La investigación reportada en esta tesis es parte de los programas de investigación del CICESE (Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California).

La investigación fue financiada por el SECIHTI (Secretaría de Ciencia, Humanidades, Tecnología e Innovación).

# Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California


## Doctorado en Ciencias en Electrónica y Telecomunicaciones con orientación en Telecomunicaciones

# Enrutamiento inteligente orientado a la Calidad de Servicio en redes definidas por software

Tesis
para cubrir parcialmente los requisitos necesarios para obtener el grado de
Doctor en Ciencias

Presenta:

**José Enrique González Trejo**

Ensenada, Baja California, México
2025

Tesis defendida por

**José Enrique González Trejo**

y aprobada por el siguiente Comité

**Dr. Raúl Rivera Rodríguez**
Codirector de tesis

**Dr. Andrey Chernykh**
Codirector de tesis

Dr. José Eleno Lozano Rizk

Dr. Gabriel Alejandro Galaviz Mosqueda

Dr. Salvador Villarreal Reyes

Dr. José Luis González Compeán

Dra. María del Carmen Maya Sánchez
Coordinadora del Posgrado en Electrónica y Telecomunicaciones

**Dra. Ana Denise Re Araujo**
Directora de Estudios de Posgrado

Resumen de la tesis que presenta José Enrique González Trejo como requisito parcial para la obtención del grado de Doctor en Ciencias en Electrónica y Telecomunicaciones con orientación en Telecomunicaciones.

**Enrutamiento inteligente orientado a la Calidad de Servicio en redes definidas por software**

Resumen aprobado por:

**Dr. Raúl Rivera Rodríguez**
Codirector de tesis

**Dr. Andrey Chernykh**
Codirector de tesis

Las redes de telecomunicaciones enfrentan un crecimiento exponencial en el tráfico de datos, impulsado por aplicaciones de alto consumo como el streaming, las videoconferencias y los servicios en la nube. Esta creciente demanda ejerce una presión considerable sobre la infraestructura de red tradicional, la cual no fue diseñada para gestionar cargas de tráfico tan variables y exigentes. Esta situación genera problemas críticos de latencia, pérdida de paquetes y fluctuaciones en el retardo (jitter), afectando directamente la calidad de servicio (QoS) y comprometiendo los Acuerdos de Nivel de Servicio (SLA) de los proveedores de internet. Las Redes Definidas por Software (SDN) han surgido como una solución innovadora al separar el plano de control del plano de datos, permitiendo una administración centralizada y dinámica. Sin embargo, el enrutamiento en SDN sigue limitado por métodos tradicionales como OSPF (Open Shortest Path First), que selecciona rutas solo en función de la distancia, sin considerar aspectos como la congestión o el ancho de banda disponible, lo cual restringe la capacidad de SDN para optimizar los recursos en condiciones de alta demanda. Para abordar estas limitaciones, este trabajo propone dos métodos para mejorar el desempeño del enrutamiento: en primer lugar, el uso de algoritmos evolutivos, como MOEA/D y MOCell, que evalúan múltiples métricas de la red, tales como ancho de banda, latencia y pérdida de paquetes, permitiendo optimizar rutas que maximizan el rendimiento y minimizan los cuellos de botella; el segundo enfoque implementa de Redes Neuronales de Grafos (GNN) como un método adicional para mejorar la selección de rutas mediante su capacidad de modelado de la topología de la red, lo que permite anticipar problemas de congestión y ajustar dinámicamente las rutas. A través de GNN, se exploran mejoras en la rapidez de convergencia y en la adaptabilidad de la red, aprovechando sus capacidades predictivas para responder a demandas de QoS. La propuesta se validó en topologías multidominio simuladas en Mininet bajo diferentes condiciones de tráfico. Los resultados muestran que los algoritmos evolutivos alcanzan hasta un 50% de mejora en la transmisión de tráfico, además de reducir de manera significativa la pérdida de paquetes y optimizar la latencia en flujos críticos, en comparación con OSPF. A su vez, el enfoque con GNN muestra que su integración en la administración de rutas permite una mejora superior al 23% en precisión y eficiencia, destacándose como una opción viable para optimizar el rendimiento en redes SDN en escenarios de alta demanda. Estos hallazgos evidencian que la incorporación de algoritmos evolutivos y GNN, en el enrutamiento de redes SDN no solo mejora la eficiencia en la gestión del tráfico, sino que también establece un nuevo estándar de adaptabilidad y resiliencia en redes de alta demanda, asegurando una infraestructura más eficiente y orientada a satisfacer las necesidades de aplicaciones actuales y futuras.

**Palabras clave: Enrutamiento, Redes Definidas por Software, Optimización, Algoritmos Evolutivos, Redes Neuronales de Grafos**

Abstract of the thesis presented by José Enrique González Trejo as a partial requirement to obtain the Doctor of science degree in Electronics and Telecommunications with orientation in Telecommunucations.

**Quality of Service-oriented intelligent routing in software defined networks**

Abstract approved by:

**Dr. Raúl Rivera Rodríguez**
Thesis Co-director

**Dr. Andrey Chernykh**
Thesis Co-director

Telecommunication networks face exponential growth in data traffic, driven by high-consumption applications such as streaming, video conferencing, and cloud services, placing significant pressure on a traditional network infrastructure not designed to handle such fluctuating and demanding loads. This situation leads to critical issues like latency, packet loss, and jitter variability, directly affecting the Quality of Service (QoS) and compromising Service Level Agreements (SLA) of internet service providers. Software Defined Networking (SDN) has emerged as an innovative solution that decouples the control plane from the data plane, enabling centralized and dynamic management. However, SDN routing is still limited by traditional methods such as Open Shortest Path First (OSPF), which selects routes solely based on distance without considering factors like congestion or available bandwidth. This limitation restricts SDN's ability to optimize resources under high-demand conditions. This work proposes two methods to address these limitations and enhance routing performance. The first approach leverages evolutionary algorithms, such as MOEA/D and MOCell, which evaluate multiple network metrics such as bandwidth, latency, and packet loss, enabling route optimization that maximizes performance and minimizes bottlenecks. The second approach implements Graph Neural Networks (GNN) as an additional method to improve route selection through network topology modeling capabilities, which allow for congestion anticipation and dynamic route adjustments. Through GNNs, improvements in convergence speed and network adaptability are explored, utilizing their predictive capabilities to meet QoS demands. The proposed methods were validated in multi-domain topologies simulated in Mininet under diverse traffic conditions. The results show that evolutionary algorithms achieve up to a 50% improvement in traffic throughput while significantly reducing packet loss and optimizing latency in critical flows compared to OSPF. Additionally, the GNN-based approach demonstrates that integrating GNNs into route management leads to an improvement of over 23% in accuracy and efficiency, positioning it as a viable option for optimizing SDN network performance in high-demand scenarios. These findings highlight that incorporating evolutionary algorithms and GNNs in SDN routing not only enhances traffic management efficiency but also establishes a new standard for adaptability and resilience in high-demand networks, ensuring a more efficient infrastructure capable of meeting the needs of current and future applications.

**Keywords: Routing, Optimization, Software Designer Network, Evolutionary Algorithms, Graph Neuronal Network**

## Dedicatoria

Dedico este trabajo con profundo agradecimiento a mi familia, cuyo apoyo incondicional, paciencia y amor han sido mi mayor fuente de fortaleza a lo largo de este camino. Gracias por acompañarme e inspirarme a dar siempre lo mejor.

Este logro es también suyo.

# Agradecimientos

A la secretaría de Ciencia, Humanidades, Tecnología e Innovación (SECIHTI), con número de becario 765196, por el apoyo económico brindado durante los cuatro años de mi doctorado, así como al Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California (CICESE) por recibirme y permitirme realizar mis actividades de investigación.

A mis codirectores de tesis, el Dr. Raúl Rivera Rodríguez y el Dr. Andrey Chernykh, a quienes quiero expresar mi profundo agradecimiento por todo el apoyo y guía que me brindaron durante el desarrollo de mi tesis doctoral. Gracias por compartir sus conocimientos y valiosa experiencia, los cuales han contribuido en mi formación académica y profesional.

A mis sinodales, el Dr. José Eleno Lozano Rizk, el Dr. Gabriel Alejandro Galaviz Mosqueda, el Dr. Salvador Villarreal Reyes y el Dr. José Luis González Compeán, a quienes expreso mi gratitud por todo su apoyo y sus valiosas recomendaciones para el desarrollo de mi trabajo de investigación doctoral.

A mis amigos y compañeros, por haber compartido conmigo momentos de aprendizaje, intercambio de ideas y actividades durante mi estancia en el CICESE.

Por último, deseo expresar mi agradecimiento a mis padres, José Enrique y Claudia, a mis hermanos Oscar y Nallely y a mi familia Jassive y Sabina, quienes han estado a mi lado, brindándome su apoyo a lo largo de esta aventura y formación profesional.

# Tabla de contenido

Página

# Lista de figuras

Figura Página

# Lista de tablas

Tabla Página

# Capítulo 1. Introducción

En la actualidad, el crecimiento exponencial del tráfico de datos derivado de aplicaciones en la nube, servicios de streaming, videoconferencias y video juegos en línea ha generado una presión considerable sobre la infraestructura de red. Las redes, que originalmente fueron concedidas para operar volúmenes de tráfico más moderados, ahora enfrentan serias dificultades para adaptase a la naturaleza fluctuaciones del tráfico moderno y a los estrictos requerimientos de las aplicaciones emergentes (Xu et al., 2021). Estas limitaciones se traducen en aumentos de latencia, pérdidas de paquetes y variaciones en el retardo (jitter), perjudicando directamente la calidad de servicio (QoS por sus siglas en inglés Quality of Service) y compromete tanto los Acuerdos de Nivel de Servicio (SLA por sus siglas en inglés Service Level Agreement) de los Proveedores de Servicio de Internet (ISP por sus siglas en inglés Internet Service Provider) como la calidad de experiencia de los usuarios (QoE por sus siglas en inglés Quality of Experience) (García-Tadeo et al., 2022; Qazi et al., 2024). Estas circunstancias muestran la urgente necesidad de desarrollar redes más resilientes y flexibles, que puedan responder eficientemente a las demandas actuales y futuras de tráfico de datos.

A medida que las aplicaciones continúan evolucionando, los requerimientos de las redes se vuelven cada vez más estrictos, en términos de mayor consumo de ancho de banda y fiabilidad. Sin embargo, las redes aún continúan operando bajo políticas de mejor esfuerzo, lo que genera problemas de congestión, especialmente en escenarios de alta demanda y tráfico fluctuante (Medhi & Ramasamy, 2018; Hajjaji et al., 2021). Por lo tanto, el desafío actual radica en adaptar las tecnologías de red para ofrecer mayor flexibilidad y un rendimiento más eficiente.

En este panorama, las Redes Definidas por Software (SDN por sus siglas en inglés Software Defined Network) han surgido como una propuesta prometedora, ya que permiten separar el plano de control del plano de datos, facilitando la asignación dinámica de recursos y una gestión centralizada de la red (Manguri & Omer, 2022). Esta flexibilidad resulta clave para adaptarse a las crecientes demandas de tráfico y aplicaciones modernas, pero aún enfrenta desafíos importantes, particularmente en el área del enrutamiento. Muchos sistemas SDN continúan empleando algoritmos de enrutamiento tradicionales como OSPF (por sus siglas en inglés Open Shortest Path First), que seleccionan rutas en función de la menor distancia, sin considerar factores críticos como la congestión en los enlaces o el ancho de banda disponible. Estas limitaciones se hacen evidentes en escenarios de alta demanda, donde OSPF no puede distribuir el tráfico de manera eficiente ni priorizar flujos críticos, ya que opera bajo el principio del mejor esfuerzo

limitando las capacidades de SDN para optimizar los recursos de manera efectiva (Alsaeedi et al., 2019; Amin et al., 2021a).

Un ejemplo claro de estas limitaciones se observa cuando una aplicación requiere de una ruta con requerimientos específicos como se muestra en la Figura 1, donde una aplicación requiere una ruta con un ancho de banda de 80 Mbps y un retardo máximo de 7 ms. En este caso, OSPF selecciona la ruta R1, R2, R3, R7, que proporciona 70 Mbps y un retardo de 8 ms, lo que no cumple con los requisitos y puede causar problemas de rendimiento y congestión. En cambio, al seleccionar una ruta que considere el ancho de banda disponible, como R1, R2, R5, R6, R7, que ofrece 90 Mbps y un retardo de 6 ms, cumple con los requerimientos, mejorando significativamente el rendimiento del servicio. Este ejemplo destaca las limitaciones de los algoritmos de enrutamiento tradicionales en SDN y la necesidad de integrar enfoques más robustos de enrutamiento que consideren múltiples parámetros, como ancho de banda disponible y retardos, para que SDN pueda alcanzar un mayor potencial.

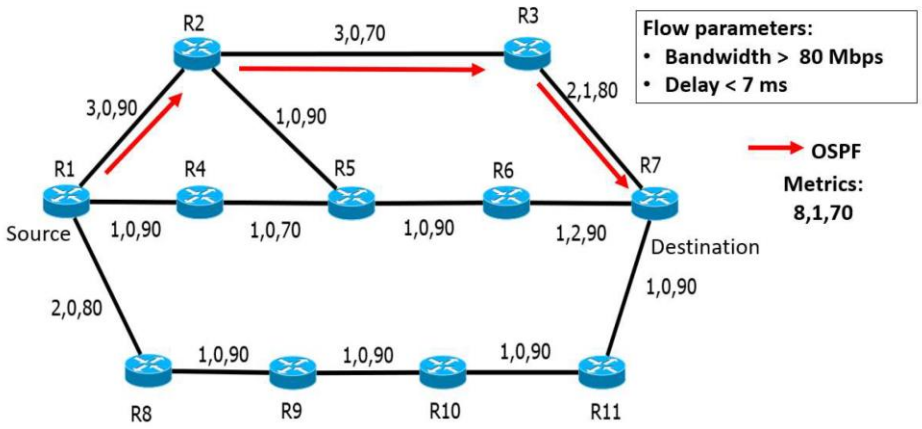


**Figura 1.** Problemas de enrutamiento. Métricas: retardo (ms), pérdida de paquetes (%) y ancho de banda (Mbps).

En este ámbito, la inteligencia artificial (IA por sus siglas en inglés Artificial Intelligence) se posiciona como una herramienta clave para superar las limitaciones de los enfoques tradicionales de enrutamiento, que no siempre son capaces de adaptarse a las crecientes demandas de tráfico ni garantizar QoS (Amin et al., 2021a). Este trabajo de investigación tiene como enfoque explorar una combinación de tecnologías de IA para mejorar la eficiencia y rendimiento de enrutamiento en redes SDN. Los algoritmos evolutivos, como los algoritmos genéticos, son particularmente relevantes en la optimización multi-objetivo, ya que permiten evaluar simultáneamente métricas de red como la disponibilidad de ancho de banda y la latencia. Estos algoritmos pueden seleccionar rutas que maximizan el rendimiento y minimizan los cuellos de botella, adaptándose a entornos con condiciones de tráfico cambiantes.

Además de los algoritmos evolutivos, el Aprendizaje por Refuerzo Profundo (DRL por sus siglas en inglés Deep Reinforcement Learning) juega un papel fundamental en este trabajo. DRL permite que los agentes de red aprendan de manera autónoma a seleccionar las mejores rutas mediante la interacción continua con el entorno. Sin embargo, DRL enfrenta desafíos como la necesidad de grandes volúmenes de datos de entrenamiento y la tendencia al sobreajuste, lo que puede limitar su capacidad de generalización (Casas-Velasco et al., 2021). Para superar estas limitaciones, se propone la integración de redes neuronales de grafos (GNN por sus siglas en inglés Graph Neural Network), que ofrecen una representación más precisa de las relaciones entre nodos y enlaces dentro de la red. Al combinar GNN con DRL, es posible mejorar la predicción de congestión y optimizar la selección de rutas, ajustando dinámicamente las decisiones de enrutamiento (He et al., 2024).

El objetivo de este trabajo es demostrar cómo la integración de agentes inteligentes basados en IA interactuando con el controlador SDN pueden mejorar significativamente el rendimiento de las redes, garantizando una distribución más eficiente de recursos y una mejora notable en la calidad de servicio a través del proceso de enrutamiento. Al combinar tecnologías avanzadas de optimización, como los algoritmos evolutivos, el Aprendizaje por Refuerzo Profundo y las redes neuronales de grafos, se pueden abordar con mayor eficiencia las crecientes exigencias del tráfico en las redes. Esta investigación no solo busca superar los obstáculos de los enfoques tradicionales de enrutamiento, sino también fomentar el desarrollo de soluciones innovadoras que ofrezcan mayor adaptabilidad y rendimiento en entornos con condiciones de alta demanda y tráfico dinámico.

## 1.1 Antecedentes

En los últimos años, la rápida expansión de tecnologías como 5G, el Internet de las Cosas (IoT por sus siglas en inglés Internet of Things) y Big Data, junto con el aumento constante de dispositivos conectados, ha provocado un incremento exponencial en el tráfico de datos, poniendo en evidencia las limitaciones de las redes tradicionales (Shukla et al., 2023). Estas redes, originalmente diseñadas para manejar volúmenes de tráfico más moderados, presentan dificultades para adaptarse a las exigencias de nuevas aplicaciones que requieren mayor ancho de banda y baja latencia, como el video streaming. Este incremento en la demanda de recursos ha resultado en problemas como una mayor latencia, pérdida de paquetes y, en algunos casos, congestión de la red, lo que afectando tanto el funcionamiento de las aplicaciones como el rendimiento de la infraestructura (Yang et al., 2022). En situaciones más críticas, esta sobrecarga puede llevar al colapso total de la red, ya que las arquitecturas convencionales, basadas en tablas de enrutamiento estáticas y del

principio de mejor esfuerzo, no son suficientes para gestionar las exigencias de QoS que requieren tanto las aplicaciones actuales como las futuras.

Para abordar estas limitaciones, las Redes Definidas por Software (SDN) han surgido como una solución prometedora para superar las limitaciones de las redes tradicionales. La principal innovación de SDN radica en la separación del plano de control y el plano de datos, lo que permite una mayor flexibilidad en la gestión de los flujos de tráfico (Gelberger et al., 2013). Mientras que las redes convencionales dependen de dispositivos autónomos (enrutadores y conmutadores) para tomar decisiones de enrutamiento, SDN centraliza estas decisiones en un controlador inteligente, lo que permite una visión global de la red. Sin embargo, a pesar de los avances que SDN representa, aún continúa dependiendo de algoritmos tradicionales de enrutamiento, como OSPF, que selecciona rutas basadas únicamente en la menor distancia y no considera otros parámetros críticos como la congestión, la latencia o el ancho de banda disponible. Esto significa que, en situaciones de alta demanda, las SDN pueden seguir experimentar problemas de rendimiento similares a los de las redes tradicionales, lo que implica la necesidad de implementar soluciones más avanzadas para optimizar el enrutamiento.

Uno de los mayores desafíos en el enrutamiento con calidad de servicio (QoS) en redes SDN es la gestión eficaz de los flujos de datos, que requieren evaluar múltiples parámetros simultáneamente, como la latencia, el ancho de banda y la pérdida de paquetes. Estos parámetros a menudo están en conflicto, lo que convierte el enrutamiento en un problema de optimización multi-objetivo (MOP por sus siglas en inglés Multi-Objective Optimization). Las soluciones tradicionales, como la programación lineal o la teoría de juegos, aunque son opciones útiles en algunos contextos, suelen ser demasiado intensivas en recursos computacionales y, a menudo, no ofrecen soluciones satisfactorias en escenarios dinámicos. Para superar estas limitaciones, los algoritmos heurísticos, y en particular los algoritmos evolutivos (EA por sus siglas en inglés Evolutionary Algorithm), han mostrado ser una solución eficiente, proporcionando resultados de alta calidad en tiempos razonables (Cui & Srivastava, 2022).

Los algoritmos evolutivos, como los algoritmos genéticos (GA por sus siglas en inglés Genetic Algorithm), han sido ampliamente utilizados para optimizar el enrutamiento en SDN. A diferencia de los enfoques convencionales que tienden a centrarse en una única métrica, los EA tienen la capacidad de manejar múltiples criterios simultáneamente, generando soluciones óptimas en entornos donde las condiciones de la red son cambiantes. Esta flexibilidad es fundamental para garantizar la QoS, ya que permite que los EA se adapten a métricas diversas y optimicen no solo la latencia, sino también la congestión y la pérdida de paquetes, factores cruciales para mejorar el rendimiento global de la red.

Dentro de los EA, los algoritmos genéticos se destacan por su capacidad para explorar amplios espacios de búsqueda y generar múltiples soluciones en una sola iteración. Basados en principios de selección natural, mutación y cruce, los GA son una herramienta ideal para la optimización de enrutamiento en redes SDN. Estudios como OpenQoS (Cui & Srivastava, 2022) y CECT (Tajiki et al., 2018) han demostrado cómo los GA pueden mejorar significativamente el rendimiento de las redes al seleccionar rutas que no solo tienen en cuenta la distancia, sino también otros factores críticos como el retardo y el ancho de banda disponible. Por ejemplo, OpenQoS prioriza el retardo para los flujos multimedia, mientras que CECT utiliza un algoritmo genético secundario para reasignar recursos en situaciones de congestión, optimizando el uso del ancho de banda.

Investigaciones recientes, como las de (D. Li et al., 2020) han implementado variantes avanzadas como los algoritmos evolutivos multi-objetivo, específicamente NSGA-II, para optimizar el enrutamiento teniendo en cuenta tanto el retardo como la pérdida de paquetes. Estos enfoques permiten no solo evitar cuellos de botella, sino también mejorar la experiencia del usuario mediante una gestión más eficiente de los recursos de la red.

Los algoritmos evolutivos ofrecen ventajas significativas sobre los enfoques tradicionales en la optimización del enrutamiento en redes SDN. Su capacidad para generar soluciones óptimas en entornos de alta demanda y su flexibilidad para adaptarse a condiciones de red cambiantes, los convierten en una herramienta clave para mejorar la QoS y garantizar un rendimiento eficiente.

Dentro del ámbito de la inteligencia artificial, los métodos de Aprendizaje por Refuerzo Profundo (DRL) han surgido como una solución eficiente para abordar los problemas de congestión y optimización de enrutamiento en redes SDN. Tradicionalmente, las redes han sido gestionadas mediante enfoques estáticos, donde las rutas de tráfico se definen con reglas preestablecidas. Sin embargo, estos enfoques no permiten una adaptación dinámica a las condiciones cambiantes del tráfico, lo que resulta insuficiente frente al crecimiento de tráfico de datos como el de la infraestructura de red (Amin et al., 2021b).

Es aquí donde los algoritmos de DRL presentan una ventaja significativa. DRL permite que los agentes de red aprendan directamente del entorno, ajustando las rutas de enrutamiento en función de las condiciones actuales de la red. A través de la retroalimentación continua, los agentes optimizan el uso de los recursos, adaptándose a las fluctuaciones del tráfico y mejorando el rendimiento de la red (Wumian et al., 2024). A diferencia de los enfoques tradicionales, DRL ofrece una capacidad reactiva y adaptativa, lo que lo convierte en una herramienta poderosa en la gestión de redes SDN (Dhandapani et al., 2024).

Estudios como (L. Zhang et al., 2022), utilizando el algoritmo Deep Deterministic Policy Gradient (DSOQR), han demostrado cómo DRL puede mejorar la flexibilidad y eficiencia en la selección de rutas, permitiendo la gestión dinámica de las mismas. Sin embargo, también se han identificado problemas relacionados con la escalabilidad y el sobreajuste, donde el modelo se especializa en ciertos contextos, lo que reduce su capacidad para adaptarse a nuevas situaciones o escenarios de tráfico dinámico.

Para superar estas limitaciones, los investigadores han propuesto las Graph Neural Networks (GNN), que pueden modelar la topología de red con gran precisión al capturar las interrelaciones entre los nodos y enlaces. Esta capacidad permite a los sistemas considerar múltiples factores críticos, como el ancho de banda disponible, para distribuir el tráfico de manera más eficiente. Estudios como los de (C. Zhang et al., 2019) y (Chen et al., 2020) han demostrado que la integración de GNN con DRL no solo mejora significativamente el rendimiento de la red, sino que también resuelve problemas de escalabilidad y optimiza la asignación de recursos. Además, esta combinación ayuda a reducir la latencia y aumentar el uso del ancho de banda, mejorando la calidad del servicio (QoS) (J. Li et al., 2020, Kim et al., 2021).

En resumen, aunque las redes SDN han representado un avance considerable frente a las redes tradicionales, aún existen limitaciones que deben abordarse para satisfacer las crecientes demandas de tráfico y garantizar una QoS adecuada. La integración de inteligencia artificial, particularmente mediante DRL y GNN, ofrece un camino prometedor para mejorar el enrutamiento y la gestión de recursos en redes SDN. Este trabajo de investigación se centra en explorar cómo la integración de estas tecnologías puede crear un agente inteligente que optimice el enrutamiento, proporcionando una mayor adaptabilidad, eficiencia y rendimiento en entornos de alta demanda y tráfico variable. Con estas innovaciones, se espera superar las limitaciones actuales y avanzar hacia redes más resilientes y eficientes.

## 1.2 Planteamiento del problema

El enrutamiento es un aspecto clave en las redes de telecomunicaciones, ya que asegura la transmisión de datos entre los nodos de la red. Sin embargo, los métodos tradicionales de enrutamiento se han centrado únicamente en determinar la ruta más corta y transmitir datos, sin contemplar factores críticos como la congestión, la latencia o los requerimientos específicos de las aplicaciones que requieren una Calidad de Servicio (QoS) diferenciada. Este enfoque limitado genera ineficiencias en la gestión del tráfico, en particular en redes con alto nivel de heterogeneidad y aplicaciones críticas como videoconferencias o transmisión en tiempo real (Lin et al., 2017; Ghosh et al., 2024).

El principal problema radica en que estos métodos tradicionales no se adaptan dinámicamente a las condiciones cambiantes del tráfico. Tampoco tienen en cuenta la sobreasignación de rutas o las diferentes exigencias de QoS, lo que puede conducir a problemas de congestión y una distribución ineficiente del tráfico (Let et al., 2023). Esto no solo afecta la experiencia del usuario final, sino que también comprometen la eficiencia global de la red.

Ante el crecimiento exponencial del tráfico de datos y la proliferación de aplicaciones que requieren baja latencia y alta fiabilidad, las redes actuales se enfrentan a retos cada vez más complejos. Las redes Definidas por Software (SDN) han traído consigo una serie de avances al permitir una gestión centralizada y una visión global de la red, proporcionando información valiosa sobre el estado de los nodos y enlaces. Pero a pesar de esto, los métodos de enrutamiento en SDN continúan basándose en enfoques tradicionales de mejor esfuerzo, lo que limita su capacidad de adaptarse a las fluctuaciones dinámicas del tráfico de red.

La incapacidad de los métodos de enrutamiento tradicionales para adaptarse a las condiciones cambiantes del tráfico de red resalta la necesidad urgente de implementar algoritmos que gestionen de manera más eficiente los recursos y garanticen una Calidad de Servicio (QoS) adecuada. Esto plantea la importancia de desarrollar enfoques de enrutamiento más dinámicos, capaces de ajustar el tráfico, mejorando el uso de los recursos de red y ofreciendo una mejor experiencia del usuario. Esta mejora es especialmente crucial para aplicaciones críticas, donde la QoS es esencial para garantizar su rendimiento como fiabilidad, incluso en escenarios de alta demanda.

## 1.3 Motivación

A pesar de los avances que las Redes Definidas por Software (SDN) han aportado a la gestión centralizada, los métodos de enrutamiento continúan basándose en enfoques de mejor esfuerzo, lo que limita significativamente su capacidad cd adaptación a las condiciones dinámicas del tráfico. Estos enfoques, diseñados para redes más estáticas, no abordar adecuadamente problemas críticos como la congestión ni garantizan la Calidad de Servicio (QoS) requerida para aplicaciones sensibles, como videoconferencias y transmisión en tiempo real. Como resultado, se genera una sobrecarga en ciertos enlaces, aumentando la latencia y afectando negativamente tanto la experiencia del usuario como la fiabilidad de los servicios (Elbasheer et al., 2021; Henni et al., 2020).

Esta falta de adaptabilidad de los métodos de enrutamiento tradicionales resalta la urgente necesidad de implementar algoritmos capaces de gestionar eficientemente los recursos y de asegurar una QoS consistente. Con el incremento del tráfico de datos y las exigencias cada vez más estrictas de las aplicaciones, es crucial desarrollar soluciones que no solo respondan a las fluctuaciones del tráfico, sino que también prioricen la estabilidad y el rendimiento de la red.

En este sentido, la implementación de algoritmos de enrutamiento multi-objetivo representa una oportunidad clave. Estos algoritmos buscan optimizar múltiples factores, tales como la latencia, el ancho de banda y la congestión, mejorando el desempeño general de la red. Este trabajo de investigación se enfoca en abordar estas limitaciones, explorando la integración de algoritmos adaptativos que no solo mejoren la eficiencia del tráfico y minimicen la congestión, sino que también garanticen una conectividad más estable y confiable, con el objetivo de asegurar una QoS apropiada para las aplicaciones.

## 1.4 Justificación

Al aprovechar las capacidades de las Redes Definidas por Software (SDN), como la gestión centralizada y la visión global del estado de la red, se cuenta con la capacidad de integrar nuevos algoritmos para mejorar el enrutamiento y optimizar los recursos disponibles. SDN proporciona información detallada sobre el estado de los nodos y enlaces, lo que permite tomar decisiones basadas en el estado actual de la red. Al combinar esta información con algoritmos de inteligencia artificial, como los algoritmos genéticos y las Graph Neural Networks (GNN), se puede mejorar la gestión del tráfico y los recursos de red.

El valor de este enfoque radica en la capacidad de SDN para monitorear constantemente el tráfico y ajustar dinámicamente las rutas según las condiciones actuales. Los algoritmos de IA, en particular los algoritmos genéticos, pueden aprovechar la vista global de la red proporcionada por SDN para identificar rutas óptimas que distribuyan el tráfico de manera eficiente. Esto evita la sobrecarga en enlaces críticos y mitiga los problemas de congestión, lo que es fundamental para mejorar el rendimiento general de la red. La naturaleza iterativa de estos algoritmos les permite adaptarse a las fluctuaciones del tráfico en tiempo real, brindando una respuesta ágil y eficiente en situaciones de alta demanda.

Por otro lado, la integración de GNN con Aprendizaje por Refuerzo Profundo (DRL) incrementa aún más las capacidades de las redes SDN. Las GNN permiten un modelado preciso de la topología de la red, capturando las interacciones entre nodos y enlaces para ofrecer una visión más detallada de la estructura

y el estado del tráfico. Esto permite que los agentes de DRL ajusten dinámicamente las rutas de enrutamiento, anticipándose a problemas como la congestión antes de que ocurran. Este enfoque no solo optimiza la utilización de los recursos, sino que también mejora la calidad del servicio (QoS), al reducir la latencia, aumentando la disponibilidad del ancho de banda y garantizar un rendimiento estable en aplicaciones críticas que requieren alta fiabilidad y baja latencia. La sinergia entre las capacidades centralizadas de SDN y los algoritmos de inteligencia artificial ofrece una solución integral para la gestión del tráfico y el enrutamiento, abordando los problemas de congestión y brindando una QoS consistente, lo que se traduce en una mejor experiencia para el usuario final.

## 1.5 Objetivos

### 1.5.1 Objetivo general

Desarrollar un algoritmo de enrutamiento inteligente para redes definidas por software (SDN) que garantice la calidad de servicio (QoS) y administre eficientemente los recursos de la red, reduciendo los problemas de congestión.

### 1.5.2 Objetivos específicos

- Analizar las limitaciones de los métodos de enrutamiento basados en QoS en redes SDN bajo diferentes niveles de congestión.

- Aplicar métodos de optimización multi-objetivo al proceso de enrutamiento en redes SDN, utilizando métricas como: ancho de banda disponible, retardos y pérdida de paquetes.

- Desarrollar un método de enrutamiento basado en Graph Neural Networks (GNN) para predecir congestiones y ajustar rutas de forma dinámica.

- Generar escenarios de congestión y bases de datos de tráfico en redes SDN, para evaluar tanto los métodos tradicionales como los propuestos.

## 1.6 Organización de la tesis

El presente trabajo de investigación se organiza de la siguiente manera: en el capítulo 2, se describe detalladamente los conceptos fundamentales relacionados con las Redes Definidas por Software (SDN) y sus características clave para garantizar la calidad de servicio (QoS). El capítulo 3 profundiza en los algoritmos multi-objetivo, explicando su relevancia y el papel de la inteligencia artificial en la resolución de problemas de enrutamiento. En el capítulo 4, se brinda una visión general de cómo las redes neuronales han evolucionado para adaptarse a la creciente necesidad de modelar relaciones complejas en los datos, pasando de arquitecturas convencionales a GNN que capturan de manera más efectiva las interacciones entre nodos y enlaces dentro de las redes. Se resalta el impacto que estas arquitecturas tienen en la mejora del enrutamiento en SDN, mientras que el capítulo 5 describe la metodología empleada para resolver el problema propuesto, centrándose en la implementación de un algoritmo evolutivo por descomposición, el algoritmo genético MOCell y las redes neuronales de grafos para optimizar el enrutamiento y la QoS en SDN. En el capítulo 6 se presentan los resultados obtenidos de la implementación de los algoritmos mencionados y las simulaciones realizadas. Finalmente, el capítulo 6 expone las conclusiones, resaltando las ventajas de estos enfoques en comparación con los métodos tradicionales de enrutamiento en SDN, así como recomendaciones y posibles líneas de investigación futura en este ámbito.

# Capítulo 2. Redes Definidas por Software (SDN)

Con el crecimiento acelerado y sin precedentes de la demanda de tráfico de datos, impulsado por el auge de tecnologías como el internet de las cosas (IoT) y el aumento de dispositivos interconectados, han evidenciado las limitaciones inherentes de la arquitectura convencional de internet. Estas nuevas exigencias han vuelto cada vez más complejas y especializadas las tareas de control y gestión de dispositivos, así como la administración del flujo de datos (Cheng et al., 2024; Lozano-Rizk et al., 2020).

En sus orígenes, la infraestructura de red fue concebida para manejar tráfico textual y como consecuencia, ha demostrado ser rígida y estática frente a los nuevos desafíos de aplicaciones interactivas y flujos multimedia, los cuales cada vez son más sensibles a retrasos y demandan un alto consumo de ancho de banda. En este tipo de arquitecturas, donde la lógica de control es distribuida, para resolver un problema o ajustar una política específica requiere la intervención manual en los dispositivos afectados, lo que aumenta la posibilidad de cometer errores humanos y afecta negativamente en la escalabilidad y fiabilidad de la red (Guck et al., 2018). Estos factores limitan la capacidad de las redes tradicionales para ofrecer el nivel adecuado de Calidad de Servicio (QoS) y, aún más crítico, de Calidad de Experiencia (QoE) para el usuario final. Mientras que la QoS está orientada a métricas técnicas como el ancho de banda y la latencia, la QoE se enfoca en la satisfacción del usuario, lo que introduce nuevos retos para las redes modernas.

En respuesta a estas necesidades tecnológicas, se han planteado diversas investigaciones que destacan, la importancia de automatizar la infraestructura de red para afrontar los requerimientos actuales, incluidas las demás de IoT (Alsaeedi et al., 2019, Filipe et al., 2020). Este interés ha conducido a la formulación de estrategias automáticas y adaptativas capaces de mitigar problemas de escalabilidad y disponibilidad en el tráfico en tiempo real, y así garantizar los niveles deseados de QoS. Sin embargo, alcanzar los estándares de QoS y QoE implica un proceso de optimización altamente complejo, el cual con frecuencia se clasifica como un problema NP-completo (Tache et al., 2024; Vesselinova et al., 2020a). Gestionar de manera efectiva el tráfico en tiempo real y asegurar el rendimiento de las aplicaciones sensibles a la latencia se vuelve, por tanto, cada vez más difícil conforme aumentan el volumen de datos y la dimensionalidad de las redes.

Ante estos retos, la infraestructura de red ha comenzado a evolucionar hacia un enfoque más automatizado y adaptable, como lo sugieren diversos estudios(Alidadi et al., 2022; Egilmez et al., 2012; Sun et al., 2021). En lugar de optar por un rediseño radical de las redes tradicionales, se han propuesto

soluciones graduales que permitan adaptaciones sin causar interrupciones significativas en las operaciones actuales. Una de estas alternativas es la introducción de redes superpuestas, que añaden una capa adicional de control sobre la red convencional (Xie et al., 2013). Sin embargo, este enfoque tiende a aumentar la complejidad de las redes debido a las capas virtuales adicionales, lo que puede generar nuevos desafíos.

Como alternativa, el paradigma de redes definidas por software (SDN) ofrece la capacidad de programar la red, facilitando así la implementación de métodos de control automático y adaptativo mediante la separación del plano de datos (hardware) del plano lógico (software), permitiendo una evolución independiente (Bannour et al., 2018; Gelberger et al., 2013). El objetivo principal de SDN es la centralización del control de la red, ofreciendo mayor visibilidad y flexibilidad para la gestión de la red y optimización de su desempeño. En contraste con otras opciones como redes superpuestas, SDN posee la capacidad de gestionar la totalidad de la red, no solo un conjunto seleccionado de nodos y ahorrar la tarea complicada de crear servicios temporales como lo realizan las redes superpuestas. Ofreciendo un marco de programación inherente para alojar aplicaciones de control y gestión que se desarrollan de forma centralizada.

## 2.1 Estructura de SDN

A diferencia de la arquitectura descentralizada de control que caracteriza a la arquitectura tradicional de internet, basada en una ardua interacción de protocolos en capas y soluciones integradas verticalmente, el paradigma de Redes Definidas por Software (SDN) propone separar la lógica de control del hardware y centralizar la toma de decisiones en controladores basados en software. Este principio es clave para promover nuevas oportunidades de innovación en la gestión de la red, al permitir la incorporación de un control automático y adaptativo, lo cual simplifica la operatividad de la red.

La visión principal de SDN es simplificar la operatividad de las redes, optimizar su gestión y proporcionar una mayor flexibilidad en comparación de las arquitecturas de red heredadas. De acuerdo con la visión de (Feamster et al., 2014), existen cuatro problemas claves que afectan en la gestión de redes actuales:

1. Configuración de red compleja y de bajo nivel: La configuración manual y distribuida de los dispositivos de red continúa siendo una tarea complicada y propensa a errores, especialmente cuando las normas de configuración son específicas de cada proveedor.

2. Estado dinámico de la red: Con el crecimiento de tecnologías como 5G, IoT y la computación en la nube, las redes se han vuelto más dinámicas y complejas. Los patrones de tráfico y las condiciones de la red cambian rápidamente debido a la movilidad y la creciente demanda de servicios.

3. Complejidad en la gestión: Muchas funciones de control y gestión están embebidas en el hardware de red, lo que dificulta la administración debido a interfaces descentralizadas y a bajo nivel, agravando el problema con el aumento de la escala de las redes.

4. Dispositivos heterogéneos: La gran variedad de dispositivos de red, cada uno con configuraciones propietarias, añade una capa de complejidad e ineficiencia a la gestión, ya que cada uno de estos dispositivos debe configurarse y gestionarse de forma individual.

En consecuencia, las infraestructuras de red heterogéneas, caracterizadas por su rigidez y limitada flexibilidad, no están bien adaptadas a los entornos dinámicos actuales, lo que complica garantizar la Calidad de Servicio (QoS). Ante esta situación, es crucial implementar políticas de alto nivel y automatizar la configuración de la red para simplificar su gestión y reducir la carga de trabajo.

La propuesta de SDN, liderada por la Open Networking Foundation (ONF), propone una arquitectura abierta con el objetivo de enfrentar los retos contemporáneos mediante la automatización de configuraciones y la programación flexible de la red. A diferencia de la arquitectura convencional (Figura 2 a) en la que los dispositivos de red se encuentran cerrados e integrados verticalmente, agrupando software con hardware, la arquitectura de SDN (Figura 2 b) abstrae la capa de datos de la de control. En este enfoque, los dispositivos de red se convierten en simples dispositivos de reenvió, mientras que toda la lógica de control se centraliza en controladores de software. Esto proporciona un esquema de programación adaptativa que facilita el desarrollo de aplicaciones especializadas y la implementación de nuevos servicios (Son & Buyya, 2018).

Con estos atributos, SDN posibilita una personalización y gestión más eficiente de la red, basada en políticas de alto nivel definidas como programas centralizados. SDN no solo supera las limitaciones inherentes de la arquitectura tradicional, sino que también establece una base robusta para la implementación eficiente de políticas de red y la adaptación a las cambiantes demandas del entorno tecnológico, haciendo la gestión de la infraestructura de red más ágil y eficaz.

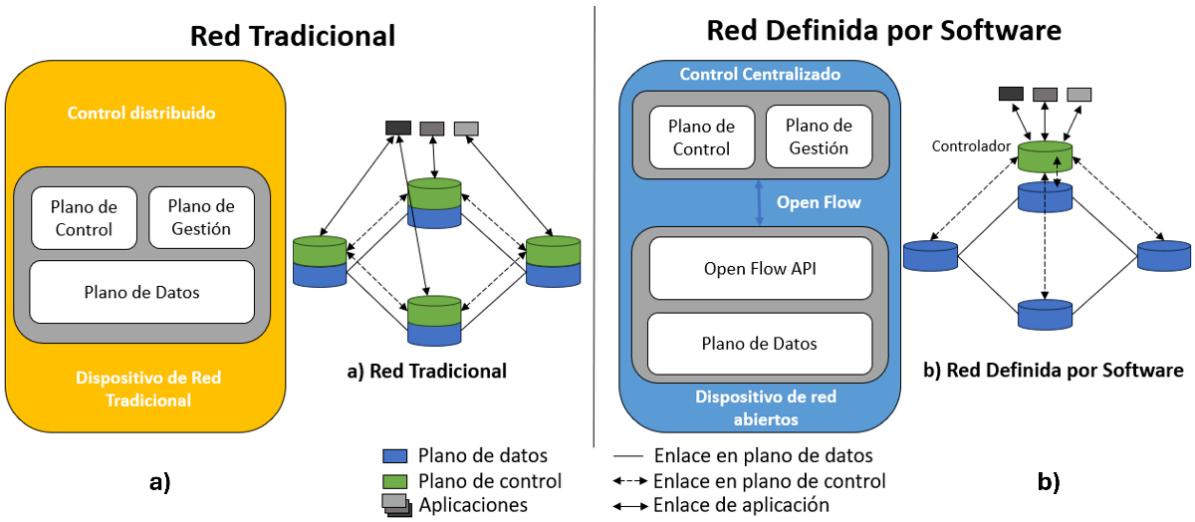


**Figura 2.** Arquitecturas de redes: a) Red Tradicional b) Red Definida por Software (SDN).

## 2.2 Arquitectura de SDN

La arquitectura de SDN se organiza verticalmente en tres capas (Figura 3): plano de datos, plano de control y controlador.

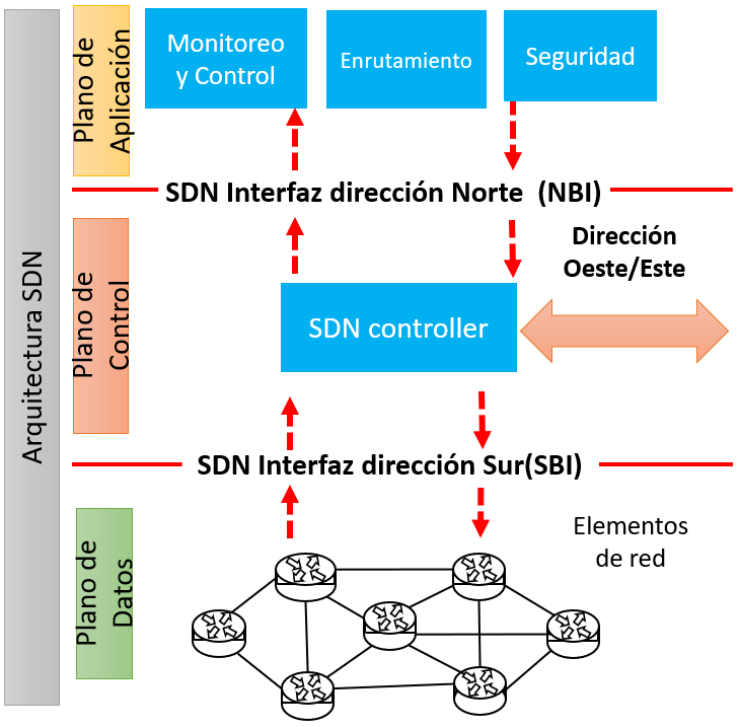


**Figura 3.** Arquitectura de SDN.

### 2.2.1 Plano de datos

El plano de datos, también conocido como plano de reenvío, está compuesto por los dispositivos de red encargados de la retransmisión de paquetes, tales como enrutadores y conmutadores. La separación de este plano respecto al plano de control permite que los dispositivos sean accesibles de manera remota a través de una interfaz abierta conocida como interfaz hacia el sur, independiente al proveedor de los dispositivos.

Para facilitar la comunicación entre el plano de control y el plano de datos, se emplean protocolos como OpenFlow (Kobayashi et al., 2014) y ForCES (Forwarding and Control Element Separation) (Kodera et al., 2021). De estos, Open Flow se ha convertido en la tecnología más adoptada por los proveedores de servicios de internet. Ambos protocolos siguen el principio básico de dividir el plano de control y el plano de datos, estandarizando la comunicación entre ellos.

OpenFlow permite que el plano de control defina de manera centralizada el comportamiento de reenvío en el plano de datos. Las decisiones de reenvío de tráfico se reflejan en políticas específicas que los dispositivos openflow traducen en reglas de reenvío de paquetes transformándolas en tablas de flujos. De acuerdo con los estándares definidos por la ONF en 2009, un dispositivo OpenFlow (Figura 4) incluye al menos una tabla de flujo y un canal seguro de comunicación con el controlador. La tabla de reenvío contiene una lista de entradas de flujo, cada una con campos coincidentes que determinan cómo deben procesarse los paquetes. Además, mantiene contadores que recopilan estadísticas sobre los flujos y acciones específicas que se ejecutan cunado un paquete coincide con una entrada en la tabla (Hu et al., 2014a; Lara et al., 2014a).

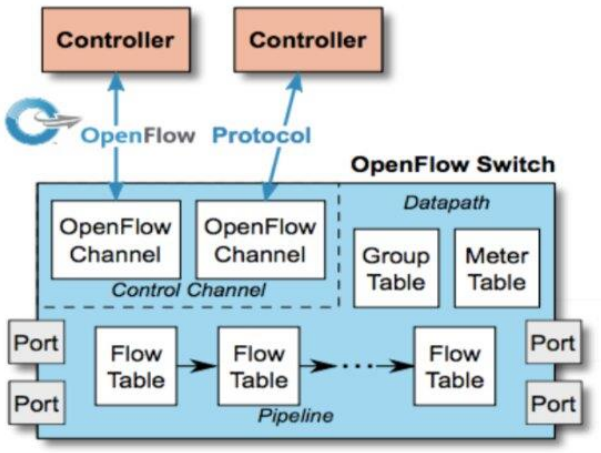


**Figura 4.** Estructura de conmutador OpenFlow.

Cuando un paquete ingresa al conmutador, este se compara con las entradas de la tabla de flujos siguiendo un orden de prioridad establecido por el controlador. Si se encuentra una coincidencia, se activan las reglas del flujo correspondiente, aplicándose las acciones asociadas al paquete. En caso de no encontrar coincidencias, el paquete es enviado al controlador a través del canal seguro, para que este determine cómo debe procesarse. Las posibles acciones incluyen definir un nuevo flujo insertando entradas en la tabla de flujos, descartar el paquete, o modificar su encabezado. Este enfoque permite una gestión centralizada y programable de la red, mejorando su eficiencia y capacidad de adaptación.

Dada la creciente adopción del protocolo OpenFlow, se han desarrollo múltiples versiones, cada una con mejoras distintivas para incrementar su flexibilidad y funcionalidad. La Tabla 1 resume las características clave de las versiones más relevantes de OpenFlow desarrolladas hasta el momento (Hu et al., 2014b; Lara et al., 2014b; Miguel-Alonso, 2023).

**Tabla 1.** Versiones OpenFlow.

| Versión | Características |
|---|---|
| 1.1 | Múltiples tablas de flujo, tablas de grupo y colas, soporte de MPLS y VLAN. |
| 1.2 | Soporte para múltiples controladores y IPv6. |
| 1.3 | Cabecera extendida de IPv6, soporte de indicadores adicionales (QoS) y soporte de túneles independientes para flujos. |
| 1.4 | Mecanismos de sincronización de tablas de flujo, agrupación de mensajes y gestión de puertos ópticos. |
| 1.5 | Identificador de paquetes de datos, tablas de salida e incorporación de banderas TCP. |

### 2.2.2 Plano de control

El plano de control es considerado el núcleo o cerebro de las redes SDN, siendo la entidad de construcción esencial en SDN. Este plano está compuesto por un controlador de software centralizado encargado de gestionar la comunicación entre las aplicaciones de red mediante la interfaz norte y los dispositivos mediante la interfaz sur. Su función principal es traducir los requisitos provenientes de la capa de aplicación hacia los dispositivos del plano de datos, así como proporcionar información relevante a las aplicaciones SDN para la toma de decisiones en la red.

Por otra parte, la capa de control SDN, conocida también como sistema operativo de red (NOS, por sus siglas en inglés), desempeña un papel fundamental al soportar la lógica de control de la red. esta capa

ofrece a las aplicaciones una visión global de la infraestructura de la red, lo que permite definir políticas de red sin la necesidad de detallar aspectos específicos de su implementación.

Para facilitar la interconexión de múltiples controladores SDN, principalmente de diversos proveedores de servicio de internet, se distinguen principalmente dos categorías de arquitecturas de control SDN distribuidas, basadas en la organización física de los controladores de acuerdo con la Figura 5: la arquitectura plana y una arquitectura de jerárquica. Ambas arquitecturas dependen de una interfaz de comunicación este-oeste, que facilita el intercambio de información entre múltiples controladores SDN, permitiendo una coordinación eficiente y una gestión centralizada de la infraestructura de red, incluso en entornos heterogéneos (Almadani et al., 2021).

En la estructura plana del plano de control, la red se segmenta horizontalmente en múltiples áreas, cada una administrada por un único controlador que gestiona un subconjunto de conmutadores openflow. Este enfoque tiene varias ventajas, como la reducción de la latencia en el control y una mayor resiliencia debido a la independencia de los controladores en cada área. Esta distribución horizontal permite que los controladores respondan de manera más eficiente a los cambios locales en la red. En cambio, la arquitectura jerárquica organiza el plano de control en múltiples niveles, cada uno dedicado a diferentes servicios o funcionalidades. Según los estudios de Liu et (Khan et al., 2023; Mekky et al., 2014), esta estructura jerárquica puede mejorar tanto la escalabilidad como el rendimiento de la red SDN, permitiendo una mejor coordinación entre los controladores y una gestión más eficiente de la red en grandes infraestructuras.

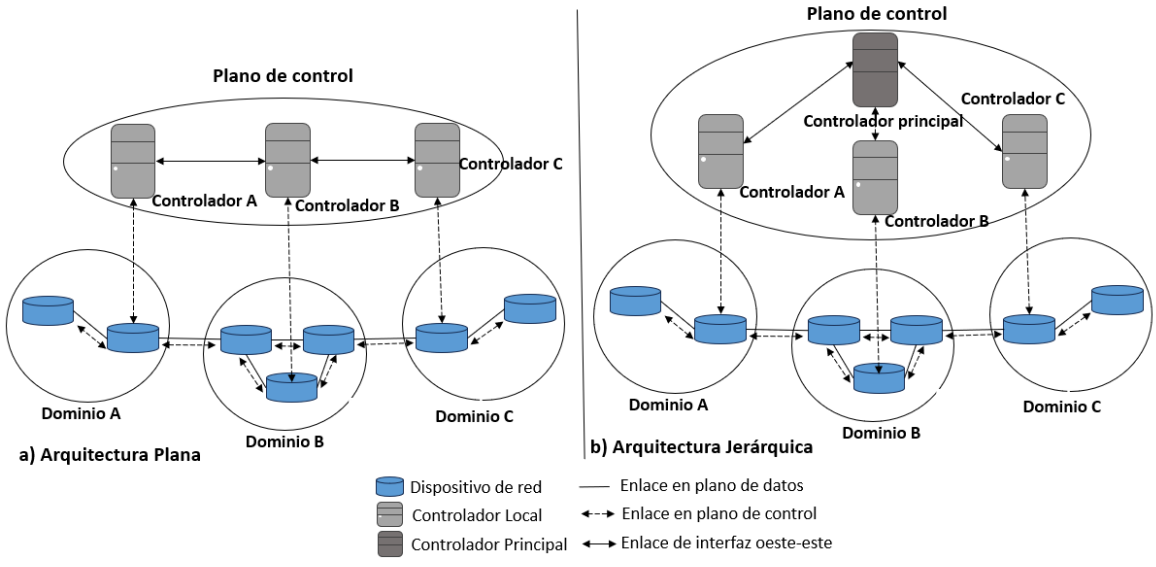


**Figura 5.** Arquitectura de controlador: a) Plana. b) Jerárquica.

La Tabla 2 presenta los principales controladores empleados tanto en ámbitos académicos como industriales, mostrando los lenguajes de programación en los que se encuentran implementados y su arquitectura (Badotra & Singh, 2017; RYU Project Team, 2014; Zhu et al., 2019).

**Tabla 2.** Controladores SDN.

| Controlador | Implementación | Arquitectura |
|---|---|---|
| Beacon | Java | Plana |
| NOX | C++ | Plana |
| POX | Python | Plana |
| Ryu | Python | Plana |
| OpenDayLight | Java | Jerárquica |
| FloodLight | Java | Plana |
| ONOS | Java | Jerárquica |
| ONIX | C++ | Jerárquica |
| Kandoo | C++ | Jerárquica |
| OpenContrail | C, C++, Python | Plana |

### 2.2.3 Plano de aplicación

En la arquitectura de SDN, el plano de aplicación abarca los programas que implementan la lógica y estrategias de control de la red. Este plano interactúa con el plano de control a través de una de una API abierta en dirección norte. Las aplicaciones en SDN transmiten sus requisitos de red al controlador, que los traduce en comandos específicos hacia la interfaz sur, utilizando reglas de reenvío que determinan el comportamiento de los dispositivos en el plano de datos. Entre las aplicaciones más comunes en SDN se incluyen enrutamiento, ingeniería de tráfico (TE por sus siglas en inglés Trafic Engineering), firewalls y balanceo de carga.

Una de las mayores ventajas de SDN es la independencia entre la lógica de las aplicaciones y el hardware de la red, lo que permite definir políticas y objetivos sin estar restringidos por la infraestructura subyacente. A su vez, las aplicaciones de SDN aprovechan la vista global de la red proporcionada a través de la interfaz en dirección norte para construir servicios y funciones de red proporcionados por el plano de control, de acuerdo con sus propósitos.

La API en dirección norte resulta clave para la programabilidad de la red, pues simplifica las tareas de control y gestión, además de fomentar la innovación. A pesar de que esta API no está estandarizada, se clasifica principalmente en dos categorías. El primer grupo incorpora APIs simples y primitivas que están directamente vinculadas a los servicios internos del controlador.

- Api ad hoc de bajo nivel: permiten a los desarrolladores implementar aplicaciones directamente en el controlador, dependiendo del lenguaje nativo del controlador, como NOX en C++ y POX en Python.

- API basado en servicios web: permite que aplicaciones externas accedan a las funciones y servicios del controlador mediante API como REST, siendo ampliamente utilizadas por su compatibilidad con múltiples lenguajes de programación.

Una segunda categoría incluye APIs de nivel superior, basadas en lenguajes de programación específico, como Frenetic (Kulkarni et al., 2021), Procera (Voellmy et al., 2013) y Pyretic (Reich et al., 2013). Estas ofrecen mayor flexibilidad para el desarrollo de aplicaciones y la especificación de políticas de red, ya que su interacción con el controlador se realiza de manera indirecta.

## 2.3 Calidad de Servicio (QoS)

De acuerdo con la Unión Internacional de Telecomunicaciones (ITU por sus siglas en inglés International Telecommunication Union), la Calidad de Servicio en el ámbito de las telecomunicaciones se define como "la totalidad de las características de un servicio de telecomunicaciones que determinan su capacidad para satisfacer las necesidades explícitas e implícitas del usuario o servicio" (Ibarrola et al., 2010).

Los administradores de red emplean la Calidad de Servicio (QoS) para garantizar el rendimiento de aplicaciones que requieren un trato preferencial y condiciones específicas. La QoS se utiliza para gestionar la prioridad y asignar recursos en la red, asegurando que las aplicaciones obtengan los recursos necesarios. Puesto que no todas las aplicaciones presentan exigencias similares, la QoS se adapta a los requerimientos particulares de tipo de tráfico.

Un ejemplo ilustrativo se observa en la transmisión de documentos electrónicos, que no puede tolerar la pérdida de paquetes, ya que el destinatario no podría reconstruir el archivo sin la totalidad de los

paquetes. Por el contrario, en llamadas de VoIP, se admite un nivel moderado de cierta pérdida de paquetes, ya que la llamada puede mantenerse comprensible; sin embargo, un retardo prolongado podría interrumpir la comunicación y hacer incomprensible el mensaje. En cambio, para el correo electrónico, la latencia no suele tener un impacto significativo.

El objetivo fundamental de la calidad de servicio es ofrecer un servicio de entrega preferencial a las aplicaciones que así lo requieran, garantizando suficientes recursos, controlando los retardos y minimizando la pérdida de datos. Para evaluar la QoS en un entorno de red, se emplean las siguientes métricas formales:

- Disponibilidad del servicio: mide la fiabilidad de la conexión de los usuarios, como el porcentaje de conexión exitosos.

- Retardo: cuantifica el tiempo que tarda un paquete en viajar de un extremo a otro de la red.

- Fluctuación del retardo (Jitter): evalúa la variabilidad en el tiempo de llegada de los paquetes.

- Caudal eficaz (Throughput): mide la tasa de transmisión real de datos a través de la red, en bits por segundo.

- Tasa de transmisión (Bit rate): velocidad máxima de transmisión de datos, en bits por segundo.

- Tasa de pérdida de paquetes (Packet loss): Porcentaje de paquetes que se pierden o se dañan durante la transmisión de datos.

- Latencia: Tiempo que tarda un paquete en ser procesado dentro de un dispositivo, desde su arribo hasta su salida.

Uno de los propósitos centrales de esta investigación es la implementación de un enrutamiento multi-objetivo en SDN, centrándose en asegurar el cumplimento de los parámetros de QoS en aplicaciones específicas. Para evaluar el rendimiento de las rutas, se consideran tres métricas clave: retardo, tasa de transmisión y tasa de pérdida de paquetes. A continuación, se proporciona una descripción detallada de cómo se calculan los parámetros de QoS en las rutas establecidas por el algoritmo de enrutamiento, siguiendo la metodología de (Raayatpanah et al., 2014).

### 2.3.1 Tasa de transmisión

La tasa de transferencia se define como el flujo de datos que se transita a lo largo de un enlace y se expresa mediante la ecuación (1):

$$x_{e,m,k} \leq u_e \tag{1}$$

Donde:

$\boldsymbol{x}$: Flujo de información.

$\boldsymbol{e}$: Enlace.

$\boldsymbol{m}$: Sesión entre el nodo de origen y el nodo destino.

$\boldsymbol{k}$: Nodo destino, ($\boldsymbol{k} \in \boldsymbol{T_m}$donde $\boldsymbol{T_m}$ es el conjunto de destinos para la sesión $\boldsymbol{m}$).

$\boldsymbol{u_e}$ : Capacidad del enlace.

En el caso de múltiples sesiones compartiendo el mismo enlace, la suma de las velocidades de transmisión hacia cada destino debe ser menor o igual a la capacidad total del enlace, conforme a la ecuación (2):

$$\sum_{m \in T_m} x_{(e,m,k)} \leq u_e \tag{2}$$

Para determinar la velocidad máxima de transmisión $x_{(e,m,k)}$ en una trayectoria específica que va desde el nodo de origen ($s_m$) hasta el nodo destino ($T_m$) atravesando nodos intermedios $k$, se toma como referencia la capacidad mínima disponible a lo largo de dicha trayectoria, de acuerdo con la ecuación (3):

$$x_{(e,m,k)} \leq \min(u_e) \tag{3}$$

### 2.3.2 Retardo

En una ruta establecida de extremo a extremo, el retardo total ($D_{m,k}(p)$) se define como la suma de los retardos ($d_e$) de cada enlace y las latencias ($l_k$) de los dispositivos en la ruta ($p$), como se muestra en la ecuación 4. Simplificando la expresión se puede representar como la sumatoria de los retardos y latencias en una ruta, como se indica la ecuación 5.

$$D_{m,k}(p) = (d_{e1}) + (d_{en}) + \cdots + (d_{en}) + (l_{k1}) + (l_{k2}) + \cdots + (l_{km}) \tag{4}$$

$$D_{m,k}(p) = \sum_{e \in p} d_e + \sum_{k \in p} l_k \tag{5}$$

La probabilidad de que el retardo en una trayectoria sea igual o menor a la máxima tolerancia de retardo en un enlace $D_{\max m,k}$ se modela en la ecuación 6, donde $\beta_{m,k}$ denota la probabilidad de violación de la restricción de retardo entre el nodo de origen al de destino.

$$Pr\left(D_{m.k}(p) \leq D_{Max\, m,k}\right) = 1 - \beta_{m,k} \tag{6}$$

Dado que $D_{m,k(p)}$ es una variable aleatoria con valores positivos y con una media finita, se aplican las desigualdades de Markov para estimar la probabilidad de violación, obteniendo la ecuación 7.

$$\Pr\left(D_{m,k}\right)(p) \leq\ D_{Max\, m,k}) \leq \frac{E(D_{m,k}(P))}{D_{Max\, m,k}} \tag{7}$$

Para acotar la probabilidad de violación de extremo a extremo, se aplica la restricción de la ecuación 6 en cada camino $p \in p_{m,k}$ , como se muestra en la ecuación 8.

$$\frac{E[D_{m,k}(P)]}{D_{Max\, m,k}} \leq\ \beta_{m,k} \tag{8}$$

El retardo promedio de extremo a extremo ($E[D_{m,k}(P)]$) se considera como la suma de los retardos promedio en cada enlace ($\overline{d_e}$) y las latencias promedio en cada nodo de la ruta ($\overline{l_k}$), como se indica en la ecuación 9.

$$\frac{E[D_{m,k}(P)]}{D_{Max\, m,k}} \leq\ \sum \overline{d_e} + \sum \overline{l_k} \tag{9}$$

De esta manera, si el camino $p$ se utiliza para transmitir un flujo hacia el nodo $k$, se debe de cumplir la desigualdad de la ecuación 10:

$$\sum \overline{d_e}\ +\ \sum \overline{l_k}\ \leq\ D_{Max\, m,k} \tag{10}$$

Por último, la ecuación (11) define el retardo en la trayectoria $p$ en función de la restricción de flujo $F(p)$. Si $F(p) > 0$, se consideran los retardos y latencias de todos los enlaces del camino; en caso contrario, se asume un valor de cero.

$$D(p) = \begin{cases} \sum \overline{d_e}\ +\ \sum \overline{l_k} & \text{si F (p)} > 0 \\ 0 & si\ no\ hay\ flujo \end{cases} \tag{11}$$

### 2.3.3 Throughput

Para evaluar el rendimiento global de la ruta, el throughput se considera como la medida que refleja la entrega exitosa de datos $E_{s,t}$ a través de un canal de comunicación. La ecuación 12, expresa el cálculo del throughput $t_h$ como la cantidad de datos entregados durante un intervalo de observación $T$:

$$t_h = \frac{E_{s,t}}{T} \tag{11}$$

En este caso, $E_{s,t}$ corresponde a los datos recibidos $d_{rx}$ desde el origen $S_{tx}$, sin considerar los datos retransmitidos $r_{data}$, de acuerdo con la ecuación (13):

$$E_{s,t} = \frac{S_{tx} - (d_{rx} - r_{data})}{S_{tx}} \tag{12}$$

De este modo, el throughput cuantifica la capacidad real de transmisión efectiva en la ruta seleccionada, permitiendo determinar la eficiencia con que el sistema gestiona los flujos de datos y cumple los requisitos de QoS en escenarios de alta demanda.

# Capítulo 3. Optimización

La optimización es el proceso de encontrar la mejor solución posible dentro de un conjunto de alternativas factibles, dadas ciertas condiciones y criterios específicos. En esencia, este proceso se orienta en maximizar o minimizar una función objetivo que representa el criterio de interés, sujeta a una serie de condiciones que acotan el espacio de soluciones factibles. Su relevancia radica en la capacidad de mejorar el rendimiento, la eficiencia o la efectividad de un sistema, especialmente cuando se dispone de recursos limitados o se enfrentan a condiciones complejas (Coello et al., 2007; Durillo et al., 2009).

Formular un problema de optimización implica definir una función objetivo $f(x)$, que cuantifica el criterio que se desea maximizar o minimizar. Las variables de decisión $x = (x_1, x_2, \dots x_n)$, son los elementos que pueden ajustarse para alcanzar el objetivo deseado. No obstante, estas variables están sujetas a un conjunto de restricciones que delimitan el espacio de decisiones factibles, denotadas como $\Omega$. Dichas restricciones, pueden expresarse a menudo como igualdades o desigualdades que establecen los limites en los cuales deben buscarse las soluciones.

De acuerdo con la naturaleza del problema, se suelen dividir en dos categorías principales: optimización mono-objetivo y optimización multi-objetivo. La primera se encuentra enfocada en maximizar o minimizar una única función objetivo, mientras que la segunda persigue equilibrar múltiples funciones objetivo que frecuentemente están en conflicto entre sí (Durillo et al., 2009).

Para profundizar estos conceptos, se presenta en primer lugar la optimización mono-objetivo, describiendo su formulación matemática y los métodos usuales para resolver este tipo de problemas. Posteriormente, se aborda la optimización multi-objetivo, incorporando el concepto de soluciones de Pareto y los enfoques de resolución más apropiados para problemas con múltiples criterios en competencia.

## 3.1 Optimización mono-objetivo

La optimización mono-objetivo se centra en la identificación del valor máximo o mínimo de una sola función objetivo $f(x)$. Este tipo de problema es adecuado cuando se busca mejorar un único aspecto del sistema, como la eficiencia, los costos o los tiempos de procesamiento. El objetivo es encontrar el extremo

máximo o mínimo de $f(x)$, cumpliendo con las restricciones impuestas que limitan el espacio de soluciones posibles (Ammeri et al., 2011; Chiandussi et al., 2012).

la formulación de un problema de optimización mono-objetivo se expresa como:

$$Min|Max\ f(x) \tag{13}$$

Este sistema está sujeto a restricciones que definen las condiciones en las que la solución es válida:

$$g_i(x) \leq 0,\ \ i = 1, \dots, m \tag{14}$$
$$h_j(x) = 0,\ \ j = 1, \dots, p \tag{15}$$

Donde:

- $x = (x_1, x_2, \dots x_n)$ es el vector de variables de decisión de dimensión n, cuyos valores se deben ajustar para optimizar el criterio de optimización.

- $f(x)$ es la función objetivo que cuantifica el criterio a maximizar o minimizar.

- $g_i(x)$ y $h_j(x)$ representa la restricción de desigualdad e igualdad, respectivamente que delimitan el espacio de búsqueda.

- El universo $\Omega$ representa el conjunto de todas las soluciones factibles, es decir, las configuraciones de x que satisfacen las restricciones $g_i(x) \leq 0$ y $h_j(x) = 0$.

El objetivo en la optimización mono-objetivo es encontrar la solución óptima global $x^*$que minimice o maximice el valor de $f(x)$ en el espacio de decisión $\Omega$. Para un problema de minimización, la condición de optimalidad se expresa como:

$$f(x^*) \leq\ f(x) \qquad \forall x\ \in \Omega \tag{17}$$

Esta condición garantiza que no exista ningún valor de $x$ dentro del espacio de decisión que proporcione un valor más bajo para $f(x)$ que $f(x^*)$. En el caso de maximización, la desigualdad.

Cuando se busca determinar el valor extremo (máximo o mínimo) de la función objetivo en todo el espacio de búsqueda, se habla de optimización global. Formalmente, sea $\Omega\ \subseteq \mathbb{R}^n$ un espacio de decisión y $f: \Omega \rightarrow \mathbb{R}$ una función objetivo. Se considera que $f^* = f(x^*)$ un óptimo global si y solo si se cumple la ecuación

17, donde $x^*$representa la solución óptima global. La variable de decisión $x$ como la función objetivo $f(x)$ puede ser continua o discreta, lo que determina si se pueden aplicar técnicas analíticas, como el cálculo diferencial, o si es necesario recurrir a métodos numéricos o algoritmos de búsqueda.

## 3.2 Optimización multi-objetivo

La optimización multi-objetivo (MOOP, por sus siglas en inglés Multi-objetive Optimization Problem) aborda situaciones en las que involucran varias funciones objetivo, cada una representando criterios distintos que se deben optimizar simultáneamente. A diferencia de la optimización mono-objetivo, donde el enfoque se centra en maximizar o minimizar una única función, en la optimización multi-objetivo los objetivos generalmente están en competencia, de manera que mejorar uno puede implicar empeorar otro. Esta naturaleza conflictiva hace necesario encontrar un equilibrio o compromiso entre objetivos, en lugar de una única solución óptima (Caramia & Dell'Olmo, 2020; H. Li & Zhang, 2021; Sharma & Kumar, 2022).

Los métodos de búsqueda tradicionales, que van desde técnicas aleatorias hasta métodos exhaustivos, tienden a ser ineficientes ante espacios de búsqueda grandes, debido a su alto coste computacional. Evaluar todas las soluciones posibles para seleccionar la mejor opción es inviable. En estos casos, es esencial adoptar enfoques más eficientes, priorizando tanto la rapidez como la calidad de las soluciones obtenidas. De este modo, el enfoque de optimización multi-objetivo se convierte en una herramienta crucial para manejar estos problemas de forma eficaz.

El frente de Pareto es concepto fundamental en la optimización multi-objetivo. En términos simples, el frente de Pareto consiste en un conjunto de soluciones donde ninguna domina completamente a otra en todos los objetivos, sino que representan un equilibrio eficiente entre las métricas en conflicto. En lugar de buscar una única solución que maximice todos los objetivos, se identifican soluciones que equilibran los compromisos, permitiendo que las decisiones se basen en el trade-off entre los diferentes criterios.

Por ejemplo, es nuestro caso de estudio de enrutamiento en redes SDN que debe satisfacer varias métricas de Calidad de Servicio (QoS), como el ancho de banda, el retardo, las fluctuaciones de retardo y la pérdida de paquetes. Estas métricas están generalmente en conflicto entre sí; maximizar el ancho de banda podría incrementar el número de enlaces en una ruta, lo que a su vez podría aumentar el retardo. La clave está

en encontrar un equilibrio entre estas métricas para garantizar un rendimiento adecuado de la red y satisfacer las necesidades de tráfico y aplicaciones.

Para describir el proceso de optimización multi-objetivo matemáticamente, consideremos un problema donde se buscan soluciones óptimas para $M$ funciones objetivo. La formulación se plantea como:

$$\min\{f_m(x) | \, m = 1,2, \dots, M\} \tag{18}$$

Este sistema está sujeto a restricciones que definen las condiciones en las que la solución es válida:

$$g_j(x) \geq 0, \;\; j = 1,2, \dots, J \tag{19}$$
$$h_k(x) = 0, \;\; k = 1,2, \dots, K \tag{20}$$
$$x_{i_j} \leq x \leq x_{i_u}, \;\; i = 1,2, \dots, N \tag{21}$$

Donde cada $f_m(x)$ representa un objetivo a optimizar. En este contexto, una solución $x_1$ domina a otra solución $x_2$ si no es peor que $x_2$ en todos los objetivos y es estrictamente mejor en al menos uno. Si alguna de estas condiciones no se cumple, $x_1$ no domina a $x_2$. Este concepto es clave para identificar el frente de Pareto, donde las soluciones no dominadas representan el equilibrio óptimo entre los objetivos. Las soluciones que no son dominadas por otras dentro de la población forman este frente. Durante el proceso de optimización, se seleccionan interactivamente las soluciones no dominadas, mientras que las que han sido superadas por otras son eliminadas del conjunto.

La Figura 6, que ilustra el caso de dos objetivos, muestra cómo el frente de Pareto se sitúa en la región inferior (para minimización). Las soluciones dominantes se encuentran en la región inferior izquierda, mientras que las soluciones dominadas se encuentran en la parte superior derecha. Las soluciones en el frente de Pareto representan el mejor compromiso posible entre los objetivos, ya que no es posible mejorar uno sin empeorar otro.

Este análisis de dominancia se realiza iterativamente, comparando soluciones entre sí. Si dos soluciones no se pueden comparar en términos de dominancia, se consideran incomparables y ambas pueden formar parte del frente de Pareto. Este conjunto de soluciones no solo representa las mejores opciones disponibles, sino que también ofrece una visión completa de las soluciones óptimas, considerando las compensaciones entre los diferentes objetivos en conflicto.

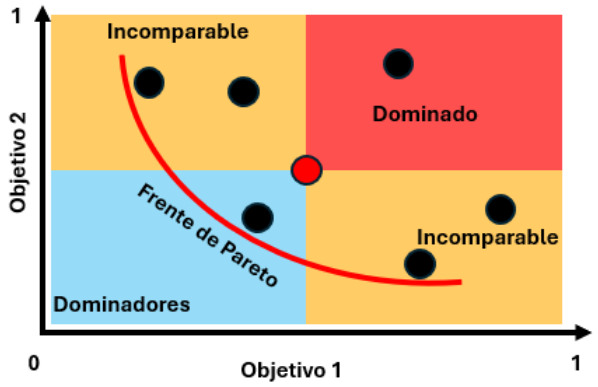


**Figura 6.** Relación de dominancia, Frente de Pareto.

## 3.3 Clasificación de métodos de solución

La optimización multi-objetivo puede abordarse principalmente a partir de métodos exactos y métodos heurísticos, cada uno con características y aplicaciones particulares. Los métodos exactos garantizan soluciones óptimas a través de una evaluación rigurosa de todas las opciones disponibles en el espacio de decisión; sin embargo, resultan poco factibles para problemas NP-complejos debido a su alto costo computacional. Por el contrario, los métodos heurísticos, tales como el recocido simulado, la búsqueda tabú y los algoritmos evolutivos, proporcionan aproximaciones rápidas y eficaces, aunque no garantizan el óptimo global.

Para asegurar la eficacia de estos métodos en la optimización multi-objetivo, deben cumplir con dos criterios esenciales: convergencia hacia el frente de Pareto, logrando aproximarse lo máximo posible a las soluciones óptimas, y diversidad en dicho frente, proporcionando una variedad representativa de compromisos entre objetivos. A continuación, se describen los principales métodos de optimización multi-objetivo, clasificados de acuerdo con estos criterios y con el grado de intervención del tomador de decisiones.

### 3.3.1 Métodos a priori

En los métodos a priori, el tomador de decisiones (DM, por sus siglas en inglés Decision Maker) establece sus preferencias antes de iniciar el proceso de optimización, definiendo metas o un orden de prioridad para los objetivos. De este modo, la búsqueda se orienta específicamente hacia las soluciones conocidas

que reflejan directamente las prioridades de DM. Entre los métodos a priori más representativos se encuentran:

#### 3.3.1.1 Método del Criterio Global:

Este método minimiza una función que mide la cercanía de una solución al vector ideal $f^*$, el cual representa el estado deseado para cada objetivo. La función más común para evaluar esta proximidad es:

$$f(x) = \sum_{i=1}^{k} \frac{f_i^* - f_i(x)^p}{f_i^*} \tag{22}$$

Donde $k$ es el número de objetivos y $p$ es un parámetro que ajusta la importancia relativa de la proximidad. La selección de $p$ es crucial para el resultado, siendo común el valor de $p = 1$ o $p = 2$ para diferentes grados de sensibilidad.

#### 3.3.1.2 Programación por metas

En este método, el DM define un conjunto de metas $T^i$ para cada objetivo $f_i(x)$. La función objetivo se centra en minimizar las desviaciones absolutas de cada objetivo respecto a sus metas:

$$Min \ \sum_{i=1}^{k} |f_i(x) - T^i| \,, \qquad x \in \Omega \tag{23}$$

Donde $\Omega$ es el espacio de soluciones factibles. Este método también puede utilizar la programación de metas generalizada, en la que se ponderan las desviaciones usando potencias p-ésimas de $|f_i(x) - T^i|$, para reflejar diferentes grados de importancia en las desviaciones de cada objetivo.

#### 3.3.1.3 Método lexicográfico

En este enfoque, El DM asigna un orden de importancia a los objetivos. Se optimiza el primer objetivo, y la solución obtenida se establece como una solución de restricción en el siguiente paso, y así

sucesivamente. Este enfoque garantiza que los objetivos de menor prioridad no comprometan la optimalidad de los de mayor prioridad. Para el primer objetivo $f_1(x)$, el problema se formula como:

$$Min f_1(x) \text{ sujeto a } g_j(x) \leq 0, \quad \text{j = 1, 2, ..., m} \tag{24}$$

Una vez obtenida la solución $x_1^*$, para este primer problema, se formula el siguiente problema incluyendo una restricción adicional que fija el valor de $f_1(x)$ como un óptimo:

$$f_1(x) = f_1^* \tag{25}$$

Este proceso se repite hasta que todos los objetivos se han optimizado en el orden de prioridad establecido. El método lexicográfico es útil cuando el DM tiene una clara jerarquía de objetivos y desea asegurar que los objetivos de mejor prioridad no afecten la optimalidad de los de mayor importancia. Sin embargo, este enfoque puede limitar las soluciones alcanzables si las restricciones impuestas por los objetivos de mayor prioridad son demasiado estrictas.

## 3.4 Métodos a posteriori

En los métodos a posteriori la optimización multi-objetivo se caracterizan por no requerir que el DM proporcione información o preferencias específicas antes de iniciar con el proceso de optimización. En su lugar, estos métodos generan un conjunto de soluciones no dominadas que forman el frente de Pareto, entre las cuales el DM puede seleccionar aquella que mejor se ajuste a sus necesidades. Al no tener preferencias previas, estos métodos son útiles para explorar distintas soluciones antes de decidir cuál objetivo priorizar. A continuación, se presentan algunos de los métodos a posteriori más populares:

#### 3.4.1.1 Combinación lineal de pesos

Propuesto por Zadeh (1963), este método convierte el problema multi-objetivo en uno escalar, creando una función objetivo única mediante la suman de los objetivos individuales, creando uno ponderado por un peso específico. La función objetivo se describe de la siguiente manera:

$$Min \sum_{i=1}^{k} w_i f_i(x), \quad x \in \Omega \tag{26}$$

Donde, $w = (w_1, \dots w_n)$, es el vector de pesos, con $w_i \geq 0$ para cada $i$ y $\sum w_i = 1$. Variando los pesos $w_i$, se pueden obtener diferentes soluciones en el frente de Pareto, ya que la combinación lineal de pesos tiende a generar soluciones en regiones convexas del frente de Pareto. A pesar de esta limitación, la simplicidad del método lo hace adecuado en escenarios donde el frente es aproximadamente convexo y los objetivos pueden combinarse de manera lineal.

#### 3.4.1.2 Método ∈-Restricciones

El método de ∈-restricciones permite optimizar una función objetivo específica mientras se convierten las demás funciones en restricciones, cada una con un límite permisibles $\in_l$, teniendo la función:

$$Min\ f_r(x) \tag{27}$$

Sujeta a:

$$f_l(x) \leq \in_l, \quad l = 1, 2, \dots, k, l \neq r \tag{28}$$

Donde $f_r(x)$ es la función objetivo seleccionada para optimizarse. Este enfoque permite obtener una variabilidad de soluciones no dominadas en el frente de Pareto, al variar los valores de $\in_l$ y alterar las funciones objetivo seleccionada para optimización. Esta flexibilidad en la sección de objetivos y restricciones es útil en situaciones donde el DM necesita mantener ciertos parámetros dentro de límites específicos, mientras optimiza una función en particular. Además, este método, es efectivo para tratar problemas con frentes cóncavos y con restricciones que representan limitaciones en el problema multi-objetivo.

#### 3.4.1.3 Algoritmos Evolutivos Multi-Objetivo (MOEA)

Los Algoritmos Evolutivos Multi-Objetivo (MOEA, por sus siglas en inglés Multi-Objective Evolutionary Algorithms) son una clase de algoritmos inspirados en procesos de selección natural y evolución. A diferencia de los métodos anteriores, los MOEA generan un conjunto de soluciones en paralelo, explorando el espacio de decisión mediante operadores genéticos como selección, cruzamiento y

mutación. Este proceso iterativo permite mantener la diversidad en la población de soluciones y cubrir la exploración de distintas zonas del frente de Pareto incluidas aquellas con frentes cóncavos y discontinuos.

#### 3.4.1.4 Métodos interactivos

Los métodos interactivos permiten la participación continua del DM a lo largo de múltiples iteraciones del proceso de optimización. Este enfoque tiene la particularidad de ajustar sus preferencias o explorar el impacto de diferentes parámetros durante cada iteración. Los métodos interactivos suelen estructurase en tres etapas:

1. Generación de una solución no dominada inicial: en la primera etapa, se produce una solución no dominada que representa una posible compensación entre los objetivos.

2. Revisión y ajuste de preferencia por parte del DM: el DM evalúa la solución obtenida y ajusta sus preferencias, proporcionando retroalimentación sobre los objetivos que desea priorizar o las restricciones que considera fundamentales.

3. Repetición del proceso hasta alcanzar satisfacción: este proceso iterativo continúa hasta que el DM encuentra una solución que satisfaga sus requisitos o el algoritmo no pueda mejorar más las soluciones.

Este método adaptativo permite que el DM experimentar y refinar sus preferencias durante el proceso de optimización, facilitando ajustar la búsqueda de soluciones a los cambios en las condiciones o en las expectativas del DM.

## 3.5 Algoritmos evolutivos

La computación evolutiva constituye un área de la informática inspirada en los principios de la evolución natural, donde las especies sobreviven a través de la adaptación continua a su entorno. Este principio de adaptación se traslada a problemas de optimización y se utiliza para mejorar iterativamente un conjunto de soluciones candidatas hasta cumplir con una condición de parada. De esta manera, la computación evolutiva busca construir soluciones óptimas a partir de un conjunto inicial de soluciones propuestas,

generando así un proceso de mejora continua mediante prueba y error. Este principio está basado en la metaheurística de población, como se ilustra en la Figura 7. En primer lugar, se aplica una búsqueda local a una solución inicial, y en cada iteración, se realiza una perturbación de cada óptimo local obtenido. Posteriormente, se realiza una búsqueda local a cada solución perturbada, repitiendo el ciclo hasta alcanzar la condición de parada (Eiben et al., 2015; Xavier et al., 2013).

La base conceptual de la adaptación y evolución de sistemas biológicos se atribuye a Darwin (1987) y fue incorporada al ámbito de la inteligencia artificial por Alan Turing en 1948. Posteriormente en 1962, Bremermann realizo experimentos computacionales sobre optimización a través de la evolución y recombinación, lo cual sirvió como base para los avances de Rechenberg (1965), Forgel et al. (1966), Holland (1975) y Koza (1992), quienes desarrollaron estrategias evolutivas y genéticas que constituyen la base de lo que hoy conocemos como computación evolutiva (CE) (Kennedy & Eberhart, 2001; de Castro & Timmis, 2002; Olague et al., 2006). En particular, los algoritmos genéticos (GA), una de las técnicas evolutivas más reconocidos, no solo retoma el marco conceptual de la supervivencia del más apto de Darwin, sino que también incorporan las teorías de herencia de rasgos a través de los genes formuladas por Mendel. Holland (1975) fue pionero en su formulación, mientras que Goldberg (1989) potenció su consolidación como una herramienta robusta y flexible para resolver problemas de búsqueda, optimización y aprendizaje dentro de la inteligencia artificial.

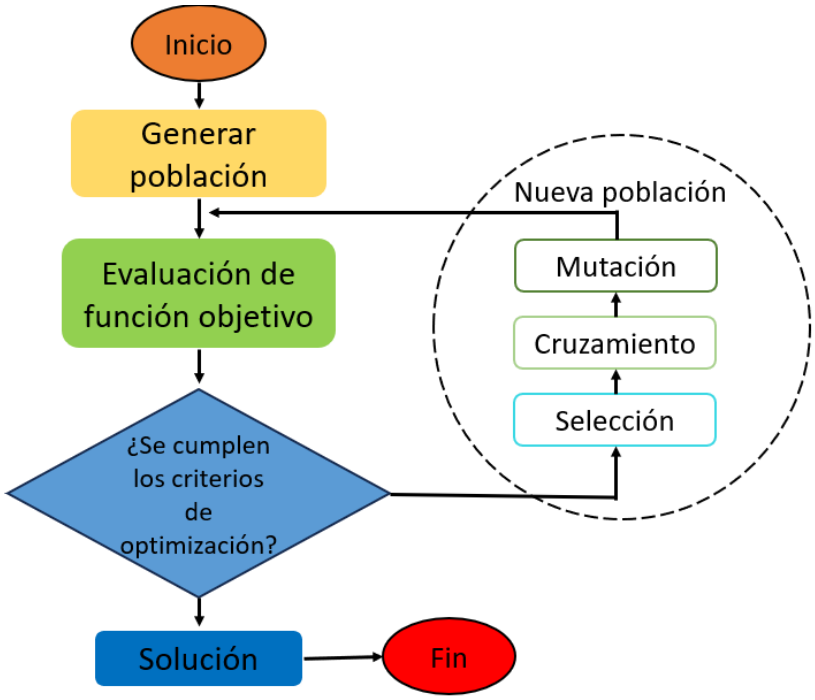


**Figura 7.** Diagrama de algoritmos evolutivos.

### 3.5.1 Estructura

Los algoritmos evolutivos son una técnica de computación inspirada en la evolución biológica, que se utiliza para resolver problemas de optimización complejos. Para abordar un problema específico con EA, define una población, compuesta por múltiples soluciones candidatas, también conocidas como individuos, cada una de las cuales representa una solución al problema. La representación de cada individuo se realiza mediante un cromosoma que codifica su información genética, o un conjunto de características, y se evalúa a través de una métrica conocida como función de aptitud. Esta función cuantifica la calidad de cada individuo en función del objetivo que se desea alcanzar, ayudando a identificar las soluciones más prometedoras que merecen ser mejoradas y combinadas(Katoch et al., 2021; Lambora et al., 2019).

El proceso evolutivo comienza con una selección de las soluciones con mayor aptitud, preservando aquellas con mejores características y aumentado sus probabilidades de transmitir estos rasgos a las próximas generaciones, mientras que las menos prometedoras suele eliminarse. Durante la etapa de reproducción, los cromosomas de los individuos seleccionados se combinan mediante el proceso de cruce, intercambiando material genético entre sí para crear descendencias que hereden características de ambos progenitores. Para promover la diversidad genética y explorar nuevos espacios de soluciones se induce el proceso de mutación, el cual consiste en realizar múltiples copias de las soluciones más prometedoras y realizar cambios en los cromosomas de forma aleatorios para obtener nuevos individuos. Este mecanismo ayuda a los algoritmos evolutivos a evitar quedarse en óptimos locales, permitiendo explorar otras regiones del espacio de búsqueda. La nueva generación de soluciones o descendencia resultante se evalúa nuevamente, y aquellas con mayor aptitud reemplazan a las generaciones anteriores. Este ciclo se repite hasta que se alcanza un criterio de parada, como la obtención de una solución satisfactoria o la superación de un número máximo de generaciones (Hruschka et al., 2009; Shukla et al., 2015).

### 3.5.2 Representación genética en algoritmos evolutivos

Para que el algoritmo evolutivo pueda manejar cada solución, cada individuo (solución) es representada mediante un cromosoma, una estructura que contiene la información genética o los rasgos particulares de una solución. Este cromosoma está compuesto por genes, donde cada gen codifica una característica específica. Dentro de este marco, existen tres espacios fundamentales que ayudan a definir el proceso evolutivo:

- Espacio de genotipo: este espacio contiene los cromosomas y los genes que codifican los rasgos de los individuos. En este espacio, se producen los procesos de cruce y mutación, los cuales permiten la recombinación y modificación de genes.

- Espacio de Fenotipo: este espacio representa la manifestación o expresión de los cromosomas en el contexto del problema, el cual corresponde a la manifestación de la solución una vez que el cromosoma ha sido decodificado.

- Estado de aptitud: en este espacio, las soluciones so evaluadas de acuerdo con la función de calidad o desempeño utilizando la función de aptitud. Los procesos de selección y supervivencia tienen lugar en este espacio.

En los algoritmos genéticos clásicos, los cromosomas se representan comúnmente mediante cadenas de bits de longitud fija. Los operadores de cruce y mutación desempeñan un papel esencial en la generación de nuevas soluciones. El cruce, o recombinación, toma los cromosomas de dos individuos padres y construye uno o más individuos hijos a partir de ellos. Se espera que los padres con una aptitud alta transmitan los genes necesarios al hijo para obtener una aptitud aún mayor. Por otro lado, la mutación se aplica como un proceso aleatorio que modifica el cromosoma de algunos individuos con el propósito de introducir genes que no existían en ningún individuo de la población. Esto funciona como un mecanismo para que los algoritmos genéticos puedan salir de los óptimos locales.

En las próximas secciones se profundizarán los procesos clave de los algoritmos genéticos, incluyendo las estrategias de selección, los métodos de cruce y mutación para mantener la diversidad genética, y los criterios de terminación que definen el alcance de una solución óptima. Este análisis permitirá entender los factores que determinan el rendimiento y éxito de estos algoritmos.

#### 3.5.2.1 Métodos de selección

Existen diferentes técnicas para seleccionar a los individuos que deben copiarse hacia la siguiente generación. Algunos de estos métodos son mutuamente exclusivos, pero otros pueden utilizarse en combinación. Los más empleados son:

- Elitista: Garantiza la selección de los miembros más aptos de cada generación, copiando el mejor o mejores individuos hacia la siguiente generación si no surge un individuo mejor.

- Proporcional a la aptitud: Los individuos más aptos tienen más probabilidades de ser seleccionados, aunque no hay certeza. Se otorga una mayor probabilidad de selección a los individuos más aptos.

- Ruleta: Es una forma de selección proporcional a la aptitud en la que la probabilidad de que un individuo sea seleccionado es proporcional a la diferencia entre su aptitud y la de sus competidores.

- Torneo: Se eligen subgrupos de individuos de la población, y los miembros de cada subgrupo compiten entre sí. Solo se selecciona un individuo de cada subgrupo para la reproducción.

- Por rango: A cada individuo de la población se le asigna un rango numérico basado en su aptitud, y la selección se basa en esta asignación en lugar de las diferencias absolutas en aptitud. Este método puede evitar que individuos muy aptos dominen a los menos aptos.

- Generacional: Sustituye todos los individuos de una generación para obtener la siguiente.

#### 3.5.2.2 Métodos de cruzamiento

El cruzamiento implica seleccionar a dos individuos para que intercambien segmentos de su código, produciendo una descendencia cuyos individuos son combinaciones de sus padres. Las formas comunes de cruzamiento son:

- Operador de cruce simple o de un punto: Se establece un punto de intercambio en un lugar aleatorio del genoma de los dos individuos, y se produce una descendencia a partir de este punto.

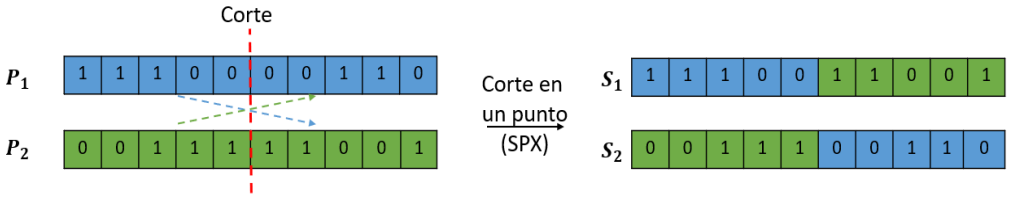


**Figura 8.** Operadores de cruce simple (un corte).

- Operador con dos puntos de corte: Es similar al anterior, pero se establecen dos puntos de intercambio en lugares aleatorios del genotipo de los individuos, produciendo dos hijos.

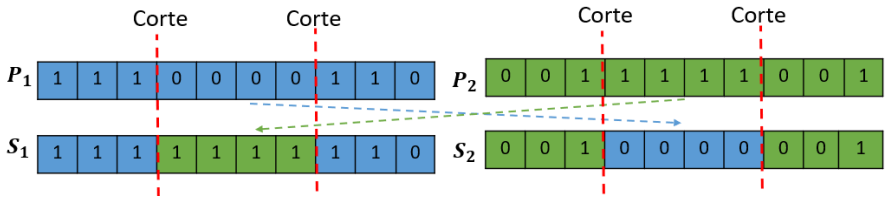


**Figura 9.** Operadore de dos cruces de corte.

- Operador de cruce uniforme: Los genes de los descendientes se obtienen según una máscara, intercambiando los bits de los dos cromosomas que coincidan donde hay un 1 en la máscara.

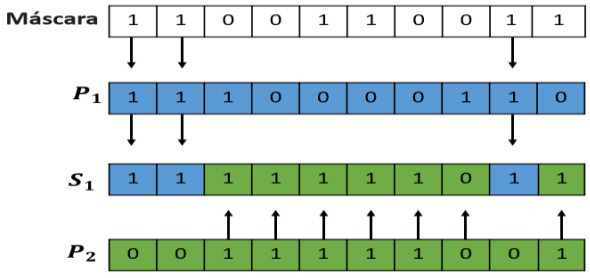


**Figura 10.** Operador de cruce uniforme.

### 3.5.2.3 Métodos de mutación

La mutación es un operador genético utilizado para proporcionar diversidad en las generaciones de una población, alterando el estado de una secuencia genética de un individuo mediante el intercambio de un elemento de su cadena. Su propósito es evitar mínimos locales que puedan ralentizar o detener el proceso evolutivo.

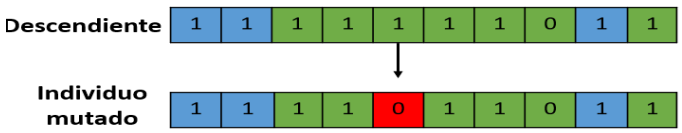


**Figura 11.** Operación de mutación.

#### 3.5.2.4 Criterios de terminación

- El proceso evolutivo es cíclico hasta que se cumpla uno de los siguientes criterios:
- Se encuentra una solución que satisface un criterio mínimo/máximo.
- Se alcanza un número de generación predeterminado.
- No se producen mejoras en los individuos hijos después de un determinado número de generaciones.
- Terminación manual.
- Una combinación de los anteriores.

## 3.6 Algoritmo Multi-objetivo por Descomposición (MOEA/D)

El algoritmo evolutivo multi-objetivo basado en descomposición (MOEA/D por sus siglas en inglés Multi-objective Evolutionary Algorithm Base on Decomposition) utiliza un enfoque de descomponer un MOP en un conjunto de subproblemas de optimización escalar. Cada uno de estos subproblemas es optimizado de manera independiente pero coordinada, mediante un conjunto de vectores de peso distribuidos uniformemente (Figura 12), representados como $w$. Estos vectores de peso sirven para orientar cada subproblema hacia una región específica de la solución de Pareto, buscando una representación amplia y diversa del espacio de soluciones.

En cada iteración del MOEA/D, se emplean operadores genéticos, como el cruce y la mutación, para mejorar las soluciones asociadas a cada subproblema. Cada vector de peso define un vecindario o $T_{región}$, que agrupa soluciones relacionadas dentro del espacio de Pareto. La meta del algoritmo es localizar

soluciones óptimas dentro de cada vecindario, promoviendo simultáneamente el progreso en todas las regiones del espacio. Si una solución generada mejora el valor de aptitud en un subproblema vecino, esta reemplaza a la solución anterior, fomentando un intercambio continuo de información y acelerando la convergencia hacia el frente de Pareto.

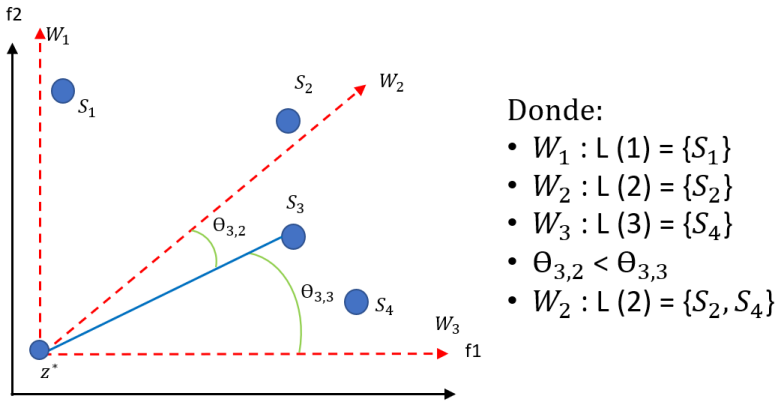


**Figura 12.** Representación de espacio de búsqueda del Algoritmo Multi-objetivo por descomposición (MOEA/D).

### 3.6.1 Métodos de descomposición en MOEA/D

Un aspecto crucial del MOEA/D es la elección del método de descomposición, ya que determina cómo se desglosa el problema original y se guía la búsqueda. Según Zhang y Li, existen tres métodos de descomposición principales que se aplican comúnmente en MOEA/D:

- Suma Ponderada: Cada objetivo es ponderado según el vector de pesos, y los valores ponderados se suman para formar una función de aptitud escalar. Este método es adecuado para problemas donde los objetivos tienen escalas similares y no existen regiones cóncavas en el frente de Pareto.

- Tchebysheff: La función de Tchebysheff busca minimizar la máxima desviación ponderada de cada objetivo respecto al vector de referencia. Esto es útil para problemas que presentan frentes de Pareto complejos y no convexos, ya que ayuda a equilibrar la diversidad en las soluciones.

- Interacción de Límites Basada en Penalizaciones: Este método integra penalizaciones para manejar las restricciones de una manera más flexible, promoviendo la exploración de áreas del espacio de soluciones que otros métodos podrían pasar por alto.

En el contexto del enrutamiento de redes, esta investigación se centra en el método de suma ponderada debido a su simplicidad y eficacia. Este enfoque descompone el problema en tres vectores de peso, cada uno asociado a un parámetro crítico de la red, como latencia, ancho de banda y fiabilidad. Los vectores de peso no solo priorizan métricas específicas, sino que también dividen el espacio de soluciones en vecindarios donde los subproblemas se resuelven en paralelo. Esto permite abordar simultáneamente diversas partes del espacio de Pareto, logrando un equilibrio entre las métricas de rendimiento de la red.

El MOEA/D utiliza un proceso iterativo en el que las soluciones dentro de cada vecindario se actualizan en función de su aptitud. Cuando una solución secundaria mejora el rendimiento de un subproblema vecino, reemplaza a la solución existente, asegurando un intercambio dinámico de información entre los vecindarios. Este mecanismo facilita la convergencia hacia el frente de Pareto, maximizando el rendimiento global de la red al tiempo que se optimizan las métricas individuales (Z. Wang et al., 2016; Q. Zhang & Li, 2007).

El pseudocódigo 1 describe este enfoque de manera detallada, ilustrando cómo los vectores de peso guían la optimización en cada subproblema y cómo las soluciones se ajustan iterativamente para mejorar los parámetros de la red. Este método es especialmente efectivo en aplicaciones que requieren un equilibrio preciso entre múltiples métricas en escenarios complejos y dinámicos. Al descomponer el problema en subproblemas manejables y actualizar las soluciones iterativamente, el MOEA/D asegura una representación robusta y eficiente del espacio de soluciones, incluso bajo las condiciones más exigentes. Esto lo convierte en una herramienta ideal para optimizar redes con objetivos en conflicto y mantener un rendimiento adaptable y confiable.

Algoritmo 1. Seudocódigo de metaheurísticas en MOEA/D

$\boldsymbol{Data} = \boldsymbol{parámetros}$ //Configuración de parámetros

$\boldsymbol{pop} = \boldsymbol{solución}$ //Población inicial

$\boldsymbol{W}\ (\boldsymbol{t} = \boldsymbol{0})$ //Inicialización de vectores de peso

$\boldsymbol{T} = \boldsymbol{W_i} \in \boldsymbol{W_t}$ //Calculo de vecindarios de acuerdo con los vectores de peso

$\boldsymbol{P_F} = []$ //creación de Frente de Pareto = vacío

**While** (condición terminal== false) **do**

$\boldsymbol{For\ neighborhood} \in \boldsymbol{pop\ do}$ //cada vector crea un vecindario

$\boldsymbol{For\ individual} \in \boldsymbol{neighborhood}$ //parámetros seleccionados por vecindario

$\mathbf{parents} = (\mathbf{neighborhood})$ //selección de padres por vecindario

$\mathbf{offpring} = \mathbf{recombination}\ (\mathbf{data}, \mathbf{parents})$ //cruzamiento

offpring = mutation (data, offspring)

$$P_F = Insert\ (Offspring)$$

$$End$$

$$End$$

$$pop = Feedback\ (Pareto\ Front)$$

$$End$$

## 3.7 Algoritmo genético MOCell

El algoritmo genético celular multi-objetivo (por sus siglas en inglés Multi-objective Cellular genetic algorithm), propuesto por (Nebro et al., 2009), organiza a los individuos de una población en una estructura espacial conocida como malla toroidal. Esta disposición asegura que cada individuo tenga un vecindario fijo, permitiendo una interacción limitada con sus vecinos más cercanos. El objetivo principal de esta estructura es fomentar la exploración local y mantener la diversidad en el espacio de soluciones. A diferencia de otros enfoques donde todos los individuos interactúan globalmente, MOCell restringe la interacción a vecindarios locales, lo que ayuda a explorar de manera más efectiva regiones específicas del espacio de Pareto.

En MOCell, la población inicial se evalúa y clasifica utilizando criterios de ranking y distancia de apiñamiento, para establecer un conjunto de soluciones no dominadas conocido comoPND (por sus siglas en inglés Pareto Non-Dominated set). Este conjunto sirve como referencia para la calidad de las soluciones durante las iteraciones del algoritmo. El proceso iterativo del algoritmo se desarrolla en varias etapas, como se describe a continuación:

1. Selección de padres: Cada individuo es evaluado en su vecindario utilizando un torneo de selección. Esta técnica permite identificar a los padres que serán utilizados en la reproducción, priorizando aquellos con un mejor desempeño según los objetivos del problema. La evaluación considera no solo el ranking de dominancia, sino también la distancia de apiñamiento, asegurando que las soluciones seleccionadas mantengan la diversidad en el vecindario.

2. Reproducción: Los individuos seleccionados como padres generan descendencia mediante el operador de recombinación (cruzamiento). Este operador combina las características de dos padres para producir nuevos individuos que heredan propiedades de ambos. Esto permite explorar nuevas regiones del espacio de soluciones.

3. Mutación: Para introducir variabilidad genética y evitar estancarse en óptimos locales, se aplica un operador de mutación a los descendientes generados. Este operador modifica aleatoriamente uno o más genes del cromosoma de un individuo, lo que puede resultar en soluciones potencialmente novedosas.

4. Evaluación de descendencia: Los nuevos individuos generados son evaluados en función de su desempeño utilizando las mismas métricas que los padres, es decir, ranking y distancia de apiñamiento. Esto garantiza que las soluciones prometedoras se integren efectivamente en la población.

5. Actualización del conjunto PND: Una vez evaluados todos los individuos, el conjunto PND se actualiza incluyendo las soluciones no dominadas más recientes. Al final de cada iteración, NNN individuos del conjunto PND son seleccionados aleatoriamente y reemplazan a individuos de la población principal. Este intercambio asegura un equilibrio entre la preservación de las mejores soluciones y la exploración continua de nuevas posibilidades.

6. Intercambio aleatorio: Para finalizar cada iteración, se realiza un intercambio aleatorio entre individuos de la población principal y el conjunto PND. Este paso introduce nuevas dinámicas en la población, reforzando la interacción entre las mejores soluciones y los individuos restantes.

La estructura toroidal de la población permite que cada individuo mantenga una interacción equilibrada con sus vecinos, independientemente de su posición en la malla. Esta propiedad garantiza que ninguna región del espacio de soluciones reciba un tratamiento preferencial, lo que resulta en una exploración uniforme y diversa del frente de Pareto.

El pseudocódigo del algoritmo MOCell refleja estas etapas (algoritmo 2), mostrando cómo cada iteración integra la selección, reproducción, evaluación y actualización del conjunto de soluciones. La Figura 13 ilustra gráficamente la estructura y funcionamiento del algoritmo, destacando cómo los vecindarios locales permiten a MOCell optimizar múltiples objetivos de manera eficiente y mantener una representación diversificada del espacio de Pareto.

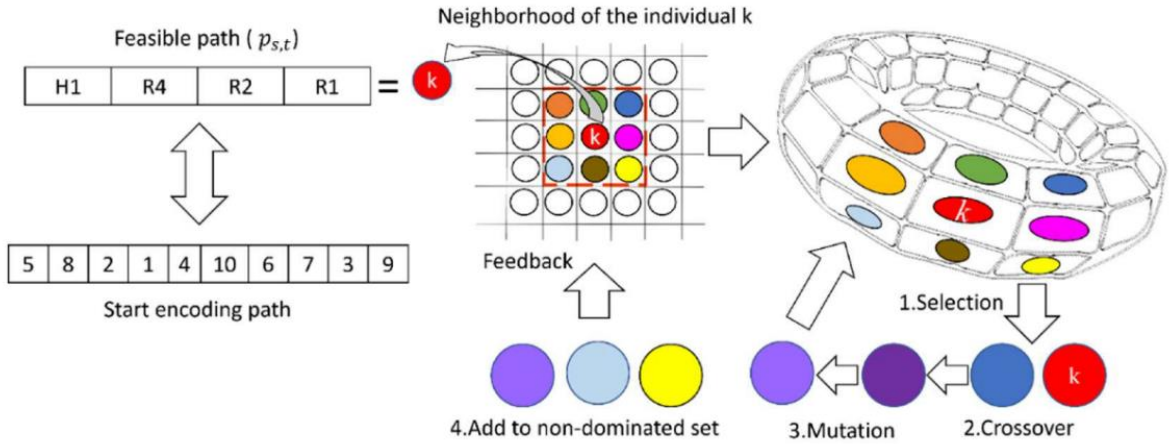


**Figura 13.** Representación del algoritmo MOCell.

Algoritmo 2. Seudocódigo de metaheurísticas en MOEA/D

$\boldsymbol{Data} = \boldsymbol{parámetros}$ //Configuración de parámetros

$\boldsymbol{pop} = \boldsymbol{solución}$ //Población inicial

$\boldsymbol{W}\ (\boldsymbol{t} = \boldsymbol{0})$ //Inicialización de vectores de peso

$\boldsymbol{T} = \boldsymbol{W_i} \in \boldsymbol{W_t}$ //Calculo de vecindarios de acuerdo con los vectores de peso

$\boldsymbol{P_F} = []$ //creación de Frente de Pareto = vacío

***While*** (condición terminal== false) ***do***

$\boldsymbol{For\ neighborhood} \in \boldsymbol{pop\ do}$ //cada vector crea un vecindario

$\boldsymbol{For\ individual} \in \boldsymbol{neighborhood}$ //parámetros seleccionados por vecindario

$\mathbf{parents} = (\mathbf{neighborhood})$ //selección de padres por vecindario

$\mathbf{offpring} = \mathbf{recombination}\ (\mathbf{data}, \mathbf{parents})$ //cruzamiento

offpring = mutation (data, offspring)

$\boldsymbol{P_F} = \boldsymbol{Insert}\ (\boldsymbol{Offspring})$

$\boldsymbol{End}$

$\boldsymbol{End}$

$\boldsymbol{pop} = \boldsymbol{Feedback}\ (\boldsymbol{Pareto\ Front})$

$\boldsymbol{End}$

# Capítulo 4. Redes Neuronales

## 4.1 Redes Neuronales clásicas (NN)

Las redes neuronales han revolucionario el campo de la inteligencia artificial al emular la complejidad y eficiencia del cerebro humano. Estas estructuras computacionales, inspiradas a partir de la arquitectura neuronal del sistema nervioso, han transformado la manera de abordar diversos problemas de procesamiento de información. Al imitar la capacidad de aprender y adaptarse, las redes neuronales permiten a las máquinas realizar tareas cognitivas avanzadas, como reconocimiento de patrones, toma de decisiones y resolución de problemas complejos (S.-C. Wang, 2003; Wu & Feng, 2018; Z. Zhang, 2018).

Las redes neuronales, inspiradas en la estructura y función del cerebro humano, son sistemas computacionales diseñados para realizar tareas específicas mediante el aprendizaje y la adaptación a partir de datos. La unidad básica de estas redes es la neurona. Su funcionamiento se describe en la ecuación 29 y se describe como función lineal de las entradas sumadas a un valor de sesgo.

$$z = w^T x + b = \sum_{i=1}^{N} w_i \cdot x_i + b \tag{29}$$

Donde cada neurona toma un conjunto de entradas $x_i$ las multiplica por sus pesos asociados ($w_i$), suma un valor de sesgo ($b$) y genera una salida, como se muestra en la representación de una neurona en la Figura 14. Sin embargo, conectar varias neuronas con funciones meramente lineales no brinda ventajas significativas frente a utilizar una sola neurona de este tipo. Por lo tanto, para que las redes neuronales puedan capturar relaciones no lineales en los datos, se utiliza un componente adicional conocido como función de activación.

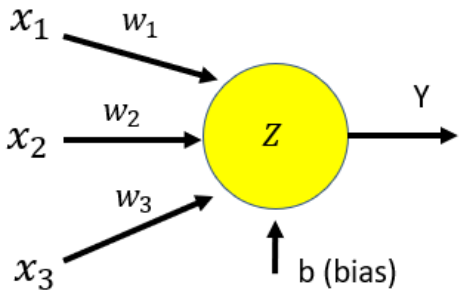


**Figura 14.** Representación de neurona artificial.

Las funciones de activación transforman la salida $z$ de una neurona en una respuesta no lineal, ampliando la capacidad de la red para modelar relaciones complejas. Matemáticamente, la función de activación se define como:

$$\mathrm{y} = \delta(\mathrm{z}) \tag{30}$$

En los primeros modelos de redes neuronales, McCulloch y Pitts propusieron una función escalón (conocida como HeavySide), la cual fue un hito en el desarrollo de funciones de activación (Chakraverty et al., 2019). Entre las funciones de activación más empleadas se encuentran: Sigmoide, Tangente hiperbólica y Unidad Lineal Rectificada (ReLU por sus siglas en inglés Rectified Linear Unit), las cuales se describen a continuación:

- Función Sigmoide: Esta función es útil para tareas donde las salidas deben interpretarse como probabilidades, ya que limita los valores entre 0 y 1.

$$\sigma(z) = \frac{1}{1 + e^{-x}} \tag{31}$$

- Tangente Hiperbólica: A diferencia de la sigmoide, produce valores entre -1 y 1, lo que facilita el entrenamiento al centrar las salidas alrededor de cero.

$$\tanh(z) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{32}$$

- Unidad Lineal Rectificada (ReLU): es ampliamente utilizada en redes profundas dado que ayuda a mitigar el problema del desvanecimiento del gradiente, permitiendo que los gradientes fluyan eficientemente durante el entrenamiento.

$$\mathrm{ReLU}(z) = \max(0, z) \tag{33}$$

Cada una de estas funciones cumple propósitos específicos. Por ejemplo, la función sigmoide es empleada frecuentemente en modelos de clasificación binaria, mientras que la tangente hiperbólica resulta útil cuando los datos requieren centrar a un torno a cero. Por otro lado, la función ReLU se ha convertido en el estándar para redes neuronales profundas debido a su simplicidad computacional y su eficiencia en la propagación de gradiente.

### 4.1.1 Arquitectura de Redes Neuronales (NN)

Una red neuronal está compuesta de múltiples capas las cuales están organizadas en una estructura jerárquica. Estas capas se agrupan en tres categorías principales con un papel en específico:

- Capa de entrada: representa las variables o características de los datos de entrada. Cada neurona en esta capa corresponde a un atributo particular del conjunto de datos, y su función principal es recibir y transmitir los datos hacia las capas siguientes sin realizar ninguna transformación.

- Capas ocultas: Situadas entre la capa de entrada y la capa de salida, estas capas intermedias son el núcleo de la red neuronal. Cada capa oculta está formada por un número configurable de neuronas, las cuales reciben como entrada las activaciones de todas las neuronas de la capa previa. Estas capas aplican transformaciones no lineales mediante funciones de activación, permitiendo que la red neuronal aprenda patrones y relaciones no lineales en los datos. Es posible disponer de tantas capas ocultas y neuronas como sea necesario.

- Capa de salida: Situada en el extremo final de la red, esta capa contiene tantas neuronas como variables de salida se requieran. Las neuronas en esta capa producen las salidas finales de la red, que pueden representar clasificaciones, valores numéricos o probabilidades.

Cada neurona en las capas de una red neuronal se conecta completamente con las neuronas de la capa previa mediante pesos, lo que facilita la transmisión y transformación eficiente de información. Este diseño permite a la red aprender patrones complejos, como se observa en configuraciones básicas con entradas, salidas y capas ocultas intermedias (ver Figura 15). Estas conexiones, junto con las funciones de activación, son clave para procesar datos y generar predicciones en tareas como clasificación, regresión y reconocimiento de patrones.

Para terminar el modelo de la red neuronal, el ultimo componente es la función de aprendizaje el cual tiene como objetivo reducir al máximo una función de coste, se pueden usar varios tipos de funciones de coste, la más común se calcula como el error cuadrático medio (MSE por sus siglas en inglés Mean Square Error) de las salidas estimadas por la red frente a las salidas teóricas como se representa en la ecuación 34.

$$C = \frac{1}{2/n}\sum_{i=1}^{N}\sum_{J=1}^{S}(y_{i,j} - \hat{y}_{i,j})^2 \quad (34)$$

Donde $N$ el número de ejemplos de entrenamiento y $S$ la cantidad de variables de salida. El objetivo de la red durante el entrenamiento busca el conjunto de parámetros $w$ y $b$ que minimizan la función de coste, a partir del método de backpropagation.

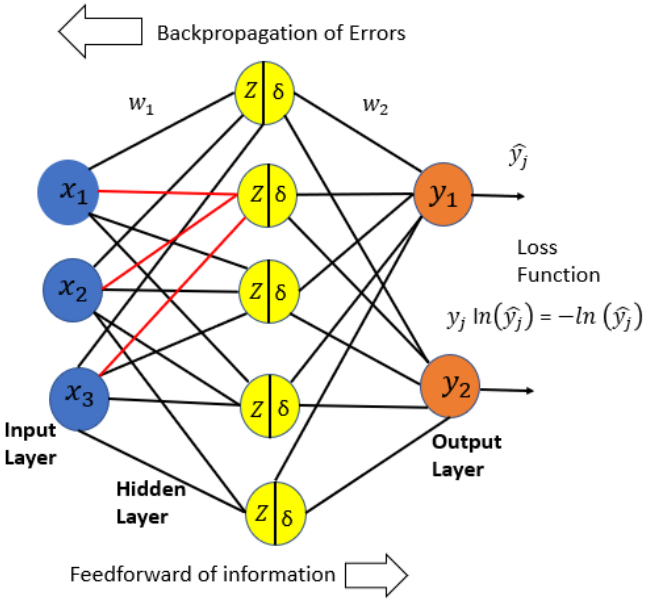


**Figura 15.** Esquema de una red neuronal de tres entradas, dos salidas y una capa oculta con cinco neuronas.

El algoritmo propagación hacia atrás (backpropagation) se centra primero en calcular la derivada parcial de la función de coste frente a la suma ponderada de cada neurona, lo que se denomina error de la neurona (Ecuación 35). El error de la neurona se interpreta el cambio que sufre la función de coste con la variación de la suma ponderada que se obtiene de ella. Para su cálculo se emplea la regla de la cadena de derivación de funciones compuestas.

$$\mathrm{y} = \delta_i^l\left(\frac{\partial \mathrm{c}}{\partial z_i^l}\right) \quad (35)$$

donde $z_i^l$, es la suma ponderada de las neuronas en cada capa representada en la ecuación 36. Para el cálculo de los errores de las neuronas de capas previas se parte del error en la capa posterior, $\delta_i^{l+1}$, representado en la ecuación 36.

$$\delta_j^l = \sum_k \frac{\partial \mathrm{c}}{\partial z_k^{l+1}} \cdot \frac{\partial z_k^{l+1}}{\partial \delta_i^l} \cdot \frac{\partial \delta_j^l}{\partial \mathrm{z}_j^l} = \sum_k \delta_k^{l+1} \mathrm{w}_{kj}^{l+1} f'(\mathrm{z}_j^l) \quad (36)$$

De esta manera es posible obtener los errores de las neuronas en todas las capas a partir de los errores de las capas posteriores y su matriz de pesos.

El entrenamiento de una red neuronal comienza con una inicialización aleatoria de los parámetros de la red, posteriormente se seguirá un proceso iterativo para cálculo del gradiente y actualizar los parámetros de la red. La forma clásica de calcular el mínimo de una función es resolver el sistema $\nabla \mathrm{f} = \vec{0}$. La ecuación 37 muestra la función del descenso del gradiente para la actualización de un peso de cada neurona.

$$\mathrm{w}_{i,j}^{l}(t+1) = \mathrm{w}_{i,j}^{l}(t) - \delta \frac{\partial \mathrm{c}}{\partial w_{i.j}^{l}}\Big|_{\mathrm{w}_{i,j}^{l}(t)} \tag{37}$$

Una tasa de aprendizaje baja implica un aprendizaje lento, dado que requiere de muchas iteraciones para encontrar el mínimo. Por otro lado, una tasa de aprendizaje elevada implica una mayor inestabilidad en el algoritmo, lo que puede significar que nunca alcance la convergencia.

## 4.2 Redes Neuronales de Grafos (GNN)

Las redes neuronales se han adaptado para trabajar con datos no euclidianos, especialmente en estructuras de grafos, dando lugar a las Redes Neuronales de Grafos (GNN por sus siglas en inglés Graph neuronal network). Estas redes amplían el alcance de las redes neuronales tradicionales al centrarse en capturar patrones y relaciones en conjuntos de datos representados mediante nodos y aristas. Gracias a esta capacidad, las GNN son particularmente útiles en tareas de predicción en diferentes niveles: nodos, aristas o incluso en el nivel completo del grafo, lo que las convierte en una herramienta poderosa para resolver problemas no lineales en múltiples disciplinas.

### 4.2.1 Estructura de GNN

Un grafo es un conjunto de objetos representado como $G = (V, E)$ donde $V$ son vértices o nodos unidos por enlaces $E$ denominados aristas o arcos, que permiten representar relaciones entre elementos de un conjunto o relaciones entre diversas entidades, como redes sociales, telecomunicaciones o sistemas biológicos. La conectividad del grafo se representa comúnmente con una matriz adyacencia $A_V$ y representa la conectividad de las aristas $e_k$ de un par de nodos $n_u$ y $n_v$ con un valor de 1 y 0 en caso

contrario. Los grafos, se puede clasificar como dirigido o no dirigido de acuerdo con sus vértices (Swaminathan et al., 2021). Además, cada conexión $e_{u,v} \in E$ tiene propiedades asignadas, como en nuestro caso particular: ancho de banda (λ) y retardo (μ) en los enlaces.

Las GNN utilizan una arquitectura única basado en un modelo de entrada-salida basado en la estructura de un grafo, lo que significa que los atributos de los nodos, las aristas y el contexto global del grafo se procesan y transforman de manera iterativa mientras se preserva la estructura de conectividad del grafo (Jiang, 2022; Y. Li et al., 2018; Vesselinova et al., 2020b; Ye et al., 2022). Este enfoque permite que las GNN integren información local y global, lo que resulta esencial para capturar las relaciones complejas inherentes a los grafos.

El núcleo del aprendizaje en las GNN radica en un proceso conocido como propagación de mensajes propuesto por (Gilmer et al., 2020) utilizando los esquemas de arquitectura Graph Nets introducidos por (Battaglia et al., 2018). Este proceso permite que los nodos intercambien información con sus vecinos, actualizando su estado en cada iteración. La actualización del estado $h_v^t$ de un nodo $v$ en el tiempo $t$ se puede describir como:

$$h_v^t = \sum_{u \in N(v)} f(x_v, x^e(u,v), x_u, h_u^{t-1}) \tag{38}$$

El estado oculto de un nodo $h_v^t$ en el tiempo $t$ representa las características internas que el nodo ha adquirido tras procesar la información de su vecindario. El conjunto de nodos $v$, denotado como $N(v)$, incluye todos los nodos con los que $v$ esta directamente conectado. Los atributos de los nodos $u$ y $v$ se representan como $x_v$ y $x_u$, respectivamente, mientras $x^e(u,v)$ describe las propiedades específicas de la arista que conecta los nodos $u$ y $v$. Además, el estado oculto previo de cada nodo vecino $u$ en el tiempo $t-1$ se denota como $h_u^{t-1}$.

La función $f$ combina los atributos de los nodos, como las características de las aristas y los estados ocultos previos para actualizar el nuevo estado oculto de cada nodo, permitiendo que recopile y procese información relevante de su entorno. Este proceso asegura la estabilidad del sistema mediante un mapeo de contracción, promoviendo la convergencia de las características a un espacio consistente como se muestra en la Figura 16.

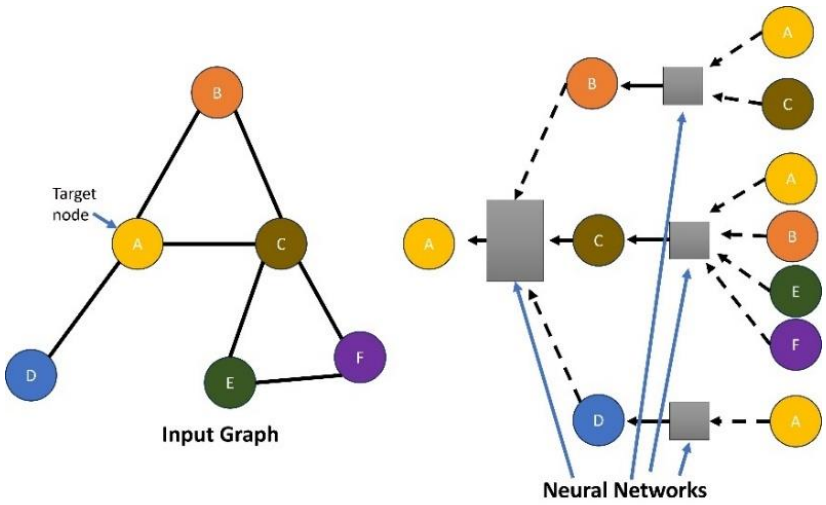


**Figura 16.** Proceso de construcción de Red Neuronal de Grafos (GNN).

### 4.2.2 Función de atención en GNN

En los modelos GNN estándar, la actualización de las características de un nodo generalmente se basa en el promedio de las características de sus vecinos. Aunque este enfoque es eficiente en grafos homogéneos, resulta limitado en escenarios de grafos heterogéneos, donde las conexiones entre nodos pueden tener diferentes grados de relevancia o importancia. Para superar estas limitaciones, se introdujeron mecanismos de atención, que asignan pesos diferenciados a las contribuciones de los vecinos en función de su importancia relativa en la tarea de aprendizaje (C. Sun et al., 2023; Vrahatis et al., 2024).

El mecanismo de atención se basa en la idea de calcular un coeficiente que mida la relevancia de cada vecino en la actualización del nodo central. Este coeficiente de atención, denotado como $\alpha_{u,v}$, se obtiene mediante una puntuación de atención $e_{i,j}$, que evalúa la importancia del nodo $v$ para el nodo $u$. Posteriormente, estas puntuaciones se normalizan utilizando la función softmax, como se muestra en la ecuación:

$$\alpha_{u,v} = \frac{exp^{e_{u,v}}}{\sum_k exp^{e_{u,k}}} \tag{39}$$

El mecanismo de atención, junto con la normalización mediante softmax, permite a las Redes Neuronales de Grafos (GNN) adaptarse eficazmente a grafos heterogéneos. Este enfoque prioriza dinámicamente las conexiones más relevantes, asignando mayor peso a las relaciones críticas y mejorando la precisión en tareas específicas como la clasificación de nodos, la predicción de enlaces y la clasificación de grafos.

Al capturar tanto interacciones locales como patrones globales, las GNN logran un equilibrio entre detalle y contexto. La normalización asegura que las contribuciones de los vecinos estén correctamente ajustadas, mejorando la estabilidad del modelo y su capacidad para generalizar a nuevos datos. Este enfoque convierte a las GNN en herramientas robustas y versátiles para el análisis de datos estructurados.

## 4.3 Redes Neuronales de Grafos Espacio-Temporales (STGNN)

En numerosos contextos reales, los datos no solo están estructurados únicamente como grafos que representan relaciones estáticas, sino que también cambian a lo largo del tiempo. Ejemplos tales como el tráfico urbano, las plataformas de redes sociales y las redes de telecomunicaciones presentan patrones que cambian en función de factores espacio-temporales. Para abordar estos escenarios, las Redes Neuronales de Grafos Espacio-Temporales (STGNN, por sus siglas en inglés) combinan la estructura gráfica con componentes que modelan la dinámica temporal, ofreciendo una solución integral para comprender y predecir patrones (Jin et al., 2022).

La principal innovación de las STGNN radica en su capacidad para integrar relaciones espaciales y dependencias temporales en un marco único. En este contexto, la representación de cada nodo se define tanto por sus interacciones con vecinos espaciales como por su evolución a través del tiempo. Este enfoque holístico permite capturar patrones en sistemas dinámicos, superando las limitaciones de las GNN tradicionales, que se centran únicamente en la estructura estática de los grafos.

Las STGNN combinan Redes Neuronales Recurrentes (RNN) para modelar las dependencias temporales con operaciones de convolución sobre grafos (GConv) para capturar relaciones espaciales. Las dependencias temporales se describen mediante el estado oculto de un nodo que evoluciona en función de su historial, de acuerdo con la Ecuación 40:

$$H^t = \delta\left(\mathrm{w}x^t + UH^{(t-1)} + b\right) \tag{40}$$

Donde $H^t$ es el estado oculto en el tiempo $t$, $x^t \in R^{n.d}$ es la matriz de características del nodo en el momento $t$, $w$ y $u$ son matrices de pesos entrenables, y $b$ es el valor del sesgo. Esta ecuación que las STGNN integren información temporal al considerar la evolución de las características de los nodos en función de su historial.

Para capturar las relaciones espaciales entre nodos, se utiliza una convolución sobre grafos (GConv), que extiende la operación de convolución a estructuras de grafo. La integración de las dimensiones espaciales y temporales se expresa en la ecuación 41:

$$H^t = \delta(\mathrm{Gconv}(x^t, A; w) + Gconv(H^{t-1}, A; u) + b) \tag{41}$$

Donde $A$ es la matriz de adyacencia que describe la conectividad del grafo, $w$ y $u$ son parámetros entrenables que manejan la información actual y pasada, respectivamente, y $b$ es el sesgo. Esta ecuación asegura que las STGNN no solo aprendan de las estructuras espaciales del grafo, sino también de la evolución temporal de los datos.

# Capítulo 5. Desarrollo experimental

En el panorama actual de las redes de telecomunicaciones, garantizar la calidad de servicio (QoS) se ha convertido en un aspecto esencial debido al crecimiento en la demanda de aplicaciones de alta exigencia, como la transmisión en tiempo real y los servicios en la nube. A pesar de que las Redes Definidas por Software (SDN) ofrecen un marco de gestión centralizada y flexible, los enfoques de enrutamiento continúan con métodos tradicionales que presentan limitaciones para asegurar la QoS bajo condiciones de tráfico extremo (Suarez-Varela et al., 2023).

El proceso de enrutamiento en SDN consiste en establecer rutas entre un nodo de origen y un nodo destino mediante la implementación de reglas de flujo en los dispositivos de red, utilizando protocolos como OpenFlow para gestionar la transmisión de datos. Sin embargo, este enfoque tiende a priorizar la selección de la ruta más corta sin considerar parámetros específicos que requieren las aplicaciones, como el ancho de banda, el retardo admisible o tolerancia en las fluctuaciones del retardo (jitter). Esto puede limitar el rendimiento en escenarios donde las aplicaciones exigen un rendimiento estricto y este enfoque puede no resultar conveniente.

Para abordar este desafío, el presente trabajo propone la integración de un agente inteligente en el proceso de enrutamiento de SDN (Figura 17). Dicho agente analiza los requerimientos de QoS de cada aplicación, clasificados previamente siguiendo la metodología propuesta por (Gonzalez-Franco et al., 2023). Con base en dicha información, el agente evalúa diversas rutas potenciales entre los nodos origen y destino, seleccionando la que mejor se ajuste a los parámetros requeridos por la aplicación.

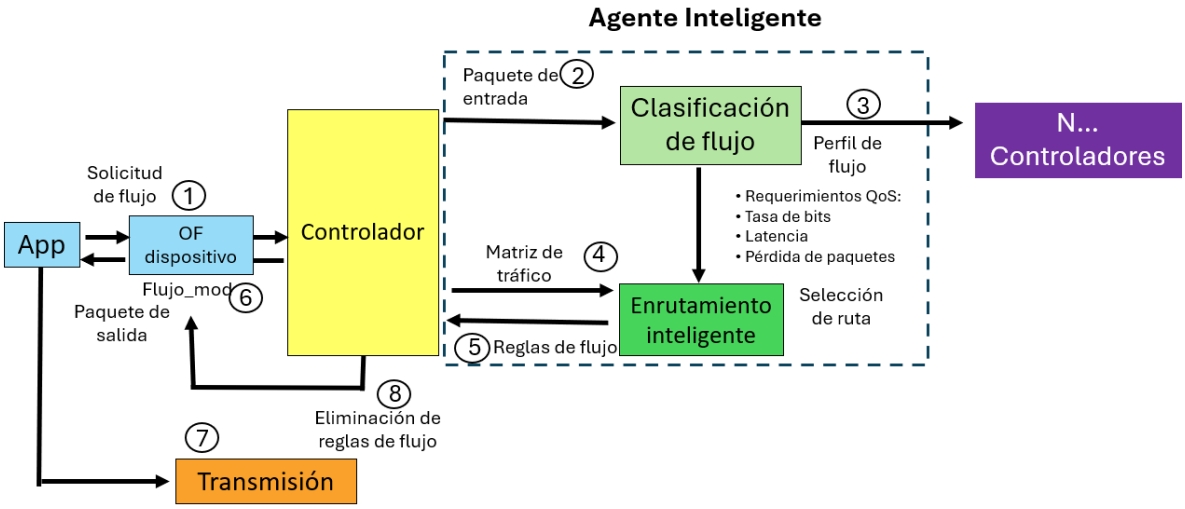


**Figura 17.** Arquitectura propuesta para el proceso de enrutamiento en SDN.

La ruta determinada por el agente se comunica al controlador SDN, el cual se encarga de configurar las reglas de flujo pertinentes en los dispositivos OpenFlow para habilitar la transmisión de datos. Una vez que concluye la transmisión, estas reglas de flujo se eliminan de los dispositivos, liberando los recursos para atender nuevas solicitudes de flujo. Este procedimiento optimiza el uso de recursos de la red y asegura que las aplicaciones reciban el nivel de QoS necesario, mejorando significativamente el rendimiento en escenarios de alta demanda.

Para desarrollar este agente inteligente, se han considerado dos enfoques distintos, ambos orientados a mejorar la eficiencia del enrutamiento en SDN y asegurar la calidad de servicio. El primer enfoque aborda el enrutamiento como un problema de optimización multi-objetivo, mediante algoritmos evolutivos para mejorar el rendimiento de la red. Estos algoritmos buscan optimizar simultáneamente diversos criterios, como minimizar la latencia y maximizar el ancho de banda. Para ello se han evaluado dos variantes: Optimización Multi-objetivo por Descomposición (MOEA/D) y el algoritmo MOCell, ambos descritos en el capítulo 3. El algoritmo MOEA/D descompone el problema en subproblemas más sencillos que se resuelven en paralelo, mientras que el MOCell emplea operadores evolutivos como la mutación y el cruce en subpoblación, generando diversas soluciones que posteriormente se evalúan de forma conjunta para para conformar un nuevo conjunto de soluciones factibles (población). Estos métodos permiten al agente identificar rutas que no solo cumplen con los requisitos de QoS, sino que también optimizan el rendimiento global de la red.

El segundo enfoque explora el uso de redes neuronales de grafos (GNN) para anticipar el estado de la red y mejorar la toma de decisiones en el enrutamiento. En este modelo, la red se representa como un grafo, donde los nodos corresponden a los vértices y los enlaces entre ellos se modelan como aristas. Gracias a esta representación, las GNN pueden capturar las relaciones topológicas de la red con gran precisión, proporcionando una base robusta para el análisis de tráfico en la red (Kumar et al., 2022). Al procesar datos históricos del tráfico y el estado actual de la red, las GNN pueden predecir el comportamiento futuro de los enlaces, como la concurrencia de congestiones en los enlaces. De esta forma, el agente es capaz de ajustar dinámicamente las rutas, adaptándolas a las condiciones previstas. Este enfoque predictivo brinda una gestión proactiva del tráfico, preservando la QoS y reforzando la estabilidad de la red.

En conjunto, ambos enfoques ofrecen perspectivas complementarias para abordar el problema de enrutamiento en SDN. Por un lado, los algoritmos evolutivos abordan el problema desde una perspectiva de optimización multi-objetivo, permitiendo equilibrar los criterios de calidad de servicio. Mientras, que las GNN brindan un enfoque predictivo que permite anticiparse a las condiciones cambiantes de la red.

Así, el agente inteligente puede seleccionar rutas que cumplan con los requerimientos de las aplicaciones, y, al mismo tiempo, mantengan el desempeño óptimo considerando la variabilidad inherente al tráfico en la red.

## 5.1 Propuesta de enrutamiento

En el contexto de las redes SDN, optimizar el uso de los recursos de la red representa un desafío crucial, especialmente debido al incremento en el tráfico y a la diversidad de aplicaciones con requerimientos específicas. La centralización y la flexibilidad características de las SDN ofrecen un entorno proactivo para administrar dinámicamente los flujos de datos. Sin embargo, también requieren métodos que aborden problemas críticos como la congestión, el uso ineficiente de los recursos y el incumplimiento de los requisitos de calidad de servicio de las aplicaciones. En este contexto, el enrutamiento inteligente representa una solución prometedora para maximizar la eficiencia de la red y garantizar un rendimiento adecuado a las necesidades de las aplicaciones.

A diferencia de un enrutamiento convencional que se limita a establecer la ruta entre nodos de origen a destino, el enrutamiento inteligente optimiza las rutas considerando múltiples criterios, como la latencia, el ancho de banda, el retardo y la estabilidad de los enlaces. Además, integra la gestión de prioridades, asignando recursos de acuerdo con los requerimientos específicos de cada flujo, lo que permite una gestión más eficiente y alineada con las necesidades de cada aplicación. En este trabajo se consideraron las siguientes métricas de enrutamiento:

- Distancia mínima**:** selecciona la ruta que minimiza la cantidad de enlaces entre el nodo de origen $s$ y el nodo de destino $d$. Cada enlace se asume con un costo unitario, y la distancia total de la ruta $P$ se calcula al sumar los enlaces que la componen:

$$H = \text{Min}(\sum_{(u,v)\in P} e) \tag{42}$$

- Costo mínimo de ancho de banda: se identifica la ruta que ofrece el mayor ancho de banda disponible. En este trabajo se modela mediante una función de minimización inversa para optimizar la capacidad de transmisión. La expresión de esta métrica se expresa como:

$$B = \text{Min}\,(\frac{1}{\lambda_e}),\ \forall e \in P \tag{43}$$

donde $\lambda_e$ representa el ancho de banda disponible en cada enlace $e$ de la ruta $P$.

- Mínimo retardo: busca la ruta que minimice el retardo total entre el origen y el destino, evaluando la suma de los retardos en cada enlace que conforman la ruta. Esta métrica se define como:

$$D = Min\ (\sum e_\mu) \tag{44}$$

Donde $e_\mu$ representa el retardo de cada enlace $e$ de la ruta $P$.

- Pérdida de paquetes: selecciona la ruta que minimiza la pérdida de paquetes, a partir de la probabilidad de pérdida en cada enlace de la ruta. Esto se representa mediante:

$$P_L = min\ (\rho(P)) \tag{45}$$

Donde $\rho(P)$ denota la probabilidad de pérdida de paquetes a lo largo de la ruta $P$.

Cada una de estas métricas desempeña un rol particular: la función de retardo es aditiva, mientras que las funciones de ancho de banda y pérdida de paquetes son cóncavas y multiplicativas, respectivamente. Estas métricas permiten garantizar la QoS de las aplicaciones, ajustándose a los requerimientos mínimos establecidos por estándares como 3GPP y 5QI (Zahid Ghadialy, 2021), para ofrecer un servicio eficiente y adaptado a las necesidades específicas de cada aplicación. Además, para evaluar el rendimiento de la ruta, se utiliza el throughput (ecuación 12) que cuantifica la entrega efectiva de datos en un intervalo de observación. Este indicador es crucial para lograr una transmisión eficiente y confiable.

La eficiencia del enrutamiento es esencial para asegurar una entrega de datos oportuna. Cuando la meta va más allá de simplemente establecer las conexiones entre nodos y se busca garantizar la calidad de la transmisión, resultando imprescindible implementar un enrutamiento que contemple la QoS. Este enfoque debe minimizar (ecuación 46) considerando las métricas descritas previamente (ecuaciones 42-45), como el retardo, la pérdida de paquetes y el ancho de banda disponible, logrando así, no solo cumplir con los requisitos técnicos de las aplicaciones, sino también maximizar el rendimiento global de la red.

$$Min_{e \in P}(D, B, P, H) \tag{46}$$

## 5.2 Implementación de algoritmos evolutivos

Los algoritmos evolutivos, tales como los genéticos, constituyen herramientas altamente efectivas para abordar problemas de optimización, como puede ser el caso del enrutamiento. Estos algoritmos inician con una población inicial de soluciones potenciales, donde cada individuo representa una posible ruta en la red. A lo largo de múltiples iteraciones (generaciones), se aplican operadores genéticos como la mutación y la recombinación, los cuales modifican y combinan las características de los individuos, para promover progresivamente la mejora en la calidad de las soluciones. Este proceso, detallado en el capítulo 3, permite explorar de manera eficiente el espacio de búsqueda y refinar las rutas conforme a los requerimientos establecidos por los flujos.

En el escenario de enrutamiento en redes, el propósito general es establecer rutas eficientes entre los nodos de origen y destino, atendiendo múltiples factores como latencia, ancho de banda disponible, pérdida de paquetes y número de dispositivos intermedios. El proceso comienza con la generación de rutas, a menudo de manera aleatoria, donde arreglos numéricos que representan diferentes trayectorias en la red. Aunque esta técnica es simple y proporciona un punto de partida, no garantiza la eficiencia ni el cumplimiento de los requisitos QoS; por ello, los mecanismos evolutivos intervienen para optimizar las rutas a lo largo de diversas generaciones.

Los operadores genéticos desempeñan un rol fundamental en la mejora de las soluciones iniciales. La mutación introduce cambios aleatorios en los individuos (rutas), promoviendo la diversidad en la población y evitando que el proceso se estanque en soluciones subóptimas, mientras que la recombinación (o cruce), por su parte, fusiona características de dos soluciones existentes para generar nuevas propuestas, explorando así variantes que no habrían sido consideradas de otra manera.

### 5.2.1 Codificación de rutas para algoritmos evolutivos

Un método básico para generar rutas de enrutamiento consiste en la búsqueda aleatoria, representando las rutas mediante matrices de números enteros generados de forma aleatoria. La población inicial se construye asignando identificadores de prioridad al azar a cada nodo, de acuerdo con el tamaño de la topología, para luego validar cada ruta utilizando un proceso de decodificación que emplea la matriz de adyacencia.

Para codificar una ruta, se construye una cadena que contiene el número total de nodos en la red en un orden aleatorio. Cada posición de la cadena representa un gen, que a su vez le corresponde un identificador de prioridad (ID). El cromosoma resultante (cadena) define una ruta potencial, donde el primer gen indica el nodo de origen, y se recorren los genes sucesivos hasta alcanzar el nodo de destino. Durante la decodificación, la matriz de adyacencia establece la secuencia de los nodos de acuerdo con el ID de prioridad de los genes. Por ejemplo, para codificar una ruta entre $H_1$ y $R_1$ de la topología de la Figura 18, el procedimiento es el siguiente:

Codificación:

1. Se genera una cadena con 10 elementos, equivalente al número de nodos de la red.

2. Cada elemento de la cadena toma un valor aleatorio entre 1 al 10.

3. Un posible cromosoma es: 5, 8, 2, 1, 4, 10, 6, 7, 3, 9.

Decodificación:

1. Mediante la matriz de adyacencia (Tabla 3), cada gen se representa en orden decreciente de prioridades para definir la ruta.

2. La ruta inicia en $H_1$ (gen de origen) que solo está conectado con $R_4$.

3. Desde $R_4$, de $R_4$ se tienen los nodos adyacentes $R_2$ (ID = 8) y $R_5$ (ID = 4), selecciona $R_2$ por tener un ID más alto.

4. En $R_2$, se elige $R_1$ como siguiente nodo, también debido a su mayor valor de ID.

5. Al contemplar la decodificación, se arriba al nodo destino propuesto ($R_1$).

6. La ruta resultante es: $H_1$, $R_4$, $R_2$, $R_1$.

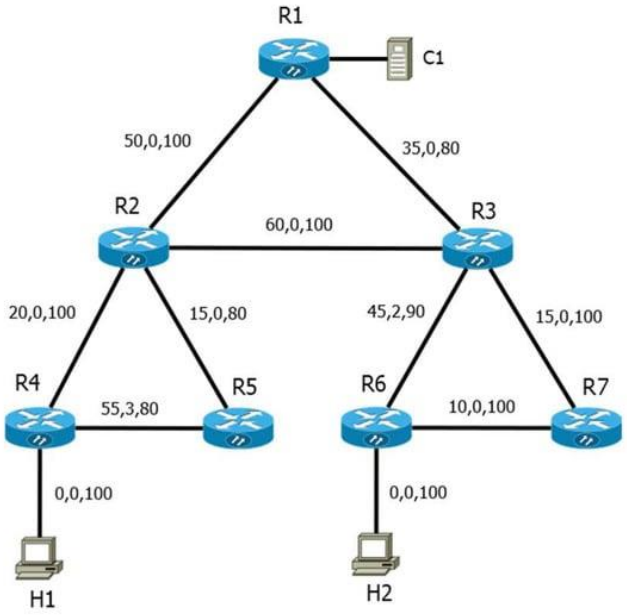


**Figura 18.** Topología para codificación: retardo (ms), tasa de pérdida de paquetes (%) y ancho de banda (mbps).

**Tabla 3**. Matriz de adyacencia para decodificación de red figura 18. Nodo de origen $\boldsymbol{H_1}$ y nodo destino $\boldsymbol{R_1}$.

| Cromosoma | 5 | 8 | 2 | 1 | 4 | 10 | 6 | 7 | 3 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Nodos** | **R1** | **R2** | **R3** | **R4** | **R5** | **R6** | **R7** | **H1** | **H2** | **H3** |
| **R1** | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **R2** | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **R3** | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| **R4** | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| **R5** | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **R6** | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| **R7** | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| **H1** | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **H2** | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **H3** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

Para refinar las soluciones en cada generación, los algoritmos evolutivos recurren a operadores de cruce (corte en un punto, corte en múltiples puntos y ciclo de cruzamiento) que combinan de forma eficiente las características de los progenitores, junto con los operadores de mutación (intercambio, inserción y mezcla) que mantienen la diversidad genética de la población y evitan la convergencia. Durante el cruce, se generan descendientes más aptos y diversos, preservando la validez de las rutas al excluir los nodos de

origen y destino. Por otra parte, la mutación introduce modificaciones controladas en los cromosomas, adaptando las rutas y permitiendo la exploración de nuevas configuraciones en el espacio de búsqueda.

En conjunto, los operadores de cruce y de mutación favorecen posibilitan un equilibrio continuo entre la exploración de nuevas soluciones y la explotación de las características genéticas más prometedoras, propiciando mejoras constantes en la calidad y adaptabilidad de las rutas generadas. Los aspectos técnicos de la implementación se describen detalladamente en el capítulo 3, mientras que los resultados específicos obtenidos se presentan en el siguiente capítulo.

### 5.2.2 Representación de la función de aptitud

En la primera parte de este trabajo el enrutamiento se modela el enrutamiento como un problema multi-objetivo, definiendo cuatro funciones objetivo ($f_1, f_2, f_3, f_4$) que representan parámetros clave de calidad de servicio: retardo, pérdida de paquetes, ancho de banda disponible y distancia. Cada función objetivo describe un aspecto específico del desempeño de la red, permitiendo identificar rutas idóneas bajo diversos criterios:

- **Función $f_1$:** minimización del retardo (Ecuación 43): Esta función tiene como objetivo minimizar el costo de retardo y latencia en la ruta, calculado como la suma de los retardos en cada enlace. Dado que el retardo sigue una regla métrica aditiva, la optimización se enfoca en reducir la latencia de extremo a extremo de las rutas candidatas.

- **Función $f_2$**: minimización de la pérdida de paquetes (Ecuación 44): La segunda función busca minimizar la pérdida acumulada a lo largo de una ruta dado que tiene un valor multiplicativo. Al optimizar esta métrica, se garantiza una transmisión más confiable.

- **Función $f_3$:** optimización del ancho de banda disponible (Ecuación 42): Esta función selecciona el ancho de banda mínimo entre todos los enlaces de la ruta. Dado que el ancho de banda en una ruta está limitado por el enlace más restrictivo. Al maximizar esta métrica se garantiza una capacidad de transmisión suficiente para las aplicaciones.

- **Función $f_4$:** optimización de la distancia (Ecuación 41): La última función objetivo busca minimizar la distancia total de la ruta, seleccionado la que presente menos enlaces entre el nodo de origen

y el de destino. Al reducir el número de enlaces intermediarios, se disminuyen las probabilidades de congestión y de fallos en dichos enlaces, lo que refuerza la estabilidad global de la red.

Estas funciones objetivo se integran de forma complementaria para identificar rutas que no solo satisfagan los requisitos de calidad de servicio (QoS), sino que también optimicen el desempeño general de la red. Al abordar múltiples parámetros, el enfoque multi-objetivo permite equilibrar prioridades y adaptarse a las necesidades específicas de cada aplicación, incluso cuando estas métricas puedan entrar en conflicto.

La formulación del problema mediante estas funciones proporciona un marco sólido, que permite a los algoritmos evolutivos evaluar con precisión la aptitud de cada solución (cromosoma) y orientar el proceso evolutivo hacia rutas más efectivas. Esto asegura que las soluciones generadas sean acordes a las exigencias de QoS de las aplicaciones.

## 5.3 Enrutamiento con GNN

La gestión de calidad de servicio (QoS) en redes SDN enfrenta el desafío constante de adaptarse a las condiciones dinámicas del tráfico y de la topología de la red. Aunque los algoritmos evolutivos han demostrado ser efectivos en la optimización de rutas mediante iteraciones genéticas, esta propuesta introduce las Redes Neuronales de Grafos (GNN) como un enfoque alternativo, orientado a predecir el estado de los enlaces y a garantizar la QoS de forma proactiva. A diferencia de la optimización interactiva convencional, esta técnica anticipa posibles problemas, como congestión o degradaciones en el rendimiento, proporcionando así una solución más ágil y efectiva.

En este modelo, la red se representa como un grafo, donde los nodos corresponden a dispositivos de red como enrutadores o conmutadores, mientras las aristas simbolizan los enlaces entre ellos. Las aristas incluyen información clave, como el ancho de banda disponible, el retardo y el nivel de utilización, permitiendo caracterizar el estado de la red en un estado específico $t$. Para cada punto de tiempo, se construye un conjunto de datos que recopila las mediciones de los atributos de la red, lo que permite a la GNN procesar y aprender patrones espaciales del tráfico (ecuación 38) utilizando el esquema de propagación de mensajes descrito en ecuación 39 del capítulo 4. Este enfoque habilita la predicción del estado futuro de los enlaces, proporcionando una herramienta poderosa para la gestión de la red.

En particular, el modelo de Red de Atención en Grafos Espacio-Temporal (ST-GAT por sus siglas en inglés Spatio-Temporal Graph Attention Network) propuesto por (Wu et al., 2022) amplía esta capacidad al combinar datos espaciales y temporales para pronosticar la tasa de transferencia promedio del tráfico. A partir de los valores históricos de dicha tasa en $F$ intervalos consecutivos, el modelo es capaz de estimar estos valores para los próximos $H$ intervalos, aplicando la predicción a todos los enlaces del grafo. Esta capacidad predictiva es crucial para anticipar condiciones críticas, como congestión o fluctuaciones en el rendimiento, y tomar medidas proactivas para mitigar sus efectos.

El modelo ST-GAT procesa los datos mediante una ventana deslizante de características que combina información espacial y temporal del tráfico. Su arquitectura se basa en dos componentes principales: una red de atención gráfica (Graph Attention Network, GAT), que identifica relaciones relevantes entre los nodos del grafo, y una red neuronal recurrente (Recurrent Neural Network, RNN), que captura dependencias temporales en el tráfico. Posteriormente, una capa lineal completamente conectada traduce estas características en predicciones específicas de rendimiento, proporcionando información clave para la toma de decisiones en el enrutamiento. Este diseño, se ilustra en la Figura 19, posibilita un procesamiento de datos dinámicos y permite una gestión más eficiente y adaptativa de la red.

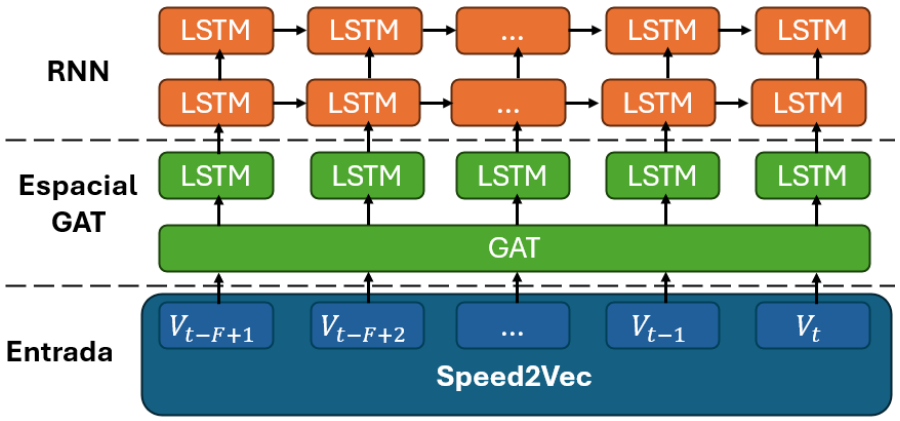


**Figura 19.** Modelo de Red ST-GAT.

El modelo propuesto para integrar el método ST-GAT en un agente inteligente para redes SDN se organiza en torno a tres componentes principales que estructuran su funcionamiento (Figura 20). El primer componente es el estado del entorno, que proporciona una descripción completa de la topología, las características de los enlaces y las métricas vigentes, como el ancho de banda disponible, el nivel de utilización de los enlaces y el retardo. Esta información es procesada por el modelo ST-GAT, capturando

patrones relevantes en el comportamiento del tráfico y generar predicciones. El segundo componente, es el sistema de recompensas, el cual evalúa las decisiones del agente en función de métricas de QoS, como el throughput y la disponibilidad del ancho de banda en los enlaces. Este sistema guía al agente para que adopte políticas de enrutamiento que optimicen el rendimiento de la red, asegurando una asignación eficiente de recursos y seleccionando rutas que cumplan con las condiciones específicos de cada aplicación. Por último, el tercer componente es la predicción del estado, que utiliza el modelo ST-GAT para anticipar eventos críticos, como congestiones o degradaciones en los enlaces, permitiendo al agente ajustar las rutas de manera proactiva, minimizando interrupciones en el tráfico y asegurando un flujo interrumpido de tráfico.

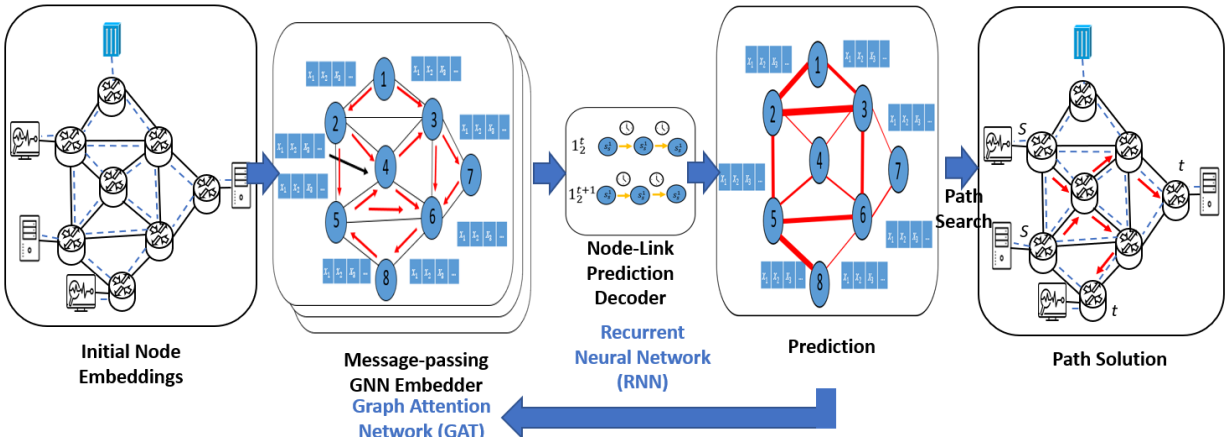


Figura 20. Representación del proceso de enrutamiento con GNN en SDN.

El funcionamiento del agente comienza con políticas básicas de enrutamiento, que se perfeccionan a través de un entrenamiento iterativo. Durante este proceso, el modelo ST-GAT analiza tanto los datos históricos como los datos en tiempo real para predecir con precisión el estado futuro de los enlaces. Este análisis permite al agente explorar diversas configuraciones de enrutamiento utilizando un enfoque de prueba y error, evaluando el impacto de las rutas propuestas con las métricas de QoS. Estas exploraciones iniciales ayudan al agente a desarrollar un entendimiento profundo de cómo las condiciones de la red interactúan con las políticas de enrutamiento, sentando las bases para estrategias más efectivas. A medida que avanza el entrenamiento, el agente refina sus decisiones, optimizando la asignación de rutas y minimizando la necesidad de ajustes reactivos, lo que resulta en un mayor rendimiento global de la red.

La integración del modelo ST-GAT en redes SDN mediante un agente inteligente ofrece beneficios considerables en comparación con métodos tradicionales y los algoritmos evolutivos. Estos últimos suelen requerir múltiples iteraciones para evaluar y seleccionar rutas óptimas, el enfoque basado en predicciones permite una respuesta más rápida y precisa al basarse en el análisis proactivo del estado de la red. Esta capacidad para anticiparse a problemas críticos reduce las interrupciones en el tráfico, asegurando una operación más estable y eficiente. Además, la capacidad del modelo ST-GAT para procesar topologías mediante grafos lo convierte en una solución altamente escalable, adecuada para manejar grandes volúmenes de tráfico y una amplia gama de tipos de flujos en la red.

# Capítulo 6. Resultados

## 6.1 Algoritmos evolutivos

Para evaluar la eficiencia y escalabilidad de los algoritmos evolutivos en problemas de enrutamiento, se optó por una topología multidominio que refleja la dificultad propia de redes de gran escala, donde abundan las rutas disponibles. Este tipo de arquitectura permiten examinar la capacidad de los métodos propuestos para seleccionar múltiples rutas que satisfagan diferentes requisitos de calidad de servicio, comparando su desempeño con los enfoques tradicionales.

Con el fin de ampliar la escala del escenario, se diseñó un escenario en el que dos centros de datos (Dominio A y B) se encuentran interconectados a través de una nube muy densa de dispositivos (Dominio C). Esta nube se caracteriza por tener una topología en malla, caracterizada por tener la mayor cantidad posible de enlaces entre los nodos, lo que incrementa significativamente la redundancia y las trayectorias disponibles entre dos nodos. La Figura 21 ilustra esta topología, organizada en tres dominios, cada uno con su propio controlador SDN, permitiendo una gestión autónoma de las políticas de enrutamiento en cada segmento de la red.

En cuanto a la distribución de los nodos, los Dominios A y B cuentan con 29 nodos cada uno, mientras que el Dominio C, lo integra una red de 196 nodos. Los enlaces entre estos nodos han sido configurados para reflejar condiciones realistas de operación propias de un proveedor de servicios de internet (ISP) se configuraron métricas como ancho de banda disponible, retardo y pérdida de paquetes, asegurando que los escenarios evaluados sean representativos de redes de gran escala.

La inicialización de la población en los algoritmos evolutivos se llevó a cabo asignando 100 individuos a los Dominios A y B, y 400 individuos al Dominio C, duplicando la cantidad de individuos (posibles soluciones) respecto al número de nodos en cada dominio. Para el algoritmo MOCell, se estableció una vecindad de tamaño 8 y se utilizó el método de selección por torneo. Además, se definió una probabilidad de cruce de 0.6 mediante el operador SPX y una probabilidad de mutación de 0.6 con un operador swap, con el fin de optimizar las rutas de acuerdo con criterios de QoS, maximizando tanto la eficiencia en la transmisión como el rendimiento global de la red.

Por otro lado, en el algoritmo MOEA/D, se implementaron cuatro vectores de peso asociados a las funciones de aptitud descritas en las ecuaciones 42-45. Dichas funciones abordan el problema multi-objetivo (Ecuación 46) mediante la minimización del retardo de extremo a extremo, la reducción de la pérdida de paquetes, la optimización del costo del ancho de banda y la selección de rutas con la distancia más corta. Esta combinación asegura que las rutas seleccionadas no solo cumplan con los requisitos de QoS, sino que también mejoren la utilización de los recursos de la red, consolidándose, así como una solución eficiente y adaptable en entornos de alta demanda.

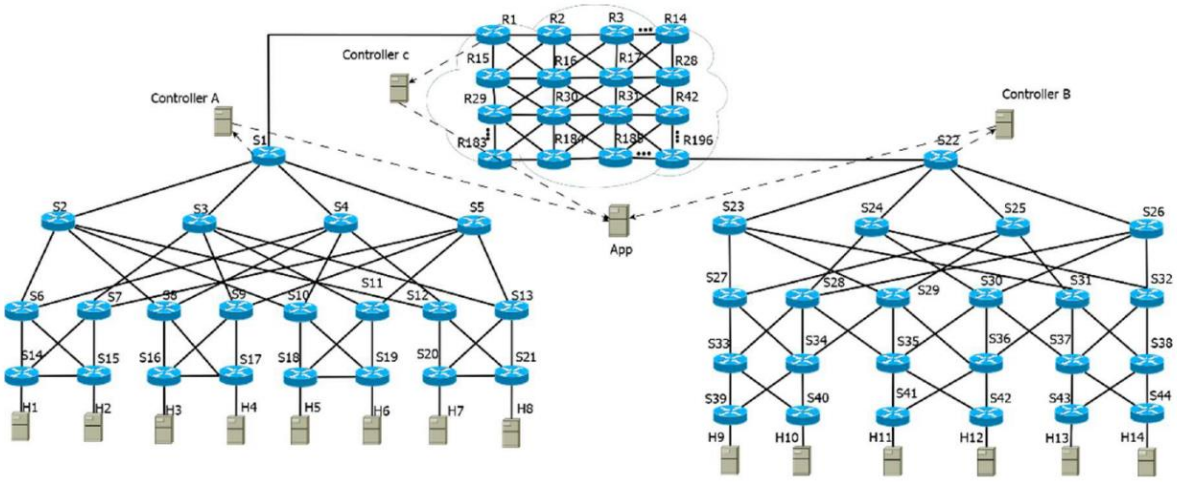


**Figura 21.** Configuración de topología para algoritmos genéticos.

La selección de rutas y la implementación de los algoritmos evolutivos se llevaron a cabo en MATLAB, utilizando el controlador OpenDayLight (ODL) con el módulo RESTCONF para la gestión de la red SDN. La simulación de la topología de red se realizó en Mininet, un simulador ampliamente utilizado en entornos de Redes Definidas por Software debido a su compatibilidad y flexibilidad con otros controladores y aplicaciones.

El entorno de simulación se ejecutó en una máquina virtual configurada con Linux Debian 9, equipada con 12 CPU, 16 GB de RAM y 100 GB de almacenamiento en disco (NLSAS). Las herramientas de software utilizadas:

- Sistema operativo: Linux Debian 9.
- Mininet versión 2.3.
- Controlador SDN OpenDayLight (ODL).

- MATLAB 2021 para la programación basada en MOCell.
- Python 3 para scripts de comunicación con API REST.
- Iperf para la medición del rendimiento en transferencias de flujo de datos.

Los controladores ODL para los Dominios A, B y C fueron configurados de acuerdo con la topología de red simulada (Figura 21) y ajustados a los parámetros de enlace, incluyendo ancho de banda, pérdida de paquetes y retardo, tal como se detalla en el apéndice A. Cada dominio y su controlador SDN fueron integrados con Mininet, permitiendo una simulación adecuada de la topología y condiciones de red en la experimentación.

### 6.1.1 Resultados de algoritmo MOEA/D y MOCell

A diferencia de los problemas con un único objetivo, donde la optimización se determina a partir de una única solución, en el enrutamiento multi-objetivo nos esforzamos en buscar un conjunto de soluciones de soluciones no dominadas que cumplan simultáneamente con las restricciones asociadas a los criterios de QoS de los flujos. Para cada dominio definido, se realizaron 30 ejecuciones independientes de los algoritmos con un punto de quiebre en 500 generaciones. En la Figura 22 se muestran las poblaciones inicial y final para las implementaciones de MOCell en cada uno de los dominios, donde los valores están normalizados de acuerdo con el valor máximo encontrado en cada métrica. Se observa una disminución significativa en el espacio de mejores soluciones, representando parámetros de QoS más altos. En cuanto a la selección de ruta de las soluciones no dominadas, es posible considerar un orden de prioridad de las funciones conforme a las exigencias de cada flujo. En este caso para el experimento, se jerarquizó primero el retardo, segundo la pérdida de paquetes y tercero el ancho de banda, de acuerdo con el desempeño de los diferentes algoritmos evaluados en la Tabla 4 y sus flujos, como se muestra en las Figuras 23 y 24.

Para evaluar el rendimiento de los métodos de enrutamiento, se analizó la transmisión de datos entre los nodos de extremo a extremo de los centros de datos A y B ($H_1$ y $H_{14}$). Para ello se consideraron diversos métodos de enrutamiento, incluyendo: la ruta más corta, la ruta con mayor ancho de banda disponible, la ruta con menor retardo, la ruta de menor pérdida de paquetes, además de una ruta con criterios de ponderación y la ruta seleccionada por MOCell. Para medir el rendimiento de las rutas, se empleó la aplicación IPERF bajo los protocolos TCP (orientado a conexión) y UDP (sin conexión) en intervalos de 200

segundos para cada transmisión de datos, para asegurar la consistencia de los resultados se replicó el experimento 30 veces.

Mientras los métodos tradicionales utilizan estrategias de exploración por descubrimiento, el algoritmo MOCell realiza consultas al controlador SDN de la topología de la red para calcular la ruta. La Tabla 4 muestra las rutas generadas por cada método de enrutamiento junto con sus respectivas métricas de QoS, proporcionando un análisis comparativo del desempeño de cada estrategia.

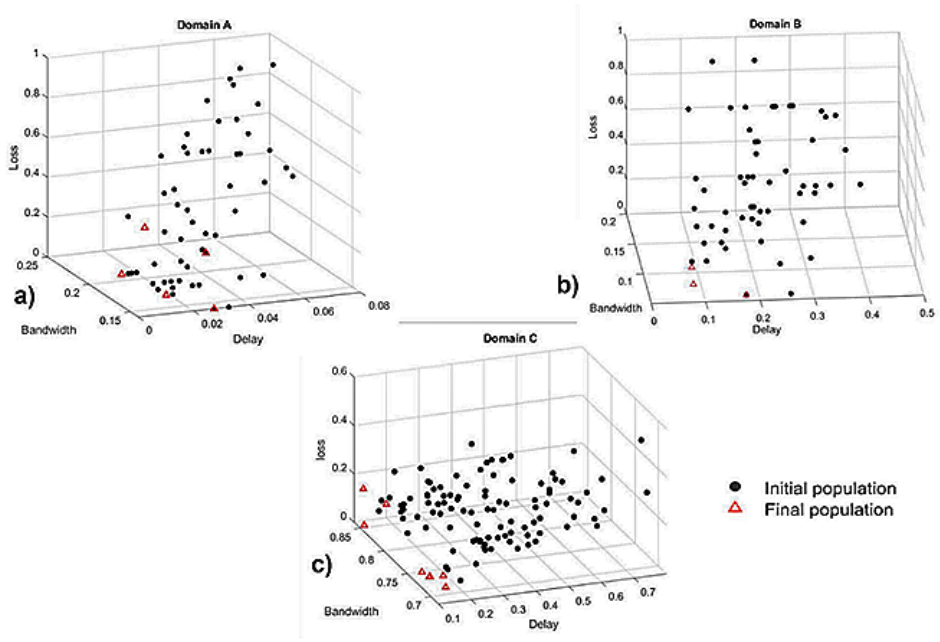


**Figura 22.** Conjunto de soluciones no dominadas obtenidas por el algoritmo genético celular multi-objetivo MOCell. a) Dominio A, b) Dominio B, y c) Dominio C.

Las condiciones experimentales fueron idénticas en todas las pruebas, aplicando los parámetros establecidos en las Tablas 10, 11, y 12 de la sección de anexos. Considerando la siguiente metodología:

1. Configurar la topología de red con tres dominios (A, B y C) y un controlador externo para cada dominio.

2. Ejecutar pruebas de rendimiento entre los nodos finales utilizando IPERF, siguiendo los criterios de optimización establecidos.

3. Transferir tráfico durante el periodo de prueba especificado.

4. Repetir el experimento utilizando la política de reenvío de flujo basada en el parámetro de optimización.

Se adoptó un enfoque interdominio para evaluar el rendimiento del flujo de datos entre $H_1$ y $H_{14}$, generando tráfico TCP y UDP con el objetivo de simular una transferencia de datos de una aplicación. Cada prueba aplicó criterios de optimización específicos para seleccionar una ruta (ver Tabla 4). Mientras los métodos tradicionales de enrutamiento consideran una única métrica, nuestra propuesta calcula y configura la ruta considerando todos los parámetros de Calidad de Servicio (QoS) disponibles. Aunque el algoritmo es adaptable a los requisitos de diversas aplicaciones, en este experimento se considera la ruta que minimizaba las métricas en orden de prioridad: retardo, ancho de banda disponible y pérdida de paquetes.

Las Figuras 23 y 24 muestran los resultados del primer experimento, con información obtenida a través de las aplicaciones IPERF y Wireshark. En negro se gráfica el tráfico total transmitido, mientras que en rojo se ilustran los paquetes retransmitidos, los cuales afectan el rendimiento de la red debido a la llegada desordenada o la pérdida de paquetes. Nuestro método mostró un rendimiento superior al minimizar todos los parámetros de QoS descritos en la Ecuación 46. Para la evaluar con MOCell, se priorizaron las funciones multi-objetivo ($f_1, f_2, f_3, f_4$ ), tal como se detalla en la sección 5.1.

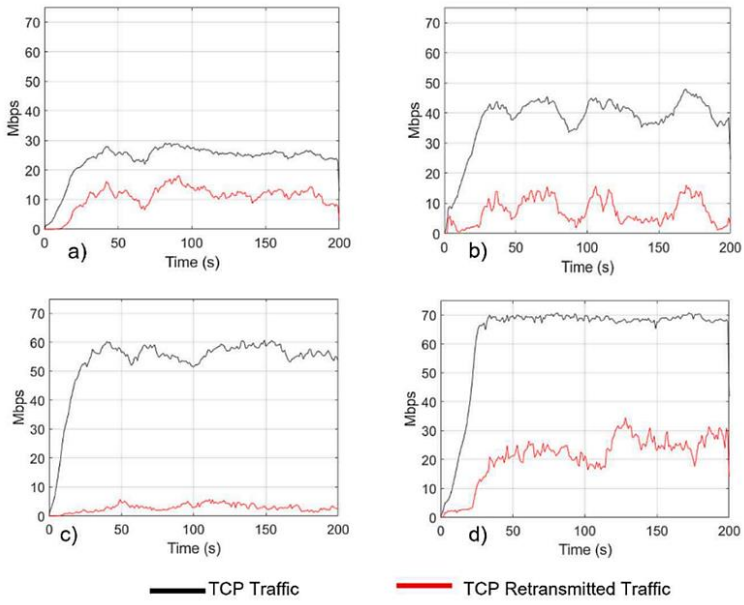


**Figura 23.** Evaluación de la capacidad de la ruta utilizando tráfico TCP. métodos: a) $\boldsymbol{H}$ (distancia), b) $\boldsymbol{B}$ (ancho de banda), c) $\boldsymbol{D}$ (retraso), y d) $\boldsymbol{P_L}$ (pérdida de paquetes).

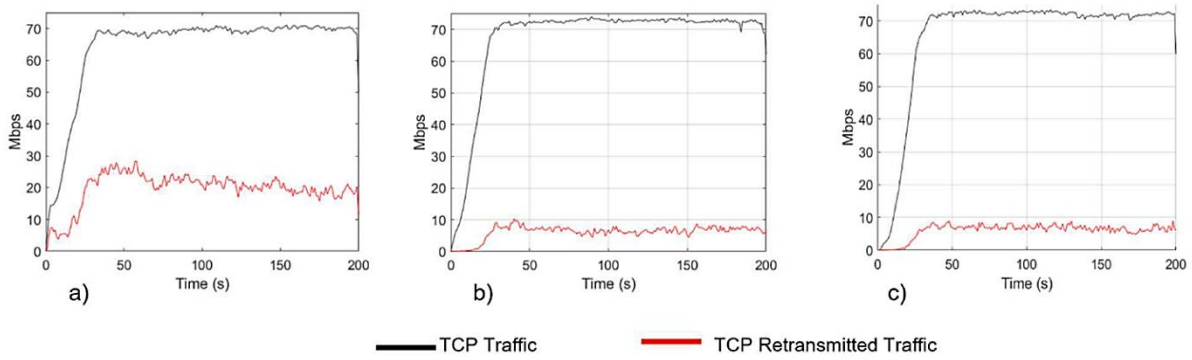


**Figura 24.** Evaluación de la capacidad de la ruta utilizando tráfico TCP. Métodos: a) $\boldsymbol{w_{B,D,P_L,H}}$ (suma ponderada), b) *MOCell* y c) *MOEA/D.*

**Tabla 4.** Resultados de rutas de enrutamiento y parámetros de QoS.

| Algoritmo | Ruta | Hops | Bandwidth (Mbps) | Delay (Ms) | Packet Loss ($10^{-3}$) | Data Rate (Mbps) |
|---|---|---|---|---|---|---|
| $\boldsymbol{H}$ | H1-S14-S6-S2-S1-R1-R16-R31-R46-R61-R76-R91-R106-R121-R136-R151-R166-R181-R196-S22-S26-S32-S38-S44-H14 | 24 | 130 | 130 | 5.5 | 23.8 |
| $\boldsymbol{B}$ | H1-S14-S6-S2-S1-R1-R16-R31-R45-R46-R59-R72-R73-R86-R101-R114-R129-R142-R155-R169-R156-R171-R172-R187-R174-R161-R147-R162-R163-R164-R178-R193-R180-R181-R196-S22-S24-S32-S37-R31-R38-S44-H14 | 42 | 166 | 166 | 3.7 | 38 |
| $\boldsymbol{D}$ | H1-S14-S7-S3-S1-R1-R16-R31-R46-R33-R47-R62-R77-R91-R106-R120-R135-R150-R164-R179-R180-R195-R196-S22-S24-S30-S37-S44-H14 | 28 | 58 | 58 | 2.6 | 53.5 |
| $\boldsymbol{P_L}$ | H1-S14-S7-S5-S1-R1-R2-R3-R17-R30-R43-R58-57-R72-R85-R99-R114-R129-R144-R145-R160-R174-R189-R190-R191-R192-R193-R194-R195-R196-S22-S26-S32-S37-S44-H14 | 35 | 138 | 138 | 0 | 63.1 |
| $\boldsymbol{w_{B,D,P_L}}$ | H1-S14-S7-S5-S1-R1-R2-R17-R32-R47-R48-R62-R77-R90-R104-R103-R116-R131-R146-R147-R162-R177-R192-R193-R194-R195-R196-S22-S24-S28-S35-S30-S37-S44-H14 | 34 | 85 | 85 | 3 | 64.4 |
| **MOCell** | H1-S14-S7-S5-S1-R1-R15-R29-R43-R57-R72-R86-R85-R100-R115-R130-R117-R118-R133-R148-R149-R164-R179-R194-R195-R196-S22-S24-S28-S35-S30-S37-S44-H14 | 33 | 75 | 75 | 2 | 66.9 |
| **MOEA/D** | H1-S14-S7-S5-S1-R1-R2-R3-R18-R33-R48-R63-R98-R92-R107-R121-R134-R149-R164-R178-R193-R194-R195-R196-S22-S24-S28-S35-S30-S37-S44-H14 | 21 | 80 | 80 | 2 | 66.37 |

El análisis de rutas seleccionados entre distintos métodos (Tabla 4) evidencia que, en promedio, los métodos tradicionales presentaron las tasas de transferencia más bajas. La ruta de menos saltos ($H$) alcanzó únicamente 23.8 Mbps, y la de ancho de banda máximo disponible ($B$) alcanzo solo 38 Mbps, frente a los 66.9 Mbps obtenidos por MOCell. Asimismo, la ruta de menor retardo ($D$) registró 53.4 Mbps, y la ruta de menor pérdidas de paquetes ($P_L$) obtuvo 63.1 Mbps. Para asegurar una comparación imparcial, se consideraron variantes basadas en el algoritmo de Dijkstra, para la selección de las rutas por los diversos métodos presentados.

A continuación, se resumen los resultados para cada ruta evaluada:

La ruta $H$ (con menor número de saltos o enlaces) registró la tasa de transferencia de datos promedio más baja, debido principalmente a la alta retransmisión del tráfico. Esto se debe a que la ruta atraviesa por múltiples nodos con elevados niveles de pérdida de paquetes, lo que redujo significativamente su eficiencia, convirtiéndola en la peor opción de esta prueba. A pesar de ser la ruta con menor número de enlaces, la frecuencia de pérdidas y retransmisiones afectó considerablemente el rendimiento, disminuyendo el tráfico efectivo.

La ruta $B$ (mayor ancho de banda disponible) presentó un alto número de saltos, lo que incrementó tanto la probabilidad de pérdida de paquetes como el retraso en la transmisión. Aunque teóricamente esta ruta ofrecía mayor capacidad, el número de nodos intermedios para alcanzar el destino aumentó la latencia y el riesgo de congestión, afectando la calidad de la transmisión, siendo un método no adecuado para aplicaciones sensibles al retardo.

Por otro lado, la ruta $P_L$ (mínima pérdida de paquetes) logró una tasa de transferencia de tráfico superior a las rutas anteriores, ya que priorizaba los enlaces con menor probabilidad de pérdida. No obstante, esta ventaja se vio compensada por la mayor cantidad de retransmisiones necesarias, ya que el retardo acumulado provocó que los paquetes llegaran en intervalos irregulares. A pesar de mejorar en términos de pérdidas, la falta de estabilidad en el tiempo de entrega disminuyó su rendimiento general.

Finalmente, la ruta ponderada, calculada a partir de una combinación de métricas de QoS como ancho de banda, retardo y pérdida de paquetes, obtuvo el mejor rendimiento en comparación con los métodos tradicionales. Al asignar pesos específicos a cada métrica, esta estrategia optimiza la selección de rutas al encontrar un equilibrio entre los distintos criterios de QoS. Gracias a su capacidad de adaptación, este método ajusta a las rutas balanceando las cargas de los pesos para brindar una mayor estabilidad y

eficiencia en comparación con los enfoques tradicionales. En conclusión, la estrategia de enrutamiento ponderado demostró ser superior, ya que maximiza el rendimiento de la ruta y minimizando los efectos adversos del retardo y la pérdida de paquetes, que son problemas recurrentes en los métodos de enrutamiento tradicionales.

Al considerar múltiples objetivos los resultados demuestran que los algoritmos evolutivos, como MOCell y MOEA/D, mejoran significativamente la selección de rutas al evaluar múltiples caminos y optimizar los parámetros de QoS. El rendimiento de la ruta se determinó con base a la cantidad de datos transmitidos exitosamente durante el ciclo de prueba (throughput), permitiendo evaluar la capacidad máxima de transmisión de la red. La eficiencia de cada ruta se calculó mediante el área bajo la curva de rendimiento usando la suma de Riemann.

El método basado en MOCell aprovechó, en promedio, el 71.08% del rendimiento de la red, superando a MOEA/D, que alcanzó un 69.23%. En comparación con los métodos tradicionales que alcanzaron una efectividad inferior al 56% (Figura 25). En términos de mejora relativa, la ruta seleccionada por MOCell obtuvo un rendimiento 54% superior a la ruta de menor distancia (H) y 35.44% mayor en comparación con la ruta de mayor ancho de banda (B). Además, MOCell superó a MOEA/D en un 1.85%, manteniendo los mismos parámetros de mutación, generaciones y selección por torneo, lo que garantizó una comparación justa entre ambos algoritmos. Durante los primeros 30 segundos de transmisión, la ruta de MOCell mostró una convergencia más rápida que la de MOEA/D, logran un rendimiento inicial superior. No obstante, tras este período, ambos métodos demostraron un desempeño similar.

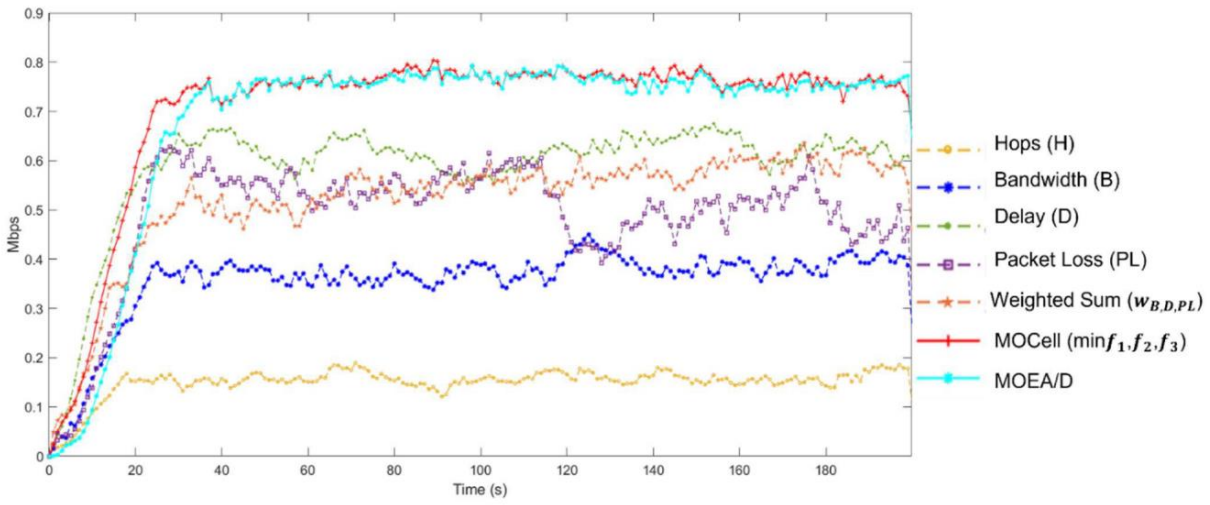


**Figura 25.** Rendimiento TCP (throughput) comparación de rutas de enrutamiento.

La Figura 26 y la Tabla 5 presentan los valores del área de rendimiento (normalizada por la capacidad máxima) obtenidos en 30 ejecuciones independientes del experimento en Mininet para cada ruta seleccionada. Los resultados muestran que las rutas basadas con la suma ponderada y la de menor pérdida de paquetes ($P_L$) presentan una mayor desviación estándar en su rendimiento en comparación con las demás rutas. Por otro lado, la ruta seleccionada con el algoritmo de retardo mínimo (D) exhibe una mediana más alta, excluyendo las rutas de los métodos MOCell y MOEA/D.

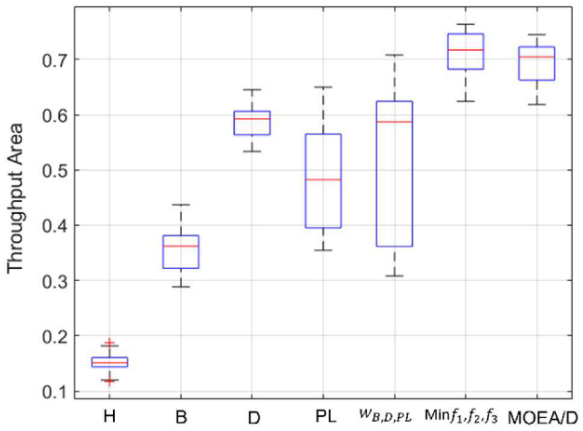


**Figura 26.** Rendimiento TCP de las rutas. Intervalo de tiempo 200 s. STDEV: $\boldsymbol{H}$= 0,01847, $\boldsymbol{B}$ = 0,04224, $\boldsymbol{D}$ = 0,02953, $\boldsymbol{P_L}$ = 0,09682, $\boldsymbol{Suma\ Ponderada}$ = 0,13832, $\boldsymbol{MOCell}$= 0,03777 y $\boldsymbol{MOEA/D}$ = 0,04104.

**Tabla 5.** Áreas de rendimiento de rutas.

| Algoritmo | Área de Rendimiento |
|---|---|
| $\boldsymbol{H}$ | 0.1511 |
| $\boldsymbol{B}$ | 0.3622 |
| $\boldsymbol{D}$ | 0.5924 |
| $\boldsymbol{P_L}$ | 0.4825 |
| $\boldsymbol{w_{B,D,P_L}}$ | 0.5872 |
| $\boldsymbol{MOCell}$ | 0.7174 |
| $\boldsymbol{MOEA/D}$ | 0.7046 |

A partir de los resultados obtenidos, se concluye que tanto la pérdida de paquetes como el retardo no dependen únicamente del ancho de banda disponible, sino también por la baja velocidad de la red, siendo

un factor determinante al asignar rutas que requieren calidad de servicio. En particular, las retransmisiones de paquetes ejercen un impacto negativo significativo el rendimiento, al aumentar la latencia y generar posibles congestiones.

Con el fin de minimizar todos los factores que impactan en el desempeño de los enlaces de extremo a extremo, es esencial aprovechar múltiples métricas de la red para obtener un mejor rendimiento en la red y QoS. Nuestra propuesta optimiza el enrutamiento mediante la evaluación de múltiples rutas disponibles, configurando los flujos en función de las necesidades de QoS en cada enlace.

En un segundo experimento, se evaluó la pérdida de datos mediante tráfico UDP entre los nodos de los extremos de los centros de datos ($H_1$ a $H_{14}$), manteniendo la misma configuración y condiciones de red del escenario anterior. Este experimento evalúa el rendimiento de las rutas seleccionadas por los mismos algoritmos presentados anteriormente. Para ello, la transmisión de datos consistió en un flujo constante de tráfico UDP con un ancho de banda de 45 Mbps durante 200 segundos, utilizando la configuración estándar de IPERF. Las condiciones de la red se mantienen consistentes con las del primer experimento, y se restablecieron para cada caso evaluado en el simulador Mininet.

La Tabla 6 muestra que todas las rutas seleccionadas por los algoritmos evaluados cumplen con el ancho de banda requerido para el flujo UDP. Sin embargo, su rendimiento varía en función de otros parámetros de red, como la pérdida de paquetes y el retardo. En esta tabla se presentan los valores promedio de rendimiento de cada ruta, mientras que en la Figura 27 ofrece una representación gráfica comparativa de la cantidad de paquetes perdidos en cada método evaluado.

**Tabla 6.** Resumen de rendimiento de flujo UDP entre H1 y H14.

| Algoritmo | Data Rate (Mbps) | Delay (ms) | Packet Loss (%) | Throughput |
|---|---|---|---|---|
| $\boldsymbol{H}$ | 43.83 | 1.081 | 0.025 | 0.9749 |
| $\boldsymbol{B}$ | 40.56 | 1.958 | 0.1067 | 0.8932 |
| $\boldsymbol{D}$ | 43.10 | 0.952 | 0.0410 | 0.9589 |
| $\boldsymbol{P_L}$ | 42.58 | 1.71 | 0.0550 | 0.9449 |
| $\boldsymbol{w_{B,D,P_L}}$ | 43.6 | 1.325 | 0.4519 | 0.9548 |
| $\boldsymbol{MOCell}$ | 44.56 | 0.986 | 0.0090 | 0.9909 |
| $\boldsymbol{MOEA/D}$ | 44.48 | 0.994 | 0.0097 | 0.9902 |

Con frecuencia, la ruta más crítica se elige a menudo por el criterio de mayor ancho de banda disponible, denominado camino B, ya que tiene la mayor capacidad para desviar flujos. Sin embargo, dicha ruta puede presentar inconvenientes con enlaces con altos niveles de jitter o pérdida de paquetes, puesto que cualquier paquete que supere la ventana de tiempo de espera o llegada desordenada afecta la calidad de la transmisión. Por otra parte, el camino $P_L$, que minimiza las pérdidas, también presenta un rendimiento limitado cuando el jitter es alto.

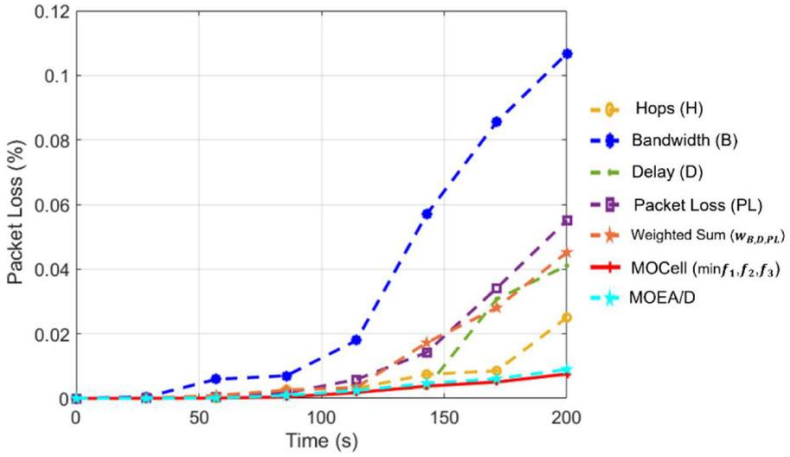


**Figura 27.** Pérdida de paquetes UDP, STDEV: $\boldsymbol{H}$= 0.00834, $\boldsymbol{B}$ = 0.03683, $\boldsymbol{D}$ = 0.01754, $\boldsymbol{P_L}$ = 0.01709, $\boldsymbol{Suma\ Ponderada}$ = 0.01343, $\boldsymbol{MOCell}$= 0.00258 y $\boldsymbol{MOEA/D}$ = 0.0021.

El método de la suma ponderada presenta un problema similar. Si bien considera varios parámetros al seleccionar la ruta, es fundamental asignar adecuadamente los pesos de cada métrica para lograr un equilibrio en el rendimiento. En este sentido, los algoritmos evolutivos, como MOCell y MOEA/D, ofrecen una alternativa más robusta, ya que proporcionan un conjunto de rutas no dominadas que optimizan múltiples métricas de forma simultánea.

En las pruebas de tráfico UDP, el método MOCell alcanzó un rendimiento del 99.09%, SUPERANDO A MOEA/D en un 0.07%. Asimismo, con otros métodos, MOCell demostró ser más efectivo: superando la ruta H (menos saltos) en 1.6%, la ruta D (menor retardo) en 3.2%, la ruta ponderada ($w_{B,D,P_L}$) en 3.6%, la ruta PL (menor pérdida) en 4.6%, y la ruta B (mayor ancho de banda) en 9.77%.

Si bien, la capacidad de enlace es un parámetro esencial para garantizar la calidad de servicio, no resulta suficiente para satisfacer las necesidades en todas las situaciones de requerimientos de las aplicaciones. Por ello, es recomendable emplear un enfoque que integre múltiples criterios en el enrutamiento, con el fin de mejorar la calidad de servicio como la experiencia del usuario, aprovechando múltiples parámetros disponibles en la red que permitan seleccionar rutas más equilibradas y eficaces.

## 6.2 Resultados GNN

Para el proceso de simulación de Redes Neuronales de Grafos (GNN), se optó por la topología de red ARPANET proveniente del conjunto de datos TopologyZoo, una base de datos que reúne más de 250 arquitecturas de diversos operadores internacionales y que se utilizada ampliamente para estudios de enrutamiento en Redes Definidas por Software (SDN). ARPANET es una de las topologías más referenciadas en la literatura, tanto por su relevancia historia como su arquitectura. Esta red contiene 34 nodos, de los cuales se configuran 7 como nodos de origen y 7 como nodos de destino, interconectados a través de 46 enlaces, tal como se ilustra en la Figura 28.

Dicho escenario proporciona un entorno adecuado para evaluar el comportamiento del tráfico de red y comparar la eficiencia de los algoritmos de enrutamiento tradicionales frente a la propuesta Smart Routing basado en GNN.

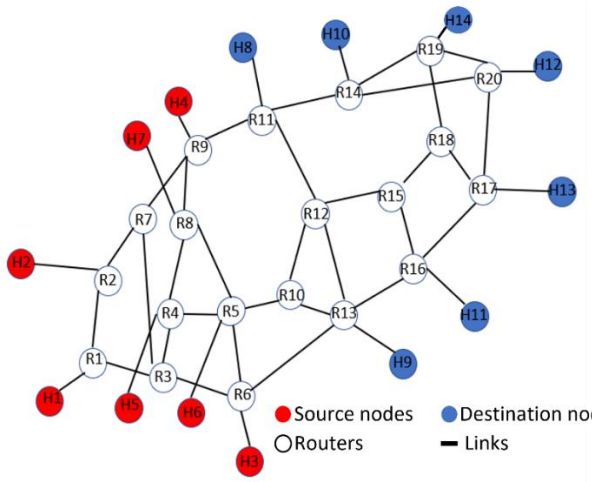


**Figura 28.** Topología experimental ARPANET de TopologyZoo.

La simulación se realizó en Mininet 2.3.0, una plataforma de código abierto que permite la emulación de redes SDN. Cada enlace se configuró con una capacidad de 150 Mbps, y los valores de retardo se asignaron de manera pseudoaleatoria entre 1 y 3 ms utilizando el algoritmo Mersenne Twister. Este algoritmo se utilizó para generar números pseudoaleatorios, tanto en la asignación de los valores de retardos como para la selección de nodos y los tiempos de envío de los flujos. El experimento fue ejecutado en un equipo con Ubuntu 20.10, provisto de un procesador Intel i7 de 12ª generación, 32 GB de RAM, y una tarjeta gráfica NVIDIA RTX A2000 de 8 GB. En este entorno, el tráfico se capturó mediante Wireshark y el modelo de predicción ST-GAT se simuló empleando PyTorch 2.0.4.

El modelo ST-GAT (Spatio-Temporal Graph Attention Network, por sus siglas en inglés), está diseñado para la predicción del estado de los enlaces en redes SDN. Su enfoque combina el aprendizaje de relaciones espaciales y temporales dentro de una red de comunicación (detallado en la sección 5.3) permitiendo anticipar posibles congestionamientos y administrar dinámicamente la gestión del tráfico de datos.

Para estructurar la información, el modelo organiza los datos en una matriz tridimensional con dimensiones $[T, F, N]$, donde $T = 5160$ (ec) corresponde al número de ventanas de tiempo disponibles, $F = 12$ indica la cantidad de características históricas utilizadas como entrada, y $N$=45 corresponde al número de enlaces dentro de la red. La Tabla 7, muestra los hyperparametros empleados en la configuración de la red que permite capturar la evolución temporal del tráfico y su distribución a través de la topología de la red.

La arquitectura del modelo combina dos enfoques: una red de Atención Gráfica (GAT, Graph Attention Networks, por sus siglas en inglés) y una Red Neuronal Recurrente (RNN, Recurrent Neural Network, por sus siglas en inglés) basa en LSTM (Long Short-Term Memory, por sus siglas en inglés). La capa GAT cuenta con $k$ cabezas de atención, que se encarga de analizar la estructura de la red, modelando la influencia de cada enlace en función de sus conexiones. Este mecanismo de atención gráfica permite que la predicción del estado de un enlace no dependa únicamente de su historial, sino que también considere el comportamiento del tráfico en enlaces vecinos, lo que resulta en una mejor representación de las interacciones dentro de la red.

Posteriormente, los datos procesados por la GAT pasan a dos capas LSTM, encargadas de modelar la evolución temporal del tráfico de la red. La primera capa, captura patrones de corto plazo en la variabilidad del tráfico en los enlaces, mientras que la segunda capa, refina la representación para detectar tendencias más complejas y relaciones a largo plazo. Finalmente, la información pasa a través de una capa lineal, cuya función es transforma la representación aprendida en predicciones para los próximos 9 intervalos de tiempo de cada enlace, proporcionando así una estimación detallada del estado futuro de la red SDN.

$$T = N_{días} * ventas/día \tag{47}$$

Donde:

$$\frac{ventanas}{día} = N_{slot\_día} - (F + P) + 1$$

**Tabla 7.** Configuración de Red ST-GAT.

| Parámetros | Valor |
|---|---|
| Capa GAT | [12, 50, 45] |
| LSTM 1 | [12, 50, 32] |
| LSTM 2 | [12, 50, 128] |
| Capa Lineal | [9, 50, 45] |

El modelo fue entrenado durante 200 épocas utilizando un tamaño de lote de 50 ejemplos, con una tasa de aprendizaje inicial $\alpha$ de $3x10^{-4}$ y un decaimiento de peso $\lambda$ de $5x10^{-5}$, con el objetivo de mejorar la estabilidad del entrenamiento y regular la actualización de los parámetros. Para mitigar el sobreajuste y mejorar la generalización del modelo, se implementó una estrategia de dropout con una tasa del 40%, asegurando que el aprendizaje del modelo no dependa excesivamente de ciertas conexiones en la red.

El conjunto de datos utilizado en el entrenamiento representa 30 días de tráfico en la red SDN, donde cada observación consta de 120 valores y una ventana de predicción de 9 valores. Esta configuración permite al modelo capturar tanto la evolución temporal del tráfico como la dependencia entre los enlaces de la red. Para garantizar una evaluación equilibrada, los datos se dividieron en un 70% para entrenamiento, proporcionando un volumen adecuado de datos para el aprendizaje; 15% para validación, utilizados para monitorear el rendimiento del modelo y prevenir sobreajuste; y el 15% restante para prueba, con el objetivo de evaluar la capacidad predictiva del modelo en escenarios no vistos previamente. Durante el entrenamiento, se emplearon las siguientes métricas para evaluar el desempeño del modelo:

- Loss (pérdida): representa el error acumulado entre las predicciones del modelo y los valores reales.

- Error Absoluto Medio (MAE, Mean Absolute Error, por sus siglas en inglés): representa la diferencia promedio entre las predicciones y los valores reales, sin importar la dirección del error.

- Raíz del Error Cuadrático Medio (RMSE, Root Mean Squared Error, por sus siglas en inglés): similar al MAE, pero penaliza los errores grandes con mayor severidad, siendo útil para evaluar si el modelo comete errores significativos.

- Coeficiente de error $R^2$: cuantifica la proporción de variabilidad en los datos por el modelo, donde valores cercanos a 1 indican una alta precisión en las predicciones.

El análisis del entrenamiento del modelo revela que, el rendimiento mejora con un mayor número de épocas, sin embargo, el costo computacional también aumenta significativamente. En la época 50, el modelo logra un $R^2$ del 73.83% con un tiempo de entrenamiento de 1.20 min. Esto indica que el modelo comienza a capturar ciertos patrones en los datos, aunque presenta un margen de errores considerables. Al llegar a 100 épocas, el coeficiente $R^2$ supera el 80% con un MAE de 0.0838 y un RMSE de 0.1095, mientras que el tiempo de entrenamiento se duplica. A este punto, el modelo es capaz de realizar predicciones confiables con una precisión aceptable. Sin embargo, al extender el entrenando hasta las 200 épocas, se obtiene un $R^2$ del 85.68%, con una disminución del error absoluto y una mejora en la estabilidad del modelo, aunque el tempo de entrenamiento aumenta a 4 min. Más allá de este punto, las mejoras son marginales en comparación con el costo computacional. Entre las épocas 200 y 500, el incremento en $R^2$ es solo del 3%, mientras que el tiempo de entramiento supera los 10 min. Por lo tanto, si se prioriza la eficiencia computacional, 100 épocas puede ser suficiente para obtener un modelo aceptable con un 80% de precisión. Sin embargo, con 200 épocas representa el mejor equilibrio entre el rendimiento y tiempo de entrenamiento, facilitando la optimización de la gestión del tráfico sin un uso excesivo de recursos. La Figura 29 y Tabla 8 presentan la evolución del rendimiento del modelo a lo largo de las épocas, mientras que la Figura 30 muestra una comparación entre los valores predichos y los valores reales en un segmento de datos.

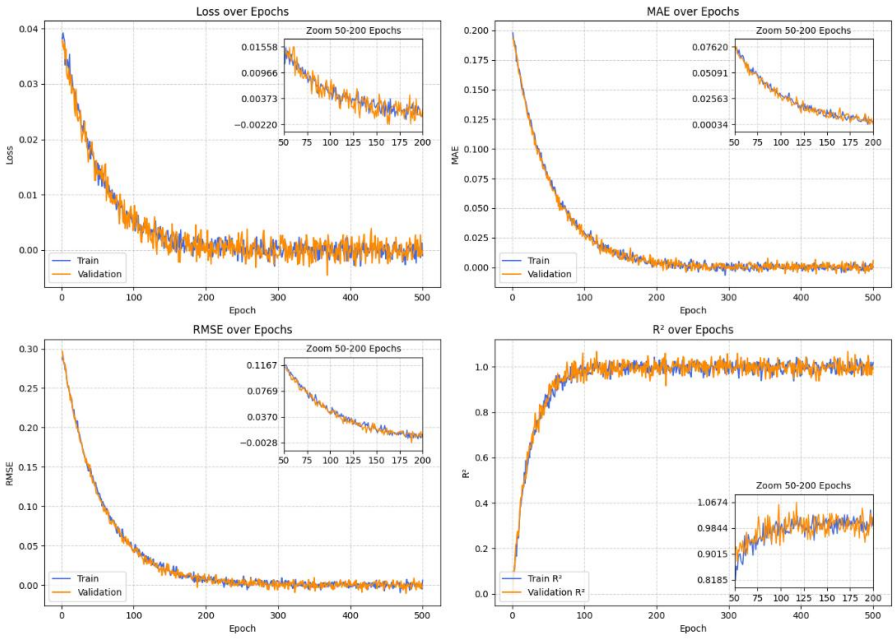


**Figura 29.** Métricas de rendimiento del modelo ST-GAT por épocas.

**Tabla 8.** Rendimiento del modelo ST-GAT por épocas.

| Época | Loss | MAE | RMSE | $R^2$ | Tiempo (min) |
|---|---|---|---|---|---|
| 50 | 0.0081 | 0.0969 | 0.1273 | 0.7383 | 1.20 |
| 100 | 0.006 | 0.0838 | 0.1095 | 0.8065 | 2.40 |
| 200 | 0.0044 | 0.0718 | 0.0942 | 0.8568 | 4.06 |
| 300 | 0.0038 | 0.0666 | 0.0877 | 0.8761 | 6.00 |
| 400 | 0.0036 | 0.0641 | 0.0843 | 0.8853 | 8.20 |
| 500 | 0.0034 | 0.0621 | 0.0819 | 0.8918 | 10.50 |

Para modelar el tráfico de red, se emplearon los perfiles de calidad de servicio (QoS) definidos en el estándar 5G, basados en los perfiles 5QI de la especificación 3GPP TS 23.501. Estos perfiles clasifican las aplicaciones en función de sus exigencias de latencia, confiabilidad y tasa de bits. En este experimento se seleccionaron cuatro perfiles que abarcan diferentes niveles de prioridad (Tabla 9), desde aplicaciones críticas como VoIP (nivel 1), hasta aplicaciones de menor prioridad, como descargas de archivos no urgentes (nivel 5). Estos perfiles representan los distintos flujos de datos que compiten por los recursos de una red en un escenario de alta demanda.

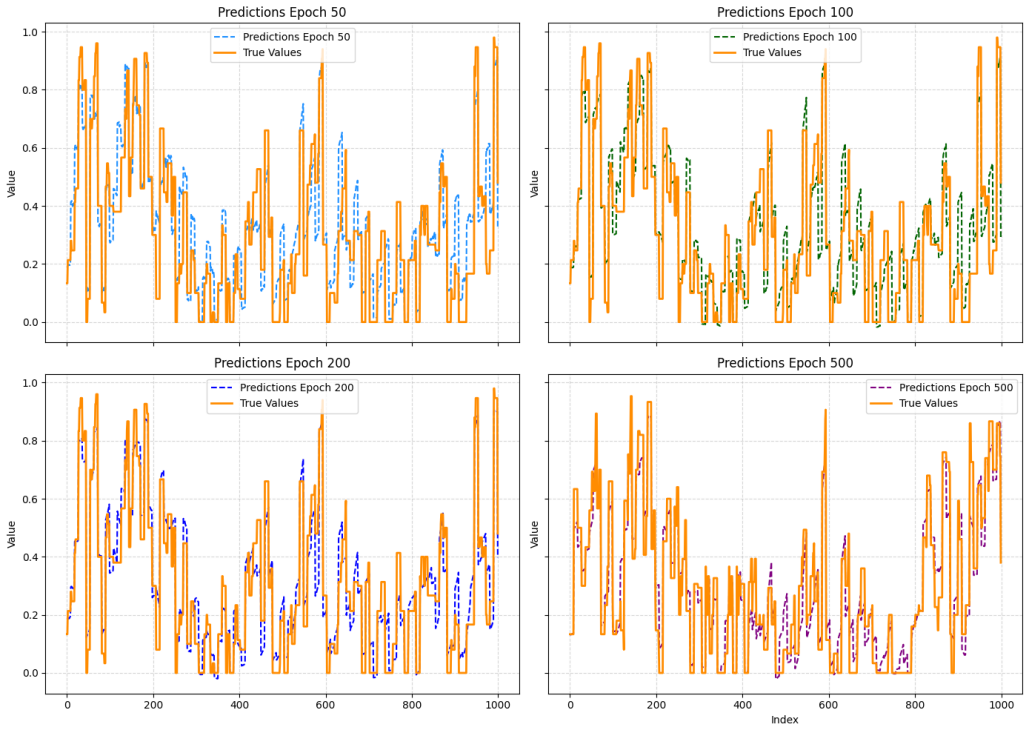


**Figura 30**. Comparación entre las predicciones y los valores reales por época en el modelo ST-GAT.

Para cumplir con los Acuerdos de Nivel de Servicio (SLA) se emplea la técnica de network slicing, dividiendo la red en segmentos lógicos a lo largo de las de las rutas y reservando los recursos específicos para cada flujo. De esta manera, los flujos mantienen su asignación de recursos incluso en condiciones de congestión, garantizando así la QoS de los flujos establecidos.

La transmisión de flujos de datos entre los nodos se llevó a cabo de forma aleatoria, seleccionando los nodos de origen y destino sin repetir combinaciones y distribuyendo el tráfico de forma uniforme. Esta asignación se basa en un patrón de distribución de tráfico del informe global de Sandvine, que indica que más del 70% del tráfico corresponde a aplicaciones de video y streaming. En el experimento, estos flujos se asocian con niveles de prioridad 3 y 4. Aproximadamente un 5% del tráfico se destinó a servicios de navegación web (prioridad 5), mientras que el 25% restante se reservó para aplicaciones sensibles a la latencia, como VoIP y videollamadas, asociadas con los niveles de prioridad 1 y 2.

**Tabla 9.** Perfiles de aplicación.

| Nivel de prioridad | Aplicación | Duración (s) | Bandwidth (Mbps) | Delay (ms) | Packet Loss (%) |
|---|---|---|---|---|---|
| 1 | VoIP | 45 | 10 | <65 | $10^{-2}$ |
| 2 | Video conferencia | 105 | 15 | <150 | $10^{-3}$ |
| 3 | Video games | 60 | 20 | <300 | $10^{-2}$ |
| 4 | Interactive Video | 120 | 25 | <100 | $10^{-4}$ |
| 5 | Best effort data | 30 | 5 | N/A | N/A |

Con el fin de simular la congestión en la red, los flujos se enviaron en intervalos temporales modelados por una distribución exponencial, reflejando la llegada aleatoria de eventos. Este enfoque permite representar las condiciones impredecibles del tráfico en escenarios de alta demanda. El intervalo máximo entre envíos de flujos fue de 15 segundos, lo cual garantiza una alta carga en la red durante el periodo de observación, permitiendo evaluar el rendimiento del algoritmo Smart Routing y el de los métodos tradicionales.

Los resultados obtenidos (ver Figura 31) evidencian que el algoritmo Smart Routing supera notablemente a los métodos tradicionales en términos de eficiencia (throughput) como en la gestión de niveles de prioridad del tráfico. Este algoritmo no solo optimiza el rendimiento en condiciones de alta demanda, sino que también asegura una asignación eficiente del tráfico acuerdo los requisitos de calidad de servicio (QoS), aspectos críticos para mantener los Acuerdos de Nivel de Servicio (SLA) en situaciones de alta demanda.

En un análisis detallado de la distribución de tráfico, muestra que Smart Routing logra direccionar eficientemente el 70% de la demanda total, mientras que los algoritmos tradicionales no superaron el 60%. En términos granulares, Smart Routing fue capaz de redirigir el 7.97% del tráfico total a aplicaciones críticas de nivel 1 y 16.38% hacia tráfico de nivel 2, ambos sensibles a la latencia. Las aplicaciones que consumen mayor ancho de banda son las de niveles 3 y 4 representaron el 46.11% del tráfico, dominando el uso de ancho de banda. A pesar de ello, el algoritmo fue capaz de dar prioridad a las aplicaciones de niveles 1 y 2, mientras proporcionaba oportunidades de transmisión al tráfico no sujeto a QoS, el cual representó el 5.46% del tráfico total en este caso.

Esta distribución inteligente de tráfico muestra la capacidad de Smart Routing para gestionar efectivamente la red de manera efectiva, maximizando el uso de los recursos y garantizando la calidad de servicio de aplicaciones prioritarias, incluso en condiciones de alta demanda y congestión.

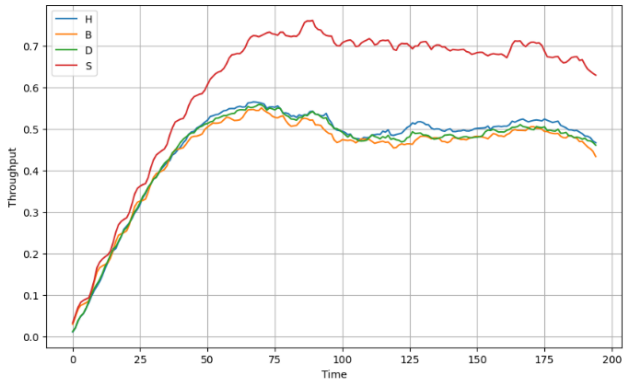


**Figura 31.** Evaluación del throughput total de algoritmos de enrutamiento. Algoritmo propuesto, denominado Smart Routing ($\boldsymbol{S}$) = 0.6488. El algoritmo de la ruta más corta ($\boldsymbol{H}$) = 0.4981, el algoritmo OSPF ($\boldsymbol{B}$) = 0.4815, y el algoritmo de ruta con menor retardo ($\boldsymbol{D}$) = 0.4930.

Asimismo, Smart Routing destaca sobre los enfoques tradicionales al gestionar el tráfico de manera más eficiente, especialmente bajo condiciones de congestión (ver Figura 32). Para los flujos de prioridad 1, Smart Routing supera al algoritmo de la ruta más corta (H) en un 23.68%, a OSPF (B) en un 27.80% y al

algoritmo de menor retardo (D) en un 20%. De igual forma, para los flujos de prioridad 2, ofrece mejoras del 23.76%, 36% y 27.81% respecto a los algoritmos H, B y D, respectivamente. En los niveles de tráfico 3 y 4, Smart Routing también demostró una eficiencia superior, siendo un 19% más eficiente que los métodos H y B, y un 12% más eficiente que el algoritmo D.

Estos hallazgos evidencian la capacidad de Smart Routing para optimizar la asignación de recursos en redes con altos niveles de tráfico, asegurando que se cumplan los Acuerdos de Nivel de Servicio y los parámetros de Calidad de Servicio (QoS), incluso bajo condiciones de tráfico de alta demanda. Esto se logra gracias a su capacidad predictiva, que permite anticipar cambios en la topología de la red y ajustar dinámicamente las rutas de tráfico para evitar sobrecargas en los enlaces críticos. Además, el análisis confirma que este enfoque es particularmente efectivo en la administración de tráfico de alto volumen, lo cual mejora la QoS en entornos de red exigentes, posicionando a Smart Routing como una solución eficiente frente a los métodos de enrutamiento tradicionales que aún prevalecen en las redes SDN.

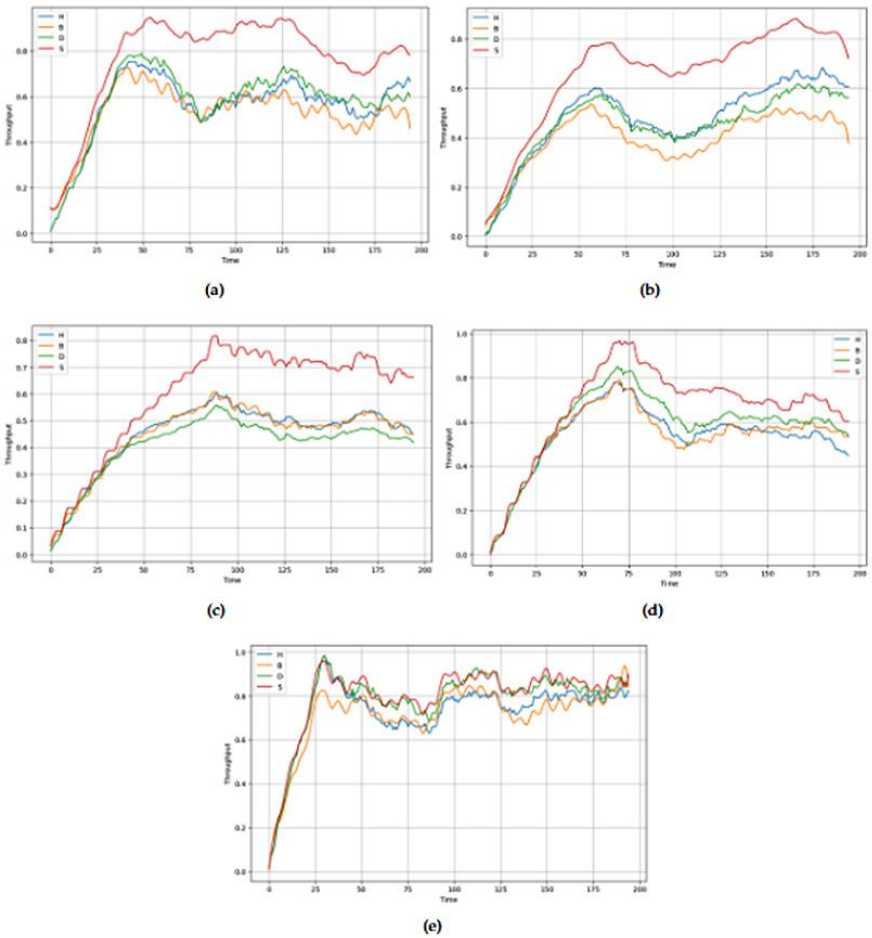


**Figura 32.** Evaluación del throughput con nivel de prioridad de tráfico: a) Nivel de prioridad 1 (mayor prioridad), b) Nivel de prioridad 2, c) Nivel de prioridad 3, d) Nivel de prioridad 4, y e) Nivel de prioridad 5. Algoritmos evaluados: (***S***) Smart Routing, (***H***) Algoritmo de la ruta más corta, (***B***) OSPF, y (***D***) Algoritmo de ruta con menor retardo.

# Capítulo 7. Conclusiones

En este trabajo se realiza una exploración exhaustiva de los algoritmos de enrutamiento empleados en controladores de red SDN, evaluando su rendimiento en topologías multidominio y centros de datos con base en métricas clave como el número de saltos (enlaces en una ruta), ancho de banda disponible, retardo y pérdida de paquetes. La propuesta de incorporar algoritmos evolutivos, específicamente MOEA/D y MOCell, en el enrutamiento para SDN ha demostrado ser altamente superior en comparación con los métodos tradicionales, mejorando considerablemente el rendimiento de la red. Los experimentos realizados señalan que estos algoritmos son capaces de optimizar las rutas de manera más eficiente, disminuyendo el costo de las métricas de red y reduciendo la probabilidad de seleccionar enlaces de bajo rendimiento, lo cual ayuda a prevenir la congestión y garantiza una mayor calidad de servicio (QoS).

La arquitectura centralizada de SDN facilita el aprovechamiento de estos algoritmos evolutivos, que han sido diseñados para escalar de manera flexible, adaptando la selección de rutas según las necesidades específicas de QoS de cada flujo de datos. Los resultados indican que la capacidad de MOEA/D y MOCell, para aprovechar una visión integral de la red permite cumplir de manera más eficaz con los Acuerdos de Nivel de Servicio (SLA) de manera más efectiva, incluso bajo condiciones de tráfico elevado y dinámico.

Asimismo, esta investigación identifica oportunidades futuras para optimizar el rendimiento de MOCell mediante ajustes en parámetros como el tamaño de la población, así como en la elección de métodos de cruce y mutación. Sería interesante realizar estudios comparativos más exhaustivos con otros algoritmos evolutivos avanzados, evaluando factores como el tiempo de convergencia y la diversidad de soluciones. La inclusión de métricas adicionales, como la dispersión y la precisión de las soluciones, permitiendo un análisis más profundo de su efectividad en entornos reales.

El análisis comparativo con los métodos de enrutamiento tradicionales confirmó que Smart Routing, basado en Redes Neuronales de Grafos (GNN), ofrece un rendimiento significativamente superior en la gestión de tráfico de red, especialmente en condiciones de alta congestión. Con la incorporación de técnicas de aprendizaje automático, como las GNN, Smart Routing no solo incrementa la transmisión de tráfico hasta en un 25% por encima de los algoritmos convencionales, sino que también aborda proactivamente los cuellos de botella y distribuye los recursos con mayor eficiencia. La capacidad predictiva de las GNN permite anticipar posibles congestiones y ajustar las rutas para mantener el flujo de datos y priorizar servicios críticos, maximizando así la confiabilidad de la red.

El uso combinado de técnicas evolutivas y redes neuronales gráficas representa un enfoque innovador y potente para la administración de redes SDN. Mientras que los algoritmos evolutivos aseguran soluciones de enrutamiento óptimas que se ajustan a múltiples métricas de QoS, las GNN ofrecen una capacidad predictiva y auto adaptativa, permitiendo a la red aprender de patrones de tráfico y condiciones cambiantes. Esta combinación facilita una gestión de red autoprogramable, que no solo optimiza el rendimiento general, sino que también asegura que se mantenga la eficiencia exigida por aplicaciones sensibles al tiempo de respuesta y cumplimiento de SLA.

En conclusión, la implementación de algoritmos de inteligencia artificial en entornos SDN marca un avance considerable en la gestión de redes, proporcionando a los operadores herramientas robustas para enfrentar los desafíos de tráfico complejo y dinámico en redes de alta demanda. Este enfoque establece nuevos parámetros de adaptabilidad, eficiencia y calidad de servicio en SDN, posicionándose como una solución integral y sumamente efectiva frente a las limitaciones de los métodos de enrutamiento convencionales. Con la combinación de algoritmos evolutivos y capacidades de GNN, el enrutamiento inteligente no solo garantiza el cumplimiento de los SLA, sino también impulsa una administración de red más autónoma y responsiva, adecuada para las crecientes exigencias de las aplicaciones modernas en redes definidas por software.

## 7.1 Trabajo futuro

Como trabajo futuro, se propone optimizar los algoritmos evolutivos empleados (MOEA/D y MOCell) con miras de mejorar la eficiencia del enrutamiento en redes SDN, centrándose en reducir el tiempo de convergencia y aumentar la diversidad de las soluciones generadas. La inclusión de métricas adicionales, como la dispersión y precisión de las soluciones, permitiría evaluar con mayor detalle la efectividad de estos algoritmos en escenarios con múltiples flujos y niveles de tráfico. Esto implicará explorar diferentes tamaños de población y técnicas de cruce y mutación, ajustando estos parámetros para maximizar la eficiencia en redes de gran escala y condiciones de tráfico altamente dinámicas.

Además, sería valioso establecer una comparación entre la optimización multi-objetivo y el enfoque basado en predicción, lo que permitirá evaluar de manera equitativa el desempeño de ambas técnicas en entornos de alta demanda. Actualmente, el enfoque basado en GNN se centra en la predicción de enlaces utilizando principalmente el ancho de banda como métrica. Futuras implementaciones podrían incorporar otras métricas críticas de QoS, como el retardo y la pérdida de paquetes, para lograr una predicción más

precisa y adaptabilidad en redes con problemas de congestión y alta demanda de tráfico que requieren brindar servicios de calidad de servicio en las que la baja latencia y estabilidad de la ruta resulta crucial.

Un área de interés importante es el análisis de la ubicación y cantidad de controladores SDN dentro de la topología de la red, lo cual repercute directamente con la carga de trabajo de los enlaces y en la prevención de cuellos de botella en los controladores. Una distribución estratégica de controladores SDN es clave para reducir la latencia en la toma de decisiones de enrutamiento y evitar la sobrecarga de ciertos enlaces. El uso de técnicas como particionamiento de grafos y optimización basada en inteligencia artificial podrían aportar soluciones más eficientes para distribuir la carga entre los controladores y mejorar la escalabilidad de la red.

Explorar técnicas adicionales de inteligencia artificial, como las redes líquidas, representa una dirección prometedora para mejorar el enrutamiento en redes SDN. Las redes líquidas, una forma avanzada de redes neuronales recurrentes, se caracterizan por su capacidad para adaptarse dinámicamente a cambios en la entrada de datos sin necesidad de reentrenamiento constante. Esta cualidad permitirá que las redes líquidas respondan a variaciones del tráfico, haciendo ajustes a las rutas de forma continua y optimizando la calidad de servicio (QoS) en función de las condiciones actuales de la red.

Además, explorar técnicas adicionales de inteligencia artificial, como las redes líquidas, presenta una vía prometedora para adaptar dinámicamente los patrones de tráfico en el enrutamiento de SDN. Las redes líquidas, una variante avanzada de las redes neuronales recurrentes, destacan por su capacidad de ajustarse en tiempo real con los cambios en la entrada de datos sin la necesidad de un reentrenamiento continuo. Esta propiedad al combinarse con algoritmos evolutivos y Redes Neuronales de Grafos (GNN), podrían mejor significativamente su capacidad de adaptación y mejorar la selección de rutas en el proceso de enrutamiento, permitiéndoles ajustar decisiones de manera constante y optimizar la estabilidad y eficiencia de la red incluso en escenarios distintos a los cuales fueron entrenados los modelos.

Finalmente, la implementación de otras métricas de evaluación para la calidad del servicio, tales como la latencia percibida por el usuario, la disponibilidad de ruta, y el tiempo de respuesta, permitirá un análisis más robusto del rendimiento de estos algoritmos en redes SDN. Implementar estas métricas en las simulaciones permitirá medir el impacto directo en la experiencia del usuario y el cumplimiento de QoS en aplicaciones críticas. Esto ampliará el alcance del enrutamiento inteligente en redes SDN, dotando a los operadores de una herramienta integral y confiable para la administración de redes.

# Literatura citada


Alidadi, A., Arab, S., & Askari, T. (2022). A novel optimized routing algorithm for QoS traffic engineering in SDN-based mobile networks. *ICT Express*, *8*(1), 130–134. https://doi.org/10.1016/j.icte.2021.12.010

Almadani, B., Beg, A., & Mahmoud, A. (2021). DSF: A Distributed SDN Control Plane Framework for the East/West Interface. *IEEE Access*, *9*, 26735–26754. https://doi.org/10.1109/ACCESS.2021.3057690

Alsaeedi, M., Mohamad, M. M., & Al-Roubaiey, A. A. (2019). Toward Adaptive and Scalable OpenFlow-SDN Flow Control: A Survey. *IEEE Access*, *7*, 107346–107379. https://doi.org/10.1109/ACCESS.2019.2932422

Amin, R., Rojas, E., Aqdus, A., Ramzan, S., Casillas-Perez, D., & Arco, J. M. (2021a). A Survey on Machine Learning Techniques for Routing Optimization in SDN. En *IEEE Access* (Vol. 9, pp. 104582–104611). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2021.3099092

Amin, R., Rojas, E., Aqdus, A., Ramzan, S., Casillas-Perez, D., & Arco, J. M. (2021b). A Survey on Machine Learning Techniques for Routing Optimization in SDN. En *IEEE Access* (Vol. 9, pp. 104582–104611). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2021.3099092

Ammeri, A., Hachicha, W., Chabchoub, H., & Masmoudi, F. (2011). A comprehensive literature review of mono-objetive simulation optimization methods. *Advances in Production Engineering & Management*, *6(4)*, 291–302. Recuperado el 14 de marzo de 2022, de: https://apem-journal.org/Archives/2011/APEM6-4_291-302.pdf

Badotra, S., & Singh, J. (2017). Open Daylight as a Controller for Software Defined Networking. *Article in International Journal of Advanced Research in Computer Science*, *8*(5). Recuperado el 7 febrero de 2023, de: www.ijarcs.info

Bannour, F., Souihi, S., & Mellouk, A. (2018). Distributed SDN Control: Survey, Taxonomy, and Challenges. *IEEE Communications Surveys and Tutorials*, *20*(1), 333–354. https://doi.org/10.1109/COMST.2017.2782482

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G., Vaswani, A., Allen, K., Nash, C., Langston, V., … Pascanu, R. (2018). *Relational inductive biases, deep learning, and graph networks*. http://arxiv.org/abs/1806.01261

Caramia, M., & Dell'Olmo, P. (2020). Multi-objective Optimization. En *Multi-objective Management in Freight Logistics* (pp. 21–51). Springer International Publishing. https://doi.org/10.1007/978-3-030-50812-8_2

Casas-Velasco, D. M., Rendon, O. M. C., & Da Fonseca, N. L. S. (2021). Intelligent Routing Based on Reinforcement Learning for Software-Defined Networking. *IEEE Transactions on Network and Service Management*, *18*(1), 870–881. https://doi.org/10.1109/TNSM.2020.3036911

Chakraverty, S., Sahoo, D. M., & Mahato, N. R. (2019). McCulloch–Pitts Neural Network Model. En *Concepts of Soft Computing* (pp. 167–173). Springer Singapore. https://doi.org/10.1007/978-981-13-7430-2_11

Chen, Y. R., Rezapour, A., Tzeng, W. G., & Tsai, S. C. (2020). RL-Routing: An SDN Routing Algorithm Based on Deep Reinforcement Learning. *IEEE Transactions on Network Science and Engineering*, *7*(4), 3185–3199. https://doi.org/10.1109/TNSE.2020.3017751

Cheng, P., Hu, J., & Song, T. (2024). A QoS-Aware Routing Mechanism with Flow Classification for SDN-IoT Architechure. *2024 5th Information Communication Technologies Conference, ICTC 2024*, 204–209. https://doi.org/10.1109/ICTC61510.2024.10602045

Chiandussi, G., Codegone, M., Ferrero, S., & Varesio, F. E. (2012). Comparison of multi-objective optimization methodologies for engineering applications. *Computers and Mathematics with Applications*, *63*(5), 912–942. https://doi.org/10.1016/j.camwa.2011.11.057

Coello, C. A. C., Lamont, G. B., Veldhuizen, D. A. Van, Goldberg, D. E., & Koza, J. R. (2007). Evolutionary Algorithms for Solving Multi-Objective Problems Springer. https://doi.org/10.1007/978-0-387-36797-2

Cui, L. S., & Srivastava, G. (2022). QoS Routing Algorithm for OBS Networks Based on a Multi-Objective Genetic Algorithm. *IEEE Access*, *10*, 12047–12056. https://doi.org/10.1109/ACCESS.2021.3138380

Dao, S. D., Abhary, K., & Marian, R. (2017). A bibliometric analysis of Genetic Algorithms throughout the history. *Computers and Industrial Engineering*, *110*, 395–403. https://doi.org/10.1016/j.cie.2017.06.009

de Castro, L. N., & Timmis, J. I. (2003). Artificial immune systems as a novel soft computing paradigm. *Soft Computing*, *7*(8), 526–544. https://doi.org/10.1007/s00500-002-0237-z

Dhandapani, M., Vetriselvi, V., & Aishwarya, R. (2024). CoopAI-Route: DRL Empowered Multi-Agent Cooperative System for Efficient QoS-Aware Routing for Network Slicing in Multi-Domain SDN. *CMES - Computer Modeling in Engineering and Sciences*, *140*(3), 2449–2486. https://doi.org/10.32604/cmes.2024.050986

Durillo, J. J., Zhang, Y., Alba, E., & Nebro, A. J. (2009). A study of the multi-objective next release problem. *Proceedings - 1st International Symposium on Search Based Software Engineering, SSBSE 2009*, 49–58. https://doi.org/10.1109/SSBSE.2009.21

Egilmez, H. E., Tahsin Dane, S., Bagci, K. T., & Murat Tekalp, A. (2012). *OpenQoS: An OpenFlow Controller Design for Multimedia Delivery with End-to-End Quality of Service over Software-Defined Networks*. In *Proceedings of the 2012 Asia Pacific signal and information processing association annual summit and conference* (pp. 1-8). IEEE. Recuperado el 8 de mayo de 2023, de: https://ieeexplore.ieee.org/document/6411795

Eiben, A. E., Agoston E and Smith, & James E. (2015). *Introduction to evolutionary computing* (2a ed.). Springer. www.springer.com/series/

Elbasheer, M. O., Aldegheishem, A., Lloret, J., & Alrajeh, N. (2021). A QoS-Based routing algorithm over software defined networks. *Journal of Network and Computer Applications*, *194*. https://doi.org/10.1016/j.jnca.2021.103215

Feamster, N., Rexford, J., Zegura, E., & Tech, G. (2014). *The Road to SDN: An Intellectual History of Programmable Networks*. https://doi.org/10.1145/2602204

Filipe, J., Ghosh, A., Prates, R. O., & Zhou, L. (2020). *Communications in Computer and Information Science 1388 Editorial Board Members*. https://www.springer.com/series/7899

Ghadialy, Z. (2021, abril 4). 3GPP 5G Specifications. 3G AND 4G Wireless Blog. Recuperado el 15 de mayo de 2023, de: https://www.3g4g.co.uk/5G/5Gtech_0003_Standards.html

García-Tadeo, D. A., Reddy Peram, D., Suresh Kumar, K., Vives, L., Sharma, T., & Manoharan, G. (2022). Comparing the impact of Internet of Things and cloud computing on organisational behavior: A survey. *Materials Today: Proceedings*, *51*, 2281–2285. https://doi.org/10.1016/j.matpr.2021.11.399

Gelberger, A., Yemini, N., & Giladi, R. (2013). Performance analysis of Software-Defined Networking (SDN). *Proceedings - IEEE Computer Society's Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems, MASCOTS*, 389–393. https://doi.org/10.1109/MASCOTS.2013.58

Ghosh, S., Antonakoglou, K., Mavromatis, I., & Katsaros, K. (2024). *Intelligent Routing as a Service (iRaaS)*. http://arxiv.org/abs/2407.05901

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2020). *Message Passing Neural Networks* (pp. 199–214). https://doi.org/10.1007/978-3-030-40245-7_10

Gonzalez-Franco, J. D., Preciado-Velasco, J. E., Lozano-Rizk, J. E., Rivera-Rodriguez, R., Torres-Rodriguez, J., & Alonso-Arevalo, M. A. (2023). Comparison of Supervised Learning Algorithms on a 5G Dataset Reduced via Principal Component Analysis (PCA). *Future Internet*, *15*(10). https://doi.org/10.3390/fi15100335

Gonzalez-Trejo, J. E., Rivera-Rodríguez, R., Tchernykh, A., Lozano-Rizk, J. E., Gonzalez-Compeank, J. L., Villarreal-Reyes, S., & Galaviz-Mosqueda, A. (s/f). *Integration of Graph Neural Networks (GNN) into Software Defined Networks (SDN)*. www.mdpi.com/journal/notspecified

Guck, J. W., Van Bemten, A., Reisslein, M., & Kellerer, W. (2018). Unicast QoS Routing Algorithms for SDN: A Comprehensive Survey and Performance Evaluation. *IEEE Communications Surveys and Tutorials*, *20*(1), 388–418. https://doi.org/10.1109/COMST.2017.2749760

Hajjaji, Y., Boulila, W., Farah, I. R., Romdhani, I., & Hussain, A. (2021). Big data and IoT-based applications in smart environments: A systematic review. En *Computer Science Review* (Vol. 39). Elsevier Ireland Ltd. https://doi.org/10.1016/j.cosrev.2020.100318

He, Y., Xiao, G., Zhu, J., Zou, T., & Liang, Y. (2024). Reinforcement learning-based SDN routing scheme empowered by causality detection and GNN. *Frontiers in Computational Neuroscience*, *18*. https://doi.org/10.3389/fncom.2024.1393025

Henni, D. E., Ghomari, A., & Hadjadj-Aoul, Y. (2020). A consistent QoS routing strategy for video streaming services in SDN networks. *International Journal of Communication Systems*, *33*(10). https://doi.org/10.1002/dac.4177

Hruschka, E. R., Campello, R. J. G. B., Freitas, A. A., & de Carvalho, A. C. P. L. F. (2009). A survey of evolutionary algorithms for clustering. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, *39*(2), 133–155. https://doi.org/10.1109/TSMCC.2008.2007252

Hu, F., Hao, Q., & Bao, K. (2014a). A survey on software-defined network and OpenFlow: From concept to implementation. En *IEEE Communications Surveys and Tutorials* (Vol. 16, Número 4, pp. 2181–2206). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/COMST.2014.2326417

Hu, F., Hao, Q., & Bao, K. (2014b). A survey on software-defined network and OpenFlow: From concept to implementation. En *IEEE Communications Surveys and Tutorials* (Vol. 16, Número 4, pp. 2181–2206). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/COMST.2014.2326417

Ibarrola, E., Liberal, F., Ferro, A., & Xiao, J. (2010). Quality of Service Management for ISPs: A Model and Implementation Methodology Based on the ITU-T Recommendation E.802 Framework. *IEEE Communications Magazine,* 48(2), 146-153. https://doi.org/10.1109/MCOM.2010.5402678

Jiang, W. (2022). Graph-based deep learning for communication networks: A survey. En *Computer Communications* (Vol. 185, pp. 40–54). Elsevier B.V. https://doi.org/10.1016/j.comcom.2021.12.015

Jin, G., Wang, M., Zhang, J., Sha, H., & Huang, J. (2022). STGNN-TTE: Travel time estimation via spatial–temporal graph neural network. *Future Generation Computer Systems*, *126*, 70–81. https://doi.org/10.1016/j.future.2021.07.012

Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, *80*(5), 8091–8126. https://doi.org/10.1007/s11042-020-10139-6

Khan, N., Salleh, R. bin, Koubaa, A., Khan, Z., Khan, M. K., & Ali, I. (2023). Data plane failure and its recovery techniques in SDN: A systematic literature review. En *Journal of King Saud University - Computer and Information Sciences* (Vol. 35, Número 3, pp. 176–201). King Saud bin Abdulaziz University. https://doi.org/10.1016/j.jksuci.2023.02.001

Kobayashi, M., Seetharaman, S., Parulkar, G., Appenzeller, G., Little, J., Van Reijendam, J., Weissmann, P., & McKeown, N. (2014). Maturing of OpenFlow and Software-defined Networking through deployments. *Computer Networks*, *61*, 151–175. https://doi.org/10.1016/j.bjp.2013.10.011

Kodera, N., Noshiro, D., Dora, S. K., Mori, T., Habchi, J., Blocquel, D., Gruet, A., Dosnon, M., Salladini, E., Bignon, C., Fujioka, Y., Oda, T., Noda, N. N., Sato, M., Lotti, M., Mizuguchi, M., Longhi, S., & Ando, T. (2021). Structural and dynamics analysis of intrinsically disordered proteins by high-speed atomic force microscopy. *Nature Nanotechnology*, *16*(2), 181–189. https://doi.org/10.1038/s41565-020-00798-9

Kulkarni, M., G. B., & Paulose, J. (2021). *Experimenting with scalability of software defined networks using pyretic and frenetic. In International Conference on Computing Science, Communication and Security*. http://www.springer.com/series/7899

Kumar, V., Jangir, S., & Patanvariya, D. G. (2022). Traffic Load Balancing in SDN Using Round-Robin and Dijkstra Based Methodology. *2022 International Conference for Advancement in Technology, ICONAT 2022*. https://doi.org/10.1109/ICONAT53423.2022.9725862

Lambora, A., Gupta, K., & Chopra, K. (2019). Genetic Algorithm- A Literature Review. *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, 380–384. https://doi.org/10.1109/COMITCon.2019.8862255

Lara, A., Kolasani, A., & Ramamurthy, B. (2014a). Network innovation using open flow: A survey. *IEEE Communications Surveys and Tutorials*, *16*(1), 493–512. https://doi.org/10.1109/SURV.2013.081313.00105

Lara, A., Kolasani, A., & Ramamurthy, B. (2014b). Network innovation using open flow: A survey. *IEEE Communications Surveys and Tutorials*, *16*(1), 493–512. https://doi.org/10.1109/SURV.2013.081313.00105

Let, G. S., Pratap, C. B., Jagannath, D. J., Dolly, D. R. J., & Evangeline, L. D. (2023). Software-Defined Networking Routing Algorithms: Issues, QoS and Models. En *Wireless Personal Communications*, 131(3), pp. 1631–1661. Springer. https://doi.org/10.1007/s11277-023-10516-y

Li, D., Wang, X., Jin, Y., & Liu, H. (2020). Research on QoS routing method based on NSGAII in SDN. *Journal of Physics: Conference Series*, *1656*(1). https://doi.org/10.1088/1742-6596/1656/1/012027

Li, H., & Zhang, L. (2021). An efficient solution strategy for bilevel multiobjective optimization problems using multiobjective evolutionary algorithms. *Soft Computing*, *25*(13), 8241–8261. https://doi.org/10.1007/s00500-021-05750-0

Li, J., Sun, P., & Hu, Y. (2020). Traffic modeling and optimization in datacenters with graph neural network. *Computer Networks*, *181*. https://doi.org/10.1016/j.comnet.2020.107528

Li, Y., Vinyals, O., Dyer, C., Pascanu, R., & Battaglia, P. (2018). *Learning Deep Generative Models of Graphs*. http://arxiv.org/abs/1803.03324

Lin, C., Wang, K., & Deng, G. (2017). A QoS-aware routing in SDN hybrid networks. *Procedia Computer Science*, *110*, 242–249. https://doi.org/10.1016/j.procs.2017.06.091

Lozano-Rizk, J. E., Nieto-Hipolito, J. I., Rivera-Rodriguez, R., Cosio-Leon, M. A., Vazquez-Briseño, M., & Chimal-Eguia, J. C. (2020). QOSCOMM: A data flow allocation strategy among sdn-based data centers for iot big data analytics. *Applied Sciences (Switzerland)*, *10*(21), 1–19. https://doi.org/10.3390/app10217586

Manguri, K. H., & Omer, S. M. (2022). SDN for IoT Environment: A Survey and Research Challenges. *ITM Web of Conferences*, *42*, 01005. https://doi.org/10.1051/itmconf/20224201005

Medhi, D., & Ramasamy, K. (2018). Network Flow Models. *Network Routing*, 114–157. https://doi.org/10.1016/B978-0-12-800737-2.00005-3

Mekky, H., Hao, F., Mukherjee, S., Zhang, Z. L., & Lakshman, T. V. (2014). Application-aware data plane processing in SDN. *HotSDN 2014 - Proceedings of the ACM SIGCOMM 2014 Workshop on Hot Topics in Software Defined Networking*, 13–18. https://doi.org/10.1145/2620728.2620735

Miguel-Alonso, J. (2023). A Research Review of OpenFlow for Datacenter Networking. *IEEE Access*, *11*, 770–786. https://doi.org/10.1109/ACCESS.2022.3233466

Nebro, A. J., Durillo, J. J., Luna, F., Dorronsoro, B., & Alba, E. (2009). MOCell: A cellular genetic algorithm for multiobjective optimization. *International Journal of Intelligent Systems*, *24*(7), 726–746. https://doi.org/10.1002/int.20358

Qazi, F., Kwak, D., Khan, F. G., Ali, F., & Khan, S. U. (2024). Service Level Agreement in cloud computing: Taxonomy, prospects, and challenges. En *Internet of Things (Netherlands)* (Vol. 25). Elsevier B.V. https://doi.org/10.1016/j.iot.2024.101126

Raayatpanah, M. A., Salehi Fathabadi, H., Khalaj, B. H., Khodayifar, S., & Pardalos, P. M. (2014). Bounds on end-to-end statistical delay and jitter in multiple multicast coded packet networks. *Journal of Network and Computer Applications*, *41*(1), 217–227. https://doi.org/10.1016/j.jnca.2013.12.004

Reich, J., Monsanto, C., Foster, N., Rexford J., & Walker, D. (2013). *Modular sdn programming with pyretic*. *Technical Report of USENIX*, 30. Recuperado el 20 de febrero de 2022, de: https://www.usenix.org/system/files/login/articles/09_reich-online.pdf

RYU Project Team. (2014). *SDN Framework RYU Using OpenFlow 1.3 RYU-English Edition. RYU* project. Recuperado el 25 de marzo de 2023, de: https://ryu-sdn.org/resources.html

Sharma, S., & Kumar, V. (2022). A Comprehensive Review on Multi-objective Optimization Techniques: Past, Present and Future. *Archives of Computational Methods in Engineering*, *29*(7), 5605–5633. https://doi.org/10.1007/s11831-022-09778-9

Shukla, A., Pandey, H. M., & Mehrotra, D. (2015). Comparative review of selection techniques in genetic algorithms. *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, 515–519. IEEE https://doi.org/10.1109/ABLAZE.2015.7154916

Son, J., & Buyya, R. (2018). A taxonomy of software-defined networking (SDN)-enabled cloud computing. En *ACM Computing Surveys* (Vol. 51, Número 3). Association for Computing Machinery. https://doi.org/10.1145/3190617

Suarez-Varela, J., Almasan, P., Ferriol-Galmes, M., Rusek, K., Geyer, F., Cheng, X., Shi, X., Xiao, S., Scarselli, F., Cabellos-Aparicio, A., & Barlet-Ros, P. (2023). Graph Neural Networks for Communication Networks: Context, Use Cases and Opportunities. *IEEE Network*, *37*(3), 146–153. https://doi.org/10.1109/MNET.123.2100773

Sun, C., Li, C., Lin, X., Zheng, T., Meng, F., Rui, X., & Wang, Z. (2023). Attention-based graph neural networks: a survey. *Artificial Intelligence Review*, *56*, 2263–2310. https://doi.org/10.1007/s10462-023-10577-2

Sun, W., Wang, Z., & Zhang, G. (2021). A QoS-guaranteed intelligent routing mechanism in software-defined networks. *Computer Networks*, *185*. https://doi.org/10.1016/j.comnet.2020.107709

Swaminathan, A., Chaba, M., Sharma, D. K., & Ghosh, U. (2021). GraphNET: Graph Neural Networks for routing optimization in Software Defined Networks. *Computer Communications*, *178*, 169–182. https://doi.org/10.1016/j.comcom.2021.07.025

Tache, M. D., Păscuțoiu, O., & Borcoci, E. (2024). Optimization Algorithms in SDN: Routing, Load Balancing, and Delay Optimization. En *Applied Sciences (Switzerland)* (Vol. 14, Número 14). Multidisciplinary Digital Publishing Institute (MDPI). https://doi.org/10.3390/app14145967

Tajiki, M. M., Akbari, B., Shojafar, M., Ghasemi, S. H., Barazandeh, M. L., Mokari, N., Chiaraviglio, L., & Zink, M. (2018). CECT: computationally efficient congestion-avoidance and traffic engineering in software-defined cloud data centers. *Cluster Computing*, *21*(4), 1881–1897. https://doi.org/10.1007/s10586-018-2815-6

Vesselinova, N., Steinert, R., Perez-Ramirez, D. F., & Boman, M. (2020a). Learning Combinatorial Optimization on Graphs: A Survey with Applications to Networking. En *IEEE Access* (Vol. 8, pp. 120388–120416). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2020.3004964

Vesselinova, N., Steinert, R., Perez-Ramirez, D. F., & Boman, M. (2020b). Learning Combinatorial Optimization on Graphs: A Survey with Applications to Networking. En *IEEE Access* (Vol. 8, pp. 120388–120416). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2020.3004964

Voellmy, A., Kim, H., & Feamster, N. (2012). Procera: A language for high-level reactive network control. In *Proceedings of the first workshop on Hot topics in software defined networks* (pp. 43-48). https://doi.org/10.1145/2342441.2342452

Vrahatis, A. G., Lazaros, K., & Kotsiantis, S. (2024). Graph Attention Networks: A Comprehensive Review of Methods and Applications. *Future Internet*, *16*(9), 318. https://doi.org/10.3390/fi16090318

Wang, S.-C. (2003). Artificial Neural Network. En *Interdisciplinary Computing in Java Programming* (pp. 81–100). Springer US. https://doi.org/10.1007/978-1-4615-0377-4_5

Wang, Z., Zhang, Q., Zhou, A., Gong, M., & Jiao, L. (2016). Adaptive Replacement Strategies for MOEA/D. *IEEE Transactions on Cybernetics*, *46*(2), 474–486. https://doi.org/10.1109/TCYB.2015.2403849

Wu, S., Sun, F., Zhang, W., Xie, X., & Cui, B. (2022). Graph Neural Networks in Recommender Systems: A Survey. *ACM Computing Surveys*, *55*(5). https://doi.org/10.1145/3535101

Wu, Y., & Feng, J. (2018). Development and Application of Artificial Neural Network. *Wireless Personal Communications*, *102*(2), 1645–1656. https://doi.org/10.1007/s11277-017-5224-x

Wumian, W., Saha, S., Haque, A., & Sidebottom, G. (2024). *Intelligent Routing Algorithm over SDN: Reusable Reinforcement Learning Approach*. https://arxiv.org/abs/2409.15226

Xavier, C. R., Dos Santos, E. P., Da Fonseca Vieira, V., & Dos Santos, R. W. (2013). Genetic algorithm for the history matching problem. *Procedia Computer Science*, *18*, 946–955. https://doi.org/10.1016/j.procs.2013.05.260

Xie, J., Kelley, S., & Szymanski, B. K. (2013). Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Computing Surveys*, *45*(4). https://doi.org/10.1145/2501654.2501657

Xu, Y., Gui, G., Gacanin, H., & Adachi, F. (2021). A Survey on Resource Allocation for 5G Heterogeneous Networks: Current Research, Future Trends, and Challenges. En *IEEE Communications Surveys and*

*Tutorials* (Vol. 23, Número 2, pp. 668–695). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/COMST.2021.3059896

Yang, S., Tan, C., Madsen, D. Ø., Xiang, H., Li, Y., Khan, I., & Choi, B. J. (2022). Comparative Analysis of Routing Schemes Based on Machine Learning. *Mobile Information Systems*, *2022*. https://doi.org/10.1155/2022/4560072

Ye, J., Zhao, J., Ye, K., & Xu, C. (2022). How to Build a Graph-Based Deep Learning Architecture in Traffic Domain: A Survey. En *IEEE Transactions on Intelligent Transportation Systems* (Vol. 23, Número 5, pp. 3904–3924). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/TITS.2020.3043250

Zhang, C., Yu, J. J. Q., & Liu, Y. (2019). Spatial-Temporal Graph Attention Networks: A Deep Learning Approach for Traffic Forecasting. *IEEE Access*, *7*, 166246–166256. https://doi.org/10.1109/ACCESS.2019.2953888

Zhang, L., Lu, Y., Zhang, D., Cheng, H., & Dong, P. (2022). DSOQR: Deep Reinforcement Learning for Online QoS Routing in SDN-Based Networks. *Security and Communication Networks*, *2022*. https://doi.org/10.1155/2022/4457645

Zhang, Q., & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, *11*(6), 712–731. https://doi.org/10.1109/TEVC.2007.892759

Zhang, Z. (2018). Artificial Neural Network. En *Multivariate Time Series Analysis in Climate and Environmental Research* (pp. 1–35). Springer International Publishing. https://doi.org/10.1007/978-3-319-67340-0_1

Zhu, L., Karim, M. M., Sharif, K., Li, F., Du, X., & Guizani, M. (2019). *SDN Controllers: Benchmarking & Performance Evaluation*. https://arxiv.org/abs/1902.04491

# Anexos

**Tabla 9.** Métricas de red en el controlador A. Métricas delay (ms), packet loss (%) y bandwidth (Mbps).

| Link | Métrica | Link | Métrica | Link | Métrica | Link | Métrica |
|---|---|---|---|---|---|---|---|
| 1,2 | 10,0,97 | 3,13 | 16,0.003,82 | 6,15 | 3,0,81 | 11,18 | 22,0,80 |
| 1,3 | 5,0,94 | 4,6 | 13,0,88 | 7,14 | 3,0,79 | 11,19 | 23,0,76 |
| 1,4 | 8,8,99 | 4,8 | 7,0,97 | 7,15 | 2,0,85 | 12,13 | 7,0,80 |
| 1,5 | 3,0,93 | 4,10 | 16,0,90 | 8,19 | 9,0,78 | 12,20 | 21,0.003,100 |
| 2,6 | 18,0.002,93 | 4,12 | 18,0.001,96 | 8,16 | 18,0.002,96 | 12,21 | 12,0,77 |
| 2,8 | 12,0,90 | 5,7 | 17,0,84 | 8,17 | 2,0,87 | 13,20 | 6,0,80 |
| 2,10 | 24,0.001,98 | 5,9 | 9,0,92 | 9,16 | 7,0,76 | 13,21 | 21,0.001,79 |
| 2,12 | 8,0,91 | 5,11 | 21,0.002,94 | 9,17 | 14,0,83 | 14,15 | 8,0,83 |
| 3,7 | 11,0,86 | 5,13 | 15,0,97 | 10,11 | 11,0,80 | 16,17 | 7,0,80 |
| 3,9 | 21,0.004,97 | 6,7 | 8,0,82 | 10,18 | 13,0,83 | 18,19 | 6,0,80 |
| 3,11 | 9,0,95 | 6,14 | 23,0,86 | 10,19 | 9,0,82 | 20,21 | 9,0,79 |
| H1,H8 | 0,0,100 | | | | | | |

**Tabla 10.** Métricas de red en el controlador B. Métricas delay (ms), packet loss (%) y bandwidth (Mbps).

| Link | Métrica | Link | Métrica | Link | Métrica | Link | Métrica |
|---|---|---|---|---|---|---|---|
| 22,23 | 2,0,94 | 25,29 | 8,0,93 | 29,35 | 18,0,82 | 33,40 | 7,0,92 |
| 22,24 | 4,0,96 | 25,31 | 13,0.001,91 | 29,36 | 12,0,97 | 34,39 | 5,0,83 |
| 22,25 | 2,0,98 | 26,28 | 9,0,92 | 30,35 | 5,0,94 | 34,40 | 12,0,95 |
| 22,26 | 16,0,99 | 26,30 | 12,0.001,91 | 30,36 | 1,0.001,86 | 35,41 | 5,0.002,84 |
| 23,27 | 8,0.001,91 | 26,32 | 18,0,94 | 30,37 | 3,0,97 | 35,42 | 4,0,96 |
| 23,29 | 15,0,93 | 27,33 | 7,0.002,93 | 31,36 | 4,0.001,96 | 36,41 | 2,0,87 |
| 23,31 | 12,0.001,95 | 27,34 | 12,0,87 | 31,38 | 6,0,96 | 36,42 | 7,0,93 |
| 24,28 | 2,0,92 | 28,33 | 11,0,88 | 31,38 | 2,0,99 | 37,43 | 6,0,92 |
| 24,10 | 7,0.001,94 | 28,34 | 6,0,93 | 32,37 | 20,0,94 | 37,44 | 3,0,94 |
| 24,32 | 12,0,97 | 28,35 | 2,0,97 | 32,38 | 5,0.001,86 | 38,43 | 5,0,94 |
| 25,27 | 12,0,98 | 29,34 | 15,0.002,97 | 33,39 | 3,0.002,82 | 38,44 | 3,0.001,95 |
| H9,H14 | 0,0,100 | | | | | | |

**Tabla 11.** Métricas de red en el controlador C. Métricas delay (ms), packet loss (%) y bandwidth (Mbps).

| Link | Métrica | Link | Métrica | Link | Métrica | Link | Métrica |
|---|---|---|---|---|---|---|---|
| 1,2 | 2,0,89 | 47,48 | 2,0,89 | 46,60 | 2,0,97 | 92,106 | 3,0.001,93 |
| 1,15 | 2,0.001,86 | 47,60 | 1,0,85 | 46,61 | 3,0,86 | 137,152 | 3,0,86 |
| 1,16 | 1,0,91 | 47,61 | 3,0,93 | 92,107 | 1,0,98 | 38,139 | 3,0,100 |

| 2,3 | 3,0,97 | 47,62 | 1,0,85 | 93,94 | 3,0,93 | 138,151 | 3,0,86 |
|---|---|---|---|---|---|---|---|
| 2,15 | 1,0,97 | 48,49 | 2,0.001,94 | 93,106 | 1,0,100 | 138,152 | 4,0.001,85 |
| 2,16 | 2,0.001,94 | 48,61 | 2,0,99 | 93,107 | 2,0.001,92 | 138,153 | 2,0,86 |
| 2,17 | 4,0.001,91 | 48,62 | 3,0,97 | 93,108 | 2,0,96 | 139,140 | 4,0.001,90 |
| 3,4 | 1,0.001,98 | 48,63 | 1,0,97 | 94,95 | 1,0.001,97 | 139,152 | 3,0,91 |
| 3,16 | 4,0,87 | 49,50 | 2,0,89 | 94,107 | 2,0,96 | 139,153 | 1,0,97 |
| 3,17 | 4,0,95 | 49,62 | 1,0,93 | 94,108 | 2,0.001,94 | 139,154 | 1,0,97 |
| 3,18 | 3,0,86 | 49,63 | 1,0.001,96 | 94,109 | 4,0,90 | 140,153 | 1,0,97 |
| 4,5 | 1,0.001,85 | 49,64 | 2,0,86 | 95,96 | 3,0,87 | 140,154 | 3,0,85 |
| 4,17 | 1,0,91 | 50,51 | 1,0,97 | 95,108 | 2,0,96 | 141,142 | 2,0.001,87 |
| 4,18 | 1,0,84 | 50,63 | 1,0,94 | 95,109 | 2,0,98 | 141,166 | 3,0,91 |
| 4,19 | 4,0.001,87 | 50,64 | 2,0.001,95 | 95,110 | 1,0,95 | 141,156 | 3,0.001,71 |
| 5,6 | 4,0.001,82 | 50,65 | 1,0,95 | 96,97 | 1,0.001,93 | 142,143 | 2,0.001,86 |
| 5,18 | 2,0,82 | 51,52 | 1,0,98 | 96,109 | 4,0,85 | 142,155 | 3,0,98 |
| 5,19 | 4,0,90 | 51,64 | 3,0,100 | 96,110 | 4,0.001,93 | 142,156 | 2,0,91 |
| 5,20 | 3,0.001,94 | 51,65 | 3,0.001,91 | 96,111 | 2,0,88 | 142,157 | 2,0,91 |
| 6,7 | 2,0,99 | 51,66 | 1,0,86 | 97,98 | 4,0,95 | 143,144 | 2,0,89 |
| 6,19 | 4,0,95 | 52,53 | 4,0.001,85 | 97,110 | 3,0,87 | 143,156 | 3,0,95 |
| 6,20 | 2,0,94 | 52,65 | 3,0,96 | 97,111 | 2,0,88 | 143,157 | 4,0.001,96 |
| 6,21 | 2,0,83 | 52,66 | 1,0.001,90 | 97,112 | 3,0,100 | 143,158 | 4,0,90 |
| 7,8 | 3,0,96 | 52,67 | 4,0,85 | 98,111 | 1,0,98 | 144,145 | 4,0,94 |
| 7.20 | 1,0,86 | 53,54 | 4,0,96 | 98,112 | 1,0,91 | 144,157 | 3,0,89 |
| 7,21 | 4,0,95 | 53,66 | 2,0,96 | 99,100 | 2,0.001,83 | 144,158 | 4,0,85 |
| 7,22 | 2,0,88 | 53,67 | 2,0.001,96 | 99,113 | 1,0,99 | 144,159 | 3,0,93 |
| 8,9 | 2,0.001,89 | 53,68 | 1,0,89 | 99,114 | 3,0,76 | 145,146 | 4,0.001,96 |
| 8,21 | 2,0,96 | 54,55 | 4,0.001,85 | 100,101 | 3,0,96 | 145,158 | 1,0,96 |
| 8,22 | 1,0,84 | 54,67 | 1,0,93 | 100,113 | 2,0,87 | 145,159 | 3,0,98 |
| 8,23 | 3,0,84 | 54,68 | 4,0,95 | 100,114 | 3,0,100 | 145,160 | 4,0,89 |
| 9,10 | 1,0,83 | 54,69 | 3,0,85 | 100,115 | 3,0,85 | 146,147 | 1,0.001,96 |
| 9,22 | 3,0,87 | 55,56 | 3,0.001,100 | 101,102 | 2,0.001,86 | 146,159 | 1,0,95 |
| 9,23 | 4,0,95 | 55,68 | 2,0,92 | 101,114 | 1,0,99 | 146,160 | 1,0,100 |
| 9,24 | 4,0.001,85 | 55,69 | 4,0.001,86 | 101,115 | 2,0,94 | 146,161 | 3,0,92 |
| 10,11 | 3,0.001,88 | 55,70 | 1,0,91 | 101,116 | 1,0,92 | 147,148 | 3,0.001,96 |
| 10,23 | 1,0,93 | 56,69 | 3,0.001,87 | 102,103 | 1,0.001,90 | 147,160 | 3,0,85 |
| 10,24 | 4,0,80 | 56,70 | 3,0,85 | 102,115 | 1,0,89 | 147,161 | 1,0,100 |
| 10,25 | 1,0.001,80 | 57,58 | 3,0,88 | 102,116 | 3,0,92 | 147,162 | 3,0,100 |
| 11,12 | 4,0.001,88 | 57,71 | 2,0,78 | 102,117 | 3,0,100 | 148,149 | 3,0,94 |
| 11,24 | 3,0,88 | 57,72 | 1,0,80 | 103,104 | 3,0,91 | 148,161 | 3,0,97 |
| 11,25 | 2,0,86 | 58,59 | 3,0,94 | 103,116 | 3,0.001,97 | 148,162 | 2,0,92 |

| 11,26 | 1,0.001,90 | 58,71 | 2,0,87 | 103,117 | 3,0,90 | 148,163 | 3,0,91 |
|---|---|---|---|---|---|---|---|
| 12,13 | 2,0,89 | 58,72 | 3,0,88 | 103,118 | 2,0,96 | 149,150 | 1,0.001,96 |
| 12,25 | 3,0,81 | 58,73 | 4,0,97 | 104,105 | 3,0.001,87 | 149,162 | 2,0,87 |
| 12,26 | 2,0,96 | 59,60 | 1,0.001,87 | 104,117 | 1,0,92 | 149,163 | 1,0.001,90 |
| 12,27 | 3,0.001,95 | 59,72 | 3,0,96 | 104,118 | 4,0,90 | 149,164 | 1,0,97 |
| 13,14 | 4,0.001,91 | 59,73 | 1,0.001,95 | 104,119 | 2,0,91 | 150,151 | 1,0.001,100 |
| 13,26 | 1,0,81 | 59,74 | 1,0,90 | 105,106 | 4,0,89 | 150,163 | 2,0,95 |
| 13,27 | 1,0,82 | 60,61 | 2,0,86 | 105,118 | 2,0,97 | 150,164 | 1,0,87 |
| 13,28 | 1,0,98 | 60,73 | 4,0,93 | 105,119 | 3,0,100 | 150,165 | 2,0,89 |
| 14,27 | 3,0,86 | 60,74 | 3,0.001,93 | 105,120 | 2,0,91 | 151,152 | 1,0.001,91 |
| 14,28 | 2,0.001,88 | 60,75 | 1,0,85 | 106,107 | 1,0.001,94 | 151,164 | 1,0,91 |
| 15,16 | 2,0.001,86 | 61,62 | 4,0,96 | 106,119 | 2,0,98 | 151,165 | 4,0.001,99 |
| 15,29 | 3,0,89 | 61,74 | 1,0,92 | 106,120 | 1,0,94 | 151,166 | 4,0,95 |
| 15,30 | 2,0.001,72 | 61,75 | 2,0,88 | 106,121 | 3,0,92 | 152,153 | 1,0.001,92 |
| 16,17 | 4,0.001,93 | 61,76 | 3,0,91 | 107,108 | 4,0,95 | 152,165 | 4,0,95 |
| 16,29 | 2,0,86 | 62,63 | 1,0.001,97 | 107,120 | 2,0,90 | 152,166 | 3,0,93 |
| 16,30 | 4,0,92 | 62,75 | 1,0,91 | 107,121 | 4,0,100 | 152,167 | 4,0,100 |
| 16,31 | 3,0,93 | 62,76 | 2,0,92 | 107,122 | 3,0,94 | 153,154 | 1,0.001,92 |
| 17,18 | 3,0.001,91 | 62,77 | 1,0,89 | 108,109 | 4,0,99 | 153,166 | 2,0,93 |
| 17,30 | 4,0,91 | 63,64 | 3,0,98 | 108,121 | 1,0,95 | 153,167 | 3,0.001,94 |
| 17,31 | 4,0,92 | 63,76 | 4,0,97 | 108,122 | 2,0.001,85 | 153,168 | 2,0,93 |
| 17,32 | 2,0,94 | 63,77 | 3,0,100 | 108,123 | 2,0,90 | 154,167 | 4,0,98 |
| 18,19 | 2,0,95 | 63,78 | 4,0,97 | 109,110 | 1,0.001,100 | 154,168 | 4,0,100 |
| 18,31 | 4,0,86 | 64,65 | 2,0.001,86 | 109,122 | 1,0,100 | 155,156 | 2,0,72 |
| 18,32 | 3,0.001,90 | 64,77 | 3,0,99 | 109,123 | 3,0,90 | 155,169 | 1,0,98 |
| 18,33 | 4,0,90 | 64,78 | 4,0,87 | 109,124 | 4,0,85 | 155,170 | 1,0,70 |
| 19,20 | 1,0,98 | 64,79 | 3,0,91 | 110,111 | 2,0.001,98 | 156,157 | 2,0.001,86 |
| 19,32 | 3,0,96 | 65,66 | 3,0.001,93 | 110,123 | 2,0,96 | 156,169 | 3,0,98 |
| 19,33 | 4,0.001,89 | 65,78 | 4,0,90 | 110,124 | 4,0,95 | 156,170 | 2,0,95 |
| 19,34 | 2,0,100 | 65,79 | 3,0.001,97 | 110,125 | 3,0,91 | 156,171 | 3,0,100 |
| 20,21 | 2,0.001,86 | 65,80 | 4,0,92 | 111,112 | 1,0,91 | 157,158 | 4,0,94 |
| 20,33 | 2,0,97 | 66,67 | 3,0.001,87 | 111,124 | 4,0,94 | 157,170 | 2,0,88 |
| 20,34 | 4,0.001,98 | 66,79 | 1,0,99 | 111,125 | 3,0,89 | 157,171 | 3,0.001,94 |
| 20,35 | 4,0,93 | 66,80 | 4,0,88 | 111,126 | 2,0,89 | 157,172 | 4,0,91 |
| 21,22 | 4,0.001,97 | 66,81 | 1,0,100 | 112,125 | 4,0.001,88 | 158,159 | 4,0.001,99 |
| 21,34 | 4,0,93 | 67,68 | 2,0.001,92 | 112,126 | 3,0,88 | 158,171 | 2,0,95 |
| 21,35 | 3,0.001,95 | 67,80 | 2,0,86 | 113,114 | 1,0.001,81 | 158,172 | 2,0,86 |
| 21,36 | 3,0,97 | 67,81 | 3,0,89 | 113,127 | 1,0,81 | 158,173 | 2,0,94 |
| 22,23 | 1,0,90 | 67,82 | 4,0,93 | 113,128 | 3,0.001,87 | 159,160 | 4,0,88 |

| 22,35 | 4,0,92 | 68,69 | 4,0.001,91 | 114,115 | 3,0,95 | 159,172 | 2,0,99 |
|---|---|---|---|---|---|---|---|
| 22,36 | 3,0.001,100 | 68,81 | 4,0,87 | 114,127 | 4,0,86 | 159,173 | 1,0,94 |
| 22,37 | 4,0,100,87 | 68,82 | 3,0.001,100 | 114,128 | 3,0.001,98 | 159,174 | 2,0,91 |
| 23,24 | 4,0,86 | 68,83 | 4,0,99 | 114,129 | 1,0,98 | 160,161 | 2,0.001,98 |
| 23,36 | 3,0,99 | 69,70 | 2,0.001,88 | 115,116 | 3,0.001,92 | 160,173 | 2,0,93 |
| 23,37 | 3,0,89 | 69,82 | 2,0,100 | 115,128 | 4,0,92 | 160,174 | 4,0,95 |
| 23,38 | 2,0,100 | 69,83 | 2,0.001,92 | 115,129 | 2,0.001,85 | 160,175 | 4,0,92 |
| 24,25 | 3,0,100 | 69,84 | 2,0,94 | 115,130 | 1,0,95 | 161,162 | 3,0,95 |
| 24,37 | 1,0,95 | 70,85 | 1,0.001,94 | 116,117 | 4,0,92 | 161,174 | 1,0,96 |
| 24,38 | 3,0.001,98 | 70,84 | 2,0,96 | 116,129 | 2,0,94 | 161,175 | 4,0,88 |
| 24,39 | 2,0,86 | 71,72 | 2,0,70 | 116,130 | 3,0,91 | 161,176 | 3,0,86 |
| 25,26 | 3,0,98 | 71,85 | 2,0,74 | 116,131 | 1,0,95 | 162,163 | 4,0.001,99 |
| 25,38 | 2,0,95 | 71,86 | 3,0.001,95 | 117,118 | 1,0,89 | 162,175 | 3,0,94 |
| 25,39 | 4,0.001,96 | 72,73 | 3,0,98 | 117,130 | 1,0,92 | 162,176 | 3,0.001,85 |
| 25,40 | 4,0,91 | 72,85 | 1,0,96 | 117,131 | 3,0,90 | 162,177 | 1,0,94 |
| 26,27 | 4,0.001,100 | 72,86 | 2,0,93 | 117,132 | 3,0,90 | 163,164 | 3,0,98 |
| 26,37 | 3,0,90 | 72,87 | 3,0,94 | 118,119 | 4,0,87 | 163,176 | 4,0,88 |
| 26,40 | 4,0,90 | 73,74 | 2,0.001,94 | 118,131 | 2,0,94 | 163,177 | 2,0.001,97 |
| 26,4 | 4,0,91 | 73,86 | 1,0,98 | 118,132 | 3,0,96 | 163,178 | 3,0,92 |
| 27,28 | 3,0.001,97 | 73,87 | 2,0,86 | 118,133 | 1,0,94 | 164,165 | 3,0.001,97 |
| 27,20 | 3,0,96 | 73,88 | 3,0,92 | 119,120 | 2,0.001,90 | 164,177 | 1,0,91 |
| 27,41 | 4,0.001,99 | 74,75 | 1,0,100 | 119,132 | 2,0,86 | 164,178 | 1,0.001,99 |
| 27,42 | 2,0,91 | 74,87 | 3,0,86 | 119,133 | 4,0.001,87 | 164,179 | 1,0,95 |
| 28,41 | 4,0.001,97 | 74,88 | 4,0.001,88 | 119,134 | 1,0,88 | 165,166 | 1,0,90 |
| 28,42 | 1,0,96 | 74,89 | 4,0,97 | 120,121 | 4,0.001,91 | 165,178 | 1,0,91 |
| 29,30 | 3,0,96 | 75,76 | 3,0,94 | 120,133 | 3,0,93 | 165,179 | 4,0,97 |
| 29,43 | 1,0,84 | 75,88 | 4,0,95 | 120,134 | 4,0,91 | 165,180 | 3,0,87 |
| 29,44 | 2,0.001,81 | 75,89 | 2,0,91 | 120,135 | 1,0.001,89 | 166,167 | 1,0,89 |
| 30,31 | 3,0,90 | 75,90 | 3,0,90 | 121,122 | 3,0,89 | 166,179 | 1,0,85 |
| 30,43 | 2,0,88 | 76,77 | 1,0.001,91 | 121,134 | 1,0,87 | 166,180 | 2,0,87 |
| 30,44 | 4,0.001,99 | 76,89 | 3,0,98 | 121,135 | 4,0,88 | 166,181 | 3,0,91 |
| 30,45 | 1,0,85 | 76,90 | 4,0.001,92 | 121,136 | 4,0,100 | 167,168 | 3,0,88 |
| 31,32 | 2,0,91 | 76,91 | 4,0,94 | 122,123 | 3,0,86 | 167,180 | 4,0,95 |
| 31,44 | 2,0,95 | 77,78 | 4,0,97 | 122,135 | 2,0,88 | 167,181 | 4,0.001,85 |
| 31,45 | 1,0,95 | 77,90 | 2,0,92 | 122,136 | 3,0,91 | 167,182 | 4,0,94 |
| 31,46 | 1,0,89 | 77,91 | 1,0.001,85 | 122,137 | 1,0.001,91 | 168,181 | 3,0.001,88 |
| 32,33 | 4,0,85 | 77,92 | 4,0,99 | 123,124 | 2,0,98 | 168,182 | 3,0,93 |
| 32,45 | 1,0,88 | 78,79 | 3,0.001,96 | 123,136 | 4,0,92 | 169,170 | 1,0.001,87 |
| 32,46 | 1,0,94 | 78,91 | 2,0,89 | 123,137 | 3,0.001,99 | 169,183 | 2,0,99 |

| 32,47 | 2,0,97 | 78,92 | 1,0.001,98 | 132,138 | 3,0,88 | 169,184 | 2,0,70 |
|---|---|---|---|---|---|---|---|
| 33,34 | 3,0.001,89 | 78,93 | 1,0,99 | 124,125 | 2,0.001,98 | 170,171 | 1,0.001,94 |
| 33,46 | 1,0,93 | 79,80 | 3,0,97 | 124,137 | 3,0,100 | 170,183 | 4,0,96 |
| 33,47 | 1,0.001,98 | 79,92 | 1,0,89 | 124,138 | 4,0.001,85 | 170,184 | 3,0,100 |
| 33,48 | 3,0,89 | 79,93 | 3,0,88 | 124,139 | 1,0,100 | 170,185 | 1,0,86 |
| 34,35 | 2,0.001,89 | 79,94 | 2,0,85 | 125,126 | 1,0,97 | 171,172 | 4,0,97 |
| 34,47 | 4,0,90 | 80,81 | 4,0,94 | 125,138 | 2,0,100 | 171,184 | 1,0,93 |
| 34,48 | 4,0,86 | 80,93 | 4,0,100 | 125,139 | 4,0.001,94 | 171,185 | 1,0.001,99 |
| 34,49 | 1,0,97 | 80,94 | 1,0.001,85 | 125,140 | 4,89,89 | 171,186 | 1,0,96 |
| 35,36 | 4,0,86 | 80,95 | 4,0,98 | 126,139 | 1,94,84 | 172,173 | 1,0.001,94 |
| 35,48 | 1,0,89 | 81,82 | 4,0,85 | 126,140 | 1,95,85 | 172,185 | 3,0.001,91 |
| 35,49 | 3,0.001,95 | 81,94 | 2,0,93 | 127,128 | 3,0.001,91 | 172,186 | 1,0,97 |
| 35,50 | 1,0,85 | 81,95 | 4,0.001,100 | 127,141 | 1,0,76 | 172,187 | 1,0.001,100 |
| 36,37 | 3,0,88 | 81,96 | 2,0,93 | 127,142 | 2,0,85 | 173,174 | 2,0,92 |
| 36,49 | 1,0,99 | 82,83 | 2,0,99 | 128,129 | 3,0.001,91 | 173,186 | 3,0.001,99 |
| 36,50 | 4,0,96 | 82,95 | 2,0,93 | 128,141 | 4,0,96 | 173,187 | 2,0,89 |
| 36,51 | 4,0,88 | 82,96 | 2,0.001,97 | 128,142 | 2,0.001,96 | 173,188 | 1,0,98 |
| 37,38 | 2,0.001,100 | 82,97 | 1,0,86 | 128,143 | 3,0,98 | 174,175 | 2,0.001,99 |
| 37,50 | 4,0,90 | 83,84 | 4,0,99 | 129,130 | 3,0.001,89 | 174,187 | 2,0.001,99 |
| 37,51 | 1,0,90 | 83,96 | 1,0,85 | 129,142 | 4,0,99 | 174,188 | 4,0,90 |
| 37,52 | 2,0,92 | 83,97 | 3,0,91 | 129,143 | 3,0.001,98 | 174,189 | 4,0,97 |
| 38,39 | 2,0.001,89 | 83,98 | 3,0,97 | 129,144 | 4,0,85 | 175,176 | 4,0.001,97 |
| 28,51 | 2,0,97 | 84,97 | 1,0.001,88 | 130,131, | 2,0.001,86 | 175,188 | 4,0,95 |
| 38,52 | 4,0,90 | 84,98 | 4,0,85 | 130,143 | 3,0,91 | 175,189 | 4,0,92 |
| 38,53 | 1,0,97 | 85,86 | 1,0,90 | 130,144 | 3,0.001,91 | 175,190 | 2,0.001,95 |
| 39,40 | 3,0,95 | 85,99 | 1,0,95 | 130,145 | 1,0,90 | 176,177 | 1,0.001,85 |
| 39,52 | 1,0,94 | 85,100 | 2,0.001,91 | 131,132 | 4,0.001,95 | 176,189 | 1,0.001,87 |
| 39,53 | 4,0.001,90 | 86,87 | 2,0,92 | 131,144 | 1,0,88 | 176,190 | 1,0,86 |
| 39,54 | 1,0,87 | 86,99 | 2,0,93 | 131,145 | 4,0.001,94 | 176,191 | 2,0,85 |
| 40,41 | 3,0.001,100 | 86,100 | 3,0.001,98 | 131,146 | 2,0,100 | 177,178 | 4,0,90 |
| 40,53 | 2,0,98 | 86,101 | 4,0,98 | 132,133 | 3,0.001,99 | 177,190 | 1,0.001,100 |
| 40,54 | 1,0,89 | 87,88 | 2,0.001,94 | 132,145 | 2,0,86 | 177,191 | 3,0,93 |
| 40,55 | 4,0,87 | 87,100 | 1,0,93 | 132,146 | 2,0,96 | 177,192 | 1,0.001,94 |
| 41,42 | 4,0,91 | 87,101 | 4,0.001,93 | 132,147 | 1,0,84 | 178,179 | 3,0.001,94 |
| 41,54 | 4,0,92 | 87,102 | 1,0,87 | 133,134 | 3,0,98 | 178,191 | 1,0.001,85 |
| 41,55 | 4,0.001,89 | 88,89 | 1,0,97 | 133,146 | 4,0,88 | 178,192 | 4,0,100 |
| 41,56 | 1,0,98 | 88,101 | 4,0,92 | 133147 | 1,0,91 | 178,193 | 2,0,97 |
| 42,55 | 1,0,96 | 88,102 | 1,0.001,100 | 133,148 | 2,0,85 | 178,180 | 2,0,91 |
| 42,56 | 1,0,98 | 88,103 | 4,0,88 | 134,135 | 3,0,89 | 179,192 | 2,0.001,86 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 43,44 | 3,0,87 | 89,90 | 4,0,96 | 134,147 | 3,0,93 | 179,193 | 1,0,98 |
| 43,57 | 1,0,86 | 89,102 | 1,0,86 | 134,148 | 1,0.001,95 | 179,194 | 1,0.001,86 |
| 43,58 | 1,0,74 | 89,103 | 2,0,91 | 134,149 | 3,0,88 | 180,181 | 1,0.001,97 |
| 44,45 | 3,0.001,91 | 89,104 | 2,0,99 | 135,136 | 2,0,86 | 180,193 | 1,0.001,98 |
| 44,57 | 2,0,86 | 90,91 | 2,0,89 | 135,148 | 1,0,91 | 180,194 | 3,0,87 |
| 44,58 | 4,0,93 | 90,103 | 4,0,99 | 135,149 | 2,0,87 | 180,195 | 1,0.001,96 |
| 44,59 | 3,0,93 | 90,104 | 2,0,94 | 135,150 | 2,0,98 | 181,182 | 2,0,86 |
| 45,46 | 4,0.001,100 | 90,105 | 3,0,90 | 136,137 | 4,0.001,91 | 181,194 | 2,0.001,94 |
| 45,58 | 1,0,88 | 91,92 | 1,0,96 | 136,149 | 1,0,88 | 181,195 | 3,0,85 |
| 45,59 | 2,0,82 | 91,104 | 2,0,95 | 136,150 | 3,0.001,98 | 181,196 | 1,0,99 |
| 45,60 | 2,0,98 | 91,105 | 2,0,95 | 136,151 | 3,0,99 | 182,195 | 1,0.001,96 |
| 46,47 | 4,0,87 | 91,106 | 2,0,94 | 137,138 | 4,0,82 | 182,196 | 1,0,85 |
| 46,59 | 4,0,100 | 92,93 | 1,0.001,89 | 92,105 | 4,0,91 | 195,196 | 1,0,92 |