



Universidad  
de Alcalá

Programa de Doctorado en Tecnologías de la  
Información y las Comunicaciones

Contribución a tecnologías  
habilitantes en redes programables y  
definidas por software

Tesis Doctoral presentada por  
**David Carrascal Acebron**

2025





Universidad  
de Alcalá

Programa de Doctorado en Tecnologías de la  
Información y las Comunicaciones

Contribución a tecnologías  
habilitantes en redes programables y  
definidas por software

Autor

**David Carrascal Acebron**

Directores

**Dr. Elisa Rojas Sánchez**

**Dr. Diego López Pajares**

Alcalá de Henares

2025



*“No os preguntarán por mí,  
que en estos tiempos a nadie le da lustre  
haber nacido segundón en casa grande;  
pero si pregunta alguno,  
bueno será contestarle  
que, español, a toda vena,  
amé, reñí, di mi sangre,  
pensé poco, recé mucho,  
jugué bien, perdí bastante.*

*Y, porque esa empresa loca que nunca debió tentarme,  
que, perdiendo ofende a todos,  
que, triunfando alcanza a nadie,  
no quise salir del mundo  
sin poner mi pica en Flandes.*

*¡Por España!*

*Y el que quiera defenderla honrado muera.  
Y el traidor que la abandone,  
no tenga quien le perdone,  
ni en Tierra Santa cobijo,  
ni una cruz en sus despojos,  
ni las manos de un buen hijo,  
para cerrarle los ojos.”*

HERNANDO DE ACUÑA



Este trabajo ha sido parcialmente financiado por la Universidad de Alcalá a través del programa FPI-UAH 2022; por la Comunidad de Madrid mediante los proyectos TAPIR-CM (S2018/TCS-4496), MistLETOE-CM (CM/JIN/2021-006), TUCAN6-CM (TEC-2024/COM-460) y VERANO (CM/DEMG/2024-038); por el MICIU y la Unión Europea NextGenerationEU/PRTR a través del proyecto ADMINISTER (TED2021-131301B-I00); por el Ministerio de Ciencia e Innovación mediante el proyecto ONENESS (PID2020-116361RA-I00); y por el Ministerio de Asuntos Económicos y Transformación Digital y la Unión Europea–NextGenerationEU a través del proyecto UNICO 5G I+D 6G-DATADRIVEN-02, coordinado por la Universidad Carlos III de Madrid.



# Agradecimientos

Es increíble cómo pasa el tiempo, y da vértigo echar la vista atrás. Pensar en cómo he llegado hasta aquí, las conferencias a las que he tenido la suerte de asistir, los viajes que he podido realizar, y, sobre todo, en las personas que he conocido por el camino. A menudo se describe el doctorado como un proceso arduo y exigente, y no diré lo contrario, pero, también merece la pena poner en valor todo lo aprendido, lo vivido y las experiencias que me han hecho crecer tanto a nivel personal como profesional.

Escribo estas líneas desde Milán, durante mi estancia doctoral junto a Marco Savi, a quien quiero agradecer sinceramente todo el tiempo, dedicación y esfuerzo que me ha brindado. Su apoyo ha hecho que esta etapa sea verdaderamente inolvidable. Esta experiencia no habría sido la misma sin todas las personas maravillosas que he conocido en Italia, que me han hecho sentir como en casa, siempre con una sonrisa y dispuestas a ayudar.

Volviendo la mirada a casa, no puedo dejar de agradecer a mis amigos, que han estado siempre a mi lado, apoyándome en cada paso. A mi familia, por su paciencia infinita, por su ayuda constante y por soportarme en los momentos de estrés y agobio de esta “empresa loca” que es el doctorado. A mis amigos del LE-34, los que están y los que ya no están, con quienes he compartido tantos cafés, confidencias, comidas y tardes de “trabajo”. Y, por supuesto, a todos mis compañeros del grupo NetIS, que tras tantos años, más que colegas se han convertido en una segunda familia. Gracias por creer en mí y por acompañarme hasta aquí.

Por último, a Elisa y Diego, mis directores de tesis. Gracias por vuestra guía, vuestro apoyo incondicional y por confiar en mí desde el primer momento. En especial a Elisa: si no fuera por aquel correo que me enviaste hace ya más de siete años, ofreciéndome una mini beca de investigación, hoy no estaría escribiendo estas líneas. Gracias por todo lo que me habéis enseñado, por lo que he aprendido con vosotros y por todo lo que aún me queda por aprender.

Sinceramente, mil gracias a todos.



# Resumen

La era contemporánea en la cual vivimos se caracteriza en mayor medida por una profunda transformación tecnológica y social, impulsada por la globalización y por el desarrollo de infraestructuras digitales interconectadas que configuran lo que se conoce como el Internet of Everything (IoE). En este nuevo paradigma, emergen redes densas y altamente heterogéneas en las que convergen dispositivos, servicios y plataformas con requerimientos funcionales muy variopintos, integrando no solo redes de comunicaciones, sino también infraestructuras energéticas, industriales y logísticas. Esta complejidad creciente, demanda nuevas metodologías para un control y gestión, que sea en la medida de lo posible, flexible y escalable. En este contexto, las redes softwarizadas y programables se postulan como un elemento tecnológico clave, al permitir una abstracción funcional de la infraestructura subyacente, facilitar su automatización y promover la integración de capacidades de control, así como, la adaptación dinámica de la red, a las necesidades intrínsecas de los nodos de la misma.

Esta Tesis contribuye al desarrollo de las redes softwarizadas y programables mediante la propuesta de soluciones orientadas a mejorar la gestión, la resiliencia y la cooperación entre nodos de dichas redes. En primer lugar, se diseñan y evalúan algoritmos de toma de decisiones inteligentes en entornos altamente dinámicos y heterogéneos, con aplicación en dominios como el Internet de las Cosas Industrial (IIoT), las redes eléctricas inteligentes (Smart Grids) y las redes de comunicaciones de nueva generación. Estas soluciones permiten una asignación dinámica de recursos, una adaptación proactiva a las condiciones del entorno y una reducción sustancial en la complejidad de gestión de la red. Además, se ha abordado la integración de modelos de Inteligencia Artificial (AI) con dichos algoritmos, con el fin de potenciar la detección temprana de fallos, lo que se traduce en una mejora significativa de la resiliencia, y la alta disponibilidad de los servicios. En segundo lugar, se propone una arquitectura software modular, orientada a servicios y alineada con estándares actuales, que permite la incorporación de herramientas emergentes, así como mecanismos automatizados con AI, ofreciendo capacidades de computación tanto en la nube, como en el edge. Esta arquitectura está concebida para proporcionar una infraestructura lógica robusta, interoperable y escalable, capaz de facilitar la orquestación autónoma de servicios distribuidos, orientado a contextos de elevada heterogeneidad tecnológica, como entornos IIoT u otros.

**Palabras clave:** Redes densas y heterogéneas, Redes programables y softwarizadas, Algoritmos, Infraestructura Cloud, IoT industrial, Smart grids.



# Abstract

The contemporary era is marked by a profound technological and social transformation, driven by globalization and the pervasive deployment of interconnected digital infrastructures that define the Internet of Everything (IoE). Within this emerging paradigm, dense and highly heterogeneous networks are formed, comprising a wide variety of devices, services, and platforms with diverse and demanding functional requirements. These systems integrate not only communication networks, but also energy, industrial, and logistics infrastructures. The increasing complexity of such environments necessitates the adoption of novel methodologies capable of ensuring the flexible and scalable control and management of distributed resources and services. In this context, softwarized and programmable networks have emerged as a pivotal technological solution, enabling functional abstraction of the underlying infrastructure, supporting automation, and fostering the integration of advanced control mechanisms, as well as the dynamic adaptation of network behavior to the intrinsic requirements of its nodes.

This Thesis contributes to the advancement of softwarized and programmable networking by proposing solutions designed to enhance the management, resilience, and cooperation between nodes within these infrastructures. Firstly, it presents and evaluates intelligent decision-making algorithms tailored for highly dynamic and heterogeneous environments, with applications in domains such as the Industrial Internet of Things (IIoT), smart grids, and next-generation communication networks. These algorithms facilitate dynamic resource allocation, proactive environmental adaptation, and a significant reduction in network management complexity. Furthermore, the integration of Artificial Intelligence (AI) models into these solutions is explored to enable early fault detection, thus improving system resilience and ensuring high service availability. Secondly, the Thesis proposes a modular, service-oriented software architecture, aligned with current standards and capable of incorporating emerging technologies and AI-driven automation mechanisms. This architecture offers computing capabilities across both cloud and edge infrastructures and is designed to deliver a robust, interoperable, and scalable logical platform that supports the autonomous orchestration of distributed services in technologically heterogeneous scenarios such as IIoT environments.

**Keywords:** Dense and heterogeneous networks, Programmable and softwarized networks, Algorithms, Cloud infrastructure, Industrial IoT, Smart Grids.



# Índice general

<b>Resumen</b>	<b>VII</b>
<b>Abstract</b>	<b>IX</b>
<b>Índice general</b>	<b>XI</b>
<b>Índice de figuras</b>	<b>XV</b>
<b>Índice de tablas</b>	<b>XIX</b>
<b>Índice de algoritmos</b>	<b>XXI</b>
<b>Lista de acrónimos</b>	<b>XXIII</b>
<b>1. Introducción y objetivos</b>	<b>1</b>
1.1. Introducción . . . . .	1
1.2. Redes programables y definidas por software . . . . .	2
1.3. Planteamiento del problema y objetivos de la tesis . . . . .	4
1.4. Estructura de la tesis . . . . .	6
1.5. Contribuciones . . . . .	8
<b>2. Estado del Arte</b>	<b>11</b>
2.1. Las redes SDN . . . . .	11
2.1.1. Arquitectura lógica de las redes SDN . . . . .	12
2.1.2. Arquitectura física de las redes SDN . . . . .	18
2.1.3. Propuestas de despliegue con control <i>in-band</i> . . . . .	21
2.1.4. Conclusiones y alineación con los objetivos de la Tesis . . . . .	26
2.2. Servicios y Tecnologías habilitantes en redes softwarizadas y heterogéneas	27
2.2.1. Servicios básicos: arranque y provisión de canales de control en entornos densos y heterogéneos . . . . .	28
2.2.1.1. Terminología básica . . . . .	30
2.2.1.2. Propuestas de etiquetado jerárquico . . . . .	34
2.2.1.3. Conclusiones y alineación con los objetivos de la Tesis . . . . .	37
2.2.2. Servicios avanzados: gestión y planificación de recursos en entornos densos y heterogéneos . . . . .	38

2.2.2.1.	Terminología básica . . . . .	39
2.2.2.2.	Propuestas de enfoques para la gestión y planeamiento de recursos . . . . .	41
2.2.2.3.	Conclusiones y alineación con los objetivos de la Tesis . . . . .	42
2.2.3.	Servicios avanzados: optimización y reconfiguración proactiva en entornos densos y heterogéneos . . . . .	44
2.2.3.1.	Terminología básica . . . . .	45
2.2.3.2.	Propuestas de optimización y reconfiguración proactiva de la red . . . . .	48
2.2.3.3.	Conclusiones y alineación con los objetivos de la Tesis . . . . .	51
2.3.	Casos de uso en entornos densos y heterogéneos . . . . .	52
2.3.1.	Redes inteligentes de distribución eléctrica . . . . .	52
2.3.1.1.	Propuestas en el contexto del EI y las PRGs/SG . . . . .	53
2.3.1.2.	Conclusiones y alineación con los objetivos de la Tesis . . . . .	55
2.3.2.	Arquitecturas inteligentes de sensores IoT/IIoT . . . . .	56
2.3.2.1.	Propuestas de arquitecturas inteligentes IIoT . . . . .	57
2.3.2.2.	Conclusiones y alineación con los objetivos de la Tesis . . . . .	58
<b>3. Planteamiento del problema</b>		<b>61</b>
<b>4. Propuesta escalable de control In-band para entornos inalámbricos y de baja capacidad</b>		<b>65</b>
4.1.	Introducción . . . . .	65
4.2.	Descripción del protocolo . . . . .	67
4.2.1.	Proceso de reconocimiento de vecindad . . . . .	69
4.2.2.	Proceso de etiquetado jerárquico . . . . .	71
4.3.	Implementación de wireless In-Band Basic OpenFlow Software Switch (win-BOFUSS) . . . . .	74
4.4.	Evaluación . . . . .	77
4.5.	Conclusiones . . . . .	79
<b>5. DEN2NE</b>		<b>81</b>
5.1.	Introducción . . . . .	81
5.2.	Definición del algoritmo . . . . .	83
5.2.1.	Terminología del algoritmo . . . . .	85
5.2.2.	Fase 1 - Etiquetado jerárquico . . . . .	87
5.2.3.	Fase 2 - Selección de las mejores etiquetas o identificadores jerárquicos . . . . .	90
5.2.3.1.	Criterio 1 - Número de saltos . . . . .	90
5.2.3.2.	Criterio 2 - Distancia . . . . .	91
5.2.3.3.	Criterio 3 - Pérdidas de enlace . . . . .	92
5.2.3.4.	Criterio 4 - Balance de potencia . . . . .	94
5.2.3.5.	Criterio 5 - Balance de potencia con pérdidas . . . . .	95
5.2.3.6.	Criterio 6 - Balance de potencia ponderado . . . . .	95
5.2.4.	Fase 3 - Balance global . . . . .	97
5.3.	Entorno de evaluación . . . . .	101

5.4.	Resultados y evaluación del algoritmo . . . . .	104
5.4.1.	Flujo de balance de recursos . . . . .	104
5.4.2.	Tiempo de convergencia de la asignación de etiquetas . . . . .	108
5.4.3.	Tiempo de convergencia del balance de recursos . . . . .	108
5.5.	Conclusiones . . . . .	111
<b>6.</b>	<b>Propuesta de reconfiguración y predicción de errores en redes de distribución eléctrica</b>	<b>113</b>
6.1.	Introducción . . . . .	113
6.2.	Búsqueda y análisis de fuentes de datos . . . . .	114
6.2.1.	Análisis del dataset - SustDataED . . . . .	117
6.2.2.	Deficiencias del dataset - SustDataED . . . . .	118
6.2.3.	Dataset final . . . . .	121
6.3.	Algoritmo de reconfiguración y propuesta de extensión . . . . .	123
6.4.	Evaluación . . . . .	125
6.4.1.	Técnicas de Machine Learning . . . . .	127
6.4.1.1.	Puntuación de características . . . . .	128
6.4.1.2.	Optimización de hiperparámetros . . . . .	130
6.4.1.3.	Reducción de dimensionalidad . . . . .	132
6.4.2.	Técnicas de Deep Learning . . . . .	133
6.4.2.1.	Optimización de hiperparámetros . . . . .	133
6.5.	Resultados y discusión de los modelos entrenados . . . . .	137
6.5.1.	Random Forest . . . . .	137
6.5.2.	Support Vector Machine . . . . .	137
6.5.3.	ANNs . . . . .	138
6.5.4.	Generalización a otras topologías y algoritmos de reconfiguración .	140
6.6.	Conclusiones . . . . .	140
<b>7.</b>	<b>BLOSTE</b>	<b>141</b>
7.1.	Introducción . . . . .	141
7.2.	Enfoque jerárquico de redistribución de energía basado en árboles . . . . .	143
7.2.1.	Construcción de la topología lógica . . . . .	144
7.2.1.1.	Procedimiento de etiquetado jerárquico basado en un árbol	146
7.2.1.2.	Procedimiento de etiquetado jerárquico basado en múltiples árboles . . . . .	148
7.2.2.	Criterios para construir la topología lógica . . . . .	149
7.2.2.1.	Número de saltos . . . . .	150
7.2.2.2.	Bajas pérdidas de enlace . . . . .	150
7.2.2.3.	Balance de potencia local a cero . . . . .	151
7.2.2.4.	Balance de potencia local a cero con pérdidas . . . . .	152
7.2.3.	Balanceo utilizando la topología lógica . . . . .	152
7.2.4.	Resiliencia y tasa de refresco . . . . .	154
7.2.5.	Implicaciones prácticas . . . . .	155
7.3.	Evaluación . . . . .	156
7.3.1.	Evaluación sobre la topología IEEE 123-Node Test Feeder . . . . .	156

7.3.1.1.	Resultados base . . . . .	160
7.3.1.2.	Resultados <i>All Roots</i> . . . . .	164
7.3.2.	Evaluación sobre la topología IEEE 34-Node Test Feeder . . . . .	167
7.3.3.	Estudio de la complejidad computacional . . . . .	171
7.3.3.1.	Complejidad computacional del mecanismo de etiquetado del algoritmo . . . . .	172
7.3.3.1.1.	MST (Prim) . . . . .	173
7.3.3.1.2.	PSO discreto . . . . .	173
7.3.3.1.3.	Algoritmo Genético (GA) . . . . .	174
7.3.3.1.4.	Resultados . . . . .	174
7.3.3.2.	Complejidad computacional del mecanismo iterativo del algoritmo . . . . .	175
7.3.3.2.1.	Resultados . . . . .	176
7.4.	Conclusiones . . . . .	178
<b>8.</b>	<b>Propuesta de arquitectura inteligente y softwarizada enfocada al IIoT</b>	<b>181</b>
8.1.	Introducción . . . . .	181
8.2.	Descripción de la arquitectura . . . . .	182
8.3.	Implementación y despliegue de la arquitectura . . . . .	184
8.3.1.	Análisis del conjunto de datos y entrenamiento del modelo de clasificación . . . . .	185
8.3.1.1.	Análisis de conjuntos de datos IIoT . . . . .	185
8.3.1.2.	Entrenamiento del modelo de clasificación del estado de eficiencia de los sensores IIoT . . . . .	188
8.3.2.	Implementación de bloques funcionales básicos . . . . .	189
8.3.2.1.	Bloque de <i>Factory Floor</i> . . . . .	189
8.3.2.2.	Bloque de Control . . . . .	191
8.3.2.3.	Bloque de Base de datos . . . . .	192
8.3.2.4.	Bloque de Inferencias . . . . .	193
8.3.2.5.	Bloque de Monitorización . . . . .	194
8.3.3.	Despliegue de arquitectura en Kubernetes . . . . .	194
8.3.3.1.	Mecanismos de escalabilidad y tolerancia a fallos . . . . .	196
8.3.3.2.	Flujo de datos entre componentes de la arquitectura . . . . .	197
8.4.	Evaluación experimental . . . . .	198
8.4.1.	Utilización de recursos y escalabilidad . . . . .	198
8.4.2.	Latencia de inferencia e ingestión de datos . . . . .	203
8.4.3.	Estimación del tiempo promedio de reconfiguración de red . . . . .	206
8.5.	Conclusiones . . . . .	206
<b>9.</b>	<b>Conclusiones y trabajo futuro</b>	<b>209</b>
9.1.	Conclusiones generales . . . . .	209
9.2.	Líneas de trabajo futuro . . . . .	210
<b>A.</b>	<b>Repositorios públicos</b>	<b>213</b>

# Índice de figuras

1.1.	Diagrama general del marco general de la Tesis. . . . .	5
1.2.	Línea de tiempo de contribuciones científicas (2023–2025): publicaciones en revistas y conferencias internacionales. . . . .	10
2.1.	Paradigma en las redes SDN. . . . .	12
2.2.	Arquitectura lógica de las redes SDN. . . . .	13
2.3.	Arquitectura básica de switch OpenFlow. . . . .	15
2.4.	Arquitectura básica de switch P4. . . . .	17
2.5.	Paradigmas de control en las redes SDN. . . . .	19
2.6.	Paradigmas de control Management–Control Continuum (MCC). . . . .	27
2.7.	Elección de una topología lógica (árbol enraizado) partiendo de una topología física. . . . .	32
2.8.	Ejemplo de difusión empleando el etiquetado jerárquico. . . . .	33
2.9.	Ejemplo de un sistema centralizado y un sistema distribuido. . . . .	40
2.10.	Resumen de los principales enfoques para la optimización y reconfiguración proactiva de la red. . . . .	47
4.1.	Estudio del crecimiento de las conexiones de dispositivos IoT desglosadas por segmento [143]. . . . .	66
4.2.	Proceso de reconocimiento de vecindad empleando las tablas locales de vecinos. . . . .	70
4.3.	Proceso de difusión de etiquetas Hierarchical Local MAC (HLMAC) en la topología de ejemplo. . . . .	72
4.4.	Establecimiento de la topología lógica. . . . .	73
4.5.	Evolución cronológica del desarrollo del software switch BOFUSS. . . . .	74
4.6.	Diagrama de flujo de la implementación sobre el software switch BOFUSS. . . . .	75
4.7.	Entorno de validación mediante emulación empleando el módulo del <i>kernel</i> <code>mac80211_hwsim</code> . . . . .	77
4.8.	Topología de ejemplo elegida para el banco de pruebas. . . . .	78
4.9.	Prueba funcional de la implementación en la topología básica de ejemplo. . . . .	79
5.1.	Dos ejemplos de redes que aprovecharían DEN2NE para equilibrar los recursos. . . . .	83
5.2.	Ejemplo de equilibrio de recursos en una red con seis nodos y un <i>gateway</i> (o nodo raíz). . . . .	86

5.3.	Procedimiento de etiquetado jerárquico en DEN2NE . . . . .	87
5.4.	Mecanismo de evitación de bucles durante el etiquetado jerárquico. . . . .	88
5.5.	Selección de ID basada en el criterio del número de saltos. . . . .	91
5.6.	Selección de ID basada en el criterio de distancia. . . . .	92
5.7.	Selección de ID basada en el criterio de pérdidas de enlace. . . . .	93
5.8.	Selección de ID basada en el criterio de balance de potencia. . . . .	94
5.9.	Selección de ID basada en el criterio de balance de potencia con pérdidas.	95
5.10.	Selección de ID basada en el criterio de balance de potencia ponderado. .	96
5.11.	Ejemplo del procedimiento de balance global. . . . .	98
5.12.	Generador de topologías para DEN2NE basado en BRITE. . . . .	101
5.13.	Topologías aleatorias empleadas en el estudio de DEN2NE. . . . .	102
5.14.	Función de densidad de probabilidad para la generación de cargas aleatorias en cada topología. . . . .	103
5.15.	Flujo de balance de recursos ( <i>abs_flux</i> ) - Tres escenarios y dos grados de red. . . . .	105
5.16.	Balance total de recursos ( <i>total_balance</i> ) - Tres escenarios y dos grados de red. . . . .	107
5.17.	Tiempo de convergencia de la asignación de etiquetas - Tres escenarios y dos grados de red. . . . .	109
5.18.	Tiempo de convergencia del balance de recursos - Tres escenarios y dos grados de red. . . . .	110
6.1.	Series temporales de los diferentes despliegues del dataset <i>SustDataED</i> . . .	117
6.2.	Comparación entre <i>Global Solar Atlas</i> y <i>PWatts</i> , valores mensuales totales de producción de energía fotovoltaica (PVOUT) frente a la irradiación directa normal (DNI). . . . .	119
6.3.	Comparación mediante matrices de correlación de los parámetros de producción fotovoltaica. . . . .	120
6.4.	Patrones de correlación entre los parámetros de producción en <i>PWatts</i> . .	120
6.5.	Esquema de procesamiento de datos para la generación del dataset final. .	122
6.6.	Ejemplo de redistribución de carga en una SG. . . . .	123
6.7.	Funcionamiento de la primera fase del algoritmo DEN2NE. . . . .	124
6.8.	Definición de la <i>pipeline</i> para detectar y predecir errores durante el proceso de distribución de energía. . . . .	126
6.9.	Puntuación de la importancia de las características para el RF utilizando el método <i>feature_importances_</i> . . . . .	129
6.10.	Puntuación de la importancia de las características para el RF utilizando el método <i>permutation_importance</i> . . . . .	129
6.11.	Puntuación de importancia de las características para el SVM utilizando los métodos <i>support_vectors_</i> y <i>dual_coef_</i> . . . . .	130
6.12.	Ánálisis de la precisión del modelo RF basado en el número de características utilizadas con el método RFECV. . . . .	132
6.13.	Ánálisis de varianza basado en el número de componentes utilizados en el PCA. . . . .	133
6.14.	Evolución de la precisión del ANN de <i>Scikit-Learn</i> . . . . .	135

6.15. Evolución de la función de pérdidas para la ANN de <i>Scikit-Learn</i> . . . . .	135
6.16. Evolución de la precisión para el ANN implementado en <i>Keras</i> . . . . .	136
6.17. Evolución de la función de pérdidas para el ANN implementado en <i>Keras</i> . .	136
7.1. Ejemplo de distribución de potencia. . . . .	144
7.2. Establecimiento de la topología lógica. . . . .	145
7.3. Procedimiento de etiquetado jerárquico. . . . .	147
7.4. Árboles jerárquicos tras el proceso de difusión del etiquetado desde el nodo raíz $G$ . . . . .	148
7.5. Árboles jerárquicos tras el proceso de difusión del etiquetado desde el nodo raíz $D$ . . . . .	149
7.6. Balanceo de potencia mediante topología lógica: un enfoque iterativo. . . . .	153
7.7. Topología IEEE 123-Node Test Feeder [163]. . . . .	157
7.8. Caracterización de la configuración de carga utilizada en la Topología IEEE 123-Node Test Feeder. . . . .	158
7.9. Jerarquía de clases para simular la Topología IEEE 123-Node Test Feeder.	158
7.10. Balance de potencia global en promedio y valor absoluto del flujo de potencia - Resultados base. . . . .	161
7.11. Balance temporal de la potencia global en el escenario con pérdidas - Resultados base. . . . .	162
7.12. Balance temporal de la potencia global en el escenario con pérdidas y capacidades - Resultados base. . . . .	163
7.13. Promedio de tiempo/iteraciones hasta la convergencia - Resultados base. .	164
7.14. Balance de potencia global en promedio y valor absoluto del flujo de potencia - Resultados <i>All Roots</i> . . . . .	165
7.15. Promedio de tiempo/iteraciones hasta la convergencia - Resultados <i>All Roots</i> .	165
7.16. Balance temporal de la potencia global en el escenario con pérdidas - Resultados <i>All Roots</i> . . . . .	166
7.17. Balance temporal de la potencia global en el escenario con pérdidas y capacidades - Resultados <i>All Roots</i> . . . . .	166
7.18. Topología IEEE 34-Node Test Feeder. . . . .	167
7.19. Perfiles generales de cargas a lo largo del día empleados con la topología IEEE 34-bus. . . . .	168
7.20. Balance de potencia global en promedio y valor absoluto del flujo de potencia - Topología IEEE 34-Node Test Feeder. . . . .	169
7.21. Promedio de tiempo/iteraciones hasta la convergencia - Topología IEEE 34-Node Test Feeder. . . . .	170
7.22. Balance temporal de la potencia global en el escenario con pérdidas - Topología IEEE 34-Node Test Feeder. . . . .	170
7.23. Balance temporal de la potencia global en el escenario con pérdidas y capacidades - Topología IEEE 34-Node Test Feeder. . . . .	171
7.24. Ejemplo de topología aleatoria generada de 70 nodos para el estudio de la complejidad computacional. . . . .	172
7.25. Complejidad computacional en la fase de etiquetado. . . . .	174

7.26. Comparativa de la complejidad computacional en tiempo con escala logarítmica. . . . .	175
7.27. Complejidad computacional del mecanismo iterativo. . . . .	177
7.28. Comparativa de la complejidad computacional del mecanismo iterativo en tiempo con escala logarítmica. . . . .	178
 8.1. Visión general de la arquitectura del sistema. . . . .	183
8.2. Mapa de calor de la correlación de las características numéricas del dataset. . . . .	186
8.3. Histogramas de las distribuciones y estadísticas principales para cada característica del conjunto de datos. . . . .	187
8.4. Intervalo de tiempo en segundos entre las marcas de tiempo de los registros. . . . .	188
8.5. Visualización del rendimiento del modelo: matriz de confusión y frontera de decisión. . . . .	189
8.6. Diagrama de flujo que representa el ciclo de vida de la instancia IIoT_Sensor, incluyendo la inicialización, los procedimientos de autenticación y la transmisión periódica de datos al módulo Base de datos. . . . .	190
8.7. Frontend de InfluxDB configurado con las series temporales de los sensores IIoT de una organización. . . . .	193
8.8. Panel de control del frontend de Grafana configurado con toda la información de los sensores IIoT. . . . .	194
8.9. Descripción física y lógica del despliegue del clúster. . . . .	195
8.10. Flujos de datos entre componentes de la arquitectura. . . . .	197
8.11. Uso de CPU de BentoML con escalado automático para 25 STA por AP. . . . .	202
8.12. Tiempo medio de inferencia de BentoML en función de las solicitudes concurrentes. . . . .	203
8.13. Tiempo medio de ingestión de datos en InfluxDB en función de las solicitudes concurrentes. . . . .	204
8.14. Probabilidad de bloqueo en BentoML bajo distintos umbrales de QoS. . . . .	205
8.15. Probabilidad de bloqueo en InfluxDB bajo distintos umbrales de QoS. . . . .	205

# Índice de tablas

2.1. Evolución de versiones del protocolo Openflow y el número de campos de cabecera soportados. . . . .	16
2.2. Características del control <i>in-band</i> y <i>out-of-band</i> . . . . .	21
5.1. Parámetros de generación de las topologías aleatorias. . . . .	102
5.2. Parámetros para la ejecución de pruebas. . . . .	104
6.1. Comparación de datasets sobre el consumo público de energía eléctrica. . . . .	116
6.2. Tipos de mediciones en el dataset <i>SustDataED</i> . . . . .	118
6.3. Características principales del dataset final ( <i>KeyMonData</i> ). . . . .	121
6.4. Configuración de enlaces basada en la topología IEEE 123 Node Test Feeder.	125
6.5. Evolución de la Precisión (%) aplicando la búsqueda ( <i>Grid Search</i> ) sobre el RF. . . . .	131
6.6. Evolución del Tiempo (s) de entrenamiento del RF en función del número de estimadores. . . . .	131
6.7. Resultados extraídos del método <i>cv_results_</i> de la búsqueda <i>Grid Search</i> en la SVM. . . . .	131
6.8. Precisión (%) ( <i>mean_test_score</i> ) extraída del método <i>cv_results_</i> del MLP.	134
6.9. Tiempo (s) ( <i>mean_fit_time</i> ) extraída del método <i>cv_results_</i> del MLP. .	134
6.10. Resumen de los resultados obtenidos del desarrollo de modelos ML y DL. .	139
8.1. Características principales del conjunto de datos. . . . .	185
8.2. Especificaciones de los recursos hardware asignados al despliegue. . . . .	196
8.3. Tiempo de despliegue (media $\pm$ IC) por cada etapa. . . . .	199
8.4. Consumo de memoria (MiB) promedio ( $\pm$ IC) por servicio en función de las STAs por AP. . . . .	200
8.5. Consumo de CPU (milicores) promedio ( $\pm$ IC) por servicio en función de STAs por AP. . . . .	202



# Índice de algoritmos

1.	Proceso de etiquetado jerárquico . . . . .	89
2.	Proceso de balance global . . . . .	100
3.	Pseudocódigo de alto nivel del controlador Ryu. . . . .	192



# Lista de acrónimos

<b>3GPP</b>	3rd Generation Partnership Project.
<b>5G</b>	fifth-generation mobile networks.
<b>6G</b>	sixth-generation mobile networks.
<b>ACO</b>	Ant Colony Optimization.
<b>AI</b>	Artificial Intelligence.
<b>ANN</b>	Artificial Neural Network.
<b>AP</b>	Access Point.
<b>API</b>	Application Programming Interface.
<b>ARPANET</b>	Advanced Research Projects Agency Network.
<b>BFS</b>	Breadth-First Search.
<b>BLOSTE</b>	Balanced Logical Overlay for Smart Topology Energy-routing.
<b>BOFUSS</b>	Basic OpenFlow User-space Software Switch.
<b>BRITE</b>	Boston university Representative Internet Topology gEnerator.
<b>CapEx</b>	Capital Expenditure.
<b>CEC</b>	Collaborative Edge Computing.
<b>CNI</b>	Container Network Interface.
<b>CNN</b>	Convolutional Neural Network.
<b>CoAP</b>	Constrained Application Protocol.
<b>CRO</b>	Coral Reefs Optimization.
<b>DAG</b>	Directed Acyclic Graph.
<b>DB</b>	Database.
<b>DC</b>	Data Center.
<b>DEN2NE</b>	Distributed ENergy ENvironments and NETworks.
<b>DER</b>	Distributed Energy Resource.
<b>DHCP</b>	Dynamic Host Configuration Protocol.
<b>DL</b>	Deep learning.
<b>DNI</b>	Direct Normal Irradiation.
<b>ECC</b>	Elliptic-curve cryptography.
<b>EI</b>	Energy Internet.
<b>eMBB</b>	Enhanced Mobile Broadband.
<b>ER</b>	Energy Router.

<b>FaaS</b>	Function-as-a-Service.
<b>FL</b>	Federated learning.
<b>GA</b>	Genetic Algorithm.
<b>GA3</b>	Generalized Automatic Address Assignment.
<b>GAN</b>	Generative Adversarial Network.
<b>GNN</b>	Graph Neural Networks.
<b>GPTs</b>	Generative pre-trained Transformers.
<b>gRPC</b>	google Remote Procedure Call.
<b>GW</b>	Gateway.
<b>HLMAC</b>	Hierarchical Local MAC.
<b>HPA</b>	Horizontal Pod Autoscaler.
<b>HVDC</b>	High-Voltage Direct Current.
<b>IEEE</b>	Institute of Electrical and Electronics Engineers.
<b>IIoT</b>	Industrial Internet of Things.
<b>IMGs</b>	Islanded Microgrids.
<b>IoE</b>	Internet of Everything.
<b>IoT</b>	Internet of Things.
<b>KNN</b>	k-Nearest Neighbors.
<b>LDA</b>	Linear Discriminant Analysis.
<b>LLDP</b>	Link Layer Discovery Protocol.
<b>LLNs</b>	Low-Power and Lossy Networks.
<b>LSTM</b>	Long Short-Term Memory.
<b>M2M</b>	Machine-to-Machine.
<b>MAC</b>	Medium Access Control.
<b>MCC</b>	Management–Control Continuum.
<b>MEC</b>	Mobile Edge Computing.
<b>MINLP</b>	Mixed Integer Nonlinear Programming.
<b>ML</b>	Machine Learning.
<b>MLP</b>	Multilayer Perceptron.
<b>mMTC</b>	Massive Machine Type Communications.
<b>MPTCP</b>	MultiPath TCP.
<b>MQTT</b>	Message Queuing Telemetry Transport.
<b>MST</b>	Minimum Spanning Tree.
<b>NFV</b>	Network Functions Virtualization.
<b>NTN</b>	Non-Terrestrial Network.
<b>ONF</b>	Open Networking Foundation.
<b>OOP</b>	Object-Oriented Programming.
<b>OpEx</b>	Operational Expenditure.
<b>OSPF</b>	Open Shortest Path First.

<b>OVS</b>	Open vSwitch.
<b>OVSDB</b>	Open vSwitch Database.
<b>P2P</b>	Peer-to-Peer.
<b>P4</b>	Programming Protocol-Independent Packet Processors.
<b>PCA</b>	Principal Component Analysis.
<b>PRG</b>	Power Router Grid.
<b>PSO</b>	Particle Swarm Optimization.
<b>PVCs</b>	Persistent Volume Claims.
<b>PVOUT</b>	Photovoltaic Power Potential.
<b>QoS</b>	Quality of Service.
<b>RELU</b>	Rectifier Lineal Unit.
<b>RF</b>	Random Forest.
<b>RFECV</b>	Recursive Feature Elimination with Cross Validation.
<b>RL</b>	Reinforcement learning.
<b>RNN</b>	Recurrent Neural Networks.
<b>RPL</b>	Routing Protocol for Low-Power and Lossy Networks.
<b>RSTP</b>	Rapid Spanning Tree Protocol.
<b>SDN</b>	Software-Defined Networking.
<b>SG</b>	Smart grid.
<b>SVD</b>	Singular Value Decomposition.
<b>SVM</b>	Support Vector Machine.
<b>TCAM</b>	Ternary Content-Addressable Memory.
<b>UAV</b>	Unmanned Aerial Vehicle.
<b>URLLC</b>	Ultra-Reliable Low Latency Communication.
<b>VLAN</b>	Virtual Local Area Network.
<b>WAN</b>	Wide Area Network.



# Capítulo 1

## Introducción y objetivos

En este primer capítulo, se presenta una introducción al tema principal de la tesis, además de dar un contexto y un marco general del problema que se va a abordar. Asimismo, se establecen los objetivos de la investigación, se describe la estructura general del documento y se enumeran las contribuciones principales que ha generado esta Tesis doctoral.

### 1.1. Introducción

Esta Tesis se enmarca dentro de las redes definidas por software (Software-Defined Networking (SDN), por sus siglas en inglés), y las redes programables, las cuales permiten la creación de redes flexibles y adaptables a las necesidades cambiantes de los usuarios y las aplicaciones finales. Este tipo de redes están ganando cada vez más importancia en la sociedad actual, dado que, con la creciente digitalización de la mayoría de los sectores, así como, tejido industrial y social, se están conformando redes cada vez más densas y heterogéneas, que requieren de nuevas tecnologías o herramientas que permitan su gestión y control. La tipología final de estas redes puede ser muy variopinta, pudiendo estar presentes en las redes de comunicaciones, en redes de sensores, en redes de distribución de energía, logística, entre otras.

Es por ello, que es difícil acotar el campo de estudio y aplicación de esta Tesis, y no solo por su naturaleza multidisciplinar, sino por la complejidad de que una herramienta o tecnología, sea completamente extrapolable a otro tipo de red. Por ejemplo, se pueden encontrar similitudes entre las necesidades de los distintos tipos de redes, en las redes de comunicaciones móviles fifth-generation mobile networks (5G) y sixth-generation mobile networks (6G), donde existen múltiples dispositivos, sensores y nodos de acceso que deben coordinarse dinámicamente para ofrecer conectividad. En el ámbito energético, donde las redes eléctricas inteligentes o Smart grids (SGs) destacan por la coordinación dinámica de la integración de fuentes de energía distribuidas, almacenamiento local y consumidores activos. También se puede ver en el campo de la logística, donde se puede considerar el caso de las redes de transporte y distribución, donde los vehículos, almacenes y sistemas de seguimiento deben coordinarse para optimizar rutas, minimizar tiempos de entrega

y reducir costes operativos. En todos estos escenarios, el uso de redes programables y softwarizadas ofrecen una base sólida para la gestión flexible y dinámica, sobre la cual se pueden desarrollar herramientas o tecnologías que optimicen cada caso de uso, consiguiendo escenarios más eficientes y adaptables a las necesidades de la red. De ahí que, la idea de esta Tesis sea la de ahondar en las tecnologías habilitantes de las redes programables y definidas por software, y cómo se pueden aplicar en redes densas y con nodos y necesidades heterogéneas.

## 1.2. Redes programables y definidas por software

Las redes programables tienen sus raíces en la historia de las redes de comunicación [1]. Desde su inicio, con la llegada de Advanced Research Projects Agency Network (ARPA-NET) en 1969, creada en Estados Unidos en un contexto de la guerra fría para que los investigadores pudieran intercambiar información, las redes de comunicación han evolucionado desde sistemas simples y estáticos, hacia arquitecturas más complejas y dinámicas. Durante este proceso, la necesidad de controlar y adaptar el comportamiento de la red ha sido un punto recurrente. Sin embargo, durante décadas, las redes tradicionales se caracterizaron por ser muy estáticas, estrechamente ligadas al hardware y al fabricante, lo que dificultaba su evolución y adaptación dinámica a nuevos protocolos o nuevas ideas. Es por ello que, la idea del SDN comenzó a gestarse en la Universidad de Stanford en 2003, cuando el profesor asociado de ese entonces, Nick McKeown, planteó las limitaciones de las redes convencionales y la necesidad de replantear cómo operaban los *backbones* [2]. En 2011, se acuñó el término SDN, al mismo tiempo que se lanzó la organización Open Networking Foundation (ONF) [3], encargada de establecer estándares y promover la difusión del SDN, la cual, a finales de 2023 fue incluida en la Linux Foundation para salvaguardar y reafirmar los proyectos y propuestas *open-source* en materia de *Networking*.

El paradigma SDN [4] radica en un concepto de arquitectura de red, en la que, se separa el plano de control (mantenimiento, gestión y control de la red) del plano de datos (lógica de *forwarding*) de la red, para centralizar toda la lógica de control en un único ente, el cual se denomina como controlador. Esta estructura permite lograr una administración de red más centralizada y flexible [4], facilitando la programabilidad de la red y la implementación de herramientas auxiliares que operan directamente sobre la Application Programming Interface (API) que expone el controlador. Esta API de alto nivel es clave en la integración de cualquier tipo de herramienta, desde monitorización, a Quality of Service (QoS), o incluso de modelos de Artificial Intelligence (AI)/Machine Learning (ML) para predecir o reconfigurar la red de forma automática.

Este paradigma se empezó a popularizar entre las grandes operadoras de telecomunicaciones, que junto a la tecnología de virtualización de funciones de red (del inglés, Network Functions Virtualization (NFV)) [5], podían desplegar, mantener y gestionar los servicios de red de forma dinámica y escalable, permitiendo al administrador de red operar desde un único de ente de control, toda la infraestructura. Grandes empresas como Google [6], NTT [7], IBM [8] o Telefónica [9] han contribuido activamente al desarrollo y adopción

de estas tecnologías. Este apoyo del sector tecnológico al despliegue de redes SDN con NFV se ve impulsado por la reutilización de hardware para el despliegue ágil de nuevos servicios y aplicaciones, lo que permite reducir significativamente el gasto en capital (del inglés, Capital Expenditure (CapEx)), así como la posibilidad de operar y gestionar la infraestructura de forma centralizada y programable, lo que se traduce en una disminución del gasto operativo (del inglés, Operational Expenditure (OpEx)).

Sin embargo, las redes softwarizadas y programables no se limitan únicamente a las redes de telecomunicaciones. En los últimos años, se ha podido ver cómo este paradigma se ha extendido a otros campos, como, por ejemplo, en las redes de sensores, donde la flexibilidad y la versatilidad son claves para optimizar el rendimiento de los equipos, que por lo general suelen tener recursos limitados. También se han visto integraciones de ecosistemas SDN en el ámbito de las redes de sensores Internet of Things (IoT) [10], donde se provee de una pila de protocolos que permite la interoperabilidad entre los equipos y controladores convencionales, pudiendo traer todos los avances de las redes de telecomunicación a entornos más agresivos, como por ejemplo el Industrial Internet of Things (IIoT) [11].

Incluso, se ha llegado a ver el uso de las redes programables en el ámbito de la distribución/encaminamiento de energía, donde la integración de fuentes de energía renovables y la gestión de la demanda dinámica requieren una coordinación meticulosa [12]. Históricamente las redes eléctricas de todos los países se han ido conformando en un modelo de top-to-down, donde las grandes centrales eléctricas generaban la energía, y esta se distribuía a los usuarios finales. Con el creciente aumento de la población, la red eléctrica se iba expandiendo y ramificando, creando redes en modo árbol cada vez más densas desde los puntos de interconexión. Sin embargo, con la llegada de las energías renovables y el cambio normativo promovido por la Unión Europea, este modelo tradicional ha comenzado a transformarse. La Directiva 2018/2001/UE sobre energías renovables [13], establece un marco regulador que permite a los ciudadanos y empresas convertirse en productores de energía (prosumidores), facilitando no solo el autoconsumo, sino también la posibilidad de inyectar el excedente energético a la red eléctrica. En España, esta directiva se materializó a través del Real Decreto 244/2019 [14], que regula el autoconsumo eléctrico y permite compensar económicamente la energía excedentaria. Por lo tanto, se ha pasado de un modelo de red eléctrica, top-to-down, a uno más distribuido y con múltiples puntos de generación, donde los usuarios finales pueden ser tanto consumidores como productores de la energía, haciendo que la red requiera de una mayor flexibilidad y adaptabilidad para gestionar este intercambio de dinámico de energía. Esta necesidad ha llevado a la creación de redes eléctricas inteligentes (SG), y a la incorporación de tecnologías de redes programables, que permitan la gestión dinámica de la energía. Estándares como el IEC 61850 [15] han sido claves en el desarrollo para facilitar la interoperabilidad y la comunicación entre diferentes subestaciones y dispositivos dentro de una SG, permitiendo integraciones con soluciones SDN [16, 17].

De forma similar, las redes de logística y transporte también han comenzado a adoptar modelos de redes softwarizadas [18], para optimizar la gestión de flotas, mejorando la eficiencia operativa. Por lo tanto, de forma análoga y sistemática se podría ir sector por

sector, viendo como en cada campo de aplicación se van dando redes densas y heterogéneas, que requieren de un enfoque programable, dando lugar a esta línea de investigación y desarrollo que se aborda en esta Tesis. A continuación, se presenta el planteamiento del problema y los objetivos de la Tesis, donde se aterriza cómo se va a abordar el estudio de las redes programables y softwarizadas, y en qué campos de aplicación se va a trabajar.

### 1.3. Planteamiento del problema y objetivos de la tesis

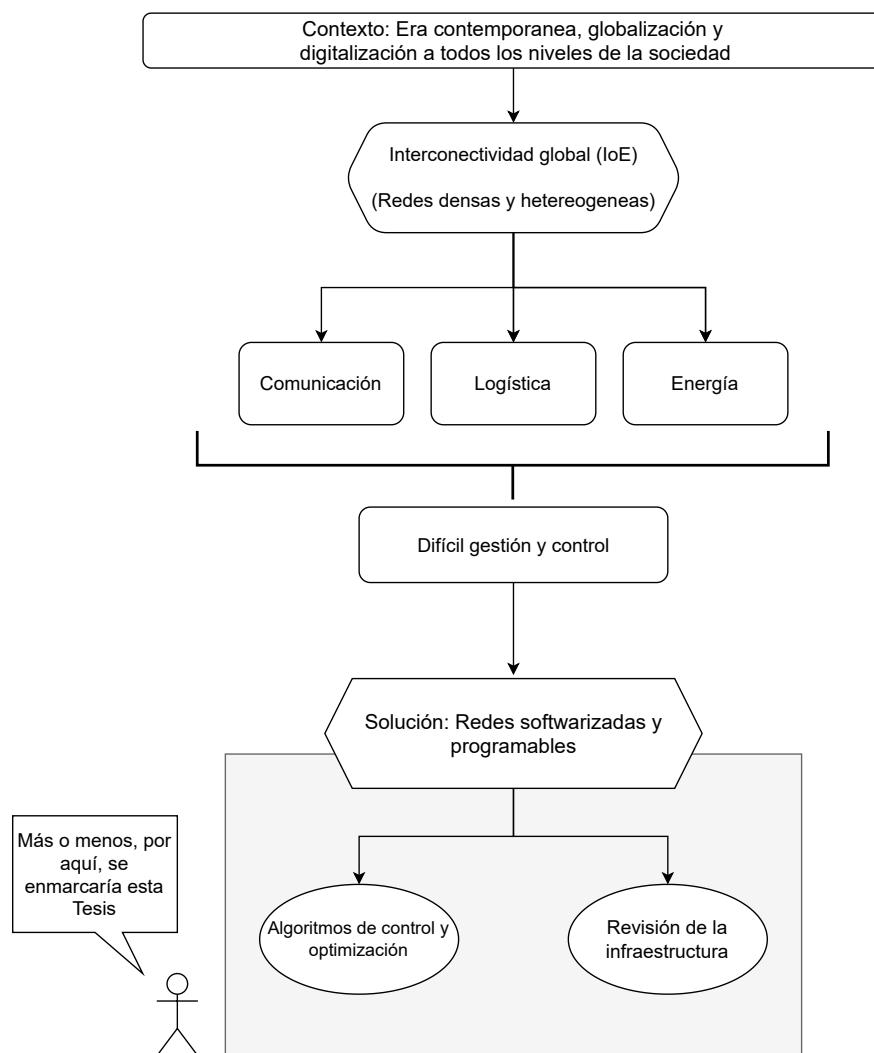
Los objetivos que se plantean en el desarrollo de la Tesis se pueden dividir en dos grandes bloques. Por un lado, se busca profundizar en el estudio de las redes programables, partiendo del escenario base de las redes SDN, y con la idea de extender este paradigma a otros campos de aplicación, como son las redes de sensores IIoT y las redes de distribución de energía. En particular, se busca profundizar en los mecanismos de control empleados en este tipo de redes, los cuales suelen organizarse en torno a dos enfoques principales: el control *out-of-band* y el control *in-band* [19]. En el modelo *out-of-band*, cada nodo de red tiene un enlace dedicado con el controlador, permitiendo una separación clara entre el plano de control y el plano de datos. Por el contrario, el modelo *in-band* asume que solo algunos nodos poseen un enlace directo con el controlador, y el resto de los dispositivos reutilizan dicho canal para transmitir información de control.

La idea de empezar profundizando este concepto radica en que, después de haber estado trabajando con ello en estudios anteriores, se ha podido ver que, en función del tipo de red y de la topología, uno u otro paradigma puede ser más adecuado, además de no haber una implementación estandarizada en el modelo *in-band*. Esto deja espacio de mejora y optimización tanto para las redes de comunicaciones, como para las redes de sensores y distribución de energía, donde este enfoque puede tener un papel importante. En redes de sensores, por ejemplo, donde los nodos suelen tener capacidades de cómputo, memoria y conectividad limitadas, implementar un enfoque *out-of-band* resulta poco viable. Asimismo, en las redes de distribución eléctrica, en las que la infraestructura sigue habitualmente una topología jerárquica de tipo árbol, no todos los nodos tienen un acceso directo al núcleo de la red, lo que hace necesario explorar soluciones basadas en el control *in-band*. Es por ello, que en el primer bloque de objetivos de la Tesis se busca profundizar en el estudio de mecanismos de control en redes densas y heterogéneas, donde el mecanismo o algoritmo pueda ser adaptado a las necesidades de la red, no solo a encañamiento sino también a la toma de decisiones de la reconfiguración de la red en aras del intercambio de recursos, como pueda ser capacidad de cómputo, o energía. Además, se contempla el uso de herramientas de AI/ML como elemento auxiliar en este proceso de control y optimización, con el fin de dotar a la red de capacidades de adaptación proactiva, pudiendo contemplar la predicción de eventos o fallos, y la mejora de las decisiones de reconfiguración y balanceo de carga en tiempo real.

Por otro lado, en el segundo bloque de objetivos de la Tesis, se busca analizar en la infraestructura que habilitan las redes softwarizadas, desde un punto de vista de la gestión y en control de la red, así como de la seguridad. En este sentido, se busca profundizar

### 1.3 PLANTEAMIENTO DEL PROBLEMA Y OBJETIVOS DE LA TESIS

en el uso de herramientas de despliegue, monitorización y gestión de red, que permitan al administrador de red tener una visión global del estado de la infraestructura, así como poder tomar decisiones automáticas sobre la reconfiguración y optimización de la red. Asimismo, se contempla el estudio de las implicaciones de rendimiento que conlleva el uso de redes programables, donde se busca identificar posibles cuellos de botella y proponer soluciones para mitigarlos. En este sentido, se contempla el uso de técnicas de AI/ML para la reconfiguración de la red, tomando métricas en la comunicación entre los nodos y el controlador. A continuación, en la Figura 1.1 se presenta un diagrama general del marco de la Tesis, donde se puede ver cómo se relacionan las distintas áreas de estudio y aplicación, así como los objetivos que se persiguen en cada una de ellas.



**Figura 1.1:** Diagrama general del marco general de la Tesis.

## 1.4. Estructura de la tesis

A continuación, se presenta la estructura general de esta memoria, describiendo brevemente el contenido de cada uno de sus capítulos. El objetivo es ofrecer una visión global del desarrollo de la Tesis, que sirva como guía para el lector y facilite la comprensión del marco completo del trabajo realizado.

El primer capítulo ha contextualizado el ámbito de las redes programables y softwarizadas, destacando su relevancia como base tecnológica para la gestión flexible y dinámica de redes heterogéneas. Asimismo, se ha introducido la problemática asociada a la creciente complejidad de las redes actuales, tanto en las redes de comunicaciones, como las de sensores o las redes de distribución energética, y se han formulado los objetivos principales que guían el desarrollo de esta Tesis. Finalmente, el capítulo concluye con una recopilación de las principales contribuciones científicas generadas a lo largo del trabajo.

El segundo capítulo se repasan los conceptos fundamentales y el estado del arte que sustentan la Tesis. El capítulo queda organizado en tres bloques principales. En el primero se abordan las redes programables y softwarizadas, partiendo del paradigma SDN, describiendo los modelos de control *out-of-band* e *in-band* y realizando un análisis detallado de las propuestas más relevantes en *in-band*. El segundo bloque se dedica a los servicios y tecnologías habilitadoras en redes softwarizadas y heterogéneas bajo la óptica del MCC (del inglés, Management–Control Continuum): aquí se examinan tanto los servicios básicos (arranque, descubrimiento y provisión de canales de control) como los servicios avanzados (gestión y planificación de recursos, optimización y reconfiguración proactiva de la red). Finalmente, el tercer bloque estudia casos de uso representativos en entornos densos y heterogéneos, con especial atención a las SG y a las arquitecturas de sensores IIoT, para mostrar cómo las tecnologías revisadas se aplican y adaptan a escenarios reales.

El tercer capítulo se sintetizan y consolidan los huecos identificados en el capítulo del Estado del Arte, con el objetivo de transformar las lecciones aprendidas en un planteamiento claro del problema, y marcar una hoja de ruta clara para la Tesis.

El cuarto capítulo se presenta la primera contribución de la Tesis, una propuesta escalable de control *in-band* para redes SDN orientada a entornos inalámbricos y de baja capacidad, directamente alineada con el primer bloque de objetivos. El capítulo describe en detalle el diseño del protocolo y su implementación práctica (**win-BOFUSS**), incluyendo los mecanismos de reconocimiento de vecindad, etiquetado jerárquico y mantenimiento de rutas de respaldo. Asimismo, se dedica una sección concreta a la validación experimental en un entorno de emulación inalámbrico, donde se analizan resultados, comportamiento operativo y consideraciones de implementación.

El quinto capítulo se presenta DEN2NE, un algoritmo escalable de distribución y reasignación automática de recursos basado en etiquetado jerárquico. El capítulo describe su motivación y principios de diseño, las tres fases principales, la implementación y una evaluación sobre topologías aleatorias que muestra baja complejidad (crecimiento apro-

ximadamente lineal), tiempos de convergencia reducidos y menor señalización en nodos con recursos limitados. DEN2NE contribuye directamente a los objetivos de la Tesis al ofrecer un mecanismo de control y encaminamiento en entornos densos y heterogéneos, y una base para la gestión/planificación de recursos en multitud de casos de uso.

El sexto capítulo presenta una propuesta novedosa que integra técnicas de aprendizaje automático (ML) y aprendizaje profundo (DL) para la predicción temprana de fallos en redes inteligentes de distribución eléctrica (SG), apoyada en algoritmos de reconfiguración basados en el etiquetado jerárquico desarrollado en DEN2NE. El capítulo revisa las principales fuentes y conjuntos de datos públicos sobre SG, describe la generación y curación de un nuevo conjunto de datos sobre la ciudad de Funchal (Madeira, Portugal) y detalla cómo se incorporaron perfiles de generación fotovoltaica simulados. A partir de esta base, se entrena y optimizan diversos modelos ML/DL orientados a operar en entornos densos y heterogéneos, y se dedica un apartado específico a la validación experimental de los modelos, evaluando su capacidad de predicción y su utilidad para guiar reconfiguraciones proactivas de la red.

El séptimo capítulo presenta BLOSTE, un algoritmo adaptativo para el encaminamiento de energía en redes de distribución malladas basado en superposiciones lógicas iniciadas desde uno o varios nodos raíz. El capítulo describe su motivación y principios de diseño, los mecanismos de etiquetado y optimización iterativa, la implementación y una evaluación exhaustiva sobre topologías IEEE de referencia y escenarios de gran escala. Los resultados muestran la capacidad de BLOSTE para mantener un balance global de potencia bajo condiciones variables, su escalabilidad (tiempos de convergencia en el orden de milisegundos y crecimiento computacional cercano a lineal), y su robustez frente a topologías y criterios de decisión heterogéneos. BLOSTE contribuye directamente a los objetivos de la Tesis al proporcionar un mecanismo distribuido, flexible y eficiente para la gestión energética en redes malladas, sirviendo como base para el diseño de arquitecturas resilientes en entornos de red complejos.

El octavo capítulo introduce una arquitectura completamente contenerizada para el soporte de inteligencia en el continuo *cloud/edge*, orientada a entornos IIoT. El capítulo describe la motivación y principios de diseño, los principales módulos funcionales (telemetría mediante InfluxDB, inferencia en tiempo real con BentoML y control adaptativo mediante SDN basado en Ryu), así como su integración en un marco reproducible de experimentación. La evaluación realizada demuestra despliegues rápidos (menos de cinco minutos), una utilización estable de recursos y tiempos de reconfiguración de red en el orden de subsegundos. Esta arquitectura contribuye directamente a los objetivos de la Tesis al proporcionar un mecanismo escalable y autónomo de monitorización, decisión y reconfiguración, que constituye una base sólida para el desarrollo de soluciones industriales más flexibles y resilientes.

Por último, el capítulo noveno recoge las conclusiones principales de la Tesis, así como un bloque que describe futuras líneas de investigación en las que se podrá seguir indagando.

## 1.5. Contribuciones

El trabajo desarrollado en esta Tesis Doctoral ha generado una contribución notable a la comunidad científica, tanto en términos de generación de conocimiento como en su difusión y transferencia. En concreto, se han publicado cuatro artículos en revistas indexadas en JCR, incluyendo una publicación en una revista de alto impacto Q1 y tres en Q2, y otra dos más que están en revisión (Q2). Además, se han presentado cuatro trabajos en conferencias internacionales organizadas por el IEEE, lo que demuestra la solidez y el interés internacional del trabajo. Como parte del compromiso con la divulgación científica, los avances de esta Tesis también han sido compartidos en eventos como las X Jornadas de Jóvenes Investigadores de la Universidad de Alcalá y la 5th EUGLOH Annual Student Research Conference 2024. Cabe destacar, como elemento diferenciador, el reconocimiento al potencial de transferencia tecnológica de los resultados de esta investigación, materializado en la obtención del Primer Premio en el Concurso de Ideas para la Creación de Empresas de Base Tecnológica de la UAH en 2024. Este premio pone de relieve la capacidad de esta Tesis no solo para generar conocimiento científico de calidad, sino también para transformarlo en soluciones con impacto real en la sociedad, alineadas con los principios de innovación y transferencia del sistema universitario.

Artículos de revista indexadas de alto impacto :

1. Carrascal, D., Rojas, E., Arco, J. M., Lopez-Pajares, D., Alvarez-Horcajo, J., & Carral, J. A. (2023). A comprehensive survey of in-band control in sdn: Challenges and opportunities. *Electronics*, 12(6), 1265. (JCR Q2)
2. Rojas, E., Carrascal, D., Lopez-Pajares, D., Alvarez-Horcajo, J., Carral, J. A., Arco, J. M., & Martinez-Yelmo, I. (2024). A Survey on AI-Empowered Softwarized Industrial IoT Networks. *Electronics*, 13(10), 1979. (JCR Q2)
3. Carrascal, D., Rojas, E., Carral, J. A., Martinez-Yelmo, I., & Alvarez-Horcajo, J. (2024). Topology-aware scalable resource management in multi-hop dense networks. *Heliyon*, 10(18). (JCR Q1)
4. Carrascal, D., Bartolomé, P., Rojas, E., Lopez-Pajares, D., Manso, N., & Diaz-Fuentes, J. (2024). Fault Prediction and Reconfiguration Optimization in Smart Grids: AI-Driven Approach. *Future Internet*, 16(11), 428. (JCR Q2)
5. Carrascal, D., Santos, C., Rojas, E., Arco, J. M., Lopez-Pajares, D. & Rodriguez-Sanchez F. J. (2025). Dynamic Energy Routing Using Tree-Based Topologies with Fast Convergence applied to Meshed Microgrids. *IEEE Access* (under review). (JCR Q2)
6. Carrascal, D., Diaz-Fuentes, J., Manso, N., Lopez-Pajares, D., Rojas, E., Savi, M. & Arco, J. M. (2025). Softwarized Edge Intelligence for Advanced IIoT Ecosystems: A Data-Driven Architecture Across the Cloud/Edge Continuum. *Applied Sciences* (under review). (JCR Q2)

Conferencias internacionales:

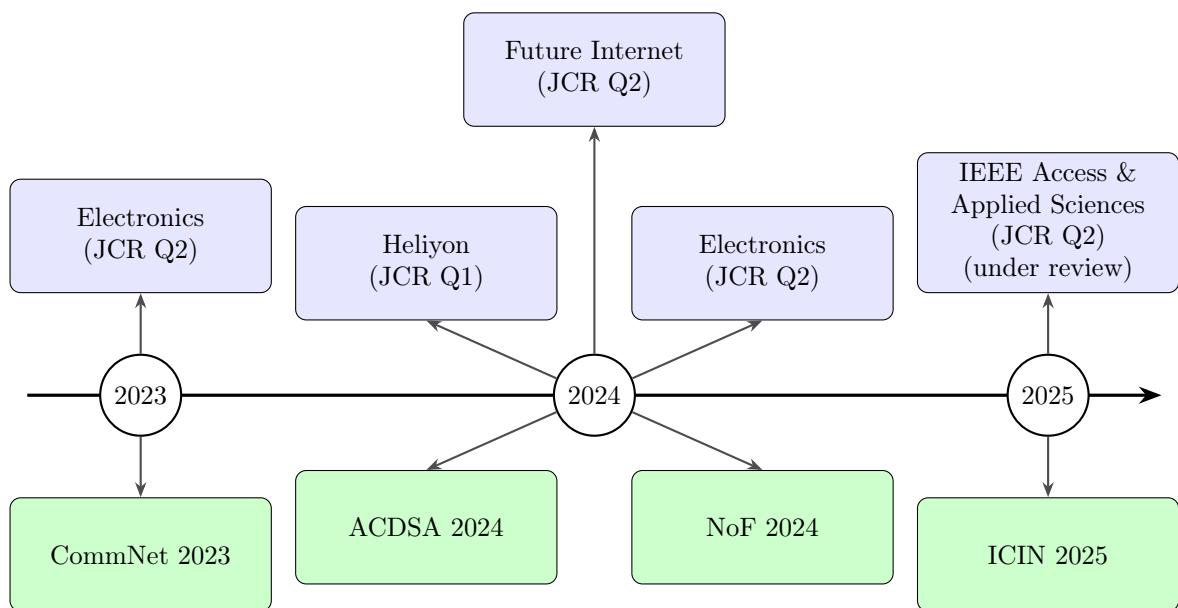
1. Carrascal, D., Rojas, E., Lopez-Pajares, D., Manso, N., & Gutierrez, E. (2023, December). A scalable SDN in-band control protocol for IoT networks in 6G environments. In 2023 6th International Conference on Advanced Communication Technologies and Networking (CommNet) (pp. 1-7). IEEE.
2. Rojas, E., Carrascal, D., Lopez-Pajares, D., Manso, N., & Arco, J. M. (2024, February). Towards ai-enabled cloud continuum for iiot: Challenges and opportunities. In 2024 International Conference on Artificial Intelligence, Computer, Data Sciences and Applications (ACDSA) (pp. 1-6). IEEE.
3. Comeron, R., Rojas, E., Carrascal, D., Alvarez-Horcajo, J., & Arco, J. M. (2024, October). Multi-hop collaborative edge computing involving constrained IoT devices at the far edge. In 2024 15th International Conference on Network of the Future (NoF) (pp. 22-24). IEEE.
4. Carrascal, D., Rojas, E., Lopez-Pajares, D., Manso, N., Alvarez-Horcajo, J., & Martinez-Yelmo, I. (2025, March). Softwarized Data-Driven Architecture for Edge Computing IIoT Environments: A Proof of Concept. In 2025 28th Conference on Innovation in Clouds, Internet and Networks (ICIN) (pp. 64-68). IEEE.

Actividades de divulgación:

1. Ponentes a las X Jornadas de Jóvenes Investigadores de la UAH, presentando el trabajo titulado “DEN2NE: origen, presente, ¿y futuro?”.
2. Participación en la 5th EUGLOH Annual Student Research Conference 2024, con el trabajo titulado “Advancements in Enabling Technologies for Programmable and Software-Defined Networks: Paving the Way to 6G”.

Premios:

1. Primer Premio - Concurso de ideas para la creación de empresas de base tecnológica - UAH (Programa propio de investigación y transferencia de la UAH 2024).



**Figura 1.2:** Línea de tiempo de contribuciones científicas (2023–2025): publicaciones en revistas y conferencias internacionales.

# Capítulo 2

## Estado del Arte

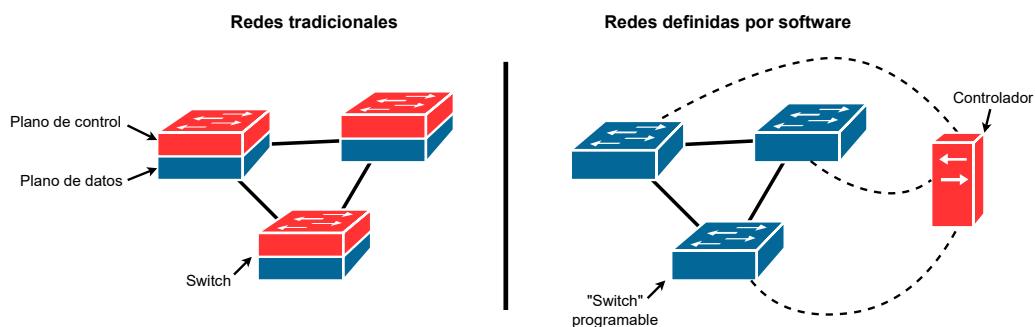
Este capítulo tiene como objetivo principal revisar los conceptos clave y el estado del arte que constituyen la base de esta Tesis. Para ello, se ha estructurado el capítulo en tres grandes bloques. En primer lugar, se explorarán las redes programables y softwarizadas partiendo de las redes SDN, siguiendo con los paradigmas de control *out-of-band/in-band*, y analizando en profundidad las propuestas más relevantes de *in-band*. En el segundo bloque, se estudiará los servicios y las tecnologías habilitantes en redes softwarizadas y heterogéneas en el contexto de abstracción del MCC (del inglés, *Management–Control Continuum*), donde se analizarán desde servicios básicos (arranque y provisión de canales de control) a servicios avanzados (gestión y planeamiento de recursos, optimización y reconfiguración proactiva de la red). Por último, se analizarán diversos casos de uso relevantes en entornos densos y heterogéneos, concretamente en los dominios de las SG y las redes de sensores IIoT.

### 2.1. Las redes SDN

En este primer bloque se revisan las redes SDN, que son la base de las redes programables y softwarizadas. Se explorarán sus características, ventajas y desventajas, así como sus paradigmas de modos de control, según se indicó anteriormente. Además, se analizarán los protocolos y así como los aspectos clave de las vertientes de trabajo del modo de control *in-band*, y cómo podemos explorar dicho modo para favorecer la flexibilidad y control en redes densas y heterogéneas.

Las redes definidas por software (SDN) representan un nuevo paradigma que rompe con las arquitecturas tradicionales de red. Antes de que apareciera el concepto de SDN, como se puede apreciar en la Figura 2.1, las redes tradicionales solían tener un plano de control unificado en los propios dispositivos, llamado generalmente *Control plane*, en el que se definía la lógica que dictaba cómo se debía llevar a cabo el forwarding de los paquetes, y un plano de datos, conocido como *Data plane*, que se implementaba definiendo su data-path, compuesto por varios bloques de procesamiento para reenviar los paquetes. Ambos planos estarían unificados en un sentido lógico, en un mismo dispositivo. Sin embargo, con la aparición del paradigma de las redes SDN, como se muestra en la Figura 2.1, los

nodos tradicionales de la red verían cómo su plano de control sería delegado a una entidad externa llamada controlador, preservando su capacidad para manejar los paquetes. En contraste con las arquitecturas tradicionales de la red, donde había que ir configurando equipo a equipo, y donde cada uno de ellos iba a desempeñar una función de red, en las redes SDN, el controlador permite configurar y supervisar de manera inteligente el comportamiento de la red a través de aplicaciones software, facilitando una programación flexible y dinámica del entorno de red. Por lo que, aunque se sigan llamando “switches” o nodos SDN, estos se comportarán según las reglas que le instale el controlador, pudiendo gestionar paquetes como un switch, un router, un firewall, etc.



**Figura 2.1:** Paradigma en las redes SDN.

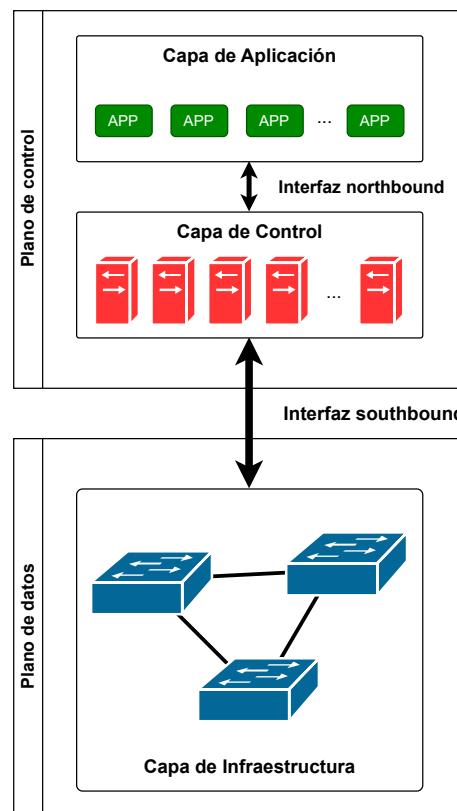
La centralización de la gestión simplifica notablemente las tareas del administrador, al proporcionar una visión global del estado de la red y un punto único desde el cual definir su funcionamiento. A través del controlador, las complejas instrucciones de bajo nivel requeridas por los dispositivos de red tradicionales, como switches y routers, las cuales podían variar en función del fabricante, se abstraen mediante interfaces con sintaxis intuitiva, reduciendo la complejidad operativa.

Estas capacidades dotan a la red de una gran agilidad y capacidad de adaptación ante cambios o nuevas necesidades, pudiendo conmutar entre distintos perfiles de funcionamiento de forma automática. El simple despliegue de una nueva aplicación sobre el controlador permite modificar de forma coherente el comportamiento de toda la infraestructura, disminuyendo así los costes asociados al mantenimiento, la operación y el despliegue. Además, SDN promueve activamente el uso de soluciones abiertas tanto a nivel de software como de hardware, fomentando ecosistemas interoperables, reduciendo la dependencia de tecnologías propietarias y eliminando barreras de entrada para nuevos actores en el sector.

### 2.1.1. Arquitectura lógica de las redes SDN

La arquitectura lógica de las redes SDN se puede dividir en dos planos, el plano de control y el plano de datos, y además, en tres capas: capa de aplicación, capa de control y capa de infraestructura. En la Figura 2.2 se muestra la arquitectura lógica de las redes SDN, así como sus interfaces principales de comunicación que más adelante se explicarán.

El plano de control, se estructura internamente en dos capas funcionales: la capa de control y la capa de aplicación. Estas se comunican mediante la interfaz norte (*northbound interface*), que permite a las aplicaciones definir políticas de alto nivel que serán interpretadas y gestionadas por el controlador. Estas capas a menudo se pueden encontrar corriendo en la misma máquina, donde conviven el controlador y las aplicaciones que interactúan con él. Sin embargo, también se puede tener un enfoque distribuido, donde el controlador está en una máquina, y las aplicaciones en otra, haciendo uso de la interfaz *northbound*. Por su parte, el plano de datos está conformado por la capa de infraestructura, que engloba los dispositivos físicos de red, principalmente switches SDN, responsables del reenvío de paquetes. La interacción entre el plano de control y el plano de datos se realiza a través de la interfaz sur (*southbound interface*), cuya función es traducir las decisiones del plano de control en instrucciones ejecutables por los dispositivos de red. En este contexto, el controlador actúa como una pieza clave del sistema, asumiendo responsabilidades esenciales como la instalación de reglas de encaminamiento, la monitorización continua del estado de la red y la recopilación de métricas operativas, las cuales serán aprovechadas por todas las aplicaciones que se ejecuten sobre el controlador.

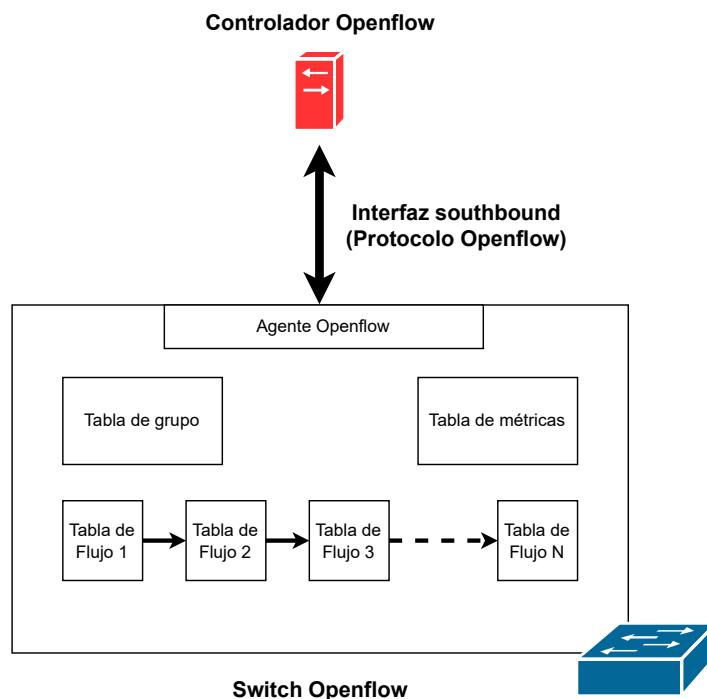


**Figura 2.2:** Arquitectura lógica de las redes SDN.

El plano de datos, por el contrario, no posee lógica de control propia, limitándose a ejecutar las reglas recibidas, como por ejemplo, hacer un reenvío, o descartar paquetes según las reglas establecidas, además de enviar estadísticas de tráfico al controlador. Esta separación de funciones establece una división clara entre la inteligencia de la red, localizada en el plano de control, y su ejecución, delegada como se ha explicado, al plano de datos. De esta manera, se rompe con el modelo tradicional en el que ambos planos coexistían en un mismo dispositivo de red. Este enfoque modular no solo mejora la escalabilidad y la flexibilidad del sistema, sino que también reduce significativamente los costes de despliegue (CapEx) y operación (OpEx), al concentrar los recursos de cómputo en un nodo centralizado, y simplificar el hardware requerido en los dispositivos de reenvío.

Según se ha visto en la Figura 2.2, la arquitectura SDN se apoya en una estructura jerárquica formada por tres capas principales: aplicación, control e infraestructura. La capa de aplicación representa el nivel de mayor abstracción dentro del ecosistema SDN. Esta capa integra un conjunto de aplicaciones que, apoyándose en los servicios ofrecidos por la capa de control, permiten definir políticas de gestión, QoS, optimizar el rendimiento de la red y adaptarla dinámicamente a diferentes contextos operativos. Un ejemplo típico de uso es la utilización los servicios de descubrimiento topológico proporcionados por la capa de control, que permiten a las aplicaciones calcular rutas óptimas entre dispositivos de red. Estas aplicaciones suelen desarrollarse empleando lenguajes de alto nivel como Python, Go o C++, con el objetivo de facilitar su portabilidad entre plataformas y maximizar la reutilización del código. No obstante, en la práctica, la existencia de APIs y entornos de desarrollo específicos para cada plataforma de control, como ONOS, OpenDaylight, Ryu o el nuevo controlador del ecosistema SDN, TeraflowSDN, introduce ciertos desafíos en la interoperabilidad y portabilidad del software entre distintas implementaciones. En este sentido, uno de los principales retos actuales de SDN sigue siendo la estandarización de interfaces *northbound* que permitan una integración más fluida y flexible entre aplicaciones y controladores heterogéneos. Descendiendo, la siguiente capa es la capa de control, la cual constituye el núcleo funcional del paradigma SDN, albergando la inteligencia centralizada de la red. Actúa como intermediario entre las aplicaciones de alto nivel y los dispositivos físicos de la capa de infraestructura, orquestando tareas críticas como el encaminamiento de flujos, la detección y resolución de fallos, la supervisión continua del estado de la red y la gestión de políticas de seguridad y QoS. Su papel como *middleware* se traduce en la capacidad de transformar políticas abstractas generadas en la capa de aplicación en instrucciones simples y concretas que pueden ser entendidas por los nodos SDN. Esta capacidad de traducir y escalar la lógica de red permite que un único controlador gobierne cientos o miles de switches de forma eficiente, garantizando escalabilidad y consistencia en entornos distribuidos. Por último, la capa de infraestructura, por su parte, está compuesta por los elementos físicos de la red, fundamentalmente nodos SDN, que ejecutan las decisiones tomadas por el plano de control. Estos dispositivos, carentes de lógica propia, cuentan con un agente SDN encargado de comunicarse con el controlador a través de la interfaz sur (*southbound interface*), como por ejemplo, OpenFlow o P4Runtime. Su funcionalidad se reduce al reenvío y descarte de paquetes o la recolección de estadísticas, lo que permite simplificar su diseño y reducir sus requisitos hardware.

En cuanto a las interfaces, hay dos, como se ha mencionado anteriormente, la interfaz *northbound* y la interfaz *southbound*. La interfaz *northbound* constituye el canal de comunicación entre la capa de control y la capa de aplicación. Su principal función es ofrecer un punto de acceso lógico al administrador de red, permitiéndole supervisar, configurar y gestionar el comportamiento de la red sin necesidad de interactuar directamente con los mecanismos de bajo nivel que gobiernan los dispositivos físicos. A través de esta interfaz, las aplicaciones pueden programar políticas o solicitudes que serán traducidas por el controlador en instrucciones comprensibles para los elementos de la infraestructura. No obstante, a diferencia de la interfaz *southbound*, la interfaz *northbound* carece de una estandarización formal. En consecuencia, la naturaleza y funcionalidad de esta interfaz varían considerablemente en función del controlador SDN empleado, cada uno de los cuales suele ofrecer su propia API con diferentes modelos de datos, protocolos y lenguajes de interacción. La interfaz *southbound* constituye el enlace entre la capa de control y la capa de infraestructura dentro de una arquitectura SDN. A diferencia de la interfaz *northbound*, esta sí cuenta con protocolos estandarizados ampliamente adoptados, que permiten la interoperabilidad entre los controladores y los dispositivos de red. Históricamente, el protocolo más representativo ha sido Openflow [20]. En la implementación del protocolo Openflow según se indica en la Figura 2.3, el concepto central es el de flujo (del inglés, *flow*), entendido como un conjunto de paquetes que cumplen determinadas condiciones definidas por el controlador. Estas condiciones se almacenan en las denominadas tablas de flujo (del inglés, *flow tables*), y suelen hacer referencia a valores específicos de campos en la cabecera del paquete o al puerto de entrada por el que se ha recibido el paquete.



**Figura 2.3:** Arquitectura básica de switch OpenFlow.

Cuando un paquete llega a un switch Openflow, empieza a atravesar de forma iterativa las tablas de flujo y cuando este coincide con los criterios de una regla definida en una tabla, se produce una coincidencia (del inglés, *match*), lo que activa un conjunto de instrucciones asociadas a dicha regla. Estas instrucciones pueden incluir el conteo de paquetes, la aplicación de acciones concretas (como reenviar o descartar el paquete), o bien su reenvío hacia otra tabla para un procesamiento adicional. En caso de no darse una coincidencia, se encapsula y se manda al controlador para que este decida cómo manejarlo. Así, mediante la instalación de estas reglas por parte del controlador SDN, se determina el comportamiento de reenvío del switch. La comunicación entre el controlador y los dispositivos se realiza a través de un canal estructurado y seguro, que admite mensajes del controlador al switch, mensajes asíncronos generados por los dispositivos, y mensajes simétricos intercambiables por ambas partes, permitiendo una gestión eficiente y dinámica del estado de la red.

No obstante, las limitaciones de flexibilidad, extensibilidad y adaptación a nuevas arquitecturas han motivado el surgimiento de alternativas a Openflow. Un ejemplo destacado es el lenguaje Programming Protocol-Independent Packet Processors (P4) [21], diseñado específicamente para superar las restricciones de OpenFlow. Una de las mayores restricciones que tiene OpenFlow es la especificación de forma explícita de los campos de cabecera sobre los que opera. Estos campos de cabecera han pasado de 12 a 41 campos de cabeceras entre sus versiones 1.0 y 1.5 como se puede ver en la Tabla 2.1. Esta evolución ha incrementado la complejidad del protocolo sin proporcionar la flexibilidad necesaria para incorporar nuevas cabeceras o funcionalidades emergentes.

Versión	Fecha	Campos de cabecera
OF 1.0	Dic. 2009	12 campos (Ethernet, TCP/IPv4)
OF 1.1	Feb. 2011	15 campos (MPLS, metadatos entre tablas)
OF 1.2	Dic. 2011	36 campos (ARP, ICMP, IPv6, etc.)
OF 1.3	Jun. 2012	40 campos
OF 1.4	Oct. 2013	41 campos
OF 1.5	Mar. 2015	44 campos

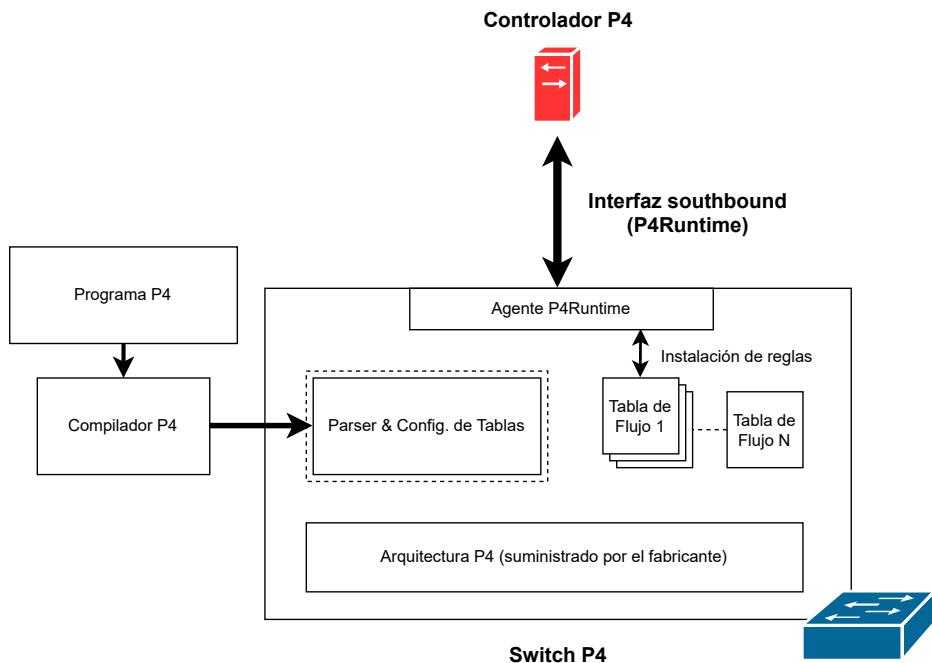
**Tabla 2.1:** Evolución de versiones del protocolo Openflow y el número de campos de cabecera soportados.

En respuesta a ello, P4 nació con tres objetivos principales:

- Permitir la reconfiguración del dispositivo en caliente, es decir, cambiar el comportamiento de los switches una vez desplegados.
- Ofrecer independencia de protocolo, desvinculando el procesamiento de paquetes de protocolos específicos que tengan que estar estandarizados para poder ser gestionados.
- Proporcionar independencia del hardware, permitiendo que las funcionalidades de procesamiento se definan sin depender de los detalles del dispositivo subyacente.

Si bien es cierto que la iniciativa de P4 nació con este objetivo en mente (*open-hardware*), la realidad es que, en la actualidad, se ha visto como cada fabricante ha implementado equipos que si cumplen con algunas de las arquitecturas de P4, pero que cada uno te ofrece unas primitivas de programación diferentes, haciendo que un programa P4 que corre en un dispositivo de un fabricante no sea totalmente compatible en otro [22].

En comparación con Openflow, si nos fijamos en la Figura 2.4, podemos apreciar que empleando P4 se puede definir cómo el switch va a manejar los paquetes, como los va a procesar y parsear, manteniendo la lógica de las tablas de flujo que teníamos en Openflow, pero ganando en flexibilidad dado que se pueden definir el propio datapath del dispositivo sin depender de un conjunto estandarizado de campos de cabecera. Esto incluso permite que P4 pueda ser implementado en dispositivos de baja capacidad [23], al poder ajustar el datapath a la mínima expresión necesaria para cumplir con las necesidades de la red. Al igual que en Openflow se tenía el protocolo de comunicación para la interfaz *southbound*, P4 también tiene su propia interfaz de comunicación, llamada P4Runtime [24], que permite a los controladores gestionar y programar dispositivos P4 de forma dinámica.



**Figura 2.4:** Arquitectura básica de switch P4.

A diferencia de Openflow, que define un conjunto cerrado de operaciones y estructuras, P4 y su interfaz P4Runtime introducen la posibilidad de reconfigurar dinámicamente el comportamiento del plano de datos mediante descripciones personalizadas del procesamiento de paquetes. Esto se logra mediante una arquitectura basada en google Remote

Procedure Call (gRPC), que ofrece cinco tipos de operaciones principales (Write, Read, Set/GetForwardingPipelineConfig y StreamChannel) para gestionar tanto el estado como la lógica interna de los switches programables. De esta forma, P4 se presenta como una propuesta de evolución de Openflow, orientada a lograr una programabilidad del plano de datos más flexible y escalable, haciéndolo ideal para *testing* y pruebas de concepto de nuevas soluciones de red.

Paralelamente, ha ido creciendo otra vía complementaria orientada a la gestión y configuración unificada de dispositivos de red llamada OpenConfig. Esta iniciativa, impulsada mayormente por un consorcio de operadores y fabricantes, propone un conjunto de modelos de datos basados en YANG que permiten describir de forma estandarizada y agnóstica el estado operativo y la configuración de dispositivos de red. A diferencia de OpenFlow o P4Runtime, que se centran en el comportamiento del plano de reenvío, OpenConfig aborda la gestión, configuración, el monitoreo y la automatización de tareas de red a través de protocolos como gNMI o NETCONF. Esto convierte a OpenConfig una herramienta clave para aquellas empresas que buscan una gestión softwarizada y programable de sus infraestructuras ya existentes, dado que permite la integración de dispositivos heterogéneos de diferentes fabricantes bajo un modelo común de gestión. Si bien es cierto que OpenConfig no permite definir explícitamente el plano de datos, a diferencia de Openflow o P4, donde se considera que todos los switches o nodos SDN equivalen a un único dispositivo lógico gestionado de forma centralizada, OpenConfig propone un enfoque diferente. Esta iniciativa busca establecer un conjunto común de modelos de datos y configuración, independientes del fabricante, para gestionar redes heterogéneas. A diferencia del enfoque SDN tradicional, en el que los dispositivos se integran como un único plano de control y datos, los dispositivos gestionados mediante OpenConfig siguen operando como entidades independientes. Ambos enfoques persiguen una mayor transparencia y facilidad de gestión de la red, pero difieren en su grado de abstracción y centralización: mientras SDN trata la red como un todo unificado, OpenConfig mantiene la identidad individual de cada dispositivo, facilitando la interoperabilidad en entornos mixtos. Sin embargo, los últimos controladores SDN como TeraflowSDN [25], han comenzado a integrar OpenConfig como una de sus interfaces *southbound* (además de P4), incluso llegando a no implementar Openflow, lo que sugiere una tendencia hacia un nuevo ecosistema de redes SDN que combina la flexibilidad de la programación del plano de datos con la estandarización y la gestión eficiente de dispositivos heterogéneos.

En este sentido, la evolución de la interfaz *southbound* no debe entenderse en términos de sustitución de unos protocolos por otros, sino como una diversificación funcional que permite combinar capacidades de reenvío programable, comunicación eficiente y gestión estandarizada según las necesidades específicas de cada red.

### 2.1.2. Arquitectura física de las redes SDN

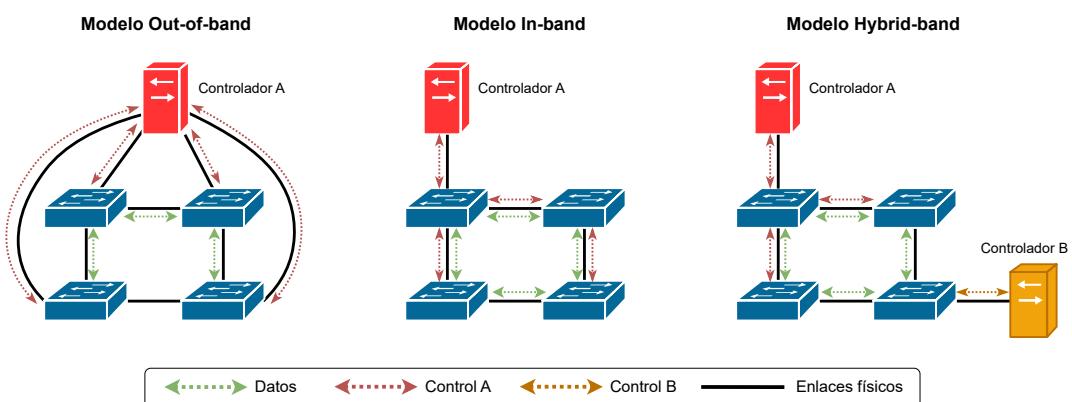
Una vez que se ha revisado la arquitectura lógica de las redes SDN, es importante entender cómo se implementa físicamente esta arquitectura, es decir, cómo se conectan los diferentes componentes que se vienen explicando en la sección anterior.

En una red SDN, según se indicó en la Figura 2.2, se compone de un elemento central, el controlador, y un conjunto de switches o nodos SDN distribuidos en la capa de infraestructura los cuales son gestionados por el controlador. Sin embargo, también es posible la implementación de múltiples controladores en una misma red SDN, lo cual aporta funcionalidades adicionales a la red, como mecanismos de respaldo y tolerancia a fallos, que incrementan su fiabilidad, y por ende, la resiliencia de la red. Por ello, se pueden clasificar las conexiones físicas en las redes SDN en dos bloques:

- Las conexiones entre los switches de la capa de infraestructura.
- Las conexiones entre el controlador y los switches de la capa de infraestructura.

Las primeras constituyen la topología física de la red, cuya estructura depende del entorno en el que se despliegue y de los objetivos funcionales de la red. Por ejemplo, en redes SDN diseñadas para centros de datos, es común adoptar una arquitectura jerárquica y regular, ya que esta facilita la escalabilidad y permite absorber incrementos en la demanda de tráfico de forma eficiente [26]. En cambio, en entornos de redes de sensores SDN, es frecuente emplear topologías en malla parcial (tendiendo hacia a una malla completa) [10], que permiten una mayor resiliencia frente a fallos y una reducción en la latencia gracias a la existencia de múltiples caminos entre nodos de la topología.

En cuanto a las conexiones entre el controlador y los switches de la capa de infraestructura, estas se pueden clasificar en principalmente en dos categorías, si bien es cierto que se puede encontrar una tercera categoría que combina ambas. Observando la Figura 2.5, se pueden distinguir dos paradigmas de control: el modo de control *in-band* y el modo de control *out-of-band*, y por último, el modo de control *hybrid-band*, el cual es una combinación de los dos anteriores.



**Figura 2.5:** Paradigmas de control en las redes SDN.

Al desplegar el canal de control en una red SDN, es posible optar por un enfoque *out-of-band* o *in-band*, como se ilustra en la Figura 2.5. En el primer caso, denominado

*out-of-band*, cada nodo SDN dispone de un enlace físico dedicado que lo conecta directamente con el controlador. De este modo, la información de control se transmite a través de una red independiente, exclusiva para dicho propósito, lo cual incrementa la seguridad y el aislamiento del canal, aunque implica un mayor coste de infraestructura al requerir al menos un enlace adicional por nodo. Por el contrario, en el enfoque *in-band*, solo algunos nodos SDN mantienen un enlace directo con el controlador, mientras que el resto accede a él a través de la propia red de datos, reutilizando los enlaces existentes para transportar la información de control. En este caso, los mensajes de control comparten la infraestructura del plano de datos, lo que puede comprometer su seguridad e integridad, al estar más expuestos a posibles interferencias o interceptaciones.

Finalmente, el enfoque *hybrid-band* contempla una solución intermedia, en la que coexisten enlaces dedicados y compartidos para la comunicación con el plano de control [27], como se muestra en la Figura 2.5. Este modelo busca equilibrar los costes operativos con los requisitos de fiabilidad y seguridad.

Cada uno de estos esquemas de despliegue presenta ventajas e inconvenientes [27], y la elección entre uno u otro depende fundamentalmente del escenario de red y del caso de uso considerado [28, 29]. No existe un paradigma mejor que otro, sino, que cada enfoque ofrece características particulares que pueden resultar más o menos adecuadas según los requisitos del entorno. Por ejemplo, el modelo *out-of-band* requiere un enlace físico adicional dedicado a la comunicación entre el controlador y cada nodo SDN, lo que incrementa notablemente los costes de despliegue y mantenimiento. No obstante, esta separación garantiza un mayor aislamiento del canal de control, lo que mejora sustancialmente la seguridad de las comunicaciones. En contraposición, el modelo *in-band* reutiliza los enlaces existentes del plano de datos para transmitir la información de control, lo que reduce significativamente el coste de infraestructura. Sin embargo, esta economía viene a expensas de una menor seguridad, ya que los mensajes de control comparten canal con el tráfico de red, quedando expuestos a posibles interferencias o ataques.

Además, uno de los principales retos del enfoque *in-band* radica en la configuración inicial, es decir, el nodo debe conocer de antemano la ruta hacia el controlador a través de la red de datos. En contraste, el modelo *out-of-band* facilita esta tarea, al disponer de una interfaz exclusiva para dicho propósito. Por ello, en *in-band*, esta información tiene que proporcionarse mediante protocolos específicos que permiten a cada nodo identificar la interfaz adecuada para reenviar los paquetes de control. Estos protocolos son de especial interés dado que no existe una solución estandarizada en la academia. Debido a lo cual, se quiere explorar en mayor medida qué opciones existen y qué metodologías se han empleado, dado que estas soluciones son fácilmente extrapolables a otros tipos de redes densas y heterogéneas que emplean entornos softwarizados con una tipología de control *in-band*. Así, por ejemplo, en entornos como el IoT, donde los dispositivos suelen disponer de una única interfaz de comunicaciones y cuentan con recursos energéticos limitados, el modelo *in-band* se presenta como una alternativa óptima, al evitar la necesidad de enlaces adicionales que aumentarían el consumo energético y reducirían la vida útil del sensor.

La Tabla 2.2 resume comparativamente las principales características de estos modelos. En ella se observa cómo el paradigma *out-of-band* destaca por su simplicidad de configuración y seguridad, mientras que el *in-band* sobresale en términos de escalabilidad y costes.

Propiedad	Control <i>out-of-band</i>	Control <i>in-band</i>
Configuración del dispositivo SDN	Sencilla	Compleja
Seguridad del canal de control	Segura, canal aislado	Riesgosa, canal compartido
Costes de mantenimiento y despliegue	Elevados	Reducidos
Escalabilidad	Limitada	Buena
Resiliencia	Costosa	Recuperación rápida

**Tabla 2.2:** Características del control *in-band* y *out-of-band*.

### 2.1.3. Propuestas de despliegue con control *in-band*

La tendencia actual indica que el control *in-band* está ganando protagonismo en los despliegues de redes SDN [30], especialmente en redes grandes y densas, donde el coste de utilizar un modelo *out-of-band* puede resultar prohibitivo. Además, el control *in-band* habilita el desarrollo de una amplia variedad de nuevas aplicaciones, sobretodo en entornos SDN híbridos o con restricciones de recursos [31, 32], donde el despliegue de enlaces dedicados de control puede ser complejo o incluso inviable. Entre los casos de uso más representativos que se benefician del control *in-band* se encuentran las redes 5G [33] y las Non-Terrestrial Networks (NTNs) [34], así como diversos escenarios del ámbito IoT, como redes submarinas [35], entornos orientados a la eficiencia energética [36] o sistemas con recursos limitados [37]. A pesar de sus numerosas ventajas, los esfuerzos dirigidos al diseño de protocolos comunes e integrales para el control *in-band* han sido escasos. Una solución efectiva debería considerar la compatibilidad con plataformas ampliamente utilizadas, tanto en los controladores como en los dispositivos SDN, a fin de garantizar una integración completa en los despliegues actuales. En este contexto, diferentes propuestas han explorado mecanismos para habilitar o mejorar el control *in-band*, con el objetivo de facilitar su adopción en entornos reales y responder a los retos que plantea, este paradigma. A continuación, se presentan algunos de los trabajos más representativos en esta línea.

En este contexto, los trabajos existentes sobre control *in-band* pueden clasificarse en función de tres aspectos clave que dicho paradigma debería proporcionar: encaminamiento automático, recuperación rápida ante fallos y arranque autónomo de la red. Además, se ha identificado un cuarto aspecto transversal, relacionado con entornos de control distribuido que también merecen la pena revisar. En primer lugar, la necesidad de contar con un mecanismo de encaminamiento automático en el control *in-band* resulta evidente. Mientras que el modelo *out-of-band* suele implementarse mediante enlaces directos entre los nodos SDN y el controlador, el control *in-band* requiere calcular rutas entre los dispositivos de red y uno o varios controladores, tanto en un sentido ascendente, como descendente. Este mecanismo de encaminamiento es esencial para permitir la comunicación de control a

través del plano de datos y suele condicionarse al tipo de despliegue o a las capacidades del entorno. En segundo lugar, una de las principales ventajas del control *in-band* es su capacidad de recuperación rápida ante fallos. En caso de que un enlace o switch sufra una avería, es posible restaurar la comunicación de control simplemente seleccionando una ruta alternativa dentro del plano de datos. No obstante, para que este proceso resulte eficaz, los mecanismos de restauración o protección de rutas deben estar bien definidos y coordinados con la lógica de encaminamiento previamente establecida. El tercer aspecto fundamental es el denominado arranque autónomo de la red (del inglés, *network bootstrapping*), que hace referencia a la capacidad del sistema para configurar automáticamente los parámetros necesarios antes del inicio de su operación. Estos parámetros incluyen, entre otros, la información de conectividad entre nodos SDN y controladores. Si bien este proceso también es deseable en entornos *out-of-band*, en el caso de control *in-band* se convierte en un requisito crítico, ya que las rutas de control pueden variar en función del despliegue y del protocolo de encaminamiento utilizado. Por último, en entornos con control distribuido, donde existen varios controladores implicados en la toma de decisiones, es necesario incorporar mecanismos de coordinación que permitan compartir rutas, realizar recuperación ante fallos de manera conjunta y gestionar el arranque de la red de forma sincronizada. Además, el control *in-band* puede ofrecer un canal adicional para la comunicación interna entre controladores, lo cual añade una dimensión interesante a su uso en arquitecturas distribuidas.

En primer lugar, varios autores han sentado las bases de un marco genérico de control *in-band* y han explorado mecanismos de encaminamiento automático con capacidad de recuperación ante fallos simples de la red. Sharma *et al.* [38] presentan un prototipo basado en OpenFlow que evalúa la viabilidad del encaminamiento automático en distintos switches, proponiendo métodos de restauración y protección frente a fallos individuales. De manera similar, Goltzmann *et al.* [39] introducen el protocolo ICS, que construye un árbol de expansión etiquetado para garantizar conectividad resiliente con un bajo sobrecoste computacional. Ambos trabajos ponen de manifiesto la factibilidad del control *in-band* en escenarios sencillos con encaminamiento automático, pero no extienden sus propuestas a la gestión simultánea de múltiples fallos ni proporcionan un estándar interoperable. Por el contrario, Khakhalin *et al.* [40], formulán un algoritmo de encaminamiento automático resistente a múltiples fallos mediante la asignación de etiquetas únicas a cada switch, sin embargo, la gestión de estas etiquetas se complica rápidamente a medida que el tamaño de la red aumenta. Este mismo problema de escalabilidad se encuentra en la propuesta de Mohan *et al.* [41], donde definen un esquema encaminamiento con rutas de control disjuntas por nodo para detectar y aislar switches maliciosos; si bien mejora la seguridad, su formulación matemática no escala bien debido al elevado número de variables de decisión. Otro grupo de autores, Raza *et al.* [42] y González *et al.* [43], investigan el uso de protocolo MultiPath TCP (MPTCP) para implementar el encaminamiento automático y aumentar la tolerancia a fallos con rutas de respaldo. No obstante, Raza *et al.* no llegaron a implementar su solución en un entorno real, y González *et al.* requieren un canal *out-of-band* para la fase inicial de arranque, lo que limita la autonomía del despliegue *in-band*. Finalmente, existen propuestas más avanzadas y específicas de encaminamiento automático para entornos heterogéneos. Asadujjaman *et al.* [44] proponen un enfoque reactivo capaz

de recuperar múltiples fallos, aunque su comunicación controlador-switch es *source-routed*, con sobrecarga que penaliza la escalabilidad. López-Pajares *et al.* [45], proponen Amaru, una solución en la cual integran exploración de red y múltiples rutas de protección con bajo coste y tiempos de recuperación reducidos, pero requieren pequeñas modificaciones en los switches para ser plenamente funcionales. Görkemli *et al.* [46] plantean un plano de control dinámico capaz de redistribuir carga entre varios controladores y adaptar el enrutamiento *in-band* según la topología y las necesidades de las aplicaciones; sin embargo, su evaluación solo contempla despliegues virtualizados. Holzmann *et al.* [47] presentan Izzy, un protocolo distribuido basado en árboles de expansión y direcciones temporales que logra tiempos de recuperación inferiores a 100 ms en simulaciones en redes de área amplia (del inglés, Wide Area Network (WAN)), aunque aún no dispone de validación en entornos reales. Por último, Kumazoe *et al.* [48] diseñan un canal *in-band* en entornos P4, embebiendo mensajes de control en paquetes de usuario; aunque innovador, su evaluación inicial revela degradaciones en el reenvío de datos que quedan pendientes de resolver.

Después de analizar los trabajos sobre encaminamiento automático y recuperación ante fallos simples de la red para control *in-band*, se ha visto que comparten una serie de supuestos y bloques funcionales recurrentes. El primero de ellos, es que parten de la premisa de que debe garantizarse la alcanzabilidad del controlador desde cada switch, por lo que incluyen algún mecanismo, ya sea centralizado o distribuido, para calcular rutas de control de forma automática; asumen la existencia de capacidades en el plano de datos (tablas de reenvío, contadores, soporte OpenFlow/P4 o uso de MPTCP) que permiten instalar reglas y supervisar el estado del tráfico; incorporan estrategias de resiliencia (protección o restauración) basadas en rutas de respaldo; emplean técnicas de identificación o etiquetado (*labels*, *locators*, identificadores temporales) para encaminar mensajes de control sin colisiones o bucles y finalmente, la mayoría valida su propuesta mediante simulación, emulación o prototipos parciales en entornos controlados. Estas piezas comunes configuran el esqueleto funcional de las propuestas y explican por qué los retos reiterados (escalabilidad, gestión de fallos múltiples, interoperabilidad y evaluación en entornos reales) aparecen de forma transversal en la literatura.

En segundo lugar, se han identificado varios trabajos que proponen protocolos concretos para el arranque autónomo (*network bootstrapping*) de la conexión de control *in-band*. Sharma *et al.* [49] fueron pioneros en implementar una configuración de control *in-band* haciendo uso del canal Openflow, que dependía en gran parte de un arranque autónomo de la red, el cual, combina el protocolo Dynamic Host Configuration Protocol (DHCP) para obtener direcciones IP y ARP para resolver direcciones MAC, mientras el controlador descubre la topología mediante mensajes sondas de tipo Link Layer Discovery Protocol (LLDP). Su prototipo, basado en el controlador NOX, demostró tiempos de arranque reducidos en topologías variadas, pero su enfoque depende de servicios clásicos de red y no aborda aspectos de interoperabilidad entre controladores heterogéneos. En la misma línea, Tu *et al.* [50] integran un control SDN *in-band* en switches convencionales de un centro de datos (del inglés, Data Center (DC)) y usan servidores de directorio para orquestar el arranque autónomo y el encaminamiento de la red, lo que funciona bien en entornos controlados pero introduce dependencias en componentes adicionales que no siempre están

disponibles en despliegues heterogéneos. Algunas propuestas requieren protocolos o canales externos para completar el arranque autónomo. Su *et al.* [51] utilizan Open vSwitch Database (OVSDB) para configurar el plano de datos y facilitar la gestión de conexiones *in-band* en Open vSwitch (OVS), lo que permite cierto control bien definido pero acopla la solución a capacidades propietarias de OVS. Sakic *et al.* [52] plantean dos mecanismos: uno basado en pre-configuración secuencial y otro apoyado en Rapid Spanning Tree Protocol (RSTP) para construir árboles de expansión de mínimos costes; ambas opciones funcionan pero resultan lentas (convergencia en segundos) y una depende del comportamiento de protocolos tradicionales de capa 2. Wu *et al.* [53] también emplean RSTP (rXstp) para estabilizar la topología previa al establecimiento de la conexión de control, requisito que obliga a intervenciones manuales (prioridades de switches) y limita el uso a escenarios con un único controlador. El tiempo de arranque y la sobrecarga de mensajes son factores clave, en este caso, Sharma *et al.* muestra buenos tiempos en ensayos, mientras que Suo *et al.* [27] propone reglas pre-instaladas para acelerar el arranque autónomo de la red en un DC pero se observa pérdida de rendimiento y cierto coste en *throughput*. Silva-Freitas *et al.* [54] abordan explícitamente la eficiencia de la red: su protocolo reduce el número de mensajes frente a LLDP, y además incorpora autenticación, mejorando seguridad y eficiencia. Li *et al.* [55] proponen un mecanismo de arranque muy rápido (50 switches en  $\approx 2$  s) construyendo un árbol en una pasada, aunque deja la recuperación rápida como trabajo futuro. En contraste, Sakic *et al.* obtienen convergencias lentas en sus opciones, lo que evidencia una tensión entre rapidez, coste y robustez.

En conjunto, la literatura sobre *network bootstrapping* para control SDN *in-band* ofrece un conjunto rico de alternativas y prototipos, pero existen huecos recurrentes: (i) soluciones agnósticas y estandarizables que no dependan de protocolos o extensiones concretas; (ii) procedimientos escalables y de baja latencia para redes grandes; (iii) arranques seguros y autenticados adecuados para entornos adversos; (iv) mecanismos que combinen arranque, recuperación ante fallos (incluyendo fallos de nodo) y soporte multi-controlador; y (v) validación en entornos reales o testbeds industriales. Estos vacíos marcan líneas claras para trabajos futuros y contextualizan por qué muchas propuestas actuales se centran en escenarios acotados o exigen compromisos prácticos entre coste, seguridad y complejidad.

Por último, en lo referente al control distribuido, varios trabajos abordan cómo establecer conectividad *in-band* tanto entre switches y controladores como entre controladores entre sí. Schiff *et al.* [56] presentan Medieval, que crea y mantiene dos árboles de expansión por controlador, uno restringido a la región del controlador y otro de alcance global, para conseguir tanto la conquista de switches como la interconexión entre controladores. La propuesta incluye un prototipo en emulación pero requiere IPs de controlador preconfiguradas y reglas iniciales en los switches; no detalla por completo la gestión de fallos de controlador y depende de una determinada configuración previa. En trabajos posteriores, los mismos autores refinan la idea con una aproximación *plug & play* que permite añadir y quitar controladores dinámicamente [57], aunque nuevamente la solución se apoya en reglas preinstaladas y en ARP para descubrimiento, por lo que queda espacio para reducir las dependencias iniciales y hacer el arranque completamente autónomo. La coordinación segura entre controladores es clave para evitar inconsistencias en el plano de control. Schiff

*et al.* proponen un marco de sincronización basado en operaciones atómicas implementadas exclusivamente con OpenFlow (aprovechando la característica de los *bundles*) [58]. El enfoque demuestra que es posible diseñar primitivas de sincronización sin modificar switches, pero su validez práctica requiere controladores y switches que soporten las primitivas de OpenFlow necesarias (p. ej. *bundles* en versiones  $\geq 1.4$ ) y su evaluación se limita a entornos emulados. Otro bloque importante de trabajo en entornos de control distribuido se centra en garantizar recuperación y convergencia tras fallos arbitrarios en el plano de control, para asegurar la consistencia de control a lo largo de la red. Canini *et al.* [59, 60], presentan Renaissance, donde introducen mecanismos que aseguran que, tras fallos arbitrarios, cada controlador no-fallido pueda alcanzar switches en tiempo acotado y que cada switch sea gestionado por al menos un controlador no-fallido. Renaissance ofrece pruebas formales de auto-estabilización y evaluaciones extensas en emulación. Estos avances resuelven muchos problemas teóricos, pero la mayoría de las pruebas siguen siendo emuladas y falta validación en testbeds a escala real. La gestión dinámica del balance de carga de control y la activación/desactivación de controladores es abordada por Görkemli *et al.* [61]. Su arquitectura permite activar controladores bajo elevada carga y reconfigurar rutas de control según la demanda; fue validada en Mininet con OVS y Floodlight, mostrando mejora frente a enfoques estáticos. Sin embargo, la evaluación se limita a entornos virtualizados y falta demostrar su resiliencia y costo real en hardware y escenarios con tráfico heterogéneo. Algunos trabajos plantean canales livianos para comunicación entre controladores. Hark *et al.* [62] proponen un servicio de inter-controladores de bajo coste que emplea mensajes embebidos en eventos de cambio de puerto y Virtual Local Area Networks (VLANs) para aislamiento; muestran crecimiento lineal de mensajes con el número de controladores. Canini *et al.* [63] proponen procedimientos basados en *timeout* frente a enfoques iterativos para que switches obtengan rutas hacia todos los controladores y para coordinar controladores entre sí. Ambos contribuyen a la coordinación, pero no resuelven plenamente la escalabilidad de la señalización ni la gestión de conflictos en topologías grandes o federadas. La detección y localización de fallos son tratadas por Kwan-Yee *et al.* [64], que describen un esquema de sondeo cíclico que identifica rápidamente fallos y localiza su origen; su evaluación en red real con ONOS y switches comerciales exhibe tiempos de recuperación por debajo de 50 ms para fallos simples. Este trabajo muestra que la detección rápida es viable en hardware, aunque el coste en sondeos periódicos y su escalado a redes muy grandes requieren una caracterización más profunda. Holzmann *et al.* [47] presentan primero Izzy (mencionado anteriormente), un esquema robusto basado en árboles y *locators*, y posteriormente Seedling [65], que agrupa controladores por proximidad para reducir coste computacional y tablas de forwarding. Estas propuestas exploran *trade-offs* entre robustez y coste: Seedling reduce la carga frente a repetir Izzy en cada controlador, pero urge analizar pérdida de generalidad y límites de agrupamiento en topologías heterogéneas.

En general, la mayoría de las propuestas se han validado en simuladores o entornos emulados (Mininet, Java, OVS), aunque hay excepciones con pruebas en hardware real (p. ej. Kwan-Yee *et al.* con ONOS y HP5900). Si bien el código de Renaissance y otras implementaciones están disponibles, la transición a testbeds industriales y despliegues a escala sigue siendo limitada. La literatura sobre control distribuido para SDN ha avanzado en

marcos conceptuales (Medieval, Renaissance), mecanismos de sincronización y algoritmos para robustez y eficiencia (Izzy, Seedling), así como en soluciones pragmáticas para detección y coordinación (Hark *et al.*, Kwan-Yee *et al.*). No obstante, persisten huecos claros: reducir las dependencias de configuración inicial, escalar la coordinación entre numerosos controladores, trasladar garantías formales a entornos reales y optimizar coste/beneficio en detección y señalización. Estas lagunas dibujan líneas de trabajo futuro necesarias para llevar los resultados de laboratorio a despliegues industriales y operativos.

#### 2.1.4. Conclusiones y alineación con los objetivos de la Tesis

El análisis crítico de la literatura sobre control *in-band*, el encaminamiento automático, arranque de red y control distribuido, ha permitido identificar una serie de huecos recurrentes que condicionan la adopción práctica de estos enfoques en entornos reales. Entre los hallazgos más relevantes destacan: la ausencia de un protocolo *in-band* estandarizado y agnóstico al fabricante/proveedor; la necesidad de mecanismos de arranque autónomo (*bootstrapping*) escalables, seguros y con baja latencia; la complejidad de coordinar controladores distribuidos sin incurrir en *overhead* excesivo; y la escasa validación en infraestructuras reales o testbeds a gran escala.

Esto encaja de forma directa con el planteamiento y los objetivos de la Tesis. El primer bloque de objetivos (profundizar en los mecanismos de control de redes programables y extender el paradigma SDN a ámbitos como IIoT y redes de distribución eléctrica) queda plenamente justificado por los *gaps* detectados: en escenarios con nodos con recursos limitados (IIoT) o topologías jerárquicas (redes eléctricas), las soluciones *out-of-band* no son viables y se requiere avanzar en protocolos *in-band* que sean seguros, autónomos y adaptativos. Además, la necesidad de algoritmos capaces de decidir dinámicamente (encaminamiento, reconfiguración, intercambio de recursos como cómputo o energía) refuerza la pertinencia de investigar técnicas que integren procedimientos clásicos con herramientas de AI/ML para predicción de fallos y toma de decisiones proactivas.

El segundo bloque de objetivos (diseño de infraestructuras software para gestión, orquestación y seguridad en redes softwarizadas) también responde a los *gaps* encontrados. La literatura muestra propuestas puntuales de arquitectura y prototipos que no siempre consideran la integración con sistemas de monitorización, gestión en el arranque, o la orquestación autónoma en entornos heterogéneos. Por tanto, resulta necesario definir arquitecturas modulares y alineadas con estándares que faciliten la interoperabilidad, permitan la incorporación de capacidades de AI/ML. A la luz de lo anterior, esta Tesis propone una hoja de ruta clara y coherente con los objetivos planteados. Desde la perspectiva metodológica, la Tesis busca cerrar la brecha entre modelado teórico y aplicabilidad práctica: no se trata solo de proponer algoritmos con buenas propiedades matemáticas, sino de validar su viabilidad operativa y sus implicaciones clave. En particular, la incorporación de escenarios aplicados (IIoT, SG) permitirá evaluar restricciones reales y topológicas que condicionan tanto la elección del paradigma de control, cómo su diseño.

## 2.2. Servicios y Tecnologías habilitantes en redes softwareizadas y heterogéneas

En este segundo bloque se quieren revisar las principales tecnologías habilitantes que permiten la creación, control y gestión de las redes programables y heterogéneas, así como identificar dónde y de que manera se integran con el diseño del paradigma SDN. Estas tecnologías son fundamentales para entender el marco de trabajo de la tesis, y cómo, posteriormente, se pueden llegar a aplicar a diferentes casos de uso.

Una vez revisado el paradigma SDN, se tiene que ahondar en el diseño arquitectónico promovido por la ONF, el cual, evolucionó desde la separación inicial entre plano de control y plano de datos hacia una visión más amplia centrada en servicios: el denominado Management–Control Continuum (MCC) [66]. En esta nueva visión, funcionalidades que implementa el controlador SDN se identifican como servicios, y los datos y los nodos a gestionar por el controlador, como recursos, según se describe en la Figura 2.6.

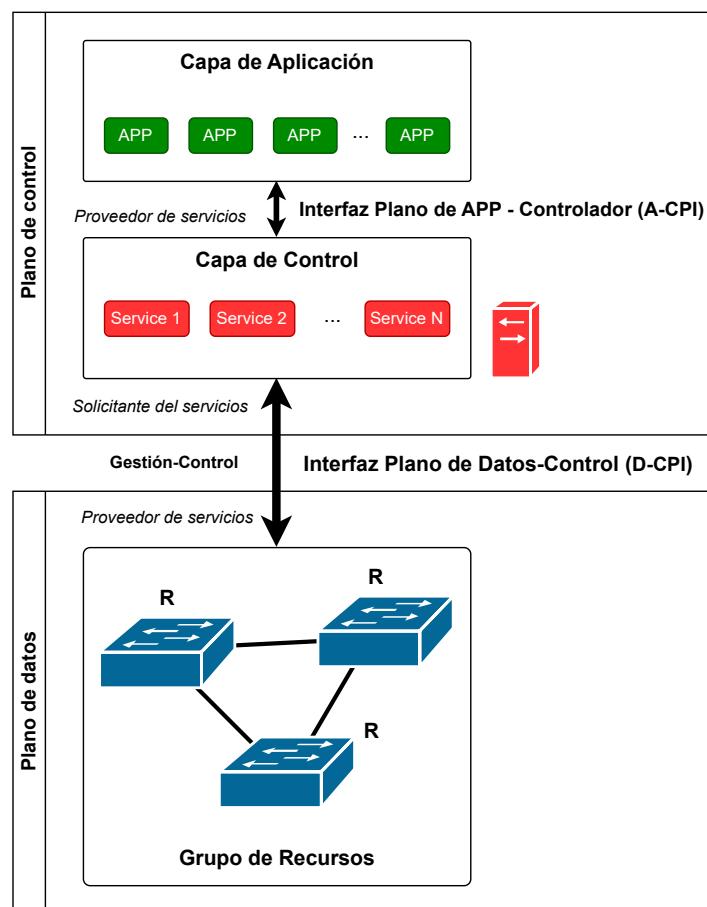


Figura 2.6: Paradigmas de control Management–Control Continuum (MCC).

Según la ONF, una arquitectura es una colección necesariamente incompleta de perspectivas sobre un conjunto de ideas subyacentes, por ello, partiendo del modelo básico de SDN, trataron de identificar todas las funcionalidades que implementaba el controlador como servicios que gestionan una serie de recursos, en este caso los nodos de la red, llevando a la mínima expresión la arquitectura de una manera agnóstica, para que esta fuera extrapolable a cualquier tipo de redes programables y heterogéneas. Por ende, un consumidor de servicio, será aquel que intercambia tanto datos como operaciones de control y gestión con un servidor o proveedor SDN (D-CPI). Aunque los datos del usuario son finalmente transmitidos o procesados por un conjunto de recursos (R) que pertenecen al proveedor SDN, el cliente final controla su servicio mediante una sesión de control y gestión, invocando acciones (A-CPI) sobre un conjunto de recursos virtuales alojados en el controlador que percibe como propios. El controlador SDN, entre otras funciones, se encarga de virtualizar y orquestar la vista de recursos y servicios que tiene el cliente, mapeándola sobre los recursos y servicios reales del proveedor. Los conceptos de “recursos” y “servicios” son intencionalmente flexibles y no tienen límites fijos. La arquitectura SDN amplía este modelo básico, al denominado como MCC, y aclara sus implicaciones, incluyendo extensiones clave como el uso compartido de recursos entre múltiples clientes, la asignación dinámica y la optimización de su uso.

El MCC organiza las capacidades de la red en una jerarquía de servicios que va desde los servicios básicos de gestión y control (descubrimiento, provisión, configuración, monitorización) hasta servicios avanzados de orquestación y automatización que permiten funcionalidades de alto valor sobre infraestructuras softwarizadas. Esta visión facilita describir la red como una plataforma de servicios en la que la infraestructura subyacente se abstrae y se pone a disposición de funciones gestionadas y componibles. Bajo el paraguas del MCC, resulta útil clasificar el estado del arte en dos bloques complementarios que recogen las tecnologías habilitantes principales que se exploran en esta Tesis: (i) servicios básicos que habilitan la operación mínima de la softwarización (descubrimiento de topología, arranque autónomo, provisión de canales de control, telemetría) y (ii) servicios avanzados que aportan capacidades de decisión, predicción y optimización (orquestación, planificación de recursos, predicción de fallos y adaptación proactiva), normalmente apoyados en técnicas de inteligencia artificial y aprendizaje automático. En las siguientes sub-Secciones, se explorarán los contenidos principales de esta Tesis englobados sobre los dos bloques de servicios anteriormente mencionados en el contexto del paradigma de control MCC.

### **2.2.1. Servicios básicos: arranque y provisión de canales de control en entornos densos y heterogéneos**

Los servicios básicos de arranque y provisión de canales de control constituyen la base operativa del MCC. Sin un mecanismo fiable para que los dispositivos descubran la topología, obtengan parámetros iniciales y establezcan conectividad con el controlador, es imposible desplegar servicios de mayor nivel. En entornos densos y heterogéneos (IIoT, micro-redes eléctricas, redes logísticas), las restricciones de recursos, la topología multi-salto y la posible ausencia de enlaces dedicados obligan a diseñar protocolos *in-band* o

híbridos que sean escalables, seguros y poco intrusivos, a la par que resilientes.

En la Sección 2.1.3 se pudo explorar todas las propuestas de mecanismos *in-band*, entre las cuales se pudo observar diferentes estrategias de arranque/provisión que emplea:

1. **Protocolos clásicos asistidos (DHCP/ARP/LDP):** usan servicios L2/L3 existentes para obtener parámetros y descubrir la topología (p. ej. [49]).
2. **Árboles raíz-centrados y exploración por difusión:** el nodo raíz (con enlace al controlador) propaga paquetes de exploración que exploran la topología y permiten instalar rutas *in-band* [45]).
3. **Etiquetado jerárquico:** asignación de identificadores temporales o jerárquicos a nodos para encaminar mensajes de control sin conocer la topología completa (p. ej. la propuesta de Izzy [47], o Amaru [45]).
4. **Híbridos con canales *out-of-band* para arranque:** soluciones que requieren un enlace auxiliar o reglas preinstaladas para la fase inicial y luego migran a *in-band* (véase [43, 51]).
5. **Mecanismos basados en multipath/TCP avanzado:** uso de MPTCP para mejorar disponibilidad del canal de control mediante rutas múltiples (p. ej. [42]).

Sin embargo, gran parte de las soluciones revisadas fueron concebidas para redes de comunicaciones convencionales y, como consecuencia, dependen de protocolos auxiliares o de implementaciones específicas en equipos SDN. Ese fuerte acoplamiento limita su aplicabilidad a entornos densos y heterogéneos, por ejemplo, redes con nodos de recursos muy reducidos (IIoT) o redes de distribución eléctrica con topologías en árbol, donde no es viable confiar en servicios adicionales ni en modificaciones de hardware/software. En contraste, los enfoques más abstractos, basados en esquemas de etiquetado jerárquico y en la construcción de árboles enraizados combinados con mecanismos de difusión/control de exploración, resultan más extrapolables y prácticos para estos escenarios: reducen dependencias infraestructurales, facilitan la escalabilidad y favorecen la resiliencia mediante rutas alternativas y reconvergencia local, por lo que constituyen las candidatas más adecuadas para los objetivos de esta Tesis.

Los esquemas basados en etiquetado jerárquico combinados con la construcción de árboles enraizados constituyen una estrategia natural y eficiente para el arranque y la provisión de canales de control en redes densas y heterogéneas. En estos enfoques, la red se organiza lógica o temporalmente en niveles mediante identificadores (etiquetas) que codifican información de pertenencia o proximidad, de forma análoga a prefijos jerárquicos; dichas etiquetas permiten encaminar mensajes de control con mínimo conocimiento global, reduciendo la necesidad de tablas extensas en los nodos. Estas etiquetas, pueden identificar a nivel de nodo, a nivel de enlace o a nivel de puerto, y dependerá del caso de uso que tipo de identificador emplear. La construcción de un árbol enraizado con la raíz colocada en el nodo, o nodos, con acceso directo al controlador facilita el arranque de la red, dado que, paquetes de exploración pueden propagarse de forma controlada desde la raíz acumulando

información topológica y asignando rutas *in-band* sin inundar la red. Esta combinación aporta ventajas prácticas relevantes para entornos con recursos limitados: localiza las decisiones de reencaminamiento, ofrece caminos de protección natural (rutas alternativas hacia la raíz dentro del árbol o a través de etiquetas vecinas) y limita el *overhead* de señalización. No obstante, conlleva retos propios, como por ejemplo, la gestión dinámica de etiquetas ante cambios topológicos (re-etiquetado) puede generar coste de señalización, y es preciso diseñar mecanismos para evitar bucles y ataques por suplantación (p. ej. autenticación ligera de mensajes de exploración). Es por ello, que el etiquetado jerárquico más árboles enraizados proporciona un marco abstracto, escalable y resistente, bien adaptado a IIoT, micro-redes y topologías jerárquicas en árbol típicas en redes de distribución eléctrica, siempre que se complemente con políticas de convergencia eficientes y escalables adecuadas para el caso de uso final.

Para analizar con mayor detalle los enfoques basados en etiquetado jerárquico y la construcción de árboles enraizados, esta sección introduce los conceptos básicos de teoría de grafos necesarios para modelar la topología de una red y para explicar cómo, mediante esquemas de etiquetado, es posible derivar una estructura de árbol enraizado adecuada para el arranque y el encaminamiento de control. A continuación se formalizan los términos y notaciones que se emplearán en el resto del capítulo; posteriormente se estudian diferentes propuestas de etiquetado jerárquico y su aplicabilidad a las redes densas y heterogéneas.

### 2.2.1.1. Terminología básica

La teoría de grafos es la disciplina matemática que estudia las propiedades y estructuras de los grafos. Un grafo es una representación visual compuesta por una serie de objetos llamados nodos, conectados a través de otro tipo de objetos denominados enlaces. Históricamente, los orígenes de la teoría de grafos se remontan al problema de los puentes de Königsberg, también conocido como problema de los siete puentes de Königsberg. Su nombre se debe al nombre de la ciudad, Königsberg, ciudad de Prusia Oriental, y el problema consistía en encontrar un camino para cruzar a pie toda la ciudad pasando solo una vez por cada uno de los siete puentes y regresando al mismo punto de partida. El problema, formulado originalmente de manera informal, se propagó a modo de juego matemático entre los intelectuales de la época, siendo resuelto por Leonhard Euler en 1736 [67] y cuya resolución dio origen y sentó las bases de la teoría de grafos.

Formalmente, un grafo se define como  $G = (\mathcal{N}, \mathcal{L})$ , siendo  $\mathcal{N}$  un conjunto de  $N$  nodos, y  $\mathcal{L} \subseteq \{\{i, j\} \mid i, j \in \mathcal{N} \text{ con } i \neq j\}$ , un conjunto de enlaces los cuales se componen de un par desordenado de nodos, es decir, cada enlace tiene dos nodos diferentes. Estos enlaces pueden tener una direccionalidad, o no. Atendiendo a la direccionalidad del conjunto de enlaces, los grafos se pueden clasificar en grafos dirigidos o no dirigidos, siendo los grafos dirigidos los que restringen el tránsito por los enlaces a una o varias direcciones específicas, mientras que los grafos no dirigidos no establecen ninguna obligatoriedad en la dirección de tránsito. Asimismo, se puede establecer un coste asociado a atravesar un enlace, pudiendo clasificar los grafos también en grafos ponderados, donde los enlaces tienen un coste asociado, y no ponderados, donde los enlaces no cuentan con un coste asociado.

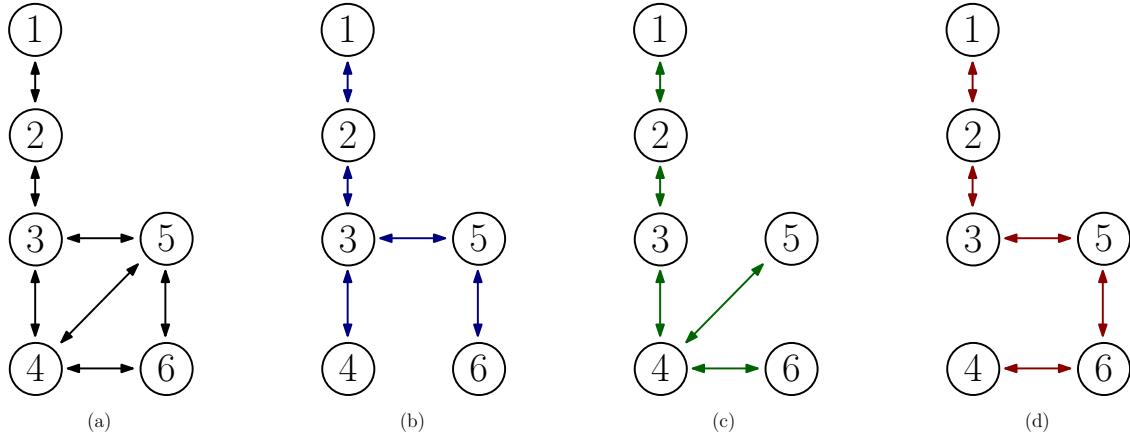
Otra característica interesante para clasificar los grafos es si tienen ciclos (bucles) o no. Un grafo se considera acíclico si no contiene ningún ciclo, esto significa que para cada nodo  $i \in \mathcal{N}$ , no hay un camino directo que empiece y termine en  $i$ .

Si, además, es conexo y no dirigido, es decir, existe un camino entre cualquier par de nodos, y acíclico, se denomina árbol. En el caso dirigido, cuando no existen ciclos dirigidos se habla de un Directed Acyclic Graph (DAG). La presencia o ausencia de ciclos es una propiedad estructural clave que condiciona los algoritmos de encaminamiento, los procesos de exploración/*bootstrapping* y los mecanismos de convergencia (por ejemplo, la prevención de bucles o la estrategia de etiquetado). Por ello, su consideración resulta esencial en el diseño de esquemas de etiquetado jerárquico y en la derivación de árboles enraizados a partir de la topología original  $G$ .

Por tanto, la teoría de grafos se puede aplicar de forma directa sobre las redes densas y heterogéneas, modelando dichas redes partiendo de una topología física a un grafo, el cual será hiperconectivo. Sobre la topología física habrá que definir cual será el nodo, o los nodos, que darán acceso al controlador, los cuales se identificarán como nodos raíz. Dichos nodos serán los encargados de iniciar el establecimiento del proceso de etiquetado jerárquico para poder construir el árbol enraizado. Este árbol, según se ha indicado, es un tipo especial de grafo acíclico conectado, el cual, también puede considerarse como un grafo con  $n$  nodos y  $n - 1$  enlaces. Esto se puede analizar como la construcción de una topología lógica sobre una topología física, la cual contendrá todos los nodos de la topología anterior, pero un subconjunto de los enlaces de la topología física. Formulando este proceso del establecimiento de la topología lógica, podemos definir que dado un grafo  $G = (\mathcal{N}, \mathcal{L})$ , siendo  $\mathcal{N}$  un conjunto de  $N$  nodos y  $\mathcal{L} \subseteq \{\{i, j\} \mid i, j \in \mathcal{N} \text{ con } i \neq j\}$  un conjunto de enlaces los cuales se componen de un par desordenado de nodos (es decir, cada enlace tiene dos nodos distintos), se puede obtener la topología lógica como un subgrafo, sea  $G' = (\mathcal{N}', \mathcal{L}')$  donde se cumple  $\mathcal{N}' \subseteq \mathcal{N}$  y  $\mathcal{L}' \subseteq \mathcal{L}$ . En este caso, de forma adicional, se cumple la relación de igualdad para los nodos, dado que  $\mathcal{N}' \subseteq \mathcal{N} \wedge \mathcal{N} \subseteq \mathcal{N}'$ , haciendo que  $\mathcal{N}' = \mathcal{N}$ . Es decir,  $a_i \in \mathcal{N} / b_i \in \mathcal{N}'$ ,  $0 \leq i \leq N$  se cumple  $\forall a_i = b_i$ .

Para ilustrar este proceso de establecimiento de la topología lógica, se presenta la Figura 2.7. Como se puede ver en la Figura 2.7 (a), se parte de un grafo inicial, el cual representará a pequeña escala una red densa y heterogénea. Sobre este grafo se pueden seleccionar distintos subgrafos que cumplan la relación de igualdad según se ha explicado anteriormente. Pudiendo elegir entre distintas topologías lógicas (b,c,d) a conveniencia, y función de que nodo se establezca como nodo raíz de la topología. En el caso de que el nodo raíz de la topología fuera el nodo 1, podemos seleccionar la topología lógica (b) que habilita al nodo 5 de una conexión más directa con el nodo que da acceso al controlador. Otra opción sería elegir la configuración (c), en caso de que los nodos 5 y 6 no tuvieran una necesidad explícita QoS en la comunicación con el controlador de la red. De igual forma podemos elegir otra configuración, como por ejemplo (d), donde el nodo 4 se encontrara al final de la topología lineal conformada. La elección concreta de la topología lógica, y por tanto la forma del árbol enraizado, depende de criterios de diseño del caso de uso. Por ejemplo, minimizar la profundidad del árbol (para reducir latencias hacia la

raíz), priorizar enlaces con mayor capacidad o menor pérdida (criterio QoS), favorecer rutas disjuntas para aumentar la resiliencia, o minimizar el *overhead* de señalización.



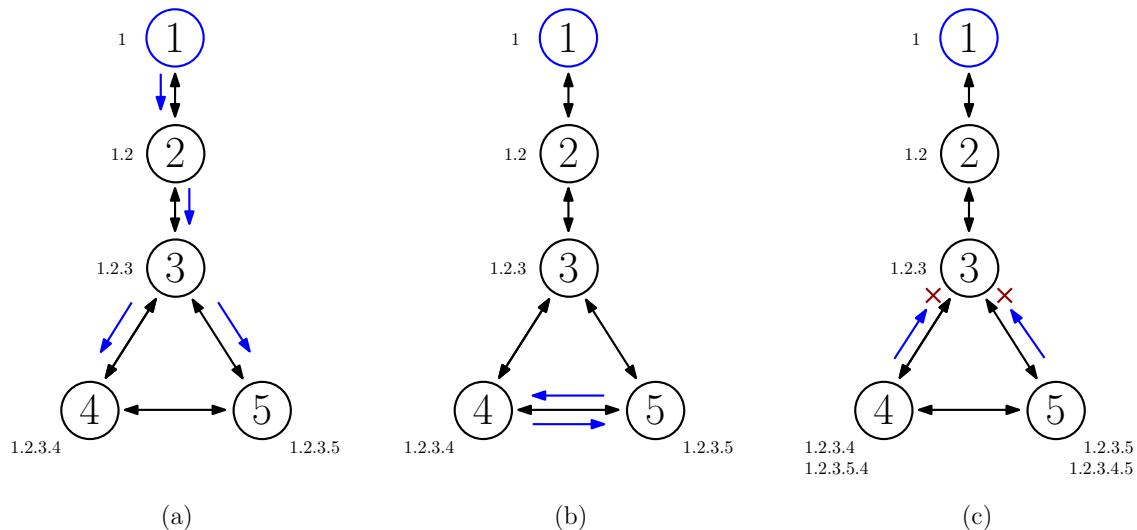
**Figura 2.7:** Elección de una topología lógica (árbol enraizado) partiendo de una topología física.

El etiquetado jerárquico es una técnica eficiente y abstracta para obtener una topología lógica, habitualmente un árbol enraizado, a partir de una topología física  $G = (\mathcal{N}, \mathcal{L})$ . En este paradigma, a cada nodo  $i \in \mathcal{N}$  se le asigna una etiqueta que codifica su posición relativa dentro de una jerarquía o partición de la red. Estas etiquetas permiten a los nodos tomar decisiones de reenvío locales, por ejemplo, mediante coincidencia de prefijos, sin requerir una visión global completa de  $G$ . El etiquetado jerárquico resulta especialmente adecuado para entornos densos y heterogéneos porque reduce la información de estado requerida en nodos con recursos limitados, facilita la derivación de rutas hacia la raíz o raíces usando información local y habilita mecanismos de agregación que simplifican la gestión. Existen diversas familias de esquemas de etiquetado jerárquico. En el etiquetado por prefijos, cada etiqueta se expresa como una secuencia jerárquica  $i_1.i_2.\dots.i_k$ . En el etiquetado por intervalos, a cada subárbol o partición se le asigna un intervalo de identificadores, de modo que un nodo conoce su pertenencia comprobando si su identificador está dentro del intervalo correspondiente. Otros enfoques utilizan identificadores compactos que codifican información relativa a la distancia o la topología respecto a la raíz, y todavía existen esquemas basados en clusterización, en los que la red se divide en grupos y cada grupo recibe una etiqueta de nivel superior con subetiquetas internas. La elección entre estas variantes suele depender de limitaciones de memoria, de la necesidad de agregar información (p. ej. capacidades o requisitos QoS) y de la facilidad de reconfiguración en caso de cambios topológicos.

Operativamente, la creación de la topología lógica mediante etiquetado se articula en varias fases. Primero se seleccionan las raíces, es decir, los nodos con acceso directo al controlador o controladores, que actuarán como iniciadores del proceso. A continuación, desde la raíz se emiten mensajes de exploración controlada que se propagan por la red física; al recibir un mensaje por primera vez, un nodo acepta la etiqueta propuesta (por ejemplo, concatenando un identificador del enlace, el puerto, su identificador o un sufijo

jerárquico) y establece una relación de “vecindad” con el emisor. Con la etiqueta asignada, cada nodo obtiene reglas locales de reenvío que permitan encaminar paquetes de control hacia su padre y, por ende, hacia la raíz. De forma complementaria, es posible manejar varias etiquetas por nodo que identifiquen rutas alternativas, facilitando así la reconvergencia rápida ante fallos de enlace o de nodo. En entornos prácticos, estas fases deben diseñarse para evitar inundaciones de mensajes y para que la aceptación de etiquetas siga políticas antibúcles.

Para ilustrar este proceso se presenta la Figura 2.8. En este caso se puede apreciar un grafo de cinco nodos, entre los cuales se ha elegido al nodo 1 como nodo raíz de la topología. Será este nodo quien empiece el proceso de difusión del etiquetado jerárquico. En este escenario se ha optado por un tipo de etiquetado basado en identificadores únicos los cuales representan de forma inequívoca al nodo dentro de la topología física. En el primer paso de la difusión, ver Figura 2.8 (a), el nodo raíz empieza a difundir su etiqueta con identificador único 1, la cual llega hasta el nodo 2, quien la recibe y acto seguido concatena su identificador único generando la etiqueta 1.2 que será almacenada y difundida por todos los enlaces disponibles menos por el que le llegó (siendo un aspecto de diseño en la estrategia de etiquetado). De forma análoga, el nodo 3 generará la etiqueta 1.2.3, almacenándola y difundiéndola a los nodos 4 y 5, obteniendo estos a su vez las etiquetas 1.2.3.4 y 1.2.3.5, respectivamente. En el segundo paso de la difusión, ver Figura 2.8 (b), los nodos de la topología ya tienen un camino directo para alcanzar el nodo raíz, el nodo 2 de forma directa, el nodo 3 a través del nodo 2, y los nodos 4 y 5 a través del nodo 3. Sin embargo, el proceso de difusión no ha concluido, dado que los nodos 4 y 5 aún tienen una etiqueta por difundir, la cual se intercambian entre ellos. En el tercer paso de la difusión, ver Figura 2.8 (c), los nodos 4 y 5 obtienen las etiquetas 1.2.3.5.4 y 1.2.3.4.5 respectivamente, lo que les habilita a una ruta adicional hacia el nodo raíz (del nodo 4 a través del nodo 5, y viceversa), pudiendo elegir entre un camino u otro.



**Figura 2.8:** Ejemplo de difusión empleando el etiquetado jerárquico.

En el último paso (Figura 2.8 (c)), se puede apreciar otro aspecto clave de los esquemas de etiquetado: los mecanismos antibucle. Los nodos 4 y 5 reciben respectivamente las etiquetas 1.2.3.5.4 y 1.2.3.4.5, la cuales tendrán que ser difundidas de forma análoga hacia el nodo 3. Al llegar dichas etiquetas al nodo 3, y al utilizar identificadores inequívocos, este tiene la capacidad de comprobar si las etiquetas recibidas contienen su propio identificador, y en tal caso descartarlas, dado que el propio significado de la etiqueta le estaría indicando al nodo que en esa ruta tendría que volver sobre él en una ocasión, reflejando un bucle (ciclo). Una vez descartadas las etiquetas, y con todas las etiquetas difundidas en la topología física, todos los nodos tendrían una o más rutas para alcanzar al nodo raíz y conocerían su posición jerárquica en la topología. De esta forma, aplicando diferentes criterios de los anteriormente mencionados, se podrían obtener tres topologías lógicas (tres árboles) a partir de una topología física: el primero donde los nodos 4 y 5 cuelgan directamente del nodo 3, el segundo donde el nodo 4 se conecta directamente al nodo 3 y el nodo 5 se conecta al nodo 4, y el tercero, donde el nodo 5 se conecta directamente al nodo 3 y el nodo 4 se conecta al nodo 5. Este proceso de etiquetado y selección de la topología lógica puede ser ejecutando de forma periódica, o configurable, para actualizar el encaminamiento automático, explorar de nuevo la topología en escenarios de movilidad o modificaciones de la topología física, siendo esto crítico en redes densas y heterogéneas.

El diseño de esquemas de etiquetado implica una serie de *trade-offs* relevantes. Etiquetas más compactas reducen la memoria y la carga de procesamiento en los nodos, pero limitan la información codificable (por ejemplo, la posibilidad de transportar metadatos de capacidad o prioridad). Los esquemas por prefijo e intervalos son naturalmente agregables y escalables, pero pueden requerir re-etiquetado amplio cuando la topología cambia; por ello, las soluciones prácticas suelen implementar re-etiquetado localizado, versionado de etiquetas o etiquetas temporales para mitigar la señalización. La prevención de bucles durante la fase de construcción exige políticas en los mensajes de exploración según se ha indicado.

Es por ello, que el etiquetado jerárquico encaja especialmente bien con los requisitos de redes IIoT, micro-redes y sistemas de distribución eléctrica y redes de comunicaciones, dado que permite realizar *bootstrapping* sin depender de servicios L2/L3 adicionales ni de extensiones propietarias, limita el estado necesario en cada nodo, escala bien con el tamaño de la topología física, y facilita la introducción explícita de rutas de respaldo. Además, este enfoque se complementa de forma natural con técnicas de AI/ML, modelos predictivos pueden priorizar la exploración de determinados subárboles, seleccionar enlaces con menor probabilidad de fallo o recomendar políticas de re-etiquetado que optimicen la resiliencia y el rendimiento de la topología lógica. En la siguiente sección, se analizarán algunas de las propuestas de etiquetado jerárquico, se compararán sus prestaciones y bondades que serán de utilidad en los entornos estudiados en esta Tesis.

### 2.2.1.2. Propuestas de etiquetado jerárquico

En el contexto de los servicios básicos, en concreto, al arranque y provisión de canales de control en entornos densos y heterogéneos, el etiquetado jerárquico aparece como una

solución consolidada dentro del catálogo de funcionalidades que el MCC clasifica como servicios de gestión y control fundamentales. Aunque el etiquetado jerárquico no constituye una novedad conceptual [68], ha sido aplicado con éxito en dominios tan diversos como enrutamiento por prefijos [69], direccionamiento en redes de sensores y esquemas de localización para IoT [70], su carácter transversal y su capacidad de abstracción lo convierten en un mecanismo idóneo para materializar las capacidades mínimas que exige el MCC en escenarios con recursos limitados y topologías heterogéneas. Por otra parte, muchas de las propuestas y variaciones prácticas que se examinan a continuación proceden de nuestro grupo de investigación, NetIS, que lleva años investigando y prototipando esquemas de etiquetado y árboles enraizados; estas aportaciones servirán de referencia a la hora de comparar enfoques, identificar *gaps* y situar las contribuciones de esta Tesis en el marco de las redes densas y heterogéneas. A continuación se presenta una taxonomía de enfoques y se analizan propuestas representativas, prestando especial atención a su aplicabilidad práctica, requisitos de implementación y limitaciones en entornos densos, distribuidos y heterogéneos.

En primer lugar, los trabajos teóricos clásicos proporcionan la base formal del problema de asignación de identificadores en redes anónimas. Uno de los primeros trabajos identificados en materia de etiquetado jerárquico lo presentan Fraigniaud *et al.* [68], donde estudian la asignación de etiquetas únicas en redes anónimas sin conocimiento previo de la topología ni de su tamaño, empleando un modelo síncrono con una fuente distinguida, un nodo raíz, que inicia el proceso. Sus resultados son valiosos porque demuestran límites y *trade-offs* fundamentales entre longitud de etiqueta, tiempo y complejidad de mensajes; sin embargo, las fuertes hipótesis (p. ej. sincronía estricta) y la ausencia de tratamiento de topologías dinámicas o con fallos hacen que su aplicabilidad directa a entornos reales y heterogéneos sea limitada. Este tipo de trabajo marca el terreno teórico, pero deja abiertos problemas prácticos como la asincronía y las restricciones de recursos en redes de baja capacidad.

En el ámbito aplicado, una familia de propuestas orientadas a los DC explora mecanismos de etiquetado jerárquico para encaminamiento sin tablas y reenvío rápido. Rojas *et al.* [69], presentan Torii-HLMAC, donde introducen direcciones posicionales, Hierarchical Local MAC (HLMAC), y un mecanismo distribuido de asignación automática que habilita encaminamiento sin tablas en *bridges* y recuperación instantánea ante fallos. Más adelante, Rojas *et al.* [71, 72], evolucionan esta idea, presentando Generalized Automatic Address Assignment (GA3) y eTorii. GA3 propone un descubrimiento distribuido que genera múltiples direcciones HLMAC ordenadas y soporta encaminamiento sin tablas de reenvío sobre árboles de rutas mínimas; eTorii combina *source routing* con asignación automática para permitir encaminamiento *on-the-fly*. Estas propuestas comparten ventajas claras para topologías jerárquicas (baja ocupación de estado, comutación rápida, balanceo por selección de etiquetas), pero su evaluación se ha centrado mayoritariamente en topologías de DC (fat-trees, topologías jerárquicas) y bajo condiciones relativamente homogéneas. Por ello, se pueden identificar una serie de debilidades evidentes, como por ejemplo, la extrapolación a topologías irregulares o muy heterogéneas, el comportamiento bajo enlaces de muy distinta capacidad/latencia, y el coste de implementación en dispo-

sitivos con capacidades limitadas requieren más estudio experimental y, en algunos casos, soporte en hardware (p. ej. P4) para validar su uso en producción.

En paralelo aparecen soluciones probabilísticas y adaptativas pensadas para redes dinámicas. El trabajo publicado por Walraed *et al.* [73], presentan la solución ALIAS bajo el protocolo *Decider/Chooser Protocol* (DCP), donde proponen algoritmos aleatorizados para la selección de etiquetas en redes con cambios frecuentes; su fortaleza es la adaptabilidad y las buenas garantías probabilísticas de convergencia, pero su evaluación se centra en escenarios de centros de datos y deja abierto cómo se comporta en entornos con recursos muy limitados o con tasas extremas de *churn* (tasa de abandono y unión masiva de nodos). Estos enfoques aleatorizados aportan resiliencia frente a dinamismo, pero su sobrecoste de mensajes y su idoneidad para nodos de baja energía deben cuantificarse para IIoT.

Respecto a redes de baja capacidad con restricciones en consumo, y en almacenamiento de información de estado, Rojas *et al.* [70], presentan IoTorii, donde se adapta el paradigma jerárquico a las Low-Power and Lossy Networks (LLNs). IoTorii se basa también en las HLMAC, reutilizando direcciones MAC típicas transformándolas en etiquetas jerárquicas y emplea sondas *broadcast* para descubrir rutas múltiples sin modificar la pila de protocolos utilizada en las LLNs. Los resultados experimentales muestran reducción de entradas en tablas de reenvío, y un menor *overhead* respecto a Routing Protocol for Low-Power and Lossy Networks (RPL) (protocolo de facto en las LLNs) en escenarios estáticos, lo que evidencia la promesa del etiquetado en entornos de baja potencia. No obstante, quedan sin resolver la robustez ante movilidad elevada, la variabilidad extrema de enlaces y la necesidad de políticas energéticas más agresivas para redes de sensores reales, densas y heterogéneas.

En cuanto a las soluciones enfocadas directamente a las redes típicas SDN, tenemos que volver a mencionar Lopez-Pajares *et al.* [45], y Holzmann *et al.* [47], vistos en la Sección 2.1.3, pero haciendo ahora hincapié en su enfoque de etiquetado jerárquico. Lopez-Pajares *et al.* [45], presentaban Amaru, donde apuestan por la exploración controlada desde un nodo raíz para recopilar la topología e instalar rutas *in-band*. Amaru combina exploración y construcción de rutas con respaldo múltiples, y demuestra tiempos de recuperación muy bajos con *overhead* casi nulo. El etiquetado que emplean se basa en los identificadores de los puertos de los nodos SDN que gestionan, haciendo que su principal limitación práctica es que requiere ligeras modificaciones en los switches SDN para soportar el protocolo, lo que plantea barreras de despliegue en entornos heterogéneos donde no es posible modificar el plano de datos de todos los equipos. Esta es una debilidad operativa importante para entornos industriales o de infraestructura crítica donde la interoperabilidad con equipos *legacy* es necesaria. De forma similar, Holzmann *et al.* [47], proponen Izzy, basado en un diseño modular, que combina un árbol de expansión con identificadores temporales dependientes de la topología para asegurar conectividad *in-band* con tiempos de recuperación por debajo de 100 ms en simulaciones de WAN. Los identificadores temporales les ayudan a detectar cuando uno de los nodos establecidos en la ruta *in-band* deja de estar operativo, lo que les permite actualizar dichas rutas y repa-

rar el canal de comunicación en un tiempo adecuado. Izzy aporta una arquitectura bien estructurada para mantener tablas y rutas de respaldo, y demuestra que la combinación del etiquetado junto al árbol de expansión puede alcanzar latencias de conmutación muy bajas. Sin embargo, su validación está limitada a simulación y queda por demostrar su factibilidad en despliegues heterogéneos reales; además, la generación y gestión de identificadores temporales plantea cuestiones de coordinación entre controladores en escenarios multi-root, es decir, donde hay más de un nodo con acceso al controlador SDN.

Recapitulando y comparando, todos los trabajos explotan la idea común de codificar información topológica en etiquetas jerárquicas para reducir estado local y facilitar el encaminamiento hacia la raíz, pero difieren en decisiones de diseño clave: (i) si la asignación es determinista (prefijos ordenados, intervalos) o probabilística; (ii) si el encaminamiento es sin tabla de reenvío (Torii/GA3) o requiere reglas instaladas (Amaru/Izzy); (iii) si la solución requiere o no modificaciones en el *dataplane*; y (iv) el alcance de la evaluación (simulación vs. prototipo vs. despliegue sobre hardware). Esos ejes determinan la aplicabilidad a entornos densos y heterogéneos: por ejemplo, las soluciones sin tabla de reenvío son atractivas para switches con escaso Ternary Content-Addressable Memory (TCAM), pero su robustez en enlaces heterogéneos no ha sido ampliamente probada; las soluciones que requieren cambios en switches presentan una barrera de adopción en infraestructuras mixtas; las propuestas probabilísticas toleran *churn* pero su coste de señalización puede ser alto para sensores con baterías limitadas.

### 2.2.1.3. Conclusiones y alineación con los objetivos de la Tesis

El análisis del estado del arte ha puesto de manifiesto que el etiquetado jerárquico y las técnicas basadas en árboles enraizados constituyen un marco fértil y consolidado para abordar el arranque y la provisión de canales de control en entornos densos y heterogéneos. Las propuestas revisadas (trabajos teóricos sobre asignación de identificadores, protocolos para centros de datos como Torii/GA3/eTorii, soluciones adaptadas a LLNs como IoTorii, protocolos *in-band* como Amaru e implementaciones modulares como Izzy) comparten la idea de explotar información codificada localmente en etiquetas para reducir estado y facilitar encaminamiento hacia la raíz o raíces. Sin embargo, de la comparación sistemática emergen limitaciones claras en relación con el objetivo de aplicar estas técnicas fuera del dominio clásico de las redes de comunicaciones. Muchas soluciones se han validado únicamente en topologías jerárquicas homogéneas o en simulación, varias requieren modificaciones del *dataplane* o soporte hardware específico, y pocas consideran nativamente la heterogeneidad de enlaces, la restricción de recursos de nodos IIoT o la coordinación multi-raíz entre controladores distribuidos.

Esto encaja de forma directa con el planteamiento y los dos objetivos de la Tesis: (i) el estudio y diseño de algoritmos y mecanismos de control (etiquetado, encaminamiento, reconfiguración proactiva) aplicables a SDN, IIoT y redes de distribución eléctrica; y (ii) el análisis de la infraestructura habilitadora (herramientas de despliegue y monitorización), incorporando técnicas de AI/ML como elemento auxiliar para predicción y toma de decisiones autónoma. En concreto, cada *gap* identificado se traduce en uno o

varios posibles trabajos dentro de la Tesis: diseño de esquemas de etiquetado tolerantes a heterogeneidad, desarrollo de una capa de compatibilidad *dataplane*-agnóstica, protocolos de *bootstrap* ligeros, algoritmos de coordinación multi-raíz, y estudios que reduzcan la señalización en nodos con recursos limitados mediante decisiones locales haciendo uso del etiquetado jerárquico (posiblemente asistidas por técnicas de AI/ML). Estas líneas se integran de manera coherente con los objetivos generales de la Tesis y marcan un camino claro desde la revisión del estado del arte hacia contribuciones concretas, evaluables y orientadas a la transferencia tecnológica en contextos reales (IIoT, smart grids y otros entornos densos y heterogéneos).

### **2.2.2. Servicios avanzados: gestión y planificación de recursos en entornos densos y heterogéneos**

En esta subsección se revisan las capacidades avanzadas que, apoyadas en el MCC y en las capas de orquestación, habilitan la gestión inteligible y la planificación de recursos en redes softwarizadas densas y heterogéneas. Estas capacidades van más allá de las funciones básicas de descubrimiento y provisión: incluyen la orquestación multi-capas (*cloud / edge*), la planificación de recursos con restricciones (computación, energía, latencia), mecanismos de autoscalado y balanceo, y técnicas de apoyo basadas en AI/ML para la predicción y la toma de decisiones proactiva.

Dentro de los servicios avanzados en el contexto del MCC, nos centraremos en la gestión y planificación de recursos en entornos densos y heterogéneos. Tradicionalmente, la planificación de recursos en redes de comunicaciones se ha focalizado en parámetros como la capacidad de computación, el consumo energético, el ancho de banda o incluso la latencia. Sin embargo, en escenarios heterogéneos este enfoque requiere una mayor abstracción, ya que el concepto de recurso depende fuertemente del dominio de aplicación. Este problema ha sido ampliamente estudiado en entornos de computación en el borde (*edge/fog computing*), donde los nodos colaboran de manera distribuida para procesar tareas con diferentes requisitos. En este contexto, los enfoques de gestión suelen estructurarse en cinco fases [74]: estimación de la carga por nodo, descubrimiento de nuevos nodos, monitoreo del estado de los mismos, orquestación global y asignación de tareas.

Si se traslada este esquema a otros dominios, la definición de recurso varía sustancialmente. En redes de distribución de energía, por ejemplo, el recurso fundamental es la potencia que cada nodo puede inyectar o demandar de la red, lo que obliga a replantear la asignación bajo restricciones de estabilidad y balance energético. En sistemas de logística, los recursos se asocian a la capacidad de transporte, las rutas disponibles o los tiempos de entrega, por lo que la planificación debe integrar criterios de optimización espacial y temporal. En entornos industriales (IIoT), los recursos incluyen tanto la capacidad de procesamiento en máquinas de control como el acceso a sensores y actuadores, condicionados por restricciones de latencia crítica y fiabilidad. En redes vehiculares, los recursos están ligados al ancho de banda disponible en entornos altamente dinámicos y al poder de cómputo en los vehículos para tareas de seguridad o navegación cooperativa. Otro campo de aplicación podría verse en redes de vehículos aéreos no tripulados (del inglés, Unman-

ned Aerial Vehicle (UAV)), el recurso más crítico suele ser la energía (limitada por las baterías), además de la conectividad intermitente en entornos de alta movilidad.

Este abanico de escenarios evidencia que, aunque las fases de planificación presentan similitudes en términos de descubrimiento, monitoreo, orquestación y asignación, la abstracción de los recursos debe adaptarse a las particularidades de cada dominio, lo que representa uno de los principales retos en la provisión de servicios avanzados sobre entornos densos y heterogéneos. En los distintos dominios analizados emergen dos grandes familias de enfoques para la gestión de recursos: los esquemas centralizados y los esquemas distribuidos (con variantes híbridas intermedias). Los modelos centralizados explotan una visión global de la infraestructura para optimizar asignaciones y políticas desde un punto de control único, lo que facilita la toma de decisiones óptimas pero puede aumentar la latencia de control, crear cuellos de botella y presentar retos de escalabilidad y tolerancia a fallos. Por el contrario, los enfoques distribuidos delegan decisiones en agentes locales, mejorando la escalabilidad, la reactividad y la resiliencia frente a fallos o particiones de la red, a costa de una menor visibilidad global y de desafíos adicionales en la coherencia y coordinación. En la práctica, las soluciones híbridas intentan aprovechar lo mejor de ambos mundos combinando coordinación global con decisiones locales autónomas. A continuación se presenta una sección dedicada a desglosar en detalle las ventajas, limitaciones y criterios de aplicabilidad de cada enfoque en el contexto de redes densas y heterogéneas.

### 2.2.2.1. Terminología básica

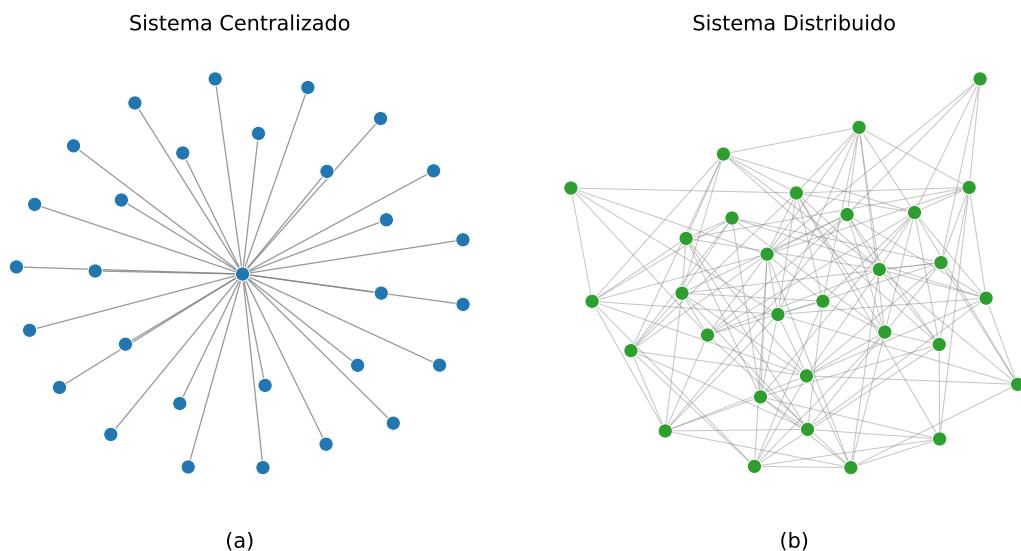
En lo relativo a los procesos de gestión de recursos, éstos pueden abordarse mediante esquemas centralizados o distribuidos (véase Figura 2.9). En un enfoque distribuido, la toma de decisiones se reparte entre los dispositivos de la red: los nodos intercambian información con sus vecinos y resuelven localmente parte del problema de asignación o reconfiguración de recursos. Por el contrario, en un esquema centralizado la responsabilidad recae en un único elemento (o un pequeño conjunto de elementos coordinados) que, con una visión global de la topología y el estado, calcula y ordena las asignaciones de recursos. Ambos modelos presentan ventajas e inconvenientes que determinan su idoneidad según las características del escenario de aplicación.

Los sistemas centralizados (Figura 2.9(a)) destacan por su simplicidad conceptual y de despliegue: la lógica de control es monolítica, lo que facilita el desarrollo, la depuración y la verificación algorítmica. Al disponer de visión global, los controladores centralizados pueden ejecutar optimizaciones más cercanas al óptimo teórico y aplicar políticas de asignación complejas con menores requisitos de coordinación inter-nodo. No obstante, este modelo sufre problemas de escalabilidad y de tolerancia a fallos: concentrar el cómputo y el estado en un único punto incrementa la carga sobre ese componente conforme crece la red, puede introducir cuellos de botella en tiempo de cómputo y en señalización, y genera un único punto de fallo que afecta a la disponibilidad del servicio.

Los esquemas distribuidos (Figura 2.9(b)) ofrecen mayor resiliencia y escalabilidad horizontal, ya que la carga de cómputo se reparte entre múltiples agentes; la caída de un

nodo no suele comprometer la operación global si existen mecanismos locales de recuperación. Además, la proximidad de la toma de decisiones a los recursos suele reducir la latencia de reacción y la necesidad de transmitir grandes volúmenes de telemetría hacia un control central. Como contrapartida, los métodos distribuidos afrontan retos en coherencia y convergencia: al disponer de visiones parciales, requieren protocolos de coordinación que pueden aumentar la latencia de convergencia y la sobrecarga de mensajes. Asimismo, la complejidad de diseñar algoritmos distribuidos correctos y eficientes (evitando bucles, inestabilidades o decisiones contraproducentes) es mayor que en el caso centralizado.

En la práctica, muchas soluciones adoptan enfoques híbridos que combinan control centralizado para políticas de alta capa y planificación global con decisiones locales o jerárquicas para la ejecución rápida y la tolerancia a fallos. Este compromiso permite aprovechar la visibilidad y la optimización del plano central sin renunciar a la reactividad y robustez del plano distribuido; sin embargo, exige diseñar mecanismos de particionado de funciones, consistencia eventual y políticas de *fall-back* que garanticen coherencia y rendimiento en presencia de heterogeneidad y restricciones de recursos.



**Figura 2.9:** Ejemplo de un sistema centralizado y un sistema distribuido.

Para abordar con mayor profundidad el reto de la gestión y planificación de recursos en redes densas y heterogéneas, en la siguiente sección se recopila y analiza las propuestas más relevantes en distintos dominios de aplicación. Para cada trabajo se identificará el enfoque de control adoptado (centralizado, distribuido o híbrido), se expondrán las motivaciones y ventajas que justifican dicha elección, y se destacarán las limitaciones detectadas en relación con escalabilidad, latencia, coste de señalización y requisitos de hardware en cada uno de ellos. El objetivo es extraer lecciones transferibles entre dominios y establecer criterios de diseño que orienten las contribuciones propuestas en esta Tesis.

### 2.2.2.2. Propuestas de enfoques para la gestión y planeamiento de recursos

En el marco de los servicios avanzados del MCC, la gestión y el planeamiento de recursos en entornos densos y heterogéneos ha dado lugar a una familia diversa de soluciones. Aunque la gestión de recursos ha sido ampliamente explorada en el campo del *edge/fog computing* [74], las implementaciones concretas en entornos más heterogéneos varían según el dominio de aplicación (centros de datos, IIoT, redes de distribución eléctrica, logística, redes vehiculares, UAVs, etc.), compartiendo objetivos comunes: optimizar la asignación de recursos (cálculo, energía, ancho de banda, latencia), garantizar la disponibilidad y la resiliencia, y minimizar el coste de señalización y la latencia de decisión. A continuación se describen las líneas principales de enfoque, si emplean un enfoque centralizado o distribuido, sus capacidades y las limitaciones más relevantes observadas en la literatura.

En el ámbito del *edge/fog computing*, Ali *et al.* [75] proponen EF DOT, una técnica de *offloading* orientada a las redes Mobile Edge Computing (MEC) basada en aprendizaje profundo (del inglés, Deep learning (DL)) que estima una función de coste (energía, retraso, recursos radio y capacidad de cálculo) y entrena una red neuronal para tomar decisiones rápidas de particionado y *offloading*. El enfoque opera de forma esencialmente centralizada a nivel del dispositivo móvil: cada dispositivo aplica localmente el modelo ya entrenado para decidir su política de *offloading* sin coordinación distribuida entre múltiples dispositivos o servidores. Esta aproximación muestra ventajas claras en latencia de decisión y eficiencia por dispositivo, pero evidencia limitaciones importantes en escenarios multi-usuario/multi-servidor, en la coordinación y en la generalización del modelo en entornos heterogéneos, lo que limita su aplicabilidad directa a redes densas con topologías rápidas y cambiantes.

Kaneva *et al.* [76] introducen un enfoque híbrido para *fronthaul offloading* en redes *fog* multi-salto: emplean Q-learning (algoritmo de aprendizaje por refuerzo basado en la idea del aprendizaje por ensayo y error) en cada nodo para descubrir rutas energéticamente eficientes, pero requieren una entidad central encargada de coordinar transmisiones y resolver colisiones. Este diseño ilustra una tensión recurrente en la literatura: la combinación de aprendizaje local (que aporta adaptabilidad) con control centralizado (que aporta garantía de QoS) mejora el rendimiento inmediato, pero sacrifica escalabilidad y robustez frente a fallos del plano central. Tong *et al.* [77], en un contexto UAV–*fog* para sistemas de transporte inteligentes, siguen la senda del control global mediante formulación Mixed Integer Nonlinear Programming (MINLP) y descomposición matemática; el resultado es una política de gran calidad, pero con pobres propiedades de escalabilidad y adaptabilidad en entornos altamente dinámicos. Frente a estas propuestas centralizadas o híbridas, Zhao *et al.* [78] muestran la promesa de enfoques distribuidos apoyados en Graph Neural Networks (GNN): su esquema *congestion-aware* permite decisiones locales informadas con menor señalización, mejor tolerancia a congestión en escenarios multi-salto y sin dependencia de un único planificador. No obstante, el entrenamiento, la eficiencia energética del modelo y la validación en entornos reales siguen siendo retos abiertos. En conjunto, los trabajos del dominio *edge/fog computing* revelan la necesidad de métodos que combinen: (i) toma de decisión local con garantías, (ii) esquemas de coordinación ligera para evitar

congestión y colisiones, y (iii) modelos de ML entrenables de forma práctica en entornos heterogéneos (por ejemplo, aprendizaje federado (del inglés, Federated learning (FL)) o modelos GNN que sean adaptativos).

En micro-redes y distribución eléctrica, Rodríguez *et al.* [79] presentan SGR-OSPF, una reconfiguración distribuida basada en agentes en subestaciones secundarias, que mezcla el mundo de las redes de distribución eléctrica, con las redes de comunicación. El enfoque distribuido mejora tiempos de respuesta y tolerancia a fallos frente a soluciones centralizadas, pero lo hace a costa de mayor ancho de banda y carga computacional, presentando además complejidad algorítmica que puede comprometer la escalabilidad a gran escala. Tenti *et al.* [80] proponen un paradigma híbrido para comunidades energéticas que mezcla control local con supervisión central (almacenamiento comunitario). Esta arquitectura permite reacondicionar sobre infraestructuras existentes y ofrece mejoras operacionales, aunque depende de recursos (p. ej. almacenamiento compartido) y su validación en despliegues reales es limitada. En este dominio se percibe un patrón claro: la prioridad es la latencia y la resiliencia local (favoreciendo soluciones distribuidas), pero hace falta optimizar la eficiencia en la comunicación, integrar medidas de seguridad y demostrar soluciones en despliegues reales con heterogeneidad de enlaces y dispositivos.

En logística urbana y reparto, Chen *et al.* [81] utilizan modelos centralizados, basados en construcción de grafos espacio-temporales y resolución de flujos máximos, para estimar capacidad de entrega urbana de los servicios de taxi; de manera análoga, Moreno-Saavedra *et al.* [82] proponen un marco multi-algorítmico centralizado para balanceo de carga operacional. Estos métodos ofrecen diagnósticos y soluciones de alta calidad desde la visión global, pero presentan limitaciones prácticas frente a variabilidad en tiempo real, demandas estocásticas y la ausencia de integración con bloques de encaminamiento activos. En logística el punto más débil es la falta de esquemas jerárquicos o distribuidos que permitan reaccionar en tiempo real y escalar sin requerir una visión global constante del escenario.

Relacionando los dominios, emergen brechas transversales que en esta Tesis se pretende atender. En primer lugar, la coordinación escalable: falta de esquemas de coordinación multi-agente que funcionen con heterogeneidad de enlaces y recursos sin generar un *overhead* de señalización prohibitivo. En segundo lugar, el coste de señalización y cómputo en nodos con recursos limitados: muchas propuestas dependen de telemetría o cómputo intensivo que no son viables en IIoT o sensores de baja capacidad. En tercer lugar, la integración práctica de técnicas de AI/ML: hay trabajos prometedores (Q-learning, GNN) pero escasa investigación sobre entrenamiento distribuido, modelos ligeros y robustez/explorabilidad en entornos reales. En cuarto lugar, la validación en testbeds más densos o despliegues reales, todavía insuficiente en la mayoría de las propuestas.

#### 2.2.2.3. Conclusiones y alineación con los objetivos de la Tesis

El análisis crítico de la literatura relativo a intercambio y gestión de recursos revela un conjunto coherente de hallazgos que justifican y orientan los objetivos planteados en esta Tesis. En términos generales, existen avances teóricos y prototipos convincentes en

## 2.2 SERVICIOS Y TECNOLOGÍAS HABILITANTES EN REDES SOFTWAREIZADAS Y HETEROGRÉNEAS

---

cada una de estas áreas, pero la mayoría de las propuestas sufren de limitaciones que impiden su adopción directa en entornos reales densos y heterogéneos (p. ej. IIoT, micro-redes o logística urbana): dependencia de sistemas centralizados, fuerte acoplamiento a plataformas concretas, requisitos de cómputo y señalización elevados, escasa atención a la seguridad en fases iniciales y validaciones limitadas en infraestructuras reales.

Una observación recurrente es la tensión entre enfoques centralizados y distribuidos para la gestión y la planificación de recursos. Los esquemas centralizados facilitan el diseño algorítmico, garantizan una visión global del sistema y permiten optimizaciones de alto rendimiento a partir de un conocimiento completo de la topología y la demanda; sin embargo, presentan problemas de escalabilidad, punto único de fallo y costes de señalización elevados, siendo limitaciones críticas en redes con nodos de recursos reducidos. Por el contrario, las propuestas distribuidas mejoran la resiliencia, reducen la dependencia de un único ente y escalan mejor en número de nodos, pero suelen pagar ese beneficio con mayor complejidad algorítmica, tiempos de convergencia superiores y una visión incompleta que dificulta optimizaciones globales.

La gestión y planificación de recursos se encuentra estrechamente ligada al encaminamiento y, en particular, al uso de esquemas de etiquetado jerárquico. El etiquetado jerárquico y la construcción de árboles enraizados facilitan la creación de topologías lógicas simples y eficientes sobre grafos hiperconectados: permiten reducir tablas de reenvío, incluso, habilitar encaminamiento sin tablas de reenvío y construir rutas de control con coste controlado. Por tanto, el etiquetado no es solo una técnica de direccionamiento: actúa como palanca para reducir señalización, simplificar el arranque de la red y habilitar mecanismos locales de toma de decisiones que, a su vez, condicionan la viabilidad de enfoques distribuidos para la gestión de recursos. En síntesis, encaminamiento, etiquetado y gestión de recursos constituyen un triángulo técnico donde la mejora en uno de los vértices influye directamente en los otros dos.

Desde la perspectiva de los dominios estudiados, emergen diferencias claras en las prioridades de diseño. En IIoT y entornos con nodos con capacidades limitadas, la minimización de señalización y la compatibilidad con *dataplanes legacy* son requisitos críticos; por ello conviene priorizar esquemas *in-band*, etiquetado simple y decisiones locales asistidas por modelos ligeros. En micro-redes eléctricas, la latencia y la resiliencia local son prioritarias, junto con garantías de seguridad en la fase de arranque y en la reconfiguración; aquí triunfan los diseños híbridos que combinan control local rápido con supervisión central para coordinación inter-nodos. En entornos de *edge/fog* o logística, la tendencia es hacia soluciones mixtas que combinen optimización centralizada con mecanismos distribuidos para la reacción en tiempo real. El papel de AI/ML en estas soluciones es prometedor pero, hoy por hoy, aún inmaduro para la adopción masiva: los trabajos existentes muestran mejoras en toma de decisiones y predicción, pero rara vez abordan el entrenamiento distribuido, la eficiencia energética, la explicabilidad o la robustez frente en entornos heterogéneos.

A la vista de estos hallazgos, la Tesis se alinea de forma natural con dos líneas de trabajo complementarias: (i) avanzar en mecanismos de control *in-band* y esquemas de etiquetado

jerárquico que sean agnósticos a la heterogeneidad de enlace, requieran modificaciones mínimas del *dataplane*; y (ii) diseñar estrategias de gestión y orquestación de recursos que combinen decisiones locales (para ahorro de señalización y latencia) con una coordinación jerárquica ligera (para optimización global), incorporando técnicas de AI/ML adaptadas al perfil de recursos del dominio. Metodológicamente, esto implica validar propuestas tanto en análisis teórico (complejidad, convergencia) como en prototipos y experimentos en topologías representativas de IIoT, micro-redes y escenarios lo suficientemente densos, de modo que se evalúen simultáneamente propiedades formales y viabilidad operativa en un entorno real.

### **2.2.3. Servicios avanzados: optimización y reconfiguración proactiva en entornos densos y heterogéneos**

Dentro del marco del MCC, los servicios avanzados constituyen la capa encargada de transformar capacidades básicas (arranque, descubrimiento, encaminamiento, telemetría) en funciones de alto valor: gestión y planificación de recursos, optimización, y toma de decisiones automáticas. En este contexto, la optimización y la reconfiguración proactiva emergen como dos servicios clave para conseguir redes más eficientes, resilientes y capaces de adaptarse a condiciones cambiantes. Estas funciones no actúan de forma aislada; se apoyan en las vistas lógicas y los recursos que expone el controlador SDN (A-CPI / D-CPI) que consumen datos recogidos por los servicios básicos del MCC para generar decisiones de alto nivel que se materializan en órdenes sobre la infraestructura.

Desde el punto de vista técnico, existen dos grandes familias de técnicas empleadas en los servicios avanzados de optimización y reconfiguración proactiva. Por un lado, métodos de optimización clásicos y metaheurísticos (programación matemática MINLP, Particle Swarm Optimization (PSO), algoritmos genéticos, temple simulado, etc.) permiten formular objetivos concretos (minimizar pérdidas energéticas, reducir latencia, maximizar utilización de recursos, equilibrar carga) y ofrecer soluciones con garantías teóricas o empíricas. Por otro lado, las técnicas basadas en AI/ML (aprendizaje supervisado para predicción de fallos en la red, Reinforcement learning (RL) para políticas de control, GNN para optimizar el control en topologías, aprendizaje federado (FL) para la distribución de entrenamiento) ofrecen capacidad de adaptación y predicción proactiva que resulta muy valiosa en entornos dinámicos. En la práctica, las soluciones más prometedoras combinan ambos enfoques: modelos predictivos que alimentan optimizadores o políticas entrenadas por RL que se encargan de la reconfiguración en tiempo real de la red.

Los casos de uso en los que estas capacidades muestran mayor impacto son especialmente relevantes para esta Tesis: las redes de distribución eléctrica (SG) y las redes de sensores/IIoT. También lo son en las redes de comunicaciones convencionales, incluso en las redes SDN, sin embargo, ya se han explorado en apartados anteriores de la Tesis algunas de estas propuestas, por lo que nos centraremos en otros casos de uso. En las SG, la optimización puede perseguir objetivos como minimizar pérdidas, balancear flujos entre micro-redes, perseguir el equilibrio del balance global de potencia, incluso, optimizar OpEx de la red. La reconfiguración proactiva basada en predicción de fallos o en predicción de

producción renovable permite reorganizar las rutas de energía y activar rutas alternativas antes de que ocurra un fallo en la red. Los fallos en las redes de distribución de energía son críticos, según se ha podido experimentar en España recientemente (*Blackout* de la Península Ibérica 2025 [83]). La característica jerárquica de la red unido a la necesidad del equilibrio constante de la misma, sumado a todos los sistemas y sectores que dependen de la energía para funcionar, hacen que proactividad en la predicción de errores sea de especial interés. En IIoT y redes de sensores densas, los objetivos cambian hacia la conservación energética, la minimización de señalización, la garantía de latencia para flujos críticos y la tolerancia a fallos de nodos con recursos limitados. Aquí, la combinación de la optimización del diseño de la arquitectura desplegada junto a decisiones locales asistidas por modelos livianos de ML es particularmente atractiva.

No obstante, el diseño e implementación de estos servicios avanzados afronta retos prácticos importantes: heterogeneidad de enlaces y dispositivos, restricciones estrictas de cómputo y energía en nodos finales, necesidad de decisiones en tiempo real, falta de datos etiquetados para entrenar modelos. Además, surge el dilema arquitectónico entre centralización (mejor optimización global pero mayor latencia y señalización) y descentralización (más resiliencia y menor sobrecarga, pero peor óptimo global). Por ello, se introduce la sub-Sección 2.2.3.1, donde se va a explorar las bases de los modelos más utilizados para la optimización y la reconfiