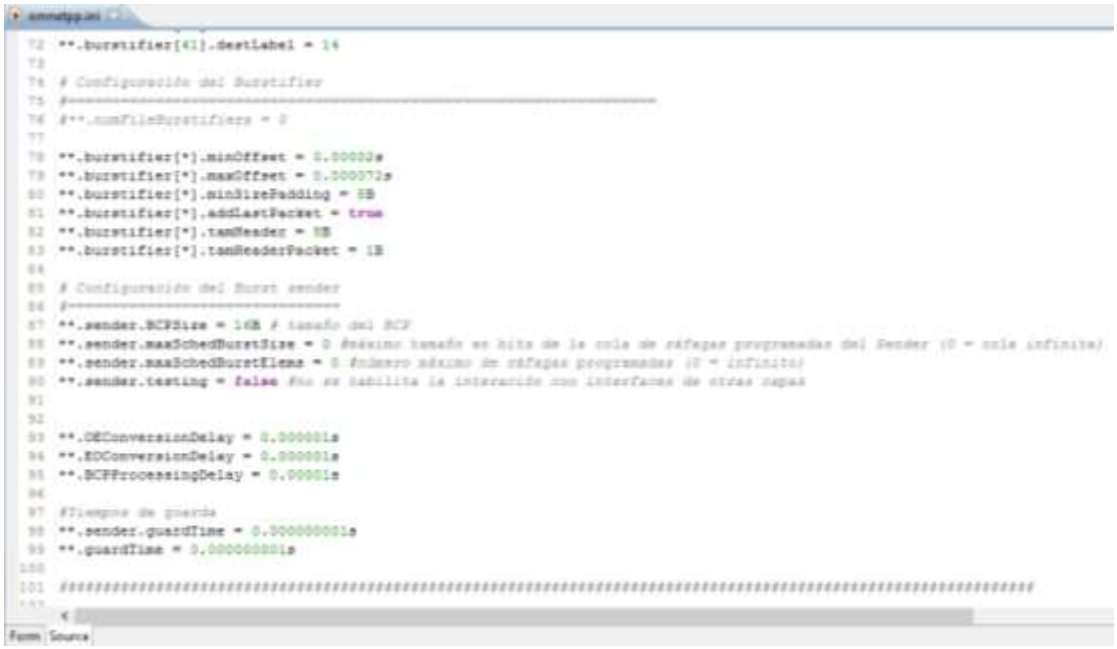
$$\text{maxOffset} = 72\mu s$$

  Ahora bien, el procedimiento para hallar el offset mínimo, parte del camino más corto que una ráfaga tendría que recorrer para llegar a su destino, esta ruta corresponde a dos nodos adyacentes. Por otra parte, sabiendo que por nodo existen tres tipos de retardo, uno debido al procesamiento del BCP y los demás, producto de la conversión O-E y E-O, las operaciones para calcular este offset son:

$$\text{minOffset} = (((1\mu s)\text{x}2) + 10\mu s)\text{x}2$$

$$\text{minOffset} = 24\mu s$$

- Se configuran además otros parámetros como: el tamaño mínimo que una ráfaga debe tener (“minSizePadding”), el tamaño de su cabecera (“tamHeader”), el de la cabecera adicionada a cada paquete dentro de la ráfaga y el del BCP; (“tamHeaderPacket”), la opción de adicionar o no el último paquete entrante antes de ensamblar la ráfaga, los criterios para llenar la cola de ráfagas y para el número de ráfagas programadas, y por último el tiempo de guarda definido como el offset entre transmisiones consecutivas de ráfagas. En la Figura 34 se muestran los parámetros descritos anteriormente.



```
**.burstifier[41].destLabel = 14

# Configuración del Burstifier
#---------------------------------------------------------------------
#**.numFileBurstifiers = 0

**.burstifier[*].minOffset = 0.000002s
**.burstifier[*].maxOffset = 0.000072s
**.burstifier[*].minSizePadding = 0B
**.burstifier[*].addLastPacket = true
**.burstifier[*].tamHeader = 0B
**.burstifier[*].tamHeaderPacket = 1B

# Configuración del Burst sender
#---------------------------------
**.sender.BCPSize = 16B # tamaño del BCP
**.sender.maxSchedBurstSize = 0 #máximo tamaño en bits de la cola de ráfagas programadas del Sender (0 = cola infinita)
**.sender.maxSchedBurstElems = 0 #número máximo de ráfagas programadas (0 = infinito)
**.sender.testing = false #no se habilita la interacción con interfaces de otras capas


**.OEConversionDelay = 0.000001s
**.EOConversionDelay = 0.000001s
**.BCPProcessingDelay = 0.00001s

#Tiempos de guarda
**.sender.guardTime = 0.000000001s
**.guardTime = 0.000000001s

##########################################################################################################################
```

Figura 34. Parámetros de configuración de la red.

- Se tienen en cuenta las longitudes de onda entre nodos frontera y centrales; para esto se crean las siguientes configuraciones:

  ✓ NSFnet1: posee dos longitudes de onda entre módulos frontera y cuatro longitudes de onda entre los módulos centrales.

  ✓ NSFnet2: presenta ocho longitudes de onda entre módulos frontera y ocho longitudes de onda entre los módulos centrales.

  ✓ NSFnet3: tiene diez y seis longitudes de onda entre módulos frontera y diez y seis longitudes de onda entre los módulos centrales.

- Velocidad de transmisión: hace referencia a la velocidad con la cual las ráfagas atraviesan la fibra óptica y viajan a través de los nodos de la red. Teniendo en cuenta los trabajos relacionados, en la mayoría de los casos, las longitudes de onda soportan una velocidad de 1, 10 o 100Gbps. A manera de ejemplo, la red óptica europea, Geant, opera a 10Gbps. [86], [104], [105], [106], [107].

- Los parámetros para determinar los criterios de ensamble son tres: el primero, denominado “maxTime”, indica que una ráfaga puede ser ensamblada, una vez se cumpla un tiempo máximo; “maxSize” establece que una ráfaga se ensambla si los paquetes agrupados cumplen con el tamaño asignado; por último, se encuentra el ensamble por paquetes o “numPackets”, el cual permite formar una ráfaga si al menos tres paquetes presentan igual destino y prioridad.

4.3 DESARROLLO DE LA RED OBS/WDM CON ENRUTAMIENTO ESTÁTICO.

El primer prototipo realizado para verificar el correcto funcionamiento del RWA en la red OBS/WDM, corresponde a una red que utiliza enrutamiento estático ideal; para lo anterior fue necesario realizar tablas de enrutamiento de forma manual, en donde las direcciones para la comunicación entre nodos corresponden al camino más corto que la ráfaga tendría que recorrer de origen a destino. Cada nodo presenta su propia tabla de enrutamiento lo cual genera 14 tablas para este prototipo.

El módulo “RoutingNuevo” es el encargado de verificar el camino que seguirá la ráfaga, comprobando si cada salto corresponde a un nodo intermedio en el camino, o al destino de la ráfaga.

Para comprobar que el RWA presenta un adecuado funcionamiento, se simula el prototipo, constatando que los caminos que recorren las ráfagas corresponden a los establecidos en las tablas de enrutamiento.

En la Figura 35 se muestra el archivo que contiene las tablas de enrutamiento.

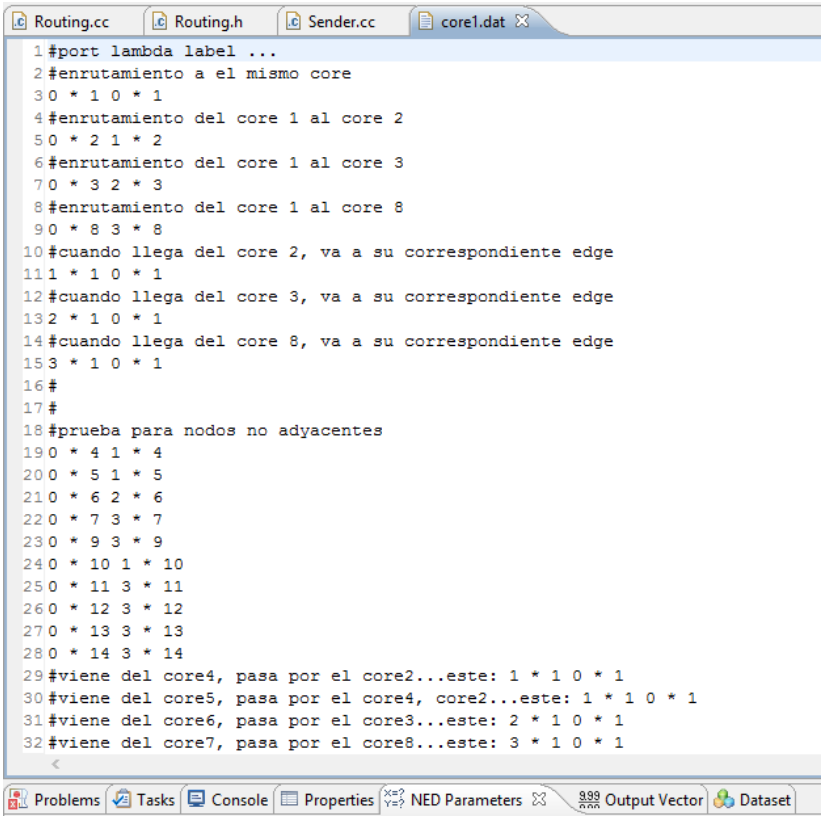


Figura 35. Tabla de enrutamiento.

Luego de verificar que el prototipo anterior funciona correctamente, se reemplazan las tablas de enrutamiento por el algoritmo del camino más corto con ayuda de la función cTopology; este proceso se implementa en el módulo “RoutingNuevo”.

Al poner en marcha el segundo prototipo, se comprueba que los caminos que sigue la ráfaga de origen a destino, correponden a los más cortos.

4.4 DESARROLLO DE LA RED OBS/WDM IMPLEMENTANDO EL ALGORITMO GENÉTICO DISEÑADO PARA EL RWA.

En esta etapa se reemplaza el algoritmo del camino más corto por el algoritmo genético, usando para esto 14 métodos que implementan las etapas del diagrama de flujo mostrado anteriormente en la Figura 23.

Cabe resaltar que en este prototipo se realiza enrutamiento dinámico, en el cual, las peticiones para el establecimiento de una ruta se generan en tiempo real, según la demanda de tráfico.

Con el fin de evidenciar que el enrutamiento y la asignación de longitudes de onda funcionan correctamente, en la interfaz gráfica de usuario se muestran las rutas después de aplicar los operadores genéticos hasta que se presenta el criterio de parada, obteniendo el camino óptimo para la petición de la ráfaga en ese instante. El término óptimo hace referencia a rutas cortas, con longitudes de onda libres entre origen y destino, y manteniendo un número de enlaces prudencial en cada nodo, según la función de aptitud realizada.

4.4.1 Parámetros de diseño del algoritmo genético para el RWA en redes OBS/WDM.

Los parámetros de diseño de la red que implementa algoritmos genéticos (además de los nombradas anteriormente) se encuentran en la clase Routing.cc, los cuales pueden ser modificados con el fin de analizar la probabilidad de bloqueo.

De acuerdo a lo anterior, las características modificables del AG son:

- Población inicial: indica el número de individuos aleatorios generados al inicio del algoritmo genético.

- Número de generaciones: hace referencia al número de iteraciones que el algoritmo genético presenta, antes de generar la solución.

4.5 DESARROLLO DE LA RED OBS/WDM USANDO EL MÉTODO DE DE CONTROL COGNITIVO BASADO EN ALGORITMOS GENÉTICOS PARA EL PROBLEMA RWA.

Las características propias de la red basada en Algoritmos genéticos con control cognitivos se encuentran también en la clase “Routing.cc”; en esta clase es posible modificar el vector de rutas cognitivas (bestRoutes) el cual almacena las soluciones generadas por el AG.

Debido a la codificación implementada en OMNeT++ y a la programación orientada a objetos C++, la misma clase Routing.cc se usa para todos los nodos de la red, sin embargo, se crean objetos diferentes para cada nodo y por lo tanto los valores, datos almacenados y los resultados son diferentes de un nodo a otro. Es por esta razón que cada nodo posee un vector de rutas cognitivas las cuales comienzan por la dirección de origen del módulo y pueden tomar direcciones de destino que varían dependiendo de los resultados generados por el algoritmo genético. El objetivo de este vector es

generar un conjunto de soluciones que pueda usar el algoritmo en la primera etapa de su proceso: la generación de rutas aleatorias. De esta forma, el método cognitivo es implementado si alguna solución perteneciente al vector, coincide con la petición proveniente de una ráfaga que desea una ruta para llegar a un destino específico. Si alguna de las rutas en el vector contiene el destino que la ráfaga desea, ésta formará parte del grupo de individuos aleatorios y realizará todas las etapas del algoritmo genético. En consecuencia, entre más grande sea el vector de rutas cognitivas, aumenta la probabilidad de que una de estas soluciones sea escogida para formar parte del proceso genético. Más adelante se analizarán las implicaciones que tiene este proceso cognitivo en la probabilidad de bloqueo de la red.

4.5.1 Parámetros de diseño del método de control cognitivo basado en algoritmos genéticos para el RWA en redes OBS/WDM.

De acuerdo a lo explicado anteriormente, el vector de rutas cognitivas es el parámetro de diseño variable que incide directamente en el desempeño de esta red.

La tabla 3 contiene los de forma detallada los parámetros de diseño implementados en la red OBS/WDM.

Tabla 3. Parámetros de diseño implementados en la red OBS/WDM.

Parámetro	Nombre de la variable en OMNeT++	Fijo	Variable	Parámetro de la red OBS/WDM	Parámetro del algoritmo genético	Parámetro del método de control cognitivo	Valores o intervalos
Funciones de distribución exponencial	sendIaTime	✓		✓			32 µs
Longitud del paquete	packetLength	✓		✓			1472 Bytes
Archivo de reglas	rulesFile	✓		✓			14 reglas
Tiempo de procesamiento de conversor OE y EO	OEConversionDelay EOConversionDelay	✓		✓			1 µs
Tiempo de procesamiento del BCP	BCPProcessingDelay	✓		✓			10 µs
Tiempo Offset	minOffset maxOffset		✓	✓			20 – 72 µs 20 – 42 µs 20 – 32 µs
Longitudes de onda	lambdasCore		✓	✓			4 λ 8 λ 16 λ
Velocidad de transmisión	dataRate		✓	✓			1 Gbps 10 Gbps
Población Inicial	numRutas		✓		✓		4 Rutas 5 Rutas 6 Rutas
Número de Generaciones	numGeneraciones		✓		✓		1 Generación 2 Generaciones 3 Generaciones
Vector cognitivo de rutas	maxCognitive	✓				✓	4 Rutas
Número de Burstifiers	numBurstifiers	✓		✓			42 Burstifiers
Tamaño mínimo permitido de la ráfaga	minSizePadding	✓		✓			8 Bytes
Tamáño de la cabecera de la ráfaga en el módulo Burstifier	tamHeader	✓		✓			8 Bytes

Tamaño de la cabecera de cada paquete en la ráfaga en el módulo Burstifier	tamHeaderPacket	✓		✓			1 Byte
Tamaño de la cabecera de la ráfaga en el módulo Sender	BCPSize	✓		✓			16 Bytes
Máximo número de bits en la cola del módulo Sender	maxSchedBurstSize	✓		✓			Sin límite
Máximo número de ráfagas permitidas en la cola del móudlo Sender	maxSchedBurstElems	✓		✓			Sin límite
Tiempo de Guarda	guardTime	✓		✓			1 ns

4.6 PLAN DE PRUEBAS.

4.6.1 Consideraciones preliminares.

- En las redes OBS/WDM las ráfagas representan la unidad básica de transporte, sin embargo, la pérdida de una ráfaga implica la eliminación de numerosos paquetes o de gran cantidad de información. Los resultados harán alusión por lo tanto a ráfagas pérdidas, recibidas o programadas.
- Cabe notar que al comenzar la simulación de las redes analizadas, éstas almacenarán datos paulatinamente, atravesando régimen transitorio. En este tiempo los paquetes y las ráfagas no alcanzan a completar la capacidad de la red, por lo tanto, los datos obtenidos en este lapso de tiempo no serán completamente confiables para analizar la probabilidad de bloqueo. El siguiente periodo corresponde al régimen permanente, lapso en el cual es ideal realizar la toma de datos y el análisis de estadísticas. Finalmente se presenta un último estado transitorio, donde para envío de paquetes; se genera entonces la siguiente situación: los datos sobre las ráfagas que ya han sido registradas y que están circulando por la red deben seguir su curso llegando al módulo de destino o perdiéndose en el camino; no obstante, esto debe realizarse sin que ocurra un disminución en los paquetes y ráfagas que se encuentran actualmente en formación, ya que se utilizan como tráfico actual circundante permitiendo obtener datos fiables de probabilidad de bloqueo. La Figura 36, muestra los periodos de tiempo analizados anteriormente.

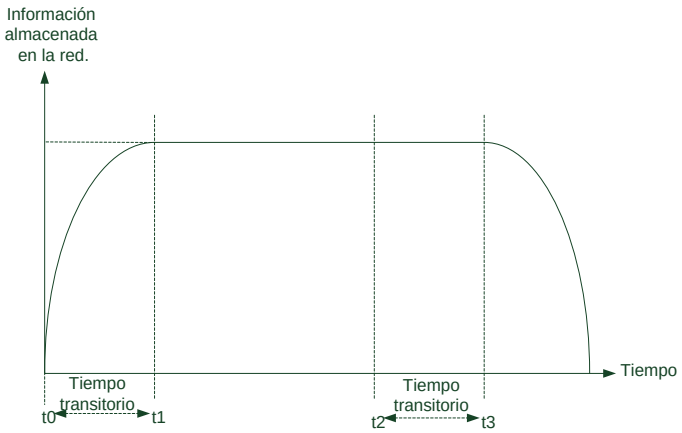


Figura 36. Estado de tiempos de la red [108].

- Tiempos de simulación: las estimaciones de los periodos de tiempo evaluados en las redes se realizan teniendo en cuenta que la función exponencial de 32us implementada, genera un tiempo de simulación inicial de 11.3 us. Se toman muestras en los tiempos mostrados en la interfaz de usuario gráfica de OMNeT++ de 0.002s, 0.004s y 0.006s obtenidos después de una, dos y tres horas respectivamente, en los cuales la red presenta un comportamiento estable.

- Prueba de probabilidad de bloqueo: permite obtener la probabilidad de bloqueo de las ráfagas en la red OBS/WDM por medio de la siguiente ecuación [105].

$$\mathrm{Pb} = \frac{\text{Número de ráfagas perdidas}}{\text{Número de ráfagas totales}} \qquad (6)$$

Donde el número de ráfagas perdidas y ráfagas totales se obtiene en el módulo “coreControlLogic” del nodo central, dicho proceso se lleva a cabo mediante contadores implementados en el código que se pueden visualizar gracias a la función WHATCH (método que permite visualizar las variables dentro de la interfaz gráfica de usuario).

- Prueba de tiempo de procesamiento: hace referencia al tiempo que toma la red para la generación de una ruta, este factor se evalúa variando algunos parámetros de la red como el número de generaciones y población inicial del algoritmo.

  En la Figura 37 se muestra la ventana sobre la cual se realiza el análisis del tiempo de procesamiento al aplicar AG con y sin métodos de control cognitivo. El tiempo es tomado desde el módulo “Sender” en el momento en el cual se ha establecido una ruta; se toman muestras para diferentes números de ráfagas enrutadas y se realizan las comparaciones respectivas.

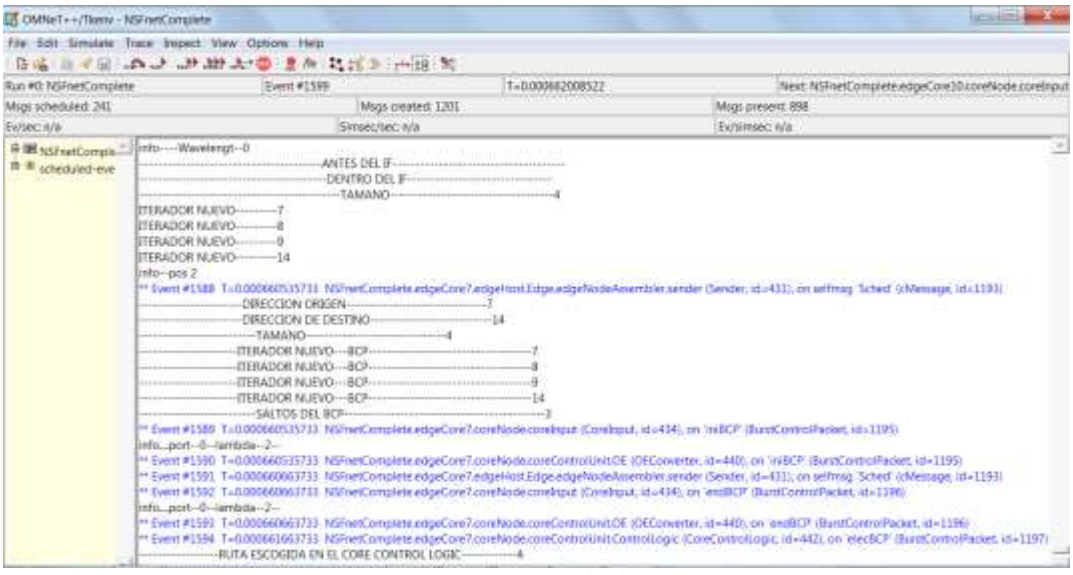


Figura 37. Interfaz gráfica de usuario para el análisis del tiempo de procesamiento.

- Otro criterio importante de evaluación es la relación de los resultados con procesos de probabilidad provenientes de la implementación del algoritmo genético y el método de control cognitivo aplicado a la red óptica OBS/WDM. Esto implica que las muestras tomadas para realizar la evaluación de resultados, deben formar parte de una población y se analizarán como conjunto para obtener el valor medio de probabilidad de bloqueo y tiempo de procesamiento originado por los procesos para el RWA.

  Para generar datos confiables se recurre a la teoría de la estimación, definida como la rama de la estadística que estudia las técnicas usadas para proporcionar un valor aproximado a un parámetro o variable a partir de datos empíricos o medidas, [109], [110], [111]. Después de realizar un análisis de los métodos de estimación, se escoge el Método Estimador de Máxima Verosimilitud ya que éste presenta las siguientes características:

  - ✓ El parámetro al que se le quiere hallar el valor esperado puede ser determinista o aleatorio.
  - ✓ Se estima que exista una función de densidad de probabilidad[11] dependiente del parámetro evaluado.
  - ✓ Se debe conocer un conjunto de parámetros cuyo comportamiento se estima al modelar los datos de forma probabilística gracias a la función de densidad aproximada.
  - ✓ El estimador de máxima verosimilitud permite obtener un valor esperado del parámetro analizado que dependa de los datos conocidos.
  - ✓ Conociendo los datos del parámetro y su comportamiento, es posible acercarse a la máxima verosimilitud del parámetro generando la distribución con la que los datos son más probables.

  En este caso, los parámetros hacen referencia a los valores de probabilidad de bloqueo y tiempo de respuesta obtenidos como resultado al analizar el comportamiento de la red usando algoritmos genéticos con y sin método de control cognitivo.

  Por criterios de diseño y para disminuir la complejidad de los procesos realizados, la función de densidad de probabilidad que siguen los datos se puede analizar individualmente para cada muestra tomada, sin embargo, las gráficas de estos resultados no son expuestos en el trabajo de grado ya que el tamaño de las muestras es numeroso [109].

  Ahora bien, es preciso estimar el número de parámetros a evaluar. Para cada caso se recurrió a un modelo experimental en cuyo proceso se tomaron N muestras, deseando obtener un margen de error del 15%. El resultado del proceso anterior indica que se requieren 20 muestras para el análisis de la probabilidad de bloqueo y 10 muestras para generar los resultados del tiempo de procesamiento.

- Los resultados de las pruebas de tiempo de procesamiento se encuentran condicionados a las características de los equipos en donde se ejecutan los modelos de red. En este caso, para realizar el plan de pruebas se usaron 4 equipos. A continuación de especifican sus características.

[11] Función de densidad de probabilidad: La distribución de probabilidad de una variable aleatoria discreta se presenta como la lista de los distintos valores xi que puede tomar la variable aleatoria X, junto con sus probabilidades asociadas f(xi) = P(X = xi), esto es, el conjunto de parejas {xi, f(xi)}. http://www.sites.upiicsa.ipn.mx/polilibros/portal/Polilibros/P_terminados/Probabilidad/doc/Unidad%202/2.4.HTM.

Tabla 4. Especificaciones Técnicas de los equipos empleados para las pruebas de los dos modelos de red.

Equipo / Especificaciones Técnicas	Equipo 1	Equipo 2	Equipo 3	Equipo 4
Microprocesador	AMD Athlon (TM) Dual-Core 2.10 GHz	Intel CORE i3 2.20 GHz	Intel CORE i3 2.3 GHz	Intel CORE i5 1.7 GHz
Memoria	2 GB	6 GB	4 GB	4 GB
Disco Duro	220 GB	700 GB	600 GB	500 GB
Sistema Operativo	64 bits, Windows 8	64 bits, Windows 8	32 bits, Windows 7	64 bits, Windows 8

4.7 Análisis de resultados sobre el desempeño del RWA para redes ópticas OBS/WDM basadas en algoritmos genéticos con y sin métodos de control cognitivo.

Después de analizar los parámetros propuestos para simular de forma adecuada la red OBS/WDM, el algoritmo genético y el método de control cognitivo, la tabla 5 permite visualizar las características que se mantendrán fijas a lo largo de las simulaciones, ya que la modificación de alguna de ellas, o bien, no incide de manera considerable en el desempeño, o no es posible realizar suficientes pruebas debido al tiempo que demandan dichos procesos, dejando el cambio de algunos de estos parámetros como trabajo futuro.

Tabla 5. Características fijas en las redes.

Parámetros	Características
Función exponencial	exp (32us)
Longitud de los paquetes	1472 bytes
Archivo de reglas	42 reglas
Tiempo de procesamiento en conversores EO y OE	1us
Tiempo de procesamiento del BCP	10us
Vector cognitivo de rutas	4 rutas

4.7.1 Escenarios de simulación para analizar la probabilidad de bloqueo en redes OBS/WDM basada en algoritmos genéticos con y sin métodos de control cognitivo.

4.7.1.1 Variación de tiempo Offset.

El primer escenario considerado, tiene en cuenta la variación del tiempo Offset máximo, empezando con el intervalo comprendido entre 20-72us, seguido por 20-42us y finalmente 20-32us; para cada rango se realizan numerosas pruebas en las que se generan los valores correspondientes al número de ráfagas perdidas, programadas y recibidas en los periodos de tiempo mencionados en el ítem

anterior. Los resultados de este proceso permiten obtener la probabilidad de bloqueo para cada tiempo analizado.
Las caracteríscias establecidas para la primera prueba se muestran en la tabla 6.

Tabla 6. Características establecidas en la primera prueba.

Parámetros	Características
Velocidad de transmisión	1 Gbps.
Número de Generaciones del AG	1 Generación
Población inicial del AG	6 rutas
Longitudes de Onda	NSFNeT 1

En las Figuras 38, 39, 40 y 41 que se muestran a continuación, se realiza la comparación de las ráfagas recibidas, programadas, perdidas, y la probabilidad de bloqueo entre las redes ópticas OBS/WDM basadas en algoritmos genéticos con y sin cognición.

- **Intervalo Offset desde 20us a 72 us.**

Tabla 7. Probabilidad de bloqueo con Offset máximo de 72us.

EXP=(32us) Población inicial 6. Offset máximo 72us.	Probabilidad de bloqueo para AG.	Probabilidad de bloqueo para AG con cognición.
0.002s	0.013398	0.00904438
0.004s	0.016826	0.01601
0.006s	0.01834	0.01888

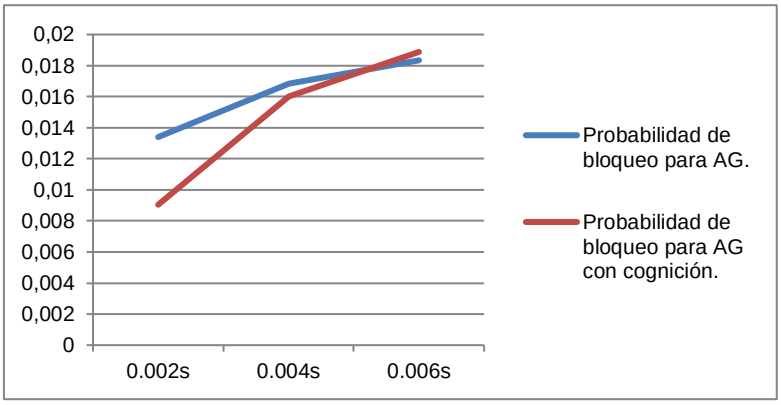


Figura 38. Probabilidad de bloqueo con Offset máximo de 72us.

La figura 38 muestra que para dos de los tres tiempos analizados la probabilidad de bloqueo correspondiente al algoritmo genético con control cognitivo es menor.

- **Intervalo Offset desde 20us a 42 us.**

Tabla 8. Probabilidad de bloqueo con Offset máximo de 42us.

EXP=(32us) Población inicial 6. Offset máximo 42us.	Probabilidad de bloqueo para AG.	Probabilidad de bloqueo para AG con cognición.
0.002s	0.0593	0.05894
0.004s	0.07066	0.0661
0.006s	0.06966	0.0666

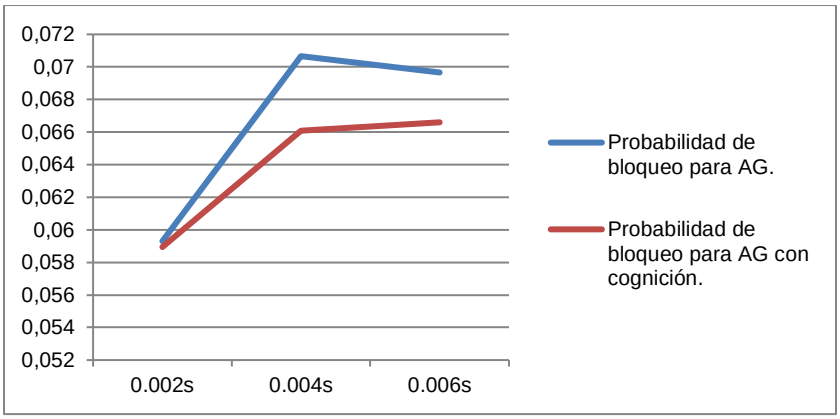


Figura 39. Probabilidad de bloqueo con Offset máximo de 42us.

- **Intervalo Offset desde 20us a 32 us.**

Tabla 9. Probabilidad de bloqueo con Offset máximo de 32us.

EXP=(32us) Población inicial 6. Offset máximo 32us.	Probabilidad de bloqueo para AG.	Probabilidad de bloqueo para AG con cognición.
0.002s	0.1342	0.1276
0.004s	0.13364	0.12998
0.006s	0.13136	0.13084

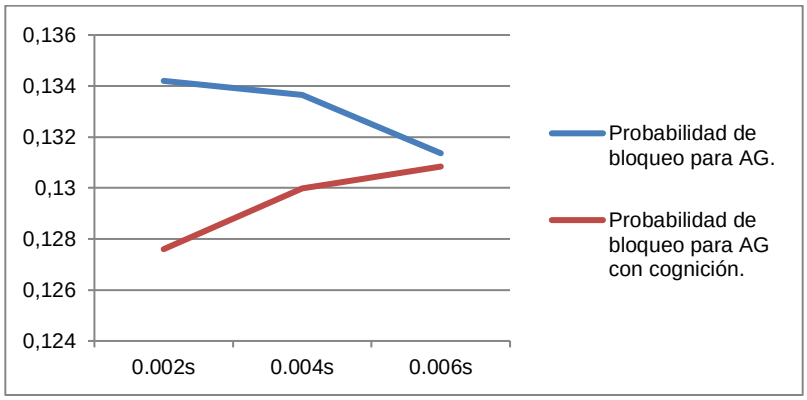


Figura 40. Probabilidad de bloqueo con Offset máximo de 32us.

Tabla 10. Probabilidad de bloqueo de la red usando algoritmos genéticos para diferentes tiempos máximos de Offset.

EXP=(0.000032s)	Probabilidad de bloqueo de la red con AG. (PBAG 72us)	Probabilidad de bloqueo de la red con AG. (PBAG 42us)	Probabilidad de bloqueo de la red con AG. (PBAG 32us)
0.002s	0.013398	0.0593	0.1342
0.004s	0.016826	0.07066	0.13364
0.006s	0.01834	0.06966	0.13136

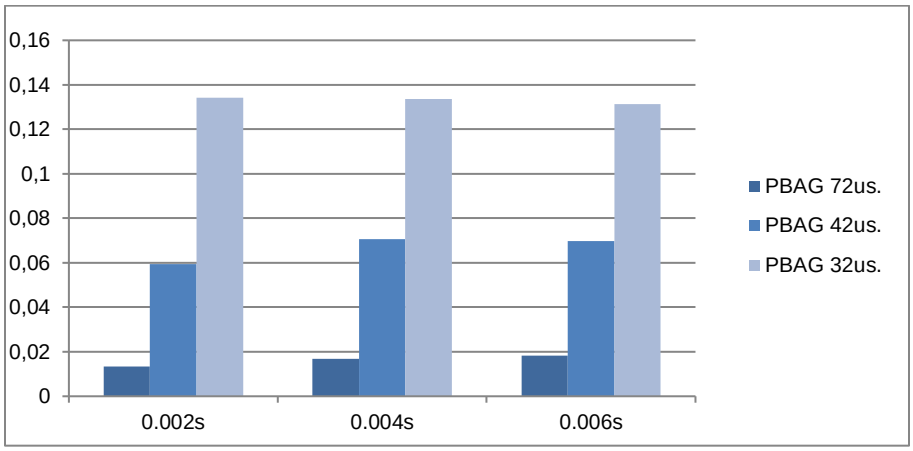


Figura 41. Probabilidad de bloqueo de la red usando algoritmos genéticos para diferentes tiempos máximos de Offset.

La Figura 41 muestra la variación de la probabilidad de bloqueo al cambiar los rangos de tiempo Offset, como se puede observar para el intervalo de tiempo más amplio (20-72us), las probabilidades de bloqueo son mucho menores en comparación con los otros intervalos, siendo el intervalo menor (20-32us) el que muestra el peor desempeño acercando sus valores de probabilidad de bloqueo a 0.14.

Tabla 11. Probabilidad de bloqueo de la red usando algoritmos genéticos con control cognitivo y diferentes tiempos máximos de Offset.

EXP=(0.000032s)	Probabilidad de bloqueo de la red con AG y control cognitivo (PBAGC 72us)	Probabilidad de bloqueo de la red con AG y control cognitivo (PBAGC 42us)	Probabilidad de bloqueo de la red con AG y control cognitivo (PBAGC 32us)
0.002s	0.00904438	0.05894	0.1276
0.004s	0.01601	0.0661	0.12998
0.006s	0.01888	0.0666	0.13084

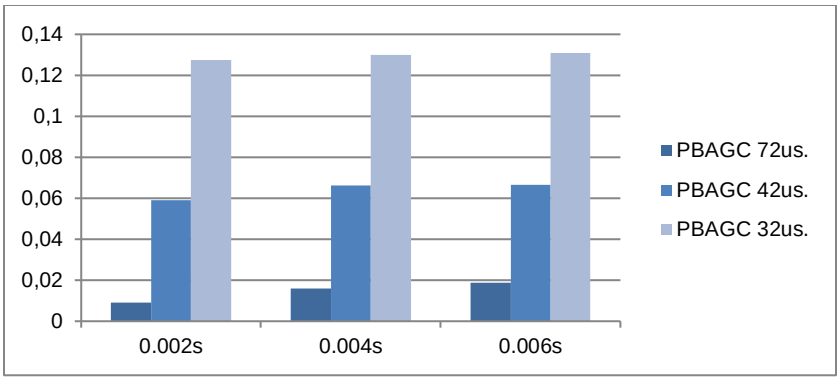


Figura 42. Probabilidad de bloqueo de la red usando algoritmos genéticos con control cognitivo y diferentes tiempos máximos de Offset.

La Figura 42 muestra la variación de la probabilidad de bloqueo al cambiar los rangos de tiempo Offset, como se puede observar para el intervalo de tiempo más amplio (20-72us), las probabilidades de bloqueo son mucho menores en comparación con los otros intervalos, siendo el intervalo menor (20-32us) el que muestra el peor desempeño acercando sus valores de probabilidad de bloqueo a 0.13.

Tabla 12. Comparación de la probabilidad de bloqueo de la red usando algoritmos genéticos con y sin cognición para diferentes tiempos máximos de Offset.

EXP=(32us) Población inicial 6.	Probabilidad de bloqueo para AG. (PBAG 72us)	Probabilidad de bloqueo para AG con cognición. (PBAGC 72us)	Probabilidad de bloqueo para AG. (PBAG 42us)	Probabilidad de bloqueo para AG con cognición. (PBAGC 42us)	Probabilidad de bloqueo para AG. (PBAG 32us)	Probabilidad de bloqueo para AG con cognición. (PBAGC 32us)
0.002s	0.013398	0.00904438	0.0593	0.05894	0.1342	0.1276
0.004s	0.016826	0.01601	0.07066	0.0661	0.13364	0.12998
0.006s	0.01834	0.01888	0.06966	0.0666	0.13136	0.13084

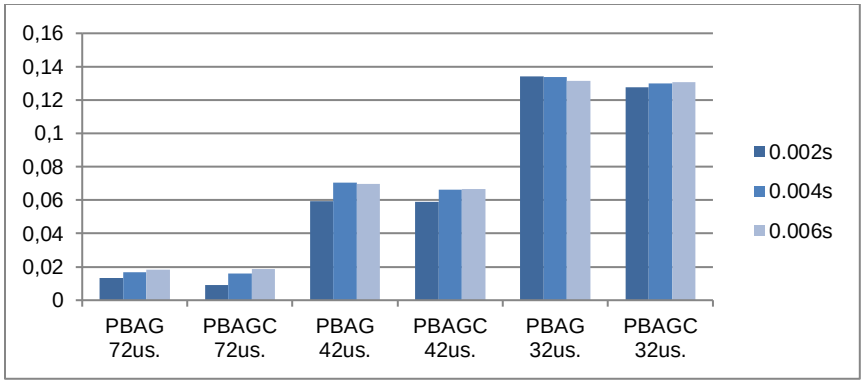


Figura 43. Comparación de la probabilidad de bloqueo de la red usando algoritmos genéticos con y sin cognición para diferentes tiempos máximos de Offset.

La Figura 43 muestra la variación de la probabilidad de bloqueo al cambiar los rangos de tiempo Offset cuando se trabaja con AG con y sin cognición, como se puede observar para todos los intervalos de tiempo se obtiene mejor respuesta usando el AG con control cognitivo, para el cual las probabilidades de bloqueo se encuentran en el mejor de los casos (para el intervalo de 20-72us) entre 0.009 y 0.018.

Del comportamiento de las redes frente a la probabilidad de bloqueo, se puede mencionar que para los valores de tiempo analizados, variando los rangos de offset entre ellos, de un total de 9 resultados, solo en un caso la probabilidad de bloqueo es menor para la red que implementa algoritmos genéticos sin control cognitivo. De esto se infiere que el existe un 11.11% de probabilidad (para la red desarrollada), que el desempeño de la red mejore usando el método de AG sin cognición en comparación con el algoritmo que implementa el método de control cognitivo. Sin embargo, se debe tener en cuenta que el margen de confiabilidad de los resultados es del 85%.

4.7.1.2 Variación de la velocidad de transmisión.

En este escenario se evalúa el desempeño en términos de la probabilidad de bloqueo al variar la velocidad de transmisión.

Las características asignadas para esta prueba se muestran en la siguiente tabla.

Tabla 13. Características establecidas en la segunda prueba.

Parámetros	Características
Intervalo Offset	(20 - 72) us
Número de Generaciones del AG	1 Generación
Población inicial del AG	6 rutas
Longitudes de Onda	NSFNeT 1

Por criterios de diseño se escogen las velocidades de 1 y 10 Gbps; en la tabla 14 se puede observar el efecto generado por este parámetro.

Tabla 14. Efecto de la variación de la velocidad de transmisión para un Offset máximo de 72us.

EXP=(0.000032s)	PBAG 1Gbps.	PBAGC 1Gbps.	PBAG 10Gbps.	PBAGC 10Gbps.
0.002s	0.013398	0.009044	0.01028	0.0034646
0.004s	0.016926	0.01601	0.01078	0.0095946
0.006s	0.01834	0.01888	0.01078	0.01078

La figura 44 permite analizar el desempeño de los algoritmos al aplicar diferentes velocidades en los caminos ópticos. Se obtiene una mejor respuesta en términos de probabilidad de bloqueo para los AG que trabajan con cognición, utilizando una velocidad de 10Gbps.

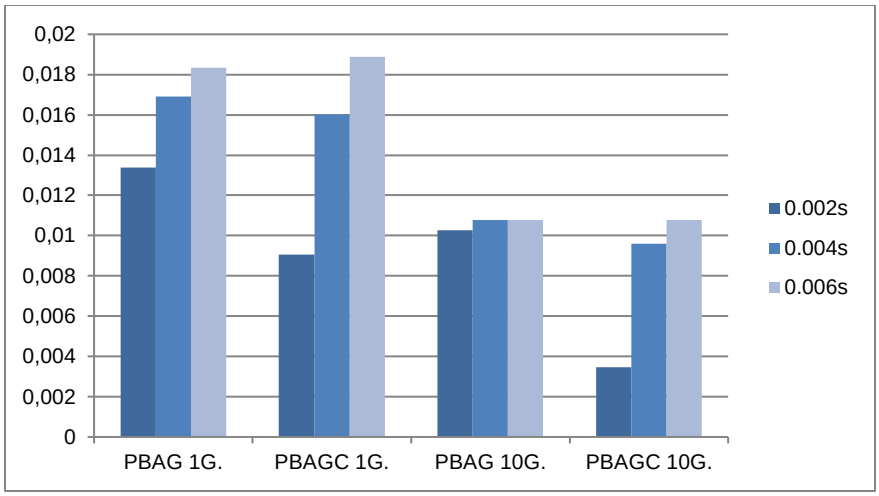


Figura 44. Efecto de la variación de la velocidad de transmisión.

Para velocidades más grandes se obtienen mejores respuestas en cuanto a probabilidad de bloqueo, ya que el tiempo de transmisión de una ráfaga entre dos nodos se define como el cociente entre el tamaño de la ráfaga sobre la velocidad de transmisión. Esto implica que a mayores tasas de velocidad, la ráfaga tarda un menor tiempo en viajar por los caminos ópticos, logrando menores pérdidas de ráfagas debido al tiempo offset.

Usando una velocidad de transmisión de 10Gbps y offset máximo de 72us, los valores de probabilidad de bloqueo disminuyen para el algoritmo genético con control cognitivo en dos de los 3 resultados. Se infiere por lo tanto que aproximadamente en el 66% ±15% de los casos analizados (usando los parámetros anteriormente mencionados), con este método heurístico es posible obtener un mejor desempeño de la red.

De los resultados obtenidos al realizar la comparación entre velocidades de transmisión de 1 y 10 Gbps, se infiere que la probabilidad de bloqueo disminuye en un 23.22%, 36.76% y 41.53% para los tiempos de 0.002, 0.004 y 0.006 segundos (en tiempo de simulación) respectivamente, al usar velocidades entre enlaces de 10Gbps.

4.7.1.3 Variación de Longitudes de Onda.

Los resultados mostrados en este escenario dependerán del efecto de la variación de las longitudes de onda, utilizando las configuraciones establecidas en el ítem 4.2.1.

Las características asignadas para estas pruebas se muestran a continuación.

Tabla 15. Características establecidas en la tercera prueba.

Parámetros	Características
Intervalo Offset	(20 - 72) us
Número de Generaciones del AG	1 Generación
Población inicial del AG	6 rutas
Velocidad de transmisión	10Gbps

Tabla 16. Efecto de la variación de las longitudes de onda.

EXP=(0.000032s)	PBAG.	PBAGC.
4L	0.008255	0.007306
8L	0.00735	0.00712
16L	0.007308	0.006724

La probabilidad de bloqueo puede observarse en la Figura 45, indicando que el AG con control cognitivo presenta un mejor desempeño. Es posible notar que para los algoritmos implementados, se presenta una disminución en la probabilidad de bloqueo a medida que aumenta el número de longitudes de onda por enlace. Esto se debe a la influencia el número de caminos ópticos libres sobre la función de aptitud aplicada.

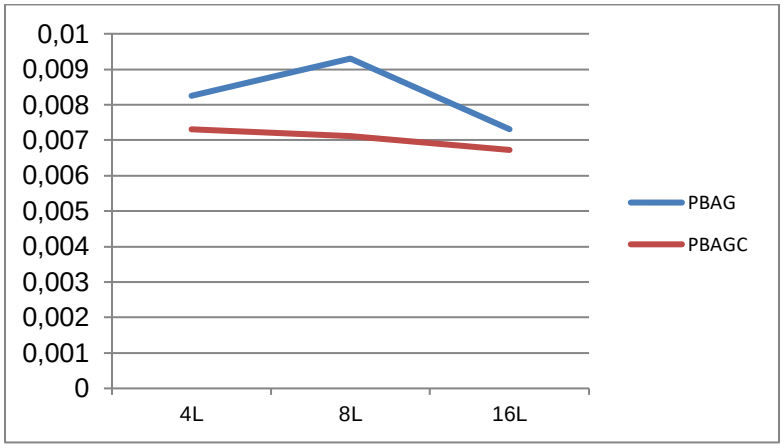


Figura 45. Efecto de la variación de las longitudes de onda.

Los resultados conseguidos al variar las longitudes de onda en las dos redes implementadas, muestran que para los tres casos analizados (es decir, el 100% de los casos con un margen de error del 15%), el comportamiento de la probabilidad de bloqueo es mejor para los AG que presentan métodos de control cognitivo. Además, este último exhibe la relación inversamente proporcional entre la probabilidad de bloqueo y el número de longitudes de onda por enlace.

4.7.1.4 Variación del número de generaciones del algoritmo genético.

El último escenario analizado tiene en cuenta el número de iteraciones que el algoritmo genético podría realizar para obtener la ruta resultante.

El escenario presenta las siguientes características.

Tabla 17. Características establecidas en la cuarta prueba.

Parámetros	Características
Intervalo Offset	(20 - 72) us
Longitudes de onda	NSFNeT 1
Población inicial del AG	6 rutas
Velocidad de transmisión	10Gbps

El efecto de la variación de las generaciones se muestra a continuación.

Tabla 18. Efecto de la variación de las generaciones en el AG.

EXP=(0.000032s)	PBAG una Generación.	PBAG dos Generaciones.	PBAG tres Generaciones.
0.002s	0.0133	0.013	0.01281
0.003s	0.01766	0.0175	0.01694
0.004s	0.02003	0.019	0.01804

En las Figuras 46, 47 y 48 se analiza el efecto en la probabilidad de bloqueo creado por el cambio de dicho parámetro.

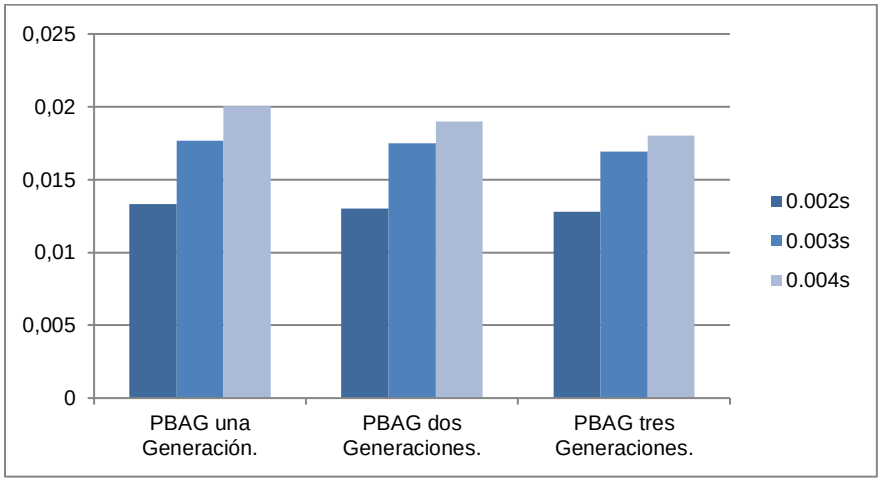


Figura 46. Efecto de la variación de las generaciones en el AG.

Tabla 19. Efecto de la variación de las generaciones en el AG con control cognitivo.

EXP=(0.000032s)	PBAGC una Generación.	PBAGC dos Generaciones.	PBAGC tres Generaciones.
0.002s	0.0102	0.0075	0.005432
0.003s	0.01357	0.00993	0.009578
0.004s	0.01489	0.01125	0.01059

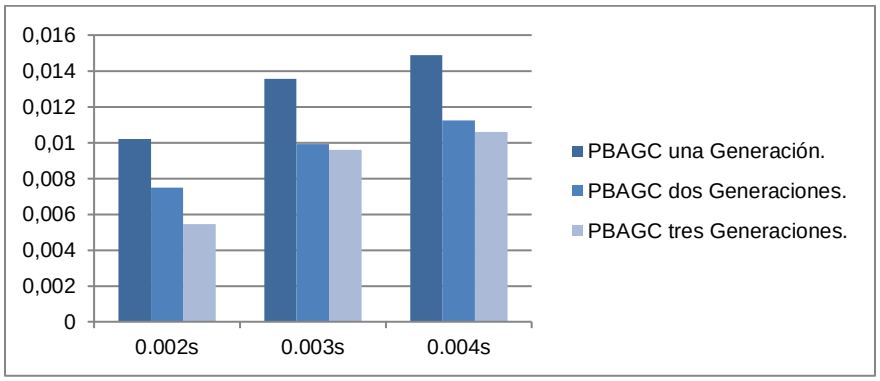


Figura 47. Efecto de la variación de las generaciones en el AG con control cognitivo.

Tabla 20. Efecto de la variación de las generaciones en el AG con y sin cognición.

EXP=(0.000032s)	PBAG una Generación.	PBAG dos Generaciones.	PBAG tres Generaciones.	PBAGC una Generación.	PBAGC dos Generaciones.	PBAGC tres Generaciones.
0.002s	0.0133	0.013	0.01281	0.0102	0.0075	0.005432
0.003s	0.01766	0.0175	0.01694	0.01357	0.00993	0.009578
0.004s	0.02003	0.019	0.01804	0.01489	0.01125	0.01059

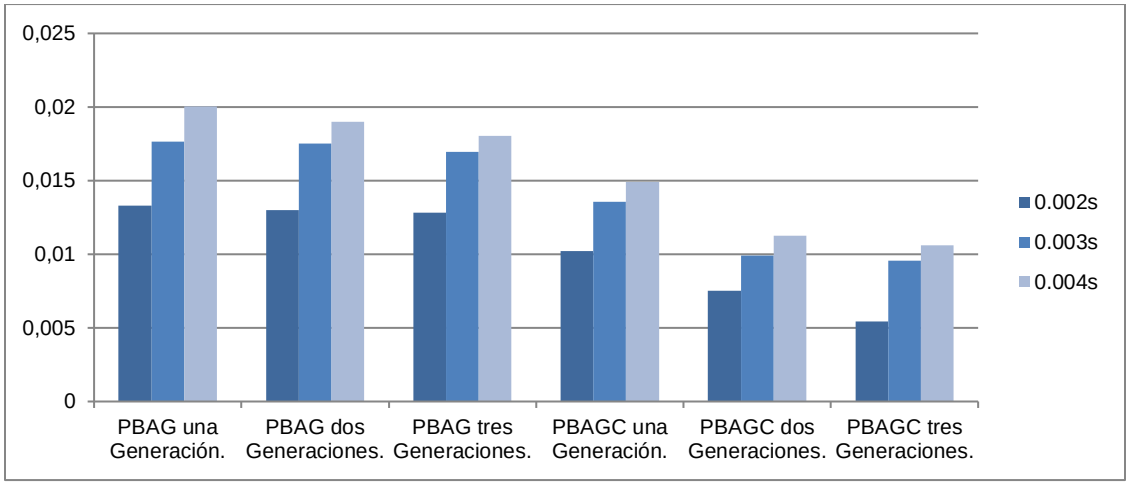


Figura 48. Efecto de la variación de las generaciones en el AG con y sin cognición.

Se puede concluir que entre más número de iteraciones se generen, la pérdida de ráfagas sobre la red es menor. Esto se debe a que aumenta la probabilidad de obtener una mejor ruta a medida que crece el número de generaciones del algoritmo.

4.7.2 Escenarios de simulación para analizar el tiempo de procesamiento en redes OBS/WDM basada en algoritmos genéticos con y sin métodos de control cognitivo.

4.7.2.1 Variación del número de rutas iniciales del algoritmo genético.

El análisis presentado a continuación permite obtener los resultados del tiempo de procesamiento generado para el establecimiento de las rutas de un número determinado de ráfagas.

El número de ráfagas utilizado para analizar los tiempos de procesamiento se limita a un valor no tan elevado debido al riguroso proceso y tiempo que demanda el análisis de las rutas desde la interfaz gráfica de usuario.

Las características asignadas al iniciar la simulación se muestran en la tabla 21.

Tabla 21. Características establecidas en la cuarta prueba.

Parámetros	Características
Intervalo Offset	(20 - 72) us
Longitudes de onda	NSFNeT 1
Número de generaciones del AG	1 Generación
Velocidad de transmisión	10Gbps

En las figuras 49, 50 y 51 se puede observar que el número de rutas iniciales tiene una relación directamente proporcional con el tiempo de procesamiento, aunque el cambio no es muy notorio debido a que los valores en los que se diferencian se encuentran en el orden de los microsegundos.

Tabla 22. Efecto de la variación del número de rutas iniciales para en análisis del tiempo de procesamiento trabajando con AG.

EXP=(0.000032s)	Tiempo de procesamiento para AG con una población inicial de 5 rutas (TPAG 5).	Tiempo de procesamiento para AG con una población inicial de 6 rutas (TPAG 6).	Tiempo de procesamiento para AG con una población inicial de 7 rutas (TPAG 7).
NR=600	0.00493	0.005179	0.005181
NR=1000	0.005545	0.005684	0.00573
NR=1200	0.006515	0.006412	0.006708

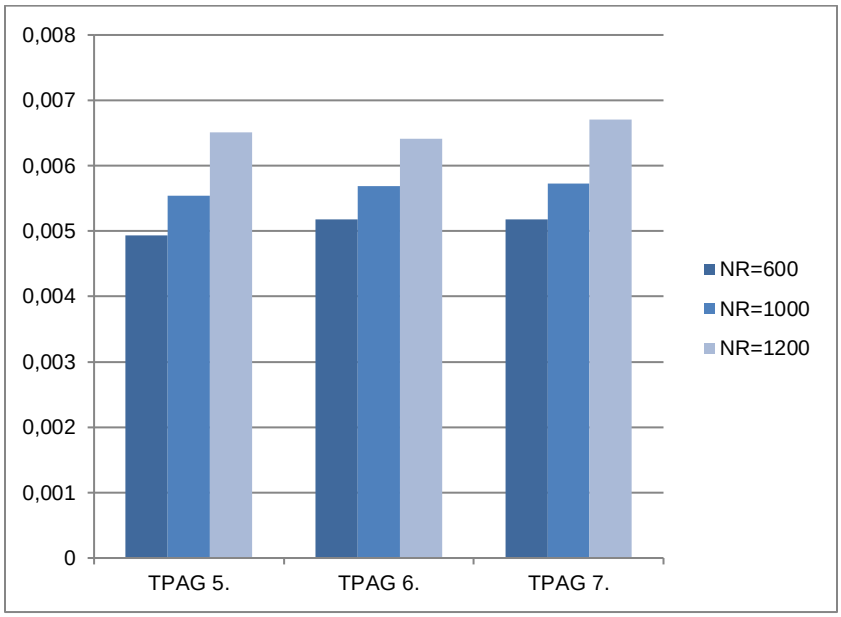


Figura 49. Efecto de la variación del número de rutas iniciales para el análisis del tiempo de procesamiento trabajando con AG.

Tabla 23. Efecto de la variación del número de rutas iniciales para en análisis del tiempo de procesamiento trabajando con AG con control cognitivo.

EXP=(0.000032s)	Tiempo de procesamiento para AG con cognición para una población inicial de 5 rutas. (TPAGC 5)	Tiempo de procesamiento para AG con cognición para una población inicial de 6 rutas. (TPAGC 6)	Tiempo de procesamiento para AG con cognición para una población inicial de 7 rutas. (TPAGC 7)
NR=600	0.005179	0.005184	0.0051997
NR=1000	0.005522	0.005475	0.005322
NR=1200	0.006433	0.006415	0.006419

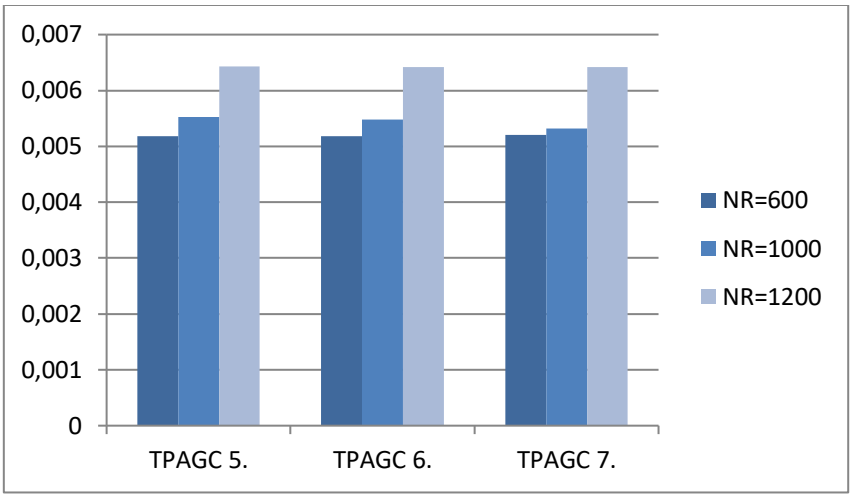


Figura 50. Efecto de la variación del número de rutas iniciales para el análisis del tiempo de procesamiento trabajando con AG con control cognitivo.

Tabla 24. Comparación para AG con y sin cognición para una población inicial de 5 rutas en la estimación del tiempo de procesamiento.

EXP=(0.000032s)	Tiempo de Procesamiento para AG con una población inicial de 5 rutas. (TPAG 5)	Tiempo de Procesamiento para AG con control cognitivo para una población inicial de 5 rutas. (TPAGC 5)
Número de Ráfagas (NR)=600	0.00493	0.005179
Número de Ráfagas (NR)==1000	0.005545	0.005522
Número de Ráfagas (NR)==1200	0.006515	0.006433

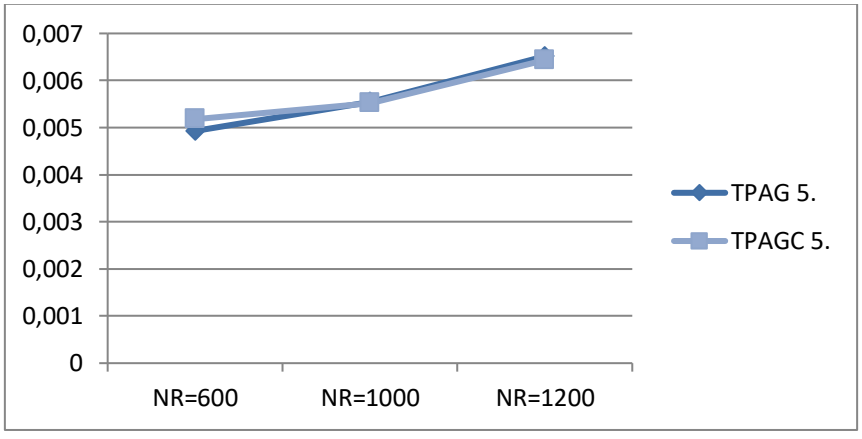


Figura 51. Comparación para AG con y sin cognición para una población inicial de 5 rutas en la estimación del tiempo de procesamiento.

Tabla 25. Comparación para AG con y sin cognición para una población inicial de 6 rutas en la estimación del tiempo de procesamiento.

EXP=(0.000032s)	Tiempo de Procesamiento para AG con una población inicial de 6 rutas. (TPAG 6)	Tiempo de Procesamiento para AG con control cognitivo para una población inicial de 6 rutas. (TPAGC 6)
Número de ráfagas (NR)=600	0.005179	0.005184
Número de ráfagas (NR)=1000	0.005684	0.005475
Número de ráfagas (NR)=1200	0.006412	0.00637

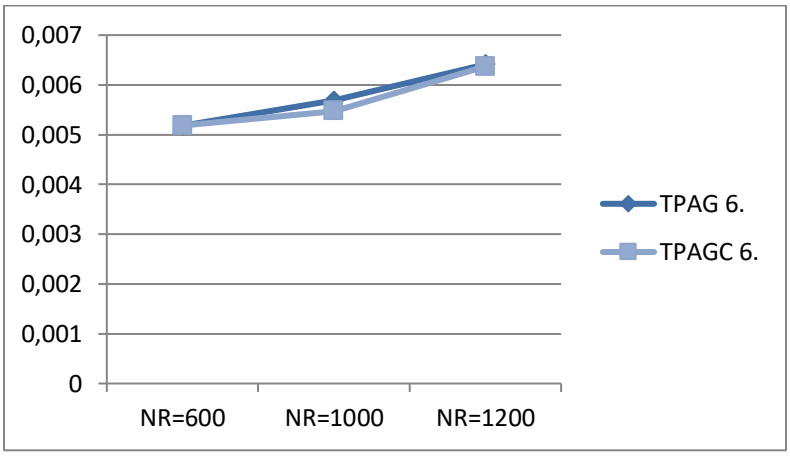


Figura 52. Comparación para AG con y sin cognición para una población inicial de 6 rutas en la estimación del tiempo de procesamiento.

Tabla 26. Comparación para AG con y sin cognición para una población inicial de 7 rutas en la estimación del tiempo de procesamiento.

EXP=(0.000032s)	Tiempo de Procesamiento para AG con una población inicial de 7 rutas. (TPAG 7)	Tiempo de Procesamiento para AG con control cognitivo para una población inicial de 7 rutas. (TPAGC 7)
Número de ráfagas (NR)=600	0.005181	0.0051997
Número de ráfagas (NR)=1000	0.00573	0.005622
Número de ráfagas (NR)=1200	0.006708	0.006699

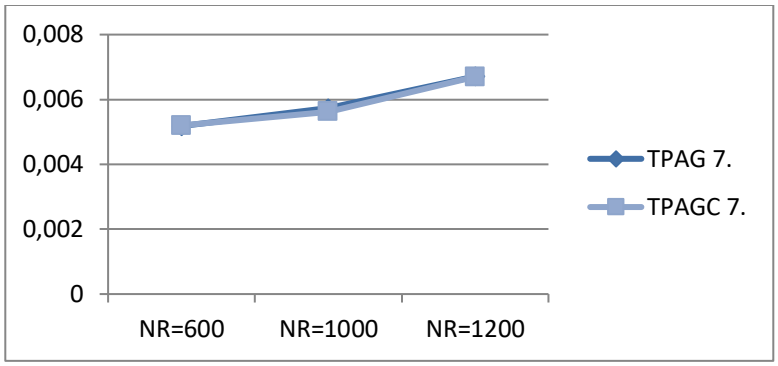


Figura 53. Comparación para AG con y sin cognición para una población inicial de 7 rutas en la estimación del tiempo de procesamiento.

En las gráficas anteriores se analiza el efecto del número de rutas iniciales en el tiempo de procesamiento, los resultados presentados para dichos tiempos por los dos métodos heurísticos, no difieren considerablemente entre sí, sin embargo, se puede observar que para el algoritmo que presenta métodos de control cognitivo los resultados son más favorables. Específicamente, en 6 de los 9 resultados, el AG con el método de control cognitivo presenta un mejor tiempo de procesamiento (menor), infiriendo que así será en el 66.66% ±15% de los casos.

5 CONCLUSIONES Y TRABAJOS FUTUROS.

5.1 Conclusiones.

Este trabajo de grado ha propuesto dos métodos basados en algoritmos genéticos (uno de los cuales implementa control cognitivo) para analizar el desempeño en los procesos de RWA dinámico en redes ópticas OBS/WDM con restricción de continuidad de longitud de onda.

El primer prototipo de red realizado en OMNeT++ incluyó el modelado de cada componente que genera los nodos frontera y centrales de la red OBS/WDM. Su agrupación forma el módulo compuesto que corresponde a alguno de los nodos de la red NSFnet. Habiendo configurado cada parámetro para garantizar el correcto funcionamiento del prototipo, se destaca entre los procesos de RWA la utilización del método LAUC para asignar a cada ráfaga una longitud de onda disponible en el tiempo más cercano donde algún canal se encuentre libre. El algoritmo genético por su parte, se encarga del método de enrutamiento, siendo el responsable de generar una ruta con el mismo origen y destino de la ráfaga que se acaba de formar. Debido a que este proceso es dinámico, cada que un conjunto de paquetes se ensamblan formando una ráfaga, ésta realiza la petición de enrutamiento, llamando al método que genera el algoritmo y devolviendo una ruta apta. La información de la ruta viaja a través de la cabecera de la ráfaga, borrándose únicamente cuando llegue a su destino.

El otro prototipo de red analizado, presenta las mismas características del anterior, pero el algoritmo genético contiene un módulo de control cognitivo adicional cuyo propósito es generar una base de información correspondiente a los resultados de rutas anteriores obtenidas por el AG; de forma que si una ráfaga entrante presenta una petición de origen y destino que coincida con alguna o algunas rutas presentes en el vector cognitivo, éstas formarán parte del conjunto de rutas aleatorias del AG.

El objetivo de la creación de los dos modelos de red presentados anteriormente, es estimar la bondad de cada proceso creado, para el RWA en términos de probabilidad de bloqueo y tiempo de procesamiento. Para conseguir este propósito, ambas redes se analizan variando parámetros tanto del AG como de la red que permiten realizar una estimación del desempeño de las redes atendiendo a los resultados obtenidos.

Por lo tanto, realizando una revisión del capítulo 4, se presentan de forma ordenada las principales conclusiones extraídas de este análisis.

- Dentro de los principales factores que deben ser analizados en un entorno se simulación al crear una red OBS/WDM con control cognitivo basado en algoritmos genéticos, se encuentra la flexibilidad e interoperabilidad que soportan los módulos de programación y el consumo de memoria (asociado con el uso de punteros en el caso del lenguaje de programación C++).

- Con el fin de obtener las pruebas necesarias del desempeño de la red OBS/WDM con control cognitivo basado en algoritmos genéticos, es indispensable realizar un manejo adecuado de la memoria en OMNeT++, ya que al usar punteros, es necesario liberar cada cierto tiempo la memoria dinámica que se encuentra en uso; de igual manera, entre mejores especificaciones técnicas presenten los equipos usados para el análisis, será posible obtener un mayor número de resultados. Para efectos de este análisis se recomienda usar equipos con memoria RAM mayor o igual a 2GB.

- La estimación del tiempo offset adecuado durante el diseño de la red, es un factor importante para optimizar los recursos y evitar el aumento en la pérdida de ráfagas, independientemente de los métodos usados para tratar el enrutamiento y la asignación de longitud de onda.

- La velocidad de transmisión permite obtener un mejor resultado en términos de probabilidad de bloqueo y eficiencia para redes OBS/WDM, ya que el descarte de ráfagas debido al tiempo offset se ve disminuido por el aumento de la velocidad de los enlaces entre los nodos de la red. Este parámetro también es independiente de los métodos establecidos para el RWA.

- La inclusión en la función de aptitud de parámetros que indicen en la probabilidad de bloqueo de la red como el número de enlaces por nodo, las longitudes de onda disponibles, el número de saltos de una ruta y los nodos del camino de origen a destino, son un factor fundamental para garantizar la convergencia del algoritmo genético hacía buenas soluciones.

- El establecimiento correcto de los valores de aptitud umbrales durante el inicio e iteraciones del AG, son factores de diseño esenciales para obtener rutas óptimas; Al escoger un umbral bajo, las rutas resultantes pueden tomar valores que estarían dentro del rango de las soluciones, pero cuyas funciones de aptitud resultan ser mediocres; o bien, si el umbral es muy alto, el valor de aptitud establecido como criterio de parada podría ser inalcanzable.

- Cuando el AG con el método de control cognitivo usa menor tiempo de procesamiento, esto ocurre debido a que la información almacenada en el vector de rutas cognitivas puede contener soluciones aptas para la generación de las siguientes ráfagas, evitando que el algoritmo genético posiblemente tenga que realizar una o más iteraciones antes de generar la ruta apta. Por otra parte, aunque el método de control cognitivo emplea dos funciones dentro del algoritmo genético (llenado del vector de rutas y validación de la(s) rutas del vector en la etapa primera etapa del algoritmo genético), si los resultados almacenados resultan ser aptos como solución, el tiempo de procesamiento debido a las iteraciones que puede evitar el algoritmo es mayor que los métodos de la base de información cognitiva.

- El comportamiento de la probabilidad de bloqueo al variar el número de generaciones del algoritmo genético, es inversamente proporcional, independientemente del método heurístico examinado. No obstante, para todos los resultados analizados, la red que usa el método de control cognitivo muestra un mejor desempeño y se observa que a mayor número de generaciones, la diferencia entre las probabilidades de bloqueo de las dos redes se hace mayor. Bajo estos criterios, la red que usa la base de información cognitiva presentará mejores resultados en el 100% de los casos, manejando un margen de confiabilidad del 85%.

- Se deduce que para todos los casos analizados, el método de control cognitivo basado en algoritmos genéticos presenta en general un mejor desempeño en términos de probabilidad de bloqueo y tiempo de procesamiento, mostrándose como una técnica heurística adecuada y que mejora los procesos de enrutamiento y asignación de longitud de onda en redes OBS/WDM.

5.2 Trabajos Futuros.

Como se mencionó anteriormente, las redes analizadas fueron diseñadas para propósitos comparativos en términos de la probabilidad de bloqueo y el tiempo de procesamiento. Se modificaron los mismos parámetros en ambos diseños de red con el fin de examinar el desempeño de cada método heurístico.

Esta este trabajo de grado puede considerarse como un estudio inicial sobre el desempeño de los algoritmos genéticos con y sin métodos de control cognitivo en los procesos de RWA dinámico para redes OBS/WDM. El trabajo propone dos métodos, de los cuales, (y bajo los parámetros analizados) el algoritmo que usa el método de control cognitivo, resulta ser el mejor comparando sus valores de tiempo de procesamiento y probabilidad de bloqueo.

De esta manera, teniendo en cuenta la línea de investigación, se plantean los siguientes trabajos futuros.

Análisis de la variación del vector de rutas cognitivas en el desempeño de la red.

Hasta el momento, todo el análisis realizado sobre la red que usa métodos de control cognitivo se efectuó usando un vector de rutas aleatorias con capacidad para almacenar 4 caminos. La incidencia del aumento o disminución de dicho vector en el desempeño de la misma red, es un proceso que puede llevarse a cabo en un futuro, teniendo amplia disponibilidad de tiempo, debido a los largos procesos de simulación y pruebas necesarias para obtener resultados válidos.

Optimización de los resultados obtenidos durante el análisis del desempeño de redes ópticas OBS/WDM con y sin métodos de control cognitivo.

Aunque en general, los resultados arrojan un margen de confiabilidad de un 85%, éste puede ser optimizado al aumentar considerablemente el número de muestras para cada modelo de análisis. Se sugiere además, aumentar los valores de los parámetros de diseño como el número de generaciones, las longitudes de onda por fibra, las rutas aleatorias, los tiempos de simulación, entre otros, para optimizar aún más estos resultados.

Comparación del algoritmo genético con y sin métodos de control cognitivo con otro algoritmo para el RWA dinámico.

Si bien, la evaluación de los dos métodos heurísticos permitió el cumplimiento de los objetivos en el presente trabajo de grado, se considera importante analizar cómo se están comportando dichos métodos en comparación con otro algoritmo para el RWA dinámico. Los resultados arrojarán qué tan aptos son los AG con y sin métodos de control cognitivo para competir con otros procesos, evaluando los factores de probabilidad de bloqueo y tiempo de respuesta.

Comparación del desempeño de redes basadas en algoritmos genéticos con y sin métodos de control cognitivo y una red que presenta enrutamiento estático.

Durante los diseños de las redes actuales, se implementó sobre la misma, un modelo basado en RWA estático que utiliza el algoritmo del camino más corto para encaminar las ráfagas. Los resultados de este trabajo futuro, permitirán visualizar las principales diferencias entre las dos clases de enrutamiento y, así mismo, examinar la comparación entre la escogencia de rutas cuando se presentan lapsos de tráfico pesado dentro de la red, tiempo en el cual la función de aptitud debería arrojar mejores rutas.

Perfeccionamiento de las etapas del algoritmo genético.

Aunque el algoritmo genético presenta un buen comportamiento, dos de los métodos que generan mayor consumo de memoria debido al número de iteraciones realizadas son: las rutas generadas de manera aleatoria y el valor de la función de aptitud para cada una de ellas. Se sugiere por lo tanto que sean optimizados estos dos procedimientos para mejorar aún más el consumo de memoria del Algoritmo genético.

Referencias

[1] K. W. Kheng, “The Role and Impact of ICT on Economy Growth, “Faculty of Business Management and Professional Studies**,** Management and Science University, February 2011.

[2] N. Antoniades, G. Ellinas and L. Roudas, “WDM Systems and Networks, Modeling, Simulation, Design and Engineering,” New York, NY: Springer, 2012.

[3] H. F. Bermúdez, D. Jiménez, “Multiplexación por división de longitud de Onda - WDM Una nueva alternativa para comunicaciones ópticas,” Revista de Investigación de la Universidad de Quindío, vol. 22, Junio 2008, pp. 49-58

[4] C. Panda, S. Narayan Patro and P. Kumar Gantayat, “Link Reliability in WDM Optical Network,” Indian Journal of Computer Science and Engineering (IJCSE), vol. 3, no. 1, February 2012.

[5] D. F. Grosz, “Sistemas de comunicación por fibra óptica de alta capacidad,” Departamento de física y matemática, Instituto Tecnológico de Buenos Aires (ITBA), Buenos Aires, Argentina, 2004.

[6] W. Puche, et al., “Tecnologías de Transporte Óptico: hacía Optical Burst Swiching (OBS),” Investigaciones Aplicadas, vol. 1, no. 4, 2008, pp. 41-52, 2008.

[7] M. Düser, I. de Miguel, and P. Bayvel, “Timescale analysis for Wavelength-Routed Optical Burst-Switched (WR-OBS) networks,” in Proc. IEEE/OSA Optical Fiber Communication Conference Exhibit, London, UK: 2002, pp. 222-224.

[8] R. Millán, “IP sobre WDM,”Comunicaciones World,” no. 174, IDG Communications S.A, 2003.

[9] T. Battestilli and H. Perros, “An introduction to optical burst switching,” IEEE Communications Magazine, vol. 41, August 2003, pp. s10-s15.

[10] M. E. Guillamón, “Diseño de Protocolos sobre Redes Ópticas de Conmutación de Ráfagas” Tesis de grado, Universidad Politécnica de Cataluña, Cataluña, España, Junio de 2005.

[11] G. Puerto, et al., “Evolución de las redes de datos: Hacia una plataforma de comunicaciones completamente óptica,” Instituto de Telecomunicaciones y Aplicaciones Multimedia, Universidad Politécnica Valencia, Valencia, España, Mayo de 2008.

[12] M. Düser and P. Bayvel, “Analysis of a Dynamically Wavelength-Routed Optical Burst Switched Network Architecture,” Journal of Lightwave Technology, vol. 20, no. 4, April 2002.

[13] F. Guitart, “Algoritmos de Routing en Redes Totalmente Ópticas,” Redes de Banda Ancha, Enero de 2009.

[14] A. B. Rodríguez, F. Saavedra, “Optimización del Algoritmo Genético para la Solución Integral de Enrutamiento en Redes Fotónicas,” Información Tecnológica, vol. 21(3), 2010, pp. 125-133.

[15] R. S. Barpanda, et al., “A Genetic Algorithm Way of Solving RWA Problem in All Optical WDM Networks,” Communications in computer and information science, Springer – Verlag Berlin Heidelberg, vol. 125, 2011, pp. 137-142.

[16] S. Bandyopadhyay, “Dissemination of Information in Optical Networks,” Ontario, Canada: University of Windsor, Springer, 2008.

[17] J. C. Francoy, B. O. Tamarit, “Redes ópticas,” Escuela Técnica Superior de Ingeniería de Telecomunicación, Universidad Politécnica de Valencia, Editorial UPV, 2006.

[18] European Commission, Cognitive Heterogeneous Reconfigurable Optical Network, CHRON, 2010 [Online]. Available: www.ict-chron.eu [Accessed: 21 Feb. 2012].

[19] Q. Mahmoud, "Cognitive Networks, towards self aware networks," University of Guelph: Canada, Wiley, 2007.

[20] W. Wei, C. Wang, J. Yu, "Cognitive Optical Networks: Key Drivers, Enabling Techniques, and Adaptive Bandwidth Services," Communications Magazine, IEEE, vol. 50, 2012, pp. 106–113.

[21] S. Lázaro, "Evaluación de prestaciones de algoritmos RWA para redes todo ópticas WDM en tiempo real," Tesis de grado, Ingeniería de Telecomunicación, Universidad autónoma de Madrid, escuela politécnica superior, Madrid, España, Octubre de 2010.

[22] M. Mejía, D. López, "Aplicación Práctica de Algoritmos Genéticos al Diseño de Redes," Revista Iberoamericana de sistemas, cibernética e informática, Vol. 1, no. 2, 2004, pp. 13-18.

[23] Universidad del País Vasco, Algoritmos Genéticos, 2004, [Online]. Available: http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/temageneticos.pdf [Accessed: 18 Jan. 2012].

[24] K. Chiang, M. Gurusamy, Y. Liu and M. Hoang, "Quality of Service in Optical Burst Switched Networks" Singapur: National University of Singapur, 2007.

[25] A. Rodríguez "Redes Ópticas" Escuela Técnica Superior de Ingenieria-ICAI, Universidad Pontificia Comillas, 2012.

[26] Arie M.C.A. Koster, Sarah Ruepp, "Benchmarking RWA Strategies for Dynamically Controlled Optical Networks", in Proc. Of The 13th International Telecommunications Network Strategy and Planning Symposium , Budapest, pp. 1-14, September 2008.

[27] U. Bhanja, S. Mahapatra and R. Roy. "A Novel Solution to the Dynamic Routing and Wavelength Assignment Problem in Transparent Optical Networks," International journal of Computer Networks & Comunications (IJCNC), Vol.2, No.2, March 2010.

[28] D. Bisbal et al, "Dynamic Establishment of All-Optical Connections in Wavelength-Routed Optical Networks Using Genetic Algorithms", Next Generation Optical Network Design and Modelling, pp. 377-392, Kluwer Academic Publishers, 2003.

[29] R. Ramaswami, K. Sivarajan and G. Sasaki, "Optical Networks", A practical Perspective Third Edition, Elsevier, 30 Corporate Drive, Suite 400, Burlington, MA 01803, USA, 2010.

[30] I. Tomkos et al, "Next Generation Flexible and Cognitive Heterogeneous Optical Networks Supporting the Evolution to the Future Internet" The future internet lecture notes in computer science, Vol. 7281, pp. 225-236, 2012.

[31] OMNeT++ User Manual, OMNeT++ versión 4.3, OMNet, Año pub, [Online]. Disponible en: http://www.omnetpp.org/doc/omnetpp/manual/usman.html

[32] OMNeT++ documentation and tutorials, [Online]. Disponible en: http://www.omnetpp.org/documentation

[33] Adel A. M. Saleh, "All-Optical Networking – Evolution, Benefits, Challenges, and Future Vision", Proceedings of the IEEE, Vol. 100, no. 5, pp. 1105-1117, 2012.

[34] R.J. Barroso, "Diseño y reconfiguración de redes ópticas con encaminamiento por longitud de onda mediante Algoritmos Genéticos", Tesis Doctoral, Universidad de Valladolid, España, marzo 2008.

[35] H. F. Bermúdez, et al, “Multiplexación por división de longitud de Onda - WDM Una nueva alternativa para comunicaciones ópticas,” Junio 2008.

[36] Héctor Fabio Bermúdez Orozco, Wilmer Diego Jiménez Trujillo, “Multiplexación por división de longitud de Onda - WDM Una nueva alternativa para comunicaciones ópticas”, Marzo 19 de 2008.

[37] WDM. Multiplexación por división de longitud, Venezuela: Universidad de Oriente, 2009, [Online]. Disponible en:: http://comunicacionudo.jimdo.com/

[38] M. Maier, Optical Switching Netwoks, Cambridge University, Hardcover 2008**.**

[39] M. Klinkowski “Offset Time-Emulated Architecture for Optical Burst Switching – Modelling and Performance Evaluation”[Online] Ph.D. thesis, Universitat Politècnica De Catalunya, Departament d' Arquitectura de Computadors, Barcelona, España , November, 2007. [Online]. Disponible en: http://www.tesisenred.net/bitstream/handle/10803/6000/01_miroslawKlinkowski.pdf?sequence=1

[40] Xiang Yu, Jikai Li, Xiaojun Cao, Yang Chen, and Chunming Qiao, **“**Traffic Statistics and Performance Evaluation In Optical Burst Switched Networks” Journal of Lightwave Technology, vol. 22, no. 12, december 2004.

[41] Ahmad Rostami, “Traffic Shaping for Contention Control in OBS Networks”, para Obtención de título de Doctor en Ingeniería, departamento de electrónica e informática, Universidad de Berlín, Berlín, 2010.

[42] Germán Jaramillo Andrade, “Estudio de la tecnología de conmutación óptica por ráfagas – OBS y análisis de migración de rede ópticas pasivas a esta tecnología”, escuela politécnica del ejército, Sangolquí-Ecuador, 2011.

[43] M. Itoiz and J. Algueta, “Optical Burst Switching: Conmutación óptica de ráfagas” [Online] Disponible en: http://es.scribd.com/doc/43061104/Conmutacion-Optica-Por-Rafagas-OBS,2005.

[44] B. Farrell, Y. Huang, M. Iwen, T. Wang, L. Zhang and J Zheng, “Wavelength Assignment in Optical Network Design” Mathematics-in-Industry Case Studies Journal, Vol. 1, pp. 49-65, 2009.

[45] A. Barradas “Quality of Service in Optical Burst Switching Networks,” Submitted for the Degree of Doctor of Philosophy, Faculdade de Ciências e Tecnologia, Departamento de Engenharia Electrónica e Informática, Universidad de Algarve, Faro, Portugal, Novembro, 2009.

[46] E. Kozlovski, M. Düser, I. de Miguel and P. Bayvel, “Analysis of Burst Scheduling for Dynamic Wavelength Assignment in Optical Burst-Switched Networks,” University College London, Departament of Electrical and Electronic Engineering Torrington Place, London, 2001.

[47] F. Espina, J. Armendariz, M. Izal, D. Morató, E. Magaña, “Arquitectura y diseño de un modelo de red OBS para simulación”, Universidad Pública de Navarra, Campus de Arrosadía s/n, E-31006 Pamplona, España, 2009.

[48] Ó. Gonzáles “Rendimiento de TCP y Cálculo de Rutas en Redes de Conmutación Óptica de Ráfagas” Ph.D. tesis, Universidad de Valladolid, Valladolid, España, 2012.

[49] J. Turner “Terabit Burst Switching” Department of Computer Science, Washington University, St. Louis, July, 1998.

[50] D. Gourkar “Study of Routing and Wavelength Assignment problem and Performance Analysis of Genetic Algorithm for All-Optical Networks” Bachelor of Technology in Computer Science and Engineering thesis, National Institute of Technology, Rourkela, India, 2010.

[51] X. Yu, M. Gen, “Introduction to Evolutionary Algorithms”, New York, Springer, 2010.

[52] S. Yussof, “Performance analysis of genetic algorithm (GA)-based multi contrained path routing algorithm”, International Journal of the Physical Sciences Vol. 6(33), pp. 7524 - 7539, December, 2011.

[53] R. J. Duran et al, “Minimization of End-to-End Delay in Reconfigurable WDM Networks by means of Genetic Algorithms”, Proceedings of the 10th Conference on Optical Network Design and Modelling (ONDM'06), Copenhagen (Denmark), May 2006.

[54] E. Pastor, et al., “A new heuristics/GA-based algorithm for the management of the s-DRWA in IP/WDM networks”, Computer Science, managing next generation networks and services, Vol. 4773, 2007, pp. 265-275.

[55] M. Batista, M. Belén “Optimización metaheurística para la planificación de redes WDM”, Departamento de Estadística, Investigación operativa y Computación, Universidad de la Laguna, 2003, pp.30-33.

[56] C. Rodríguez, “Algoritmos heurísticos y metaheurísticos para el problema de localización de regeneradores” Proyecto fin de carrera, Ingeniería Informática superior, Universidad Rey Juan Carlos, Madrid, España, 2010.

[57] J. Mena “Las clases P y NP: problemas tratables e intratables” en Complejidad y Optimización: NP Completo, Escuela de Ingeniería de Sistemas y Computación, Universidad del Valle, Agosto, 2006.

[58] Algoritmos genéticos, [Online]. Disponible en: http://sedici.unlp.edu.ar/bitstream/handle/10915/4060/IVAlgoritmosgen%C3%A9ticos.pdf?sequence=8

[59] Carlos A. Marmelada, “Darwin y la teoría de la evolución”, [Online], 2009. Disponible en: http://www.unav.es/cryf/darwin.html

[60] L. C. Tovar, M. R. Coronell, Y. D. Meisel, "Optimización Multiobjetivo en Redes Ópticas con Transmisión Multicast utilizando Algoritmos Evolutivos y Lógica Difusa, "Ingeniería & Desarrollo, Universidad del Norte, Colombia, Barranquilla, vol. 21, 2007, pp. 39-55.

[61] S. Liang, A. N. Zincir-Heywood, M.I. Heywood, **“**Intelligent Packets for Dynamic Network Routing Using Distributed Genetic Algorithm”, [Online]. Disponible en: https://web.cs.dal.ca/~mheywood/X-files/Publications/LiangGECCO-2k2.pdf.

[62] M. Melanie, “An Introduction to Genetic Algorithms”, Cambridge, Massachusetts - London, England, Fifth printing, 1999

[63] T.A. El-Mihoub, et al.,”Hybrid Genetic Algorithms: A Review”, Engineering Letters, 13:2, Advance online publication, August 2006, pp. 124-137.

[64] T. Weise, “Global Optimization Algorithms - Theory and Application” [Online], 2009. Disponible en: http://www.it-weise.de/projects/book.pdf.

[65] “Computación Evolutiva Algoritmos Genéticos” en Inteligencia Computacional, Departamento de Informática, Ingeniería Informática, [Online], Disponible en: http://infofich.unl.edu.ar/upload/66648614acb4d750ce204b83a33bcdbfcd1a361e.pdf.

[66] E.P.Ephzibah, “Cost Effective Approach on Feature Selection using Genetic Algorithms and LS-SVM Classifier”, IJCA Special Issue on Evolutionary Computation for Optimization Techniques ECOT, Vol. 1, 2010, pp. 16–20.

[67] J. H. Holland, Adaptation in natural and artificial systems, University of Michigan Press, 1975. Reimpreso por MIT Press en 1992.

[68] X. H. Lin, et al, “A genetic algorithm based approach to route selection and capacity flow assignment”, Computer Communications, Department of Electrical and Electronic Engineering, Department of Electrical and Electronic Engineering, The University of Hong Kong, China, Hong Kong, volume 26, 2003, pp. 961–974.

[69] John R. Koza, “Genetic Programming On the Programming of Computers by Means of Natural Selection” A Bradford Book The MIT Press, Cambridge, Massachusetts - London, England, 1998, pp. 62-71.

[70] X.-S. Yang, “A New Metaheuristic Bat-Inspired Algorithm, in: Nature Inspired Cooperative Strategies for Optimization”, Studies in Computational Intelligence, Springer Berlin, volume 284, Springer, 2010, pp. 65-74.

[71] F. Gonzales et al, “Lightpath routing and wavelength assignment by means of ant colony optimization”, University of Valladolid, Valladolid, Spain, February, 2003.

[72] Luis R. “La capa de enlace de Datos”, [Online], 2008. Disponible en: http://ipref.wordpress.com/2008/09/21/la-capa-de-enlace-de-datos/

[73] G.S. Zervas and D. Simeonidou, “Cognitive Optical Networks: Need, Requirements and Architecture,” High-Performance Networks Group, School of CSEE, University of Essex, Colchester, United Kingdom, 2010.

[74] L.W. Chen, Eytan Modiano “Dynamic routing and wavelength assignment with optical bypass using ring embedding”, Optical Switching and Networking vol. 1, 2005, pp. 35–49.

[75] F. Loukdache, et al, “Modeling an OBS Network in OMNeT++ and the Impact of Data Channels in such Network”, Proceedings of the World Congress on Engineering 2012, Vol. 2, London, U.K, July 4 - 6, 2012.

[76] L. Tomkos, et al, “Next Generation Flexible and Cognitive Heterogeneous Optical Networks - Supporting the Evolution to the Future Internet”, Future Internet Assembly, Springer, pp. 225-236, 2012.

[77] J. Gond and A. Goel, “Performance Evaluation of Wavelength Routed Optical Network with Wavelength Conversion”, Journal of telecommunications, volume 2, issue 1, April, 2010.

[78] L. N. Binh, L.H. Binh and V. T. Tu, “Routing and Wavelength Assignment and Survivability of Optical Channels in Ultra-high Speed IP over DWDM Networks Under Constraints of Residual Dispersion and Nonlinear Effects”, IJCSNS International Journal of Computer Science and Network Security, Vol.9 No.2, February 2009.

[79] V. T. Le et al., “A New Fitness Function for GA-based Dynamic RWA Algorithms in Optical WDM Networks,” in Proc. of Networks, jointly held with the IEEE 7th Malaysia International Conference on Communication, 13th IEEE International Conference on, Malaysia, vol. 2, 2005.

[80] Y. S. Kavian, “Robust DWDM Optical Networks Design using Genetic Algorithms,” in Proc. of First International Conference on Communications Engineering, University of Sistan and Baluchestan, Zahedan, Iran, 2010, pp. 143-146.

[81] R. Nakkeeran, et al., “Genetic algorithm based approach for routing and wavelength assignment,” Department of Electronics and Communications Engineering, Pondicherry, Engineering College Pondicherry, India, 2004.

[82] D. Bisbal, et al., “Dynamic Routing and Wavelength Assignment in Optical Networks by Means of Genetic Algorithms,” Photonic Network Communications, vol. 7, no.1, 2004, pp. 43-58.

[83] V. Trong, et al, “Dynamic RWA Based on the Combination of Mobile Agents Technique and Genetic Algorithms in WDM Networks with Sparse Wavelength Conversion”, IEICE Trans. INF. & SYST., Vol.e88–d, No.9 September, 2005.

[84] I. de Miguel, et al “Nature-inspired Routing and Wavelength Assignment Algorithms for Optical Circuits-Switched Polymorphic Networks”, Fiber and Integrated Optics, Vol.23, No.2-3, March-June 2004, pp.157-170.

[85] E. Kozlovski, “Survivability of Wavelength-Routed Optical Burst-Switched Networks with Guaranteed IP Services,” Department of Electronic and Electrical Engineering, University College London, Torrington Place, London, United Kingdom, 2003.

[86] A. L. Barradas and M. C. R. Medeiros “An OMNeT++ Model for the Evaluation of OBS Routing Strategies”, University of Algarve, Faro, Portugal, March, 2008.

[87] F. Espina et al, “OBS network model for OMNeT++: A performance evaluation”, Public University of Navarre, Pamplona, Spain, March, 2010.

[88] R. J. Durán ,”Advantages of Using Cognition when Solving Impairment- Aware Virtual Topology Design Problems”, University of Valladolid, Athens Information Technology, Abril, 2011.

[89] P. Katsaros and C. Lazos, “Steady-state Simulation of Queuing Processes in Parallel Time Streams: Problems and Potentialities”, Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece, 2001.

[90] Universidad Politécnica de Valencia, Introducción al simulador de redes NS-2, [Online]. Disponible en: http://riunet.upv.es/bitstream/handle/10251/12735/Art%C3%ADculo%20docente%20NS2.pdf?sequence=1

[91] The Cisco Network Simulator & Router Simulator, [Online]. Disponible en: http://www.boson.com/netsim-cisco-network-simulator

[92] Dell Soft Technologies, NetSIM Software, [Online]. Disponible en: http://dellsoft.in/index.php?option=com_content&view=article&id=50&Itemid=6&product=true

[93] Departament d'Enginyeria Telemàtica, OPNET: Manual de Usuario, [Online]. Disponible en: http://www.opnet.com/university_program/teaching_with_opnet/textbooks_and_materials/materials/OPNET_Modeler_Manual.pdf

[94] NED, Network Description, OMNeT++, [Online]. Disponible en: http://www.ewh.ieee.org/soc/es/Nov1999/18/ned.htm.

[95] Capítulo 4: Simulación en OMNet++, [Online]. Disponible en: http://catarina.udlap.mx/u_dl_a/tales/documentos/lem/deschamps_e_me/capitulo4.pdf

[96] J. Gómez, Diseño e implementación de herramientas docentes de conmutación en el entorno Omnet, [Online]. Disponible en: http://repositorio.bib.upct.es/dspace/bitstream/10317/189/1/pfc1127.pdf

[97] I de Miguel, et al, “Genetic Algorithm for Joint Routing and Dimensioning of Dynamic WDM Networks”, Journal of Optical Communications and Networking, Vol. 1, Issue 7, pp. 608-621, 2009.

[98] B. Lo, et al, “Routing and Wavelength Assignment vs. Wavelength Converter Placement in All-Optical Networks”, University of Science and Technology Kazem Sohraby, University of Arkansas, IEEE Optical Communications, August 2003.

[99] U. Bhanja, S. Mahapatra, R. Roy, “A novel solution to the dynamic routing and wavelength assignment problem in transparent optical networks”, International journal of Computer Networks and Communications, Vol.2, No2, March 2010.

[100] Temas genéticos, Universidad del País Vasco, [Online]. Disponible en: http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/temageneticos.pdf

[101] Y. Kavian, et al. “Genetic Algorithm for Designing DWDM Optical Networks under Demand Uncertainty”, Electrical Engineering Department, Iran University of Science and Technology, Tehran, 16844, Tehran, Iran, 2009.

[102] H. Cheng and S. Yang, “Multi-population Genetic Algorithms with Immigrants Scheme for Dynamic Shortest Path Routing Problems in Mobile Ad Hoc Networks”, Department of Computer Science, University of Leicester, University Road, Leicester LE1 7RH, United Kingdom, 2010.

[103] Historia resumida de Internet, IES Severo Ochoa 2010/11 Informática 4ºESO Alumna: Marta Oliva Grupo: F, [Online], Disponible en: http://infoolivaf.blogspot.com/2010/12/historia-resumida-de-internet.html.

[104] A. K. Garg, “A Novel Hybrid Approach for Efficient Network Utilization of OBS” Department of Electronics and Communication Engineering, Mullana (Ambala), India, International Journal of Software Engineering and Its Applications, Vol. 6, No. 1, January, 2012.

[105] A. Goel , et al. “Performance Analysis of Continuous Optical Burst Switching Networks”, Department of Electronics & Communication MANIT, Deemed University, Bhopal-462051, International Journal of Engineering, India, Vol. 3, 2010.

[106] K. Chaing, M. Gurusamy, Y Liu, M.Hong, “Relative QoS differentiation”, Quality of service in Optical Burst Switched Networks, Biswanath Mukherjee, University of California, Davis, Springer Science Business Media, LLC, pp. 73-87, 2001.

[107] S. Abeysundara et al, “A Genetic Algorithm Approach to Solve the Shortest Path Problem for Road Maps”, Proceedings of the International Conference on Information and Automation, December 15-18, 2005, Colombo, Sri Lanka.

[108] I. C. Graña, “Simulación y comparativa de mecanismos de conmutación en redes ópticas”, Proyecto de fin de carrera, Escuela Técnica Superior de Ingenieros de Telecomunicación, 2005.

[109] B. Álvarez, “Estimación por máxima verosimilitud y conceptos de teoría asintótica”, [Online] Disponible en: http://webs.uvigo.es/alvarez/teaching_archivos/ectria2_0708/tema_introduction.pdf, 2007.

[110] G. E. Box, et al, “Fundamentos (Probabilidad, parámetros y estadísticos), Comparación de dos tratamientos: distribuciones de referencia, pruebas e intervalos de confianza” Estadísticas para Investigadores, Reverté S.A., pp. 17-94, 2008.

[111] V. A. Nolberto, et al, “Estadística Inferencial”, Estadística inferencial Aplicada, Elena Soto Loayza pp. 21-39, 2008.

ANEXOS.

ANEXO A. ALGORITMOS.

A1. Algoritmo para el manejo y envío de paquetes.

Se generan paquetes por intervalos de tiempo y le son asignados atributos que definirán su gestión a través de la red. El algoritmo abstrae la cadena numérica que representa las direcciones de los nodos en la red mediante la función **"intuniform",** encargada de escoger un número aleatorio entre cero y el tamaño de la cadena cuyo resultado será la posición del vector **"destAddresses"** que representa la dirección de destino asignada al paquete entrante, la prioridad se establece mediante la misma función escogiendo un número aleatorio entre cero y dos.

Algoritmo 1. Algoritmo para el manejo y envío de paquetes.

INPUTS: Cadenas de números que representan los nodos de la red y mensajes.

OUTPUT: Paquetes de datos con un conjunto de parámetros determinados (Origen, Destino, saltos y prioridad)

.

BEGIN

```
if (msg == generatePacket)
   {
    int destAddress = destAddresses[intuniform(0, destAddresses.size()-1)];
    char pkname[40];
    Packet *pkt = new Packet(pkname);

    pkt->setByteLength(packetLengthBytes->longValue());
    pkt->setSrcAddr(myAddress);
    pkt->setDestAddr(destAddress);

    int type = intuniform(0,2);
    pkt->setPriority(type);

    send(pkt,"out");
    scheduleAt(simTime() + sendIATime->doubleValue(), generatePacket);
   }
```

END

A2. Algoritmo para la obtención de reglas alusivas a la dirección y prioridad de los paquetes.

Este algoritmo se encarga de obtener el archivo de texto plano correspondiente a las reglas y comparar cada línea usando la clase **"Classifier_Rules"**, que verifica si la dirección y prioridad del paquete son compatibles con las reglas dentro del archivo.

Algoritmo 2. Algoritmo para la obtención de reglas alusivas a la dirección y prioridad de los paquetes.

INPUTS: Mensajes y archivo de texto **"rules.dat"**.

BEGIN

```
  if (numOuts != 0){

     rules = (Classifier_Rules*)calloc(numOuts,sizeof(Classifier_Rules));
     char *line = (char*)calloc(1500,sizeof(char));
     int i=0;
     const char *rulesFile = par("rules");
if (strlen(rulesFile) == 0){
     opp_error("El archivo de reglas no ha sido definido");
 }
  FILE *ruleFile = fopen(rulesFile,"r");
   if(ruleFile != NULL)
    {
      while(fgets(line,1500,ruleFile) != NULL)
         {
         if(strcmp(line,"\n") != 0 && line[0] != '#')
          {
            rules[i] = Classifier_Rules(line);
            i++;
          }
         }
    }
     else
        {
          opp_error("No se puede abrir a archivo de reglas");
        }
   fclose(ruleFile);
   if(!(i == numOuts))
   {
    printf("(Classifier_Rules) AVISO: El despachador de reglas no coincide con las colas de los
    módulos.\n");
    free(line);
  }
}
END
```

A3. Algoritmo para la comparación de paquetes con reglas.

Este algoritmo se encarga de verificar que las reglas y las características del paquete coinciden, si esto es así, se retorna un booleano en verdadero para que el paquete sea mandado a la primera compuerta donde coincide la regla. Se debe tener en cuenta que el número de salidas módulo y las reglas deben ser iguales.

Algoritmo 3. Algoritmo para la comparación de paquetes con las reglas.

INPUTS: Reglas y mensajes.

OUTPUT: Booleano.

BEGIN

```
bool Classifier_Rules::match(cMessage *msg)
{
   Packet *pkt = check_and_cast<Packet*>(msg);

   if(isSet[0])
    {
     if(!(pkt->getDestAddr() == destAddr)) return false;
     }
   if(isSet[1])
    {
     if(!(pkt->getPriority() == priority)) return false;
     }
}
```

END

A4. Algoritmo comparativo entre reglas y características del paquete.

Encargado de realizar una comparación entre cadenas con el fin de comprobar que éstas se encuentran en el archivo de texto **"rules.dat"**. Si la comparación resulta exitosa, se abstrae el valor de la regla en una variable y se retorna verdadero, de otra forma, el algoritmo devuelve un error en tiempo de compilación.

Algoritmo 4. Algoritmo comparativo entre reglas y características del paquete.

INPUTS: Dirección de destino y prioridad del paquete.

OUTPUT: Un atributo de tipo booleano que indica si la regla coincide o no.

BEGIN

```
while(token != NULL)
{
  if(strcmp(token,"destAddr") == 0)
   {
      token = strtok(NULL," \n");
      if(token != NULL)
       {
        destAddr = atoi(token);
        isSet[0] = true;
       }
       else
         {
           throw runtime_error("No es posible analizar sintácticamente el valor de la dirección de destino en
          el archivo de reglas");
```

```
          }
      }
 else if(strcmp(token,"priority") == 0)
     {
       token = strtok(NULL," \n");
       if(token != NULL)
         {
          priority = atoi(token);
          isSet[1] = true;
          }
         else
          {
            throw runtime_error("No es posible analizar sintácticamente el valor de la dirección de prioridad en
            el archivo de reglas");
          }
      }
      token = strtok(NULL," ");
   }
}
```
END

A5. Algoritmo que establece el criterio de ensamblaje.

Recibe paquetes y los almacena en una cola mediante tres criterios: número máximo de paquetes, tamaño en Bytes o tiempo de ensamblaje. Si se cumple alguno de los criterios mencionados, anteriormente los paquetes son ensamblados con el método **"assemblyBurst".**

Algoritmo 5. Algoritmo que establece el criterio de ensamblaje.

INPUTS: Paquetes.

BEGIN

```
 if(burst.empty())
   {
    scheduleAt(simTime() + maxTime, maxTime_msg );
    firstPacket_t = simTime();
   }
else if(desborde && !addLastPacket)
   {
     if(burst.empty())opp_error("No es posible ensamblar una ráfaga con una cola vacía");
     assemblyBurst();
     if(maxTime_msg->isScheduled())cancelEvent(maxTime_msg);
     scheduleAt(simTime() + maxTime, maxTime_msg);
     firstPacket_t = simTime();
     desborde = false;
     if(((burstBits + pqt->getBitLength()+ tamHeaderPacketBits)> maxSizeBits)) desborde=true;
    }
timeAssembled.collect(simTime()-firstPacket_t);
burst.insert(pqt);
burstBits += pqt->getBitLength() + tamHeaderPacketBits;
numPacketsInBurst++;
```

```
if(desborde && !addLastPacket)
opp_error("La sobrecarga se generó insertando el primer mensaje y los requerimientos no permiten
sobrecagra (overflowLastPacket es falso)");
if((desborde || numPacketsInBurst == numPackets) || burstBits ==maxSizeBits)
  {
   assemblyBurst();
   if(maxTime_msg->isScheduled()) cancelEvent(maxTime_msg);
  }
}
END
```

A6. Algoritmo para el desencapsulamiento de ráfagas.

Es el encargado de recibir el inicio de la ráfaga, almacenándolo en una lista, una vez recibido el **"endBCP"** se busca en la lista el identificador que le corresponde, abstrayendo la ráfaga de la lista y posteriormente desencapsulándola.

Algoritmo 6. Algoritmo para el des-encapsulamiento de ráfagas.

INPUTS: Mensaje **"iniBCP**" y **"endBCP"**.

BEGIN

```
while(!iter_list.end())
{
    item = (ScheduleBurst*)iter_list++;
   if(item->getIdBurst() == bId)
  {
    burst = check_and_cast <Burst *> (item->decapsulate());
    listSize -= burst->getBitLength();
   delete burstList.remove(item);
   delete burst;
   numElems--;
   numElemsVector.record(numElems);
   return;
   }
 }
END
```

A7. Algoritmo para programar el envío del BCP.

Corrobora que la ráfaga haya llegado desde módulo **"Burstifier**", posteriormente se encarga de encontrar la longitud de onda con el horizonte más cercano (Menor tiempo en el que una longitud de onda se encuentra disponible) mediante la función **"findNearestHorizon"**.

Los valores del BCP son asignados mediante una clase mensaje denominada **“info.msg”,** que almacena la información correspondiente a la ráfaga y auto-programa un mensaje para establecer el envío del BCP.

Se guardan las ráfagas en una lista que registra el tiempo en el que cada una debe enviarse hacia su salida, sin embargo la lista es limitada, por lo tanto, las ráfagas pueden ser descargadas si la cola se encuentra llena.

Algoritmo 7. Algoritmo para programar el envío del BCP.

INPUTS: Ráfagas de datos.

BEGIN

```
if(!(msg->isSelfMessage()))
   {
     Burst *rfg = check_and_cast<Burst*>(msg);
     burstTam.record(rfg->getBitLength());
     burstRecv++;
     int wl = 0;
     int pos=0;
     wl=findNearestHorizon();
     char pkname[40];
     rfg->setSendId(getId());
     BurstSenderInfo *myInfo = new BurstSenderInfo();
     myInfo->setIdBurstifier(rfg->getIdBurstifier());
     myInfo->setNumSeq(rfg->getSecNumor de ráfaga());
     myInfo->setAssignedLambda(wl);
     dirOri = source->myAddress;
     if(destino!=dirOri)
         {

          myInfo->setSaltos(saltos);

          myInfo->setRutaODArraySize(tamano);

          myInfo->setRutasSender(rutaOrigenDest);
          std::vector<int>hola =myInfo->getRutasSender();
         }
       BurstifierInfo *lInfo = (BurstifierInfo*) rfg->removeControlInfo();
       myInfo->setLabel(lInfo->getLabel());
       delete lInfo;
       cMessage *ctrMsg= new cMessage("Sched");
       if(horizon[wl] - rfg->getMaxOffset() >= simTime())
          {
            pos = scheduleBursts.insertBurst(rfg, horizon[wl]);
            if(pos == -1)
              {
               delete msg;
               delete ctrMsg;
               burstDroppedByQueue++;
               return;
              }
```

```
            myInfo->setBurstId(pos);
            ctrMsg->setControlInfo(myInfo);
            ctrMsg->setKind(OBS_PROGRAMAR_BCP;
            scheduleAt(horizon[wl] - rfg->getMaxOffset(), ctrMsg);
            horizon[wl] = horizon[wl] + (rfg->getBitLength()/dataRate) + guardTime;
            horizonVec[wl]->record(horizon[wl]);
          }
       else
          {
            pos = scheduleBursts.insertBurst(rfg, simTime() + rfg->getMaxOffset());
            char name[40];
            sprintf(name, "--pos %d", pos);
            if(pos == -1)
             {
              delete msg;
              delete ctrMsg;
              burstDroppedByQueue++;
              return;
             }
            myInfo->setBurstId(pos);
            ctrMsg->setControlInfo(myInfo);
            ctrMsg->setKind(OBS_PROGRAMAR_BCP);
            scheduleAt(simTime(), ctrMsg);
            horizon[wl] = simTime() + rfg->getMaxOffset() + (rfg->getBitLength()/dataRate) + guardTime;
            horizonVec[wl]->record(horizon[wl]);
          }
  }
 END
```

A8. Algoritmo para encontrar el horizonte más cercano.

Este algoritmo devuelve la posición que contiene el tiempo en el que un canal se encuentra disponible.

Algoritmo 8. Algoritmo para encontrar horizonte más cercano

OUTPUT: Horizonte.

BEGIN

```
int Sender::findNearestHorizon()
  {
   int min = 0;
   int i;
   for(i=0;i<numLambdas;i++)
     {
     if(horizon[min] > horizon[i])
     min = i;
     }
   return min;
  }
```

END

A9. Algoritmo para la búsqueda de nodos adyacentes al vector de rutas.

Se encarga de descubrir los nodos adyacentes al nodo actual perteneciente a la ruta aleatoria, formando así su trayecto de origen a destino, para esto usa la clase **"cTopology"** y el puntero **"*adyacents".**

Algoritmo 9. Algoritmo para la búsqueda de nodos adyacentes al vector de rutas.

INPUTS: Dirección y mapa de nodos anteriores.

OUTPUT: Vector de ruta.

BEGIN

```
int w = (numberPorts-1)*(lambdasC+1);
for( int y=0; y<w; y=y+(lambdasC+1)
  {
   cTopology::Node *adjacents = thisNode->getLinkOut(y)->getRemoteNode();
   dirAdjacentNode = adjacents->getModule()->getParentModule()->getSubmodule("edgeHost")-
   >getSubmodule("Host")->getSubmodule("source")->getAncestorPar("address");
   if(ant.size()>0)
       {
        if(ant.find(dirAdjacentNode)==ant.end())
          {

           addressVector[dirAdjacentNode]=pos;

           pos++;

          }

       }

   else

      {

       addressVector[dirAdjacentNode]=pos;

       pos++;
      }
   }
  return addressVector;
```

END

A10. Algoritmo de rutas aleatorias.

Este algoritmo se encarga de generar un número determinado de rutas aleatorias entre origen y destino para la ráfaga. El proceso de generación de ruta implica realizar un llamado a la función

“findAdjacentNodes” con el fin de obtener rutas válidas; el número de rutas aleatorias puede variar de acuerdo al criterio de diseño.

Algoritmo 10. Algoritmo de rutas aleatorias.

INPUTS: Dirección de destino.

OUTPUT: Vector de ruta.

BEGIN

```
if (sourceDirection != inLabel)
  {
   while(contRutas<numRutas)
     {
      bool i=false;
      int adjacent;
      std::map<int,int>antNode;
      int actNode=0;
      std::vector<int>v;
      std::map<int,int>adj;
      actNode=sourceDirection;
      v.push_back(sourceDirection);
        do
          {
           adj = findAdjacentNodes(actNode, antNode
           std::vector<int>vAdj;
             for(std::map<int,int>::iterator it = adj.begin(); it != adj.end(); ++it)
                {
                 vAdj.push_back(it->first);
                }
           if(vAdj.size()>0)
               {
                adjacent = vAdj[intuniform(0, adj.size()-1)];
                v.push_back(adjacent);
                if(adjacent==inLabel)
                  {
                   if(rutas.find(v)==rutas.end())
                        {
                         rutas[v] = contRutas;
                         contRutas++;
                         }
                   i=true;
                  }
                  else
                       {
                        antNode[actNode]=1;
                        actNode=adjacent;
                       }
                 }
                else
                     {
                      i=true;
                     }
```

```
            while(i==false);
             }
           }
      return rutas;
END
```

A11. Algoritmo para el establecimiento y aplicación de la función de aptitud.

Este algoritmo permite aplicar la función de aptitud creada a las rutas aleatorias, los parámetros necesarios para dicha aplicación son los siguientes:

Número de nodos y enlaces por nodo en la ruta: se emplea la función **"cTopology"** para descubrir el grafo de la red, abstrayendo los módulos con sus respectivos parámetros mediante le puntero "***thisNode"**, que realiza un recorrido por cada uno de los elementos pertenecientes a la ruta

Longitudes de onda libres: se emplea el algoritmo LAUC (Horizonte) que permite la extracción del tiempo en el que fue desocupado por última vez un canal y que es capturado mediante un puntero llamado **"*longitudes"**.

Finalmente se realizan las funciones aritméticas correspondientes.

Algoritmo 11. Algoritmo para el establecimiento y aplicación de la función de aptitud.

INPUTS: Rutas aleatorias.

OUTPUT: Rutas aleatorias con su correspondiente función de aptitud.

BEGIN

```
std::vector<int>:: iterator ite;
if(nodos!=0)
  {
   for( ite=infoFunction.begin()+1; ite!= infoFunction.end()-1; ite++)
      {
       cTopology *topology = new cTopology("topo");
       topology->extractByParameter("variableCoreNode");
       cTopology::Node *thisNode = topology->getNodeFor(getParentModule());
       thisNode=topology->getNode(*ite-1);
       numberPorts = thisNode->getModule()->getParentModule()->par("numPorts");
       nuevoPorts=numberPorts-1;
       sumaPorts = sumaPorts + nuevoPorts;
      }
   }
for( itedos=infoFunction.begin(); itedos!= infoFunction.end()-1; itedos++)
  {
  cTopology *topology = new cTopology("topo");
  topology->extractByParameter("variableCoreNode");
  cTopology::Node *thisNode = topology->getNodeFor(getParentModule());
  thisNode=topology->getNode(*itedos-1);
  cModule *parent3 = thisNode->getModule()->getSubmodule("coreControlUnit")
 >getSubmodule("GatesHorizon");
```

```
        numHorizon = check_and_cast<CoreOutputHorizon*>(parent3);
        numberPorts = thisNode->getModule()->getParentModule()->par("numPorts");
        lambdasC = getParentModule()->getParentModule()->par("lambdasCore");
        lambdasE = getParentModule()->getParentModule()->par("lambdasEdge");
        int rango1 = lambdasE +1;
        int rango2 = rango1 + lambdasC;
        int rango3 = rango1 + 2*lambdasC + 1;
        int rango4 = rango1 + 3*lambdasC + 2;
        int rango5 = rango1 + 4*lambdasC + 3;
        int outPort;
        int w = (numberPorts-1)*(lambdasC+1);
        for( int y=0; y<w; y=y+(lambdasC+1)

            {
             cTopology::Node *adjacents = thisNode->getLinkOut(y)->getRemoteNode();
               dirAdjacentNode = adjacents->getModule()->getParentModule()->getSubmodule("edgeHost")-
            >getSubmodule("Host")->getSubmodule("source")->getAncestorPar("address");
             if(position != infoFunction.size())
                 {
                  if( dirAdjacentNode == infoFunction.at(position) )
                     {
                      cGate *gate = thisNode->getLinkOut(y)->getLocalGate();
                      numberGate=gate->getIndex();
                      if(numberGate >=rango1 && numberGate<=rango2){outPort=1;}
                      if(numberGate >=rango2 && numberGate<=rango3){outPort=2;}
                      if(numberGate >=rango3 && numberGate<=rango4){outPort=3;}
                      if(numberGate >=rango4 && numberGate<=rango5){outPort=4;}
                      position++;
                      }
                  }
              }
         numHorizon=(CoreOutputHorizon*)thisNode->getModule()->getSubmodule("coreControlUnit")-
        >getSubmodule("GatesHorizon");
         simtime_t **longitudes = numHorizon->horizon;
         for(int j=0;j<lambdasC;j++)
            {
             if(simTime()>longitudes[outPort][j])
               {
                wLibres=wLibres+1;
               }
            }
        }
costo1 = saltos + alfa*(wTotal - wLibres);
if(totalRDestino ==0 && totalRPasantes ==0)//
   {
    costo2=1;
     }
if(totalRDestino == 0)
   {
    costo2=1;
    }
if(totalRPasantes == 0)
   {
   costo2=1;
    }
```

```
 else
    {
     costo2 = totalRPasantes/totalRDestino;
     }
aptitud1= 1/costo1;
aptitud2= 1/costo2;
aptitud = 1/costo1 + 1/costo2;
fa[infoFunction] = aptitud;
}
return fa;
END
```

A12. Algoritmo utilizado para la selección de las rutas aleatorias.

Se encarga de elegir un número de individuos del conjunto de rutas aleatorias generadas, los valores de la función de aptitud para cada individuo son normalizados, se usa la función "**random"** que devuelve números aleatorios entre cero y uno para la elección de la ruta basándose en el método de la ruleta sesgada.

Algoritmo 12. Algoritmo utilizado para la selección de las rutas aleatorias.

INPUTS: Rutas aleatorias con sus respectivas aptitudes.

OUTPUT: Rutas seleccionadas.

BEGIN

```
for(std::map<std::vector<int>,double>::iterator it = rutasAptitud.begin(); it != rutasAptitud.end(); ++it)
    {
    vectorDeRutas=it->first;
    vectorRutasOrdenadas.push_back(vectorDeRutas);
    funcionAptitud=it->second;
    sumaAptitudes = sumaAptitudes + funcionAptitud;
    vectorFAptitud.push_back(funcionAptitud);
    }
size_t pos=0;
for(int i=0; i<padres;i++)
   {
    numeroRandom = get_random(rangoMinimo, rangoMaximo);
    std::vector<double>:: iterator itedos;
    pos=0;
    double sumaRangos =0;
    for( itedos=vectorFAptitud.begin(); itedos!= vectorFAptitud.end(); itedos++)
        {

         rangos = *itedos/sumaAptitudes;

         sumaRangos = sumaRangos + rangos;
         if(sumaRangos>=numeroRandom)break;

         pos++;
```

```
        }
    rutaPadre= vectorRutasOrdenadas.at(pos);
    if(resultadoSeleccion.size()>0)
       {
        if(resultadoSeleccion.find(rutaPadre)==resultadoSeleccion.end())
          {

           resultadoSeleccion.insert(std::pair<vector<int>,double>(rutaPadre,*itedos));

          }
        else
       {
     i--;
    }
}
else
    {
     resultadoSeleccion.insert(std::pair<vector<int>,double>(rutaPadre,*itedos));
     }
}
return resultadoSeleccion;
```
END

A13. Algoritmo para la realización de cruce entre individuos.

En este algoritmo se analiza es posible aplicar o no el operador de cruce. Para esto, se realiza un recorrido sobre el vector de cada ruta seleccionada mediante un iterador **"it"** y se descartan aquellas que únicamente tiene dos elementos ya que son nodos adyacentes, seguidamente se validan las rutas para el cruce, que son almacenadas en vectores **"hijoUno"** e **"hijoUno"**.

Algoritmo 13. Algoritmo para la realización de cruce entre individuos.

INPUTS: Rutas seleccionadas.

OUTPUT: Vectores con las nuevas rutas(hijos).

BEGIN

```
for(std::map<std::vector<int>,double>::iterator it = rutasSeleccionadas.begin(); it
!=rutasSeleccionadas.end(); ++it)
    {
    vectorDeRutasSelecionadas=it->first;
    if(vectorDeRutasSelecionadas.size()>4)
      {
       rutasValidas1.insert(std::pair<vector<int>,int>(vectorDeRutasSelecionadas, i));
       rutasValidas2.insert(std::pair<vector<int>,int>(vectorDeRutasSelecionadas, i));
       i++;
       }
    }
if(rutasValidas1.size()<2 && rutasValidas2.size()<2)return totalCruce;
f1=rutasValidas1.size();
if1=0;
```

```
for(std::map<std::vector<int>,int>::iterator it1 =rutasValidas1.begin(); it1!= rutasValidas1.end(); it1++)
   {
   if(if1<(f1-1))
      {
       if2=0;
for(std::map<std::vector<int>,int>::iterator it2=rutasValidas2.begin(); it2!= rutasValidas2.end(); it2++)
   {
   if(if2>if1)
     {
      vectorResultante1 = it1->first;
      vectorResultante2 = it2->first;
      hijoUno=producirHijo1(vectorResultante1,vectorResultante2);
      if(hijoUno.empty())continue;
      std::vector<int> insertarHijo;
      for(std::map<std::vector<int>,int>::iterator it = hijoUno.begin(); it !=hijoUno.end(); ++it)
         {
          insertarHijo=it->first;
          int cont=it->second;
          totalhijos.insert(std::pair<vector<int>,int>(insertarHijo,cont));

          }

     hijoDos=producirHijo2(vectorResultante1, vectorResultante2);

     for(std::map<std::vector<int>,int>::iterator it = hijoDos.begin(); it !=hijoDos.end(); ++it)
         {
          insertarHijo=it->first;
          int cont=it->second;
          totalhijos.insert(std::pair<vector<int>,int>(insertarHijo,cont));

          }



       hijosValidos=rutaValidas1(totalhijos, rutasValidas1);

       for(std::map<std::vector<int>,int>::iterator it = hijosValidos.begin(); it !=hijosValidos.end(); ++it)
           {
            insertarHijo=it->first;
            int cont=it->second;
            totalCruce.insert(std::pair<vector<int>,int>(insertarHijo,cont));
            }
    }if2++;
   }if1++;
  }
}
if(hijosValidos.empty())noNodos = true;
END
```

A14. Algoritmo para la aplicación de reducción en las rutas generadas.

Realiza la elección de los tres mejores individuos del conjunto de soluciones compuesto por rutas padres e hijas, inicialmente se ubican las aptitudes de los individuos en un vector

“std::vector<double>aptitudes”, mediante la función “**aptitudes.push_back”** finalmente se recorre al vector de aptitudes con un iterador y se eligen los tres individuos con las mejores aptitudes, que son almacenados en el vector “**rutasElegidas”.**

Algoritmo 14. Algoritmo para la aplicación de reducción en las rutas generadas.

INPUTS: Vectores de rutas padre y vector de rutas hijos.

OUTPUT: Vector con los tres mejores individuos.

BEGIN

```
if(hijos.empty())
  {
   for(std::map<std::vector<int>,double>::iterator it = padres.begin(); it != padres.end(); ++it)
       {
        posiciones.push_back(i);
        i++;
        rutas.push_back(it->first);
        aptitudes.push_back(it->second);
       }
  }
else
  {
   for(std::map<std::vector<int>,double>::iterator it = padres.begin(); it != padres.end(); ++it)
      {
        posiciones.push_back(i);
        i++;
        rutas.push_back(it->first);
        aptitudes.push_back(it->second);
      }
  aptitudHijos=RoutingNuevo::funcionAptitud(hijos);
  for(std::map<std::vector<int>,double>::iterator it = aptitudHijos.begin(); it != aptitudHijos.end(); ++it)
     {
       posiciones.push_back(i);
       i++;
       rutas.push_back(it->first);
       aptitudes.push_back(it->second);
     }
  }
for(std::vector<double>::size_type j =1; j<aptitudes.size();j++)
    {
    for(std::vector<double>::size_type t =0; t<(aptitudes.size()-1);t++)
      {
       if(aptitudes.at(t)<aptitudes.at(t+1))
         {

          posicionNueva=posiciones.at(t);

          posiciones.at(t)=posiciones.at(t+1);
          posiciones.at(t+1)= posicionNueva;
          aptitudNueva=aptitudes.at(t);
          aptitudes.at(t)=aptitudes.at(t+1);
          aptitudes.at(t+1)=aptitudNueva;
```

```
        }
       }
     }
for(std::vector<int>::size_type t=0; t<rutasElegidas ;t++)
    {
     x= posiciones.at(t);
     rutaEnMapa = rutas.at(x);
     aptitudOrdenada = aptitudes.at(t);
     rutasResultantes[rutaEnMapa]=aptitudOrdenada;
     }
posiciones.clear();
rutas.clear();
aptitudHijos.clear();
rutaEnMapa.clear();
rutasFinales.clear();
return rutasResultantes;
}
END
```

A15. Algoritmo para la elección de la ruta de mejor aptitud.

En este algoritmo se captura el tamaño del vector mediante la función "**aptitudes.size"** se recorre todo el vector "**aptitudes"** mediante variables de tipo tamaño "**size_type"** y se realiza el método de la burbuja para obtener las rutas con mayor aptitud.

Algoritmo 15. Algoritmo para la elección de la ruta de mejor aptitud.

INPUTS: Vector de las rutas obtenidas después de la reducción.

OUTPUT: Vector de la mejor ruta.

BEGIN

```
for(std::map<std::vector<int>,double>::iterator it=ultimasRutas.begin(); it!= ultimasRutas.end(); it++)
     {
      posiciones.push_back(i);
      i++;
      rutas.push_back(it->first);
      aptitudes.push_back(it->second);
     }
for(std::vector<double>::size_type j =1; j<aptitudes.size();j++)
     {
      for(std::vector<double>::size_type t =0; t<(aptitudes.size()-1);t++)
         {
           if(aptitudes.at(t)<aptitudes.at(t+1))
             {

               posicionNueva=posiciones.at(t);

               posiciones.at(t)=posiciones.at(t+1);
               posiciones.at(t+1)= posicionNueva;
               aptitudNueva=aptitudes.at(t);
               aptitudes.at(t)=aptitudes.at(t+1);
```

```
            aptitudes.at(t+1)=aptitudNueva;
            }
          }
        }
  x= posiciones.at(cero);
  rutaEnMapa = rutas.at(x);
  std::vector<int>:: iterator j;
  for( j=rutaEnMapa.begin(); j!= rutaEnMapa.end(); j++)
     {
      rutas.clear();
      aptitudes.clear();
      posiciones.clear();
      return rutaEnMapa;
      }
  }
END
```

A16. Algoritmo para la obtención de la nueva población (Rutas padres + Rutas hijos).

Se recorren los vectores que contienen las rutas de las generaciones mediante los iteradores **"it"** y se almacenan en el mapa **"generacionPoblacion"** con la función **"generacionPoblacion. Insert".**

Algoritmo 16. Algoritmo para la obtención de la nueva población (Rutas padres + Rutas hijos).

INPUTS: Vectores de las generaciones (vector padres y vector hijos).

OUTPUT: Mapa con todas las rutas.

BEGIN

```
for(std::map<std::vector<int>,double>::iterator it=pobla.begin(); it!= pobla.end(); it++)
     {
      rutaPobla=it->first;
      fitnnessPobla = it->second;
      generacionPoblacion.insert(std::pair<vector<int>,double>(rutaPobla,fitnnessPobla));

      }
for(std::map<std::vector<int>,double>::iterator it=desce.begin(); it!= desce.end(); it++)
     {
      rutaDesce=it->first;
      fitnnessDescen = it->second;
      generacionPoblacion.insert(std::pair<vector<int>,double>(rutaDesce,fitnnessDescen));
     }
rutaDesce.clear();
rutaPobla.clear();
return generacionPoblacion;
}
END
```

A17. Algoritmo genético para el enrutamiento.

Se analiza el valor de aptitud de los individuos después de aplicar todos los operadores genéticos, si éste alcanza o pasa el valor máximo establecido, la ruta de este individuo es elegida y se envía la ráfaga, de lo contrario, se entra a un ciclo en el cual se realiza el proceso de genético incluyendo todos los individuos de la primera generación.

Algoritmo 17. Algoritmo genético para el enrutamiento.

INPUTS: Dirección de destino.

OUTPUT: Ruta elegida.

BEGIN

```
 if(aptitudMasGrande>=mayorAptitud)
   {
    rutaFinalEscogida=RoutingNuevo::rutaEscogida(rutasFinales);
    }
else
   {
   for(numGeneraciones=1; numGeneraciones<2;numGeneraciones++)
      {
       mayorAptitud=1.1;
       rutasResul=RoutingNuevo::rutasAleatorias(dirDestino);
       std::vector<int>Adj;  for(std::map<std::vector<int>,int>::iterator  it  =  rutasResul.begin();  it  !=
       rutasResul.end(); ++it)
            {
             std::vector<int>:: iterator itj;for( itj=Adj.begin(); itj!= Adj.end(); itj++) ;
             }
      rutasYAptitud = RoutingNuevo::funcionAptitud(rutasResul);
      descendenciaYAletorias = RoutingNuevo::sumaPoblacionDescendencia(rutasYAptitud, rutasFinales);
      seleccionC = RoutingNuevo::seleccion(descendenciaYAletorias);

       std::vector<int>a;  for(std::map<std::vector<int>,double>::iterator  it  =  seleccionC.begin();  it  !=
       seleccionC.end(); ++it)
           {
            std::vector<int>:: iterator i; for( i=a.begin(); i!= a.end(); i++){ev << *i << endl;}
            crucePadres = RoutingNuevo::cruce(seleccionC);
            rutasFinales=RoutingNuevo::reduccion(seleccionC, crucePadres);
            aptitudMasGrande=RoutingNuevo::mayorAptitudRuta(rutasFinales);
            if(aptitudMasGrande>=mayorAptitud)
               {
                rutaFinalEscogida=RoutingNuevo::rutaEscogida(rutasFinales);break;
                }
           else
                {
                 mayorAptitud=mayorAptitud-0.1;
                }
          }
         rutaFinalEscogida=RoutingNuevo::rutaEscogida(rutasFinales);
      }
    rutasResul.clear();
    rutasYAptitud.clear();
    seleccionC.clear();
    crucePadres.clear();
```

```
    rutasFinales.clear();
    rutasF.clear();
    descendenciaYAletorias.clear();
    return rutaFinalEscogida;
   }
END
```

A18. Algoritmo para el enrutamiento teniendo en cuenta un vector de rutas.

Permite la actualización del BCP de la ráfaga obteniendo los parámetros necesarios para su enrutamiento como dirección de origen y destino.

Algoritmo 18. Algoritmo para el enrutamiento teniendo en cuenta un vector de rutas.

INPUTS: Direcciones de origen y destino y saltos.

OUTPUT: Un puntero de la clase **"CoreRouting"** llamado **"ítem"** que contiene el puerto de salida la dirección de destino y la longitud de onda de una ráfaga, longitud de onday puerto de salida.

BEGIN

```
  if(sourceDirection==inLabel)
    {
     puertoSalida=0;
     item->setOutColor(-9);
     item->setOutPort(puertoSalida);
     item->setOutLabel(inLabel);
     return item->dup();
     }
  else
    {
    for(std::vector<int>::iterator it=rutaOD.begin(); it!=rutaOD.end(); it++)
        {
         cTopology *topo = new cTopology("topo");
         topo->extractByParameter("variableCoreNode");
         cTopology::Node *thisNode = topo->getNodeFor(getParentModule());
         if(topo->getNode(*it-1)!=thisNode)
           {
           i++;
            }
        else
           {
            encontro=true;
            int w = (numberPorts-1)*(lambdasC+1);
            for(y=0; y<w; y=y+(lambdasC+1))
                {
                 cTopology::Node *adjacents = thisNode->getLinkOut(y)->getRemoteNode();
                 dirAdjacentNode = adjacents->getModule()->getParentModule()-
                 >getSubmodule("edgeHost")- >getSubmodule("Host")->getSubmodule("source")-
                 >getAncestorPar("address");
                }
          if(posicion !=rutaOD.size())
              {
```

```
                if(dirAdjacentNode == rutaOD.at(i))
                  {
                   cGate *gate = thisNode->getLinkOut(y)->getLocalGate();
                   outGateIndex=gate->getIndex();
                   posicion++;
                  }
                }
            }
          }
       }
  if(outGateIndex >=rango1 && outGateIndex<=rango2)
     {
      puertoSalida=1;
      item->setOutColor(-9);
      item->setOutPort(puertoSalida);
      item->setOutLabel(inLabel);
      rafagasPasantes++;
      salt = salt -1;
      item->setSaltos(salt);
      item->setVRouting(rutaOD);
      return item->dup();
     }
  if(outGateIndex >rango2 && outGateIndex<=rango3)
     {
     puertoSalida=2;
     item->setOutColor(-9);
     item->setOutPort(puertoSalida);
     item->setOutLabel(inLabel);
     rafagasPasantes++;
     salt = salt -1;
     item->setSaltos(salt);
     item->setVRouting(rutaOD);
     return item->dup();
     }
 if(outGateIndex >rango3 && outGateIndex<=rango4)
    {
     puertoSalida=3;
     item->setOutColor(-9);
     item->setOutPort(puertoSalida);
     item->setOutLabel(inLabel);
     rafagasPasantes++;
     salt = salt -1;
     item->setSaltos(salt);
     item->setVRouting(rutaOD);
     return item->dup();
     }
 if(outGateIndex >rango4 && outGateIndex<=rango5)
    {
     puertoSalida=4;
     item->setOutColor(-9);
     item->setOutPort(puertoSalida);
     item->setOutLabel(inLabel);
     rafagasPasantes++;
     salt = salt -1;
     item->setSaltos(salt);
```

```
    item->setVRouting(rutaOD);
    return item->dup();
    }
 }
return NULL;
}
END
```

A19. Algoritmo para almacenar las mejores rutas resultantes del proceso del algoritmo genético (cognición).

Este algoritmo almacena las rutas resultantes del AG en un vector de vectores. Solo introduce las rutas que comparten el mismo origen, ya que la misma clase se ejecuta en cada nodo de la red. Posteriormente, tras haber asignado un número máximo de rutas al vector, el resultado es llamado durante la ejecución del algoritmo para verificar su validez y comprobar que la ruta, además, presenta el mismo destino de la ráfaga. Si se cumple este último criterio, la ruta es escogida para formar parte del conjunto de caminos aleatorios originados al comienzo del algoritmo genético.

Algoritmo 19. Algoritmo para almacenar las mejores rutas resultantes del proceso del algoritmo genético (cognición).

INPUTS: Dirección destino.

OUTPUT: Vector de rutas provenientes del desarrollo del algoritmo genético.

BEGIN

```
std::vector<std::vector<int> > vectorCognicionResultantex;
std::vector<int> rutaCognix;
size_t tamanox;

for(tamanox=0; tamanox<bestRoutes.size(); tamanox++)
    {
    if(bestRoutes.at(tamanox).at(0)==sourceDirection &&
    bestRoutes.at(tamanox).at(bestRoutes.at(tamanox).size()-1)==inLabel)
        {
        vectorCognicionResultantex.push_back(bestRoutes.at(tamanox));
        }
    }
if (vectorCognicionResultantex.size()!=0)
   {
   for(tamanox=0; tamanox<vectorCognicionResultantex.size();tamanox++)
        {
        if(contRutas<numRutas)
           {
           rutaCognix=vectorCognicionResultantex.at(tamanox);
           rutas[rutaCognix]= contRutas;
           contRutas++;
           }

          else break;
        }
```

END

A 20. Algoritmo generador de un vector de rutas dinámico (cognición).

Se encarga de definir un número máximo de rutas provenientes de los resultados anteriores del algoritmo genético. El vector almacenará dinámicamente los caminos, abstrayendo el primero de ellos, para dar cabida a una ruta entrante cuando así se requiera. El proceso de llenado ocurre cada vez que se presenta la petición de una ruta para transportar la ráfaga.

Algoritmo 20. Algoritmo generador de un vector de rutas dinámico (cognición)

INPUTS: Vector proveniente del resultado del algoritmo genético.

OUTPUT: Vector de rutas dinámico.

BEGIN

```
void RoutingNuevo::putVectorBest(std::vector<int> bVector){

   if(bestRoutes.size()<=(maxCognitive-1))
     {
      bestRoutes.push_back(bVector);
   else
     {
      bestRoutes.at(posCognitive)=bVector;
      posCognitive++;
   if(posCognitive==maxCognitive)
      {
       posCognitive=0;
      }
}
```

END

ANEXO B. DIAGRAMAS DE CLASES DE LA RED OBS/WDM.

En este anexo se realiza la descripción de las clases necesarias para el correcto funcionamiento de la red junto con sus atributos y métodos más representativos.

Tabla 1. Descripción de la clase Source.

Nombre
Source (Fuente)
Descripción
Esta clase se encarga de la asignación de direcciones y tipo de información de manera aleatoria para enviar el mensaje.
Atributos
myAddress: Atributo de tipo entero (int) que contiene la dirección de un nodo fuente. destAddresses: Atributo de tipo cadena(string) que contiene las direcciones de los nodos de la red. sendIATime: Atributo de tipo volátil que contiene la velocidad de generación de tráfico. packetLengthBytes: Atributo que contiene el tamaño de los paquetes en Bytes. type: Atributo de tipo entero (int) que contiene el tipo de información a transmitir (vídeo, voz, datos). pkCounter: Atributo de tipo entero largo (long) que permite el conteo de paquetes. endToEndDelaySignal: Átributo estadístico que permite el análisis del retardo de extremo a extremo de una señal. hopCountSignal: Atributo estadístico que permite el conteo de saltos.
Métodos
initialize(): Inicializa todos los atributos de la clase y asigna a los paquetes una dirección y prioridad, como se muestra a continuación: intdestAddress = destAddresses[intuniform(0, destAddresses.size()-1)] inttype = intuniform(0,2); handleMessage(cMessage *msg): Se encarga del manejo, envío del mensaje y de la generación de rutas aleatorias. ~Source(): Destructor, cancela el envío de paquetes.

Tabla 2. Descripción de la clase Sink.

Nombre
Sink(Destino)
Descripción
Se encarga de recibir los mensajes y eliminarlos. Por motivos de diseño ésta clase solo realiza el método de eliminación para evitar la acumulación de paquetes en el destino, ya que el objetivo es evaluar únicamente los parámetros del tercer nivel del modelo OSI.
Atributos
No tiene.
Métodos
Deletemsg: Elimina el mensaje.

Tabla 3. Descripción de la clase Classifier.

Nombre
Classifier(Clasificador)
Descripción
Se encarga de clasificar los paquetes en colas con respecto a una dirección de destino y una prioridad.
Atributos
numOuts: Atributo de tipo entero (int) que representa el número de compuertas de salida. PackSize: Atributo que almacena datos recibidos. droppedPacket: Atributo de tipo entero (int) que almacena el número de paquetes descartados, que puede visualizarse durante la ejecución de la simulación mediante una función llamada WATCH. PackSizeVec: Atributo que almacena la longitud de los paquetes recibidos (en bytes).
Métodos
initialize(): Inicializa todos los parámetros de la clase y almacena información proveniente de la clase Classifier_Rules. finish(): Almacena datos estadísticos. handleMessage(cMessage *msg): Se encarga de recibir paquetes y comparar si coinciden con alguna regla correspondiente a un archivo de texto como se describe en el ANEXO A. Si coincide, se manda a la correspondiente compuerta de salida, de lo contrario dicho paquete se descarta. ~Classifier(): Libera memoria dinámica.

Tabla 4. Descripción de la clase Burstiffier.

Nombre
Burstiffier(Generador de Ráfagas)
Descripción
Esta clase almacena los paquetes entrantes con el mismo destino y prioridad, y los ensambla en ráfagas ópticas, usando un criterio dado (tiempo de ensamblaje, número de paquetes, tamaño máximo de almacenamiento en bytes).
Atributos
maxTime: Atributo de tipo simtime_t que almacena el tiempo máximo para ensamblaje de la ráfaga. maxSize: Atributo de tipo entero (int) que almacena el tamaño máximo de la ráfaga. numPackets: Atributo de tipo entero (int) que almacena el máximo número de paquetes. minOffset: Atributo de tipo simtime_t que almacena el Offset mínimo del BCP. maxOffset: Atributo de tipo simtime_t que almacena el Offset máximo del BCP. minSizePadding: Atributo de tipo entero (int) que almacena el tamaño mínimo que una ráfaga debe tener para ser enviada. Si al programar su envío no logra alcanzar parámetro, la ráfaga se completa con bytes de relleno. addLastPacket: Atributo de tipo booleeano. Si es verdadero, permite que el último paquete se adicione a la ráfaga, incluso si la ráfaga excede su tamaño. Si es falso, la ráfaga se ensambla y el paquete será el primero de la siguiente ráfaga. tamHeader: Atributo de tipo entero (int) que almacena el tamaño de la cabecera de la ráfaga. tamHeaderPacket: Atributo de tipo entero (int) que almacena el Tamaño de la cabecera de

cada paquete insertado a la ráfaga. burstBits: Atributo de tipo entero (int) que almacena el Tamaño actual de la ráfaga, en Bytes. numPacketsInBurst: Atributo de tipo entero que almacena el Número de paquetes en la ráfaga. destLabel: Atributo de tipo entero (int) que almacena el etiqueta de destino de la ráfaga. burstCounter: Atributo de tipo entero (int) que almacena el tamaño de la ráfaga. recvPackSize: Atributo de tipo estadístico que almacena el tamaño de los paquetes recibidos. PacketTime: Promedio de retardo de paquetes, debido al ensamblaje de ráfagas. firstPacket_t: Atributo de tipo simtime_t que almacena el tiempo de llegada del primer paquete de la ráfaga actual**.**
Métodos
initialize(): Inicializa todos los parámetros de la clase. handleMessage(cMessage *msg): Recibe paquetes, analiza si cumplen con las condiciones para ser almacenados en ráfagas. assemblyBurst(): Se encarga del ensamblaje de la ráfaga. Finish(): Almacena datos estadísticos. ~Burstifier(): Destructor.

Tabla 5. Descripción de la clase Sender.

Nombre
Sender (Salida)
Descripción
Es la responsable del tratamiento de la ráfaga para la creación del paquete de control y el paquete de datos. Cada vez que una ráfaga es recibida se crea el algoritmo genético. En esta clase se asigna, además, la ruta resultante a un método que aplicará un método de control cognitivo basado en los estados anteriores de la red
Atributos
color: Vector que permite la implementación de colores para el uso de longitudes de onda. numLambdas: Atributo de tipo entero (int) que almacena el número de canales de datos. dataRate: Atributo de tipo doble (double) que almacena la velocidad de transmisión del canal óptico. BCPSize: Atributo de tipo entero (int) que almacena el tamaño del BCP. guardTime: Atributo de tipo simtime_t que almacena el Offset entre la transmisión consecutiva de ráfagas. maxSchedBurstSize: Atributo de tipo entero (int) que almacena el tamaño máximo de la cola de ráfagas programadas (0 = infinito). control_is_busy: Atributo de tipo booleano, si es verdadero el canal de control se encuentra actualmente ocupado. Horizon: Vector que muestra el tiempo de simulación en dónde cada canal se encuentra libre. vector<cOutVector*>: Vector de valor del horizonte. Uno para cada Lambda. burstRecv: Atributo de tipo entero (int) que permite el conteo de ráfagas recibidas. burstSent: Atributo de tipo entero (int) que permite el conteo de ráfagas enviadas. burstDroppedByOffset: Atributo de tipo entero (int) estadístico que permite el conteo de ráfagas descartadas porque alcanzaron el offset mínimo. burstDroppedByQueue: Atributo de tipo entero (int) estadístico que permite el conteo de ráfagas descartadas debido a que la cola de ráfagas en espera está llena. rutaOrigenDest: Atributo de tipo vector al cual le es asignado el resultado de una ruta hecha por el AG.
Métodos
initialize(): Inicializa todos los parámetros de la clase.

handleMessage(cMessage *msg): Recibe las ráfagas, genera el algoritmo genético encargado de establecer la ruta adecuada entre origen y destino, asigna un horizonte apropiado para enviarlas, si está libre, envía un BCP para el establecimiento del camino óptico y finalmente si es posible programa la ráfaga. Además, descarta la ráfaga si no cumple con el tiempo de offset apropiado o si su número en la cola para ser enviadas es muy alto. finish(): Asigna valores a los parámetros que almacenan datos estadísticos de la red. findNearestHorizon(): Devuelve el menor tiempo que encuentra descrito en el ANEXO A. ~Sender(): Libera memoria para el vector horizonte.

Tabla 6. Descripción de la clase EdgeNodeDissasembler.

Nombre
EdgeNodeDisassembler(Desensamblador)
Descripción
Recibe las ráfagas y las desensambla para enviar los paquetes a los equipos finales.
Atributos
listSize: Atributo de tipo entero (int) que almacena el tamaño de la lista de ráfagas. recvBursts: Atributo de tipo estadístico que permite el conteo de ráfagas recibidas.
Métodos
initialize(): Inicializa todos los parámetros de la clase. handleMessage(cMessage *msg): Recibe la ráfaga, la libera los paquetes desensamblando la ráfaga. finish(): Almacena los datos estadísticos de la clase ~EdgeNodeDisassembler(): Limpia la cola de las ráfagas recibidas.

Tabla 7. Descripción de la clase Classifier_Rules

Nombre
Classifier_Rules
Descripción
Representa un despachador de reglas que provee un método de comprobación de emparejamiento con respecto a una dirección y una prioridad.
Atributos
destAddr: Atributo de tipo entero (int) que almacena la dirección de destino. priority: Atributo de tipo entero (int) que almacena la prioridad de los datos.
Métodos
Classifier_Rules(char* rule):Inicia los valores en falso, Analiza si las reglas tienen un tipo deservicio y una dirección. bool match(cMessage *msg): Comprueba cuales opciones se encuentran en las reglas. ~Classifier_Rules(): Destructor.

Tabla 8. Descripción de la clase CoreInput.

Nombre
CoreInput (Entrada del Core)
Descripción
Este módulo actúa como interfaz entre los módulos **“CoreControlUnit”** y **“OXC”**. Separa los canales de datos y de control e implementa métodos para cambiar los colores a compuertas usando identificadores físicos.

Atributos
vector<map<int,int>>: Vector que contiene la entrada del mapa de colores. numPorts: Atributo de tipo entero (int) que almacena el número de fibras conectadas a un nodo. *inPortBegin: Vector que almacena el índice de la compuerta de entrada a la cada fibra óptica se encuentra conectada. outDataBegin: Vector que almacena el índice de la compuerta de salida a la cual el comienzo de los canales de datos de cada fibra óptica se encuentran conectados. portLen: Atributo de tipo entero (int) que almacena el número de canales (datos+control) para cada fibra óptica.
Métodos
initialize():Inicializa los parámetros de la clase. handleMessage(cMessage *msg): Analiza si el elemento entrante es un BCP o una ráfaga para enviarlos hacia el destino indicado. getInPort(intgateIndex): Dado el índice de la compuerta de entrada, calcula el número de fibra correspondiente. getInLambda(intgateIndex): Dado el índice de la compuerta de entrada, calcula el número del lambda dentro de la fibra correspondiente. getOXCGate(intport, int lambda): Convierte enlaces de datos, identificados por el Id. de la fibra y lambda (no color) a compuertas de salida para OXC. getLambdaByColour(intport,intcolor): Recibe un enlace identificado por el Id de la fibra y el color para establecerlo como una longitud de onda. ~CoreInput(): Libera memoria.

Tabla 9. Descripción de la clase CoreControlLogic.

Nombre
CoreControlLogic
Descripción
Crea un camino óptico basándose en el BCP entrante.
Atributos
dataRate: Atributo de tipo doble (double) que almacena velocidad de transmisión del canal óptico. dropCounter: Atributo de tipo entero (int) estadístico que almacena el número de ráfagas descartadas. recvBurstCounter: Atributo de tipo entero (int) estadístico que almacena el número de ráfagas recibidas. schedBurstCounter: Atributo de tipo entero (int) estadístico que almacena el número de ráfagas programadas. processingTime: Atributo de tipo simtime_t que contiene la unidad de control de procesamiento de tiempo para cada BCP. guardTime: Atributo de tipo simtime_t que contiene el tiempo offset entre la llegada de la ráfaga y el orden de establecimiento del canal. vector<int>recibeBCP: Vector que contiene la ruta. dirOrigen: Atributo de tipo entero (int) que almacena la dirección de la ráfaga.
Métodos
initialize(): Se inicializan todos los parámetros y datos estadísticos de la red. finish(): Registra ráfagas descargadas. handleMessage(cMessage *msg): Extrae la información contenida en el BCP, comprueba si la ráfaga se encuentra programada para llegar en algún momento, se encarga de descartar la ráfaga si los tiempos offset programados no son apropiados o enviarla si el camino óptico

pudo establecerse. ~CoreControlLogic(): Libera memoria.

Tabla 10. Descripción de la clase CoreOutputHorizon.

Nombre
CoreOutputHorizon
Descripción
Se encarga de encontrar un valor de lambda por el que se envía una ráfaga, basándose en el algoritmo LAUC basado en Horizonte descrito en el ANEXO A.
Atributos
*horizon: Vector horizonte. portLambdas: Atributo de tipo entero (int) que almacena el número de canales de datos para cada puerto.
Métodos
initialize(): inicializa los parámetros de la clase. findNearestLambda(intport,simtime_tarrivalTime): Encuentra el lambda cuyo horizonte (tiempo en el que el canal está libre) es el menor y el más cercano al valor del tiempo de llegada dados. updateHorizon(intport, int lambda, simtime_tnewTime): Actualiza el valor del horizonte al nuevo valor de tiempo. getHorizon(intport,int lambda): Obtiene el horizonte del canal seleccionado. finish(): Almacena los datos estadísticos de la clase. ~CoreOutputHorizon(): Libera la memoria de los vectores.

Tabla 11. Descripción de la clase EOconverter.

Nombre
EOconverter
Descripción
Convierte un BCP en el dominio eléctrico al dominio óptico
Atributos
dataRate: Atributo de tipo doble (double) que almacena la velocidad de transmisión de la fibra óptica. EOConversionDelay: Atributo de tipo simtime_t que almacena el retardo de conversión del módulo numPorts: Atributo de tipo entero (int) que almacena el número de fibras. control_is_busy: Atributo de tipo booleano que indica si el canal de control se encuentra ocupado o no en el momento.
Métodos
initialize(): Inicializa los parámetros de la clase. handleMessage(cMessage *msg): Actualiza la información del número de BCPs que esperan para la conversión, si el canal está ocupado, se almacena el BCP en la cola, si no, se convierte el BCP del dominio eléctrico al óptico y se envía. ~EOConverter(): Borra los mensajes pendientes en la cola para liberar memoria.

Tabla 12. Descripción de la clase OEconverter.

Nombre
OEconverter
Descripción
Convierte el BCP en el dominio óptico al dominio eléctrico
Atributos
OEConversionDelay: Atributo de tipo simtime_t que almacena el retardo de conversión en el nodo.
Métodos
initialize(): Inicializa los parámetros de la clase. handleMessage(cMessage *msg): Realiza un duplicado del BCP para pasarlo al dominio eléctrico, y envía el paquete de control. finish(): Almacena los datos estadísticos de la clase. ~OEConverter():Borra los mensajes pendientes en la cola.

Tabla 13. Descripción de la clase OpticalCrossConnect.

Nombre
OpticalCrossConnect
Descripción
Punto de interconexión donde las ráfagas ópticas pasan de forma transparente.
Atributos
* schedulingTable: Vector que representa las conexiones de las compuertas de entrada.
Métodos
setGate(intinGate, intoutGate): Compuerta de entrada del módulo OXC usada como índice. unsetGate(intinGate): Deshabilitar la conexión con la compuerta de la salida dada. initialize(): Inicializa los parámetros de la clase. handleMessage(cMessage *msg): Contiene un algoritmo sencillo que verifica si la compuerta de entrada tiene una conexión programada y manda el mensaje a la compuerta de salida asignada. ~OpticalCrossConnect(): Libera la tabla de planificación.

Tabla 14. Descripción de la clase RoutingNuevo.

Nombre
RoutingNuevo
Descripción
Permite el enrutamiento de la ráfaga. Contiene un método encargado de almacenar las n mejores rutas, producto del AG, e irlas cambiando de posición (o descartándolas) cada vez que una nueva ruta llega al método
Atributos
puertoSalida: Atributo de tipo entero (int) que almacena el valor del puerto de salida. lambdasC: Atributo de tipo entero (int) que almacena el número de lambdas entre los nodos core. lambdasE: Atributo de tipo entero (int) que almacena en número de lambdas entre nodos edge y core. miDireccion: Atributo de tipo entero (int) que almacena la dirección del nodo actual, agregado para cTopology. direccion: Atributo de tipo entero (int) que almacena la dirección de destino. numberPorts: Atributo de tipo entero (int) que almacena el número de puertos. dirAdjacentNode: Atributo de tipo entero(int) que contiene la dirección del nodo adyacente. rafagasPasantes: Atributo de tipo entero (int) que contiene el número de ráfagas que pasan pero no entran a un nodo.

map<int,int>findAdjacentNodes(intdir,std::map<int,int>ant): Vector que almacena los nodos adyacentes a cada nodo.
i=false; Atributo de tipo boleano que informa si una ruta existe o no.Está en falso mientras no se encuentre una ruta.
adjacen: Atributo de tipo entero(int) que almacena el número del nodo adyacente escogido, después de realizar el recorrido aleatorio por el vector.
map<int,int>antNode: Mapa que contiene todos los nodos por los cuales ha pasado una ruta.
actNod: Almacena el nodo actual
vector<int>v: Vector de la ruta resultante.
map<int,int>adj: Mapa de los nodos adyacentes al nodo actual.
contRutas:Atributo de tipo entero(int)que cuenta las rutas.
numRutas: Atributo de tipo entero(int) que almacena el máximo número de rutas que queremos de origen a destino.
vector<int>infoFunction: Vector que almacena los datos de la función de Aptitud.
saltos: Atributo de tipo entero(int) que almacena el número de saltos en una ruta.
nodos: Atributo de tipo entero(int) que almacena el número de nodos dentro de una ruta.
w : Atributo de tipo entero(int) que almacena el número de longitudes de onda entre nodos core.
wTotal= Atributo de tipo entero(int) que almacena el número de longitudes totales de un enlace.
alfa = 1/((double)(wTotal)-1): Atributo de tipo doble (double) que almacena el valor del parámetro de diseño alfa.
rDestino: Atributo de tipo entero(int) que almacena el número de ráfagas que van a este destino.
rPasantes: Atributo de tipo entero(int) que almacena el número ráfagas que pasan por el nodo, pero no es su nodo destino.
totalRDestino: Atributo de tipo entero (int) que almacena la sumatoria de las ráfagas que van hacia los nodos de la ruta que se está analizando.
totalRPasantes: Atributo de tipo entero (int) que almacena la sumatoria de ráfagas que pasan por los nodos presentes en la ruta que se está analizando.
costo1: Atributo de tipo entero (int) que almacena el valor de la primera función de costo.
costo2: Atributo de tipo entero (int) que almacena el valor de la segunda función de costo.
costoTotal: Atributo de tipo entero (int) que almacena el valor total de la función de costo.
aptitud: Atributo de tipo entero (int) que almacena el valor de la función de Aptitud.
numberGate: Atributo de tipo entero(int) que almacena el número de compuertas que tiene un nodo.
vector<int>,double>funcionAptitud(std::map<std::vector<int>,int>rutas):Vector que almacena los datos obtenidos cuando se aplica la función de aptitud a cada nodo.
vector<int>rutasSeleccionadas: Vector que contiene las rutas seleccionadas.
vector<int>vectorDeRutas: Vector que contiene todas las rutas.
vector<int>rutaPadre: Vector de rutas padre.
vector<double>vectorFAptitud: Vector que tiene la aptitud de los individuos
vector<double>vectorRango: Vector que almacena la aptitud normalizada de los individuos (es entre 0 y 1).
funcionAptitud: Atributo de tipo doble (double) que abstrae la función de aptitud de cada individuo.
sumaAptitudes: Atributo de tipo doble (double) que almacena la suma del total de las aptitudes.
rangos: Atributo de tipo doble (double) que almacena el valor de aptitud normalizado.
vector<int>,double>resultadoSeleccion: Vector que almacena el resultado de la selección.
rangoMinimo: Atributo de tipo doble (double) que almacena el valor mínimo de la función de aptitud.
rangoMaximo: Atributo de tipo doble (double) que almacena el valor máximo de la función de aptitud.
padres: Atributo de tipo entero (int) que almacena el número de padres.
vector<int>,double>selección (std::map<std::vector<int>,double>rutasAptitud): Vector que almacena los individuos seleccionados con el proceso de ruleta.
vector<int>,double>resultadoCruce: : Vector que almacena e resultado de la del cruce.
vector<int>, int>HijosYAptitud: Vector que almacena la aptitud de los hijos.
vector<int>vectorDeRutasSelecionadas: Vector que almacena las rutas seleccionadas.

vector<int>,int> rutasValidas1: Vector que almacena rutas válidas.
vector<int>,int> rutasValidas2: Vector que almacena rutas válidas.
posCruce1: Atributo de tipo de tipo tamaño (size_t) que se detiene en la posición en la que el número del nodo es igual a posCrece2.
posCruce2: Atributo de tipo de tipo tamaño (size_t) que se detiene en la posición en la que el número del nodo es igual a posCrece1.
vector<int>hijo1: Vector que almacena el hijo 1.
vector<int>hijo2: Vector que almacena el hijo 2.
f1=0: Atributo de tipo de tipo tamaño (size_t) que almacena el tamaño de un vector para comparar.
f2=0: Atributo de tipo de tipo tamaño (size_t) que almacena el tamaño de un vector para comparar.
nodoComun: Atributo de tipo de tipo booleano que indica si dos rutas tienen nodos comunes.
noNodos: Atributo de tipo de tipo booleano que indica si dos rutas tienen el mismo número de nodos.
vector<int>validarHijo1: Vector que almacena un individuo para compararlo con otro y validarlo si no son iguales.
vector<int>validarHijo2: Vector que almacena un individuo para compararlo con otro y validarlo si no son iguales
contHijos: Atributo de tipo entero(int) que realiza el conteo de hijos.
noValido1: Atributo de tipo booleano que invalida un hijo si ya existe otro igual a él.
noValido2: Atributo de tipo booleano que valida un hijo si este no existe en el vector de individuos.
vector<int>double cruce: Vector que almacena las rutas obtenidas después del cruce, siguiente generación.
std::vector<std::vector<int> > vectorCognicionResultantex: Vector de rutas que contendrá los caminos cuyo origen y destino sea compatible con los datos de la ráfaga
std::vector<int> rutaCognix: Vector que almacena la ruta resultante, la cual será transmitida al mapa generador de rutas aleatorias.
size_t tamanox: Contador que recorre el tamaño del vector

Métodos

initialize(): Inicializa los parámetros de la clase y llama funciones de la clase cTopology para el enrutamiento. cTopology es una clase implementada en OMNeT++ diseñada para soportar el encaminamiento en redes de telecomunicaciones. (http://www.omnetpp.org/doc/omnetpp/api/classcTopology.html)
rutasAleatorias(intinLabel): Realiza el descubrimiento de los nodos adyacentes a cada nodo que es almacenado en el vector de nodos adyacentes:
funcionAptitud(std::map<std::vector<int>,int>rutas): Primero realiza el descubrimiento de los enlaces en cada nodo que se encuentra en la ruta, el número de saltos y las longitudes presentes en los enlaces, para encontrar el valor de la función de aptitud.
get_random(double min, doublemax): Devuelve un valor aleatorio entre un máximo y un mínimo.
seleccion(std::map<std::vector<int>,double>rutasAptitud): Selecciona los mejores individuos de la generación con el método ruleta.
cruce(std::map<std::vector<int>,double>rutasSeleccionadas): Realiza el cruce entre individuos válidos.
producirHijo2(std::vector<int>padre1,std::vector<int>padre2): Genera un vector con una de las rutas producto del cruce.
rutaValidas1(std::map<std::vector<int>,int>):Elige las rutas que cumplen con los requisitos y elimina aquellas que no.
rutasHijas, std::map<std::vector<int>,int>rutasPadre): Mapa que contiene las rutas hijas.
reduccion(std::map<std::vector<int>,double>padres, std::map<std::vector<int>,int> hijos): Se reduce el tamaño de la población bajo el criterio de la ruta de mejor aptitud.
mayorAptitudRuta(std::map<std::vector<int>,double>ultimasRutas): Organiza y elige la ruta que tiene la mejor aptitud.
rutaEscogida(std::map<std::vector<int>,double>ultimasRutas): Almacena la mejor ruta obtenida, producto de la aplicación de los operadores genéticos.
sumaPoblacionDescendencia(std::map<std::vector<int>,double>pobla,std::map<std::vector<int>,double>d

esce): Obtiene el vector de rutas generadas en el primer cruce mas las rutas padres.
algoritmoGenetico(intdirDestino): Mediante la dirección de destino permite aplicar a una ráfagaealgorito genético llamando a las funciones que contienen los operadores genéticos.
getEntry(intinLabel, std::vector<int>rutaOD, intsalt):Función que permite el enrutamiento de la ráfaga a través del puerto de salida para llegar a su destino.
intVoid RoutingNuevo::putVectorBest(std::vector<int>): Vector encargado de recibir vectores y devolver un arreglo o un conjunto de rutas, las cuales se irán actualizando con el tiempo.

Tabla 15. Descripción de la clase BurstList.

Nombre
BurstList.
Descripción
Usada como campo de control del mensaje.
Atributos
counter: Atributo de tipo entero (longint) usado para asignar un único Id a cada elemento. numElems: Atributo de tipo entero (longint) para almacenar el número de elementos dentro del contador. listSize: Atributo de tipo entero (longint) que almacena el tamaño de la lista. maxSize: Atributo de tipo entero (int) indica el tamaño máximo. maxElems: Atributo de tipo entero (int) indica el número de elementos. OutVectornumElemsVector: Vector de salida. QueueburstList: Lista.
Métodos
BurstList(): Constructor, Inicializa los parámetros de la clase. ~BurstList(): Destructor, limpia la lista. insertBurst(Burst *burst,simtime_tsendTime):Inserta la ráfaga al final del vector. Retorna la posición donde la ráfaga se encuentra almacenada. removeBurst(intbId): Elimina la ráfaga. retrieveBurst(intindex):Crea un mensaje "clon" reemplazando el original. retrieveBurstSize(intindex): Obtiene el tamaño de la ráfaga. retrieveSendTime(intindex): Obtiene el tiempo de envío. retrieveMinOffset(intindex): Obtiene el tiempo mínimo de offset. retrieveMaxOffset(intindex): Obtiene el tiempo máximo de offset. setMaxSize(intsize):Establece el tamaño máximo. setMaxElems(intnumElems): Establece el tamaño máximo de la cola.

Tabla 16. Descripción de la clase CoreOutput.

Nombre
CoreOutput
Descripción
Este módulo realiza la tarea inversa de CoreInput. Reúne los canales de control y de datos y los reordena con el fin de conectar el CoreNode con otro nodo OBS.
Atributos
vector<map<int,int>>: Mapa de colores de salida (un mapa por cada fibra). gate2Colour: Atributo de tipo entero (int) que permite el mapeo entre puerto/lambda y colores. numPorts: Atributo de tipo entero (int) que almacena el número de fibras conectadas al nodo. portLen: Atributo de tipo entero (int) que almacena el número de canales (datos+control) por

cada fibra óptica. inDataBegin: Este vector almacena el índice inicial de cada entrada de la fibra. outPortBegin: Este vector almacena el índice inicial de cada salida de la fibra.
Métodos
initialize(): Inicializa los parámetros y el proceso de análisis en el mapa de colores. handleMessage(cMessage *msg): Reenvía el paquete entrante a la correspondiente compuerta de salida. getOutPort(intgateIndex):Se usa para saber cuál es la fibra a la que pertenece el canal. getOutLambda(intgateIndex): Se usa para saber cuál es la longitud de onda a la que pertenece el canal. ~CoreOutput(): Libera memoria. getOXCGate(intport,int lambda): Convierte el puerto y lambda a compuertas de salida hacia OXC. getColourByLambda(intport,int lambda): Obtiene el color de salida mediante el puerto y lambda.

Tabla 17. Descripción de la clase Burst.

Nombre
Burst.
Descripción
Clase de tipo registro que es generada para la implementación de métodos en C++ que permiten una interacción adecuada con otras clases con las otras clases
Atributos
numPackets: Atributo de tipo entero (int) que contiene el número de paquetes que forman una ráfaga. minOffset : Atributo de tipo simtime_t que almacena el valor mínimo de offset. maxOffset : Atributo de tipo simtime_t que almacena el valor máximo de offset. idBurstifier : Atributo de tipo entero (int) que almacena el identificador de la ráfaga. secNum: Atributo de tipo entero (int) que lleva el número de la ráfaga que contiene el BCP. sendId : Atributo de tipo entero (int) que lleva el identificador de la ráfaga que contiene el BCP.
Métodos
void Burst::insertMessage(cMessage *msg): Inserta mensajes. cMessage* Burst::retrieveMessage(): Toma y recupera los primeros elementos de la cola. boolBurst::hasMessages(): retorna verdadero o falso si existe o no mensaje respectivamente. Burst::~Burst(): Limpia los mensajes.

Tabla 18. Descripción de la clase Mensaje.

Nombre
Mensaje
Descripción
Un mensaje dentro del entorno de simulación de OMNeT++ puede representar un evento, cliente, puesto de trabajo o un paquete, dependiendo de las necesidades del programador, dichos mensajes se representan con la clase cMessage o con su sub clase CPacket. En el proyecto actual se crean un número determinado de mensajes que son:**“Packet.msg”, “Burst.msg”, “BurstControlPacket”, “BurstifierInfo.msg”, “BurstSenderInfo.msg”, “ScheduleBurst.msg”, “BCPControlInfo.msg”, “ControlUnitInfo.msg”, “CoreRouting.msg”.**
Atributos

numPackets: Atributo de tipo entero (int) que contiene el número de paquetes que forman una ráfaga.
minOffset: Atributo de tipo simtime_t que almacena el valor mínimo de offset.
maxOffset: Atributo de tipo simtime_t que almacena el valor máximo de offset.
idBurstifier: Atributo de tipo entero (int) que identifica a la ráfaga.
deltaArriveBurst: Atributo de tipo entero (int) que almacena la diferencia entre el tiempo actual y el tiempo estimado de llegada de la ráfaga.
colorBurst: Atributo de tipo entero (int) que indica el color de la ráfaga.
label: Atributo de tipo entero (int) que lleva la etiqueta de destino.
numSeq: Atributo de tipo entero (int) que lleva el número de secuencia dentro del paquete de control.
Send: Atributo de tipo entero (int) quelleva el identificadordelsender.
burstSize: Atributo de tipo entero (int) que lleva el tamaño asociado a la ráfaga.
rutaEscogida[]: Vector de tipo entero (int) que lleva la ruta elegida.
numSaltos: Atributo de tipo entero (int) que almacena el número de saltos.
srcAddr: Atributo de tipo entero (int) que leva la dirección de la fuente.
Priority: Atributo de tipo entero (int) que indica la prioridad de un paquete.
destAddr: Atributo de tipo entero (int) que indica la dirección de destino de la ráfaga.
numberHops: Atributo de tipo entero (int) que guarda el número de saltos.
RoutingvRouting: Vector de tipo entero que lleva la ruta.
outPort: Atributo de tipo entero (int) que lleva el puerto de salida de la ráfaga.
outLabel: Atributo de tipo entero (int) que lleva la etiqueta de salida de la ráfaga.
outColor: Atributo de tipo entero (int) que lleva el color de salida de la ráfaga.
inGate: Atributo de tipo entero (int) que contiene la compuerta de entrada OXC.
outGate: Atributo de tipo entero (int) que contiene la compuerta de salida OXC.
Port: Atributo de tipo entero (int) que guarda lazfibra de entrada y de salida del BCP.
BCPArrival: Atributo de tipo entero simtime_t que almacena el tiempo de llegada del BCPini.
assignedLambda: Atributo de tipo entero (int) que contiene la longitud de onda asignada.
sendTime: Atributo de tipo entero simtime_t que contiene el tiempo programado para enviar la ráfaga.

Métodos

Las funciones que más se emplean de esta clase son: Getter y Setter, encargadas de obtener valores de los mensajes o establecerlos en ellos.

ANEXO D. TEOREMA DE LOS ESQUEMAS Y MODELOS DE CONVERGENCIA DE LOS ALGORITMOS GENÉTICOS.

Con el fin de garantizar la veracidad del proceso evolutivo, teniendo la certeza del funcionamiento y la adecuada convergencia de los Algoritmos Genéticos, se han efectuado teoremas e hipótesis que los respaldan. Tal es el caso del Teorema de los esquemas propuesto por John Holland [66].

Para explicar este teorema, son necesarias varias definiciones.

En primer lugar, un esquema es una representación capaz de describir varios cromosomas al mismo tiempo. Técnicamente, si se supone un alfabeto binario $S = \{0,1\}$, un esquema se forma a partir del alfabeto ampliado de S, es decir, $S' = \{0,1,*\}$, donde $*$ puede ser 1 o 0 [67], [68]. Por lo tanto, un esquema forma el conjunto de cadenas que se asocian con él, evaluando todos los posibles valores de$*$. Por ejemplo, el esquema $[001\ *]$ representa los cromosomas $[001\mathbf{1}]\ y\ [001\mathbf{0}]$.

Por otra parte, el esquema $[0101]$ representa solo una cadena, mientras que $[****]$ representa todas las posibles cadenas de longitud 4. De acá se desprende que cada esquema coincide con 2^r cadenas, siendo r el número de símbolos $*$ presentes en el esquema.

Ahora bien, el orden del esquema se denota por $o(S)$, e indica el número de posiciones que no contienen el símbolo " $*$ " dentro del esquema. Para el caso de $S = [0011\ **]$, $o(S) = 4$.

Cuanto mayor sea el orden del esquema, éste se hace más específico. Además este concepto es útil para calcular la probabilidad de supervivencia de un esquema al usar el operador mutación.

δ(S) se denomina longitud del esquema. Hace referencia a la distancia entre el primer y último valor fijo del esquema, permitiendo observar al mismo tiempo qué tan compacta es su información. Así, para el ejemplo anterior, $\delta(S) = 4 - 1 = 3$.

Este concepto se usará más adelante para calcular la probabilidad de supervivencia de un esquema ante el operador mutación.

A continuación se denotan otras definiciones:

λ, representa el tamaño de la población, la cual es constante a lo largo de las generaciones.

P_C, representa la probabilidad de cruce.

P_m, representa la probabilidad de mutación.

$\xi(S,t)$, representa el número de cadenas, que en una población particular, en el tiempo t, se asocian con el esquema S.

Para comprender de una mejor manera el último apartado, se supone la siguiente población, $P_t = \{(0101),(1111),(1101),(0100)\}$. Y los siguientes esquemas, $S_1 = (** 01)$, $S_2 = (*** 0)$.

Por lo tanto $\xi(S,t)$ es: $\xi_1\ (S_1,t) = 2$, $\xi_2(\ S_2,t) = 1$.

Como se verá en la ecuación de los esquemas, existe otra propiedad llamada evaluación en el tiempo, denotada por $eval(S,T)$ y definida como la media de la función objetivo del total de cadenas que en la población se asocian con el esquema S.

Ahora bien, sea $\overline{F(t)}$ la media de la función objetivo para toda la población durante el tiempo t. Esto quiere decir que $\overline{F(t)} = F(t)/\lambda$, donde $F(t)$ es la suma de las funciones objetivo de cada individuo en la población en el tiempo t.

Con las definiciones y notaciones introducidas hasta ahora, se puede formular el Teorema de los esquemas según [65].

"Si se denota por $E_{sel,\ cru,\ mut}(\xi(S,t+1))$, el número esperado de individuos que se asocian con el esquema S, en el tiempo t + 1, después de efectuar la selección, el cruce y la mutación en un Algoritmo genético canónico[12], se tiene:

$$E_{sel,\ cru,\ mut}(\xi(S,t+1)) \geq \xi(S,t).eval(S,t)/\overline{F(t)}[1 - P_C \frac{\delta(s)}{l-1} - o(S)P_m] \quad (1)$$

Para demostrar la anterior ecuación, se realizarán tres pasos, donde en un comienzo se analizará el efecto de la selección (usando como mecanismo la ruleta sesgada[13]), en el número esperado de individuos relacionados con la cadena S. Este número se denota por $E_{sel}(\xi(S,t+1))$. A partir de este número esperado se analizará el efecto de cruce, obteniendo $E_{sel,\ cru}(\xi(S,t+1))$, finalizando con el análisis del efecto mutación y así obtener la cota inferior para $E_{sel,\ cru,\ mut}(\xi(S,t+1))$.

- Efecto selección: La probabilidad de que un individuo escogido para cruzarse (usando el criterio de selección de la ruleta sesgada) se relacione con el esquema S, se calculará con $eval(S,t)/F(t)$.

  El número esperado de individuos que a partir del anterior individuo seleccionado, se relacionarán con S, se denota por $(eval(S,t)/F(t)).\ \xi(S,t)$. Al realizarse λ selecciones, se obtendrá:

$$E_{sel}(\xi(S,t+1)) = \lambda.\xi(S,t).eval(S,t)/F(t). \quad (2)$$

Puesto que $\overline{F(t)} = F(t)/\lambda$, se tiene:

$$E_{sel}(\xi(S,t+1)) = \xi(S,t).eval(S,t)/\overline{F(t)} \quad (3)$$

Si se supone en este caso que la evaluación del esquema supera a la evaluación ampliada a la totalidad de la población en un ε%, en otras palabras si:

$$eval(S,t) = \overline{F(t)} + \varepsilon\overline{F(t)}(\varepsilon > 0). \quad (4)$$

[12]Hace referencia, en general, a todo aquello que se ajusta a las reglas.

[13]Tipo de selección de individuos usada en algoritmos genéticos, en la que los cromosomas son elegidos al girar una rueda de ruleta que asigna a cada cromosoma o individuo un sector cuyo arco es proporcional a su valor obtenido en su función de aptitud.
http://books.google.com.co/books?id=U4pwetEPmpQC&pg=PA312&lpg=PA312&dq=ruleta+sesgada+algoritmos+geneticos&source=bl&ots=FhK-NS9_HD&sig=mXFhjfUGjHhRxiq3BY8iOKSYVKQ&hl=es&sa=X&ei=cOufUfWqCoPD0gGKh4HYDg&ved=0CC8Q6AEwAQ#v=onepage&q=ruleta%20sesgada%20algoritmos%20geneticos&f=false.

El resultado será

$$E_{sel}\big(\xi(S,t)\big) = \xi(S,0).(1+\varepsilon)^t \qquad (5)$$

La anterior ecuación denominada ecuación crecimiento reproductivo de los esquemas, según la definición dada en [65] muestra que: "el número medio esperado de individuos asociados con un esquema S en el tiempo t, crece de forma exponencial en función de la diferencia entre la evaluación media de dicho esquema con respecto a la evaluación media global".

- Efecto cruce: indica la probabilidad de que el operador cruce, destruya al esquema S. Se denota por $P_{d,cruce}(S)$.

En primer lugar:

$$P_{d,cruce}(S)=\frac{\delta(S)}{l-1} \qquad (6)$$

Sin embargo, el operador cruce se efectúa bajo una cierta probabilidad, por lo tanto la ecuación es:

$$P_{d,cruce}(S)=\frac{\delta(S)}{l-1}.P_c \qquad (7)$$

Por lo tanto, la probabilidad de que el esquema S, sobreviva al cruce es:

$$P_{so,\ cruce}(S) = 1 - P_{d,cruce}(S) = 1 - \frac{\delta(S)}{l-1}.P_c. \qquad (8)$$

No obstante, es pertinente aclarar que la ecuación anterior en realidad proporciona una cota alusiva a la probabilidad de que el esquema S sobreviva, porque es posible que éste no muera aunque el punto de corte se establezca entre los símbolos * ubicados entre el primer y último elemento distintos de *. El siguiente ejemplo lo explicará mejor:

Considerando los siguientes esquemas

$$S_1 = (0 * * 1),$$

$$S_2 = (0 * * 1),$$

El punto de corte es irrelevante ya que cualquiera de los esquemas resultantes, coincide con S_1 y S_2.

Por lo tanto la ecuación (9) se redefine como:

$$P_{so,\ cruce}(S) \geq 1 - \frac{\delta(S)}{l-1}.P_c. \qquad (9)$$

Obteniendo,

$$E_{sel,\ cru}\big(\xi(S,t+1)\big) \geq \xi(S,t).eval(S,T)/\overline{F(t)}(1 - \frac{\delta(S)}{l-1}.P_c\ ). \qquad (10)$$

- Efecto mutación: hace alusión a la probabilidad de que el esquema S, sobreviva al aplicarle el operador mutación. Se denota por $P_{so,\ mut}(S)$, definido como $P_{so,\ mut}(S)=(1-\ P_m\ )^{o(Q)}$. Puesto que la probabilidad de mutación es muy pequeña, es decir $P_m \ll 1$, la ecuación de la probabilidad de sobrevivir, puede expresarse como:

$$P_{so,\ mut} \approx 1 - o(Q)P_m. \qquad (11)$$

Combinando los tres efectos ejercidos sobre el esquema, selección, cruce y mutación, esto da origen al teorema de los esquemas:

$$E_{sel,\ cru,\ mut}\big(\xi(S,t+1)\big) \geq \xi(S,t).eval(S,t)/\overline{F(t)}[1-\frac{\delta(s)}{l-1}P_C-\ o(S)P_m+\frac{\delta(s)}{l-1}o(S)P_CP_m]. \qquad (12)$$

Si se suprime el último término se obtiene:

$$E_{sel,\ cru,\ mut}\big(\xi(S,t+1)\big) \geq \xi(S,t).eval(S,t)/\overline{F(t)}[1-\frac{\delta(s)}{l-1}P_C-\ o(S)P_m]. \qquad (13)$$

Esta ecuación indica el límite inferior del número esperado de individuos relacionados con el esquema S en la siguiente generación. La ecuación se encuentra expresada en función del número de individuos afines con el esquema en la actual generación, la longitud y el orden del esquema, las longitudes de las cadenas y las probabilidades de cruce y mutación.

En [66], se explica este teorema: “esquemas con una corta longitud, aptitud superior al promedio y bajo orden, tendrían un aumento exponencial de su participación en las siguientes generaciones de un Algoritmo Genético”.
Por otra parte, varios teoremas de convergencia se han desarrollado a lo largo de los años, los cuales avalan la tendencia de los Algoritmos Genéticos hacía el óptimo global con el paso de las generaciones. Es así como Günter Rudolph en 1994 [69] genera un modelo con el que demuestra la convergencia del GA simple hacía un óptimo global. Similarmente, Joe Suzuki [70] usando cadenas de Markov, demuestra la convergencia de un Algoritmo Genético Simple específicamente examinando los casos en donde la probabilidad de mutación es baja y los procesos de selección se realizan bajo tensión. El método y demostración usada por Gunar Liepins garantiza que en el espacio-solución exista un óptimo global para poblaciones a las cuales no se les aplique procesos de mutación y en donde se asegure que los mejores individuos siempre estén presentes durante el proceso genético, haciendo posible que cualquier individuo pueda ser alcanzado por medio del operador cruce. Similarmente, autores como T. E. Davis y J. C. Principe [64] A.E. Eiben [64], o U. K. Chakraborty, D.G. Dastidar [64] realizan estudios enfocados a demostrar la capacidad de los algoritmos genéticos para alcanzar la solución óptima dentro de la población con la cual se trabaja.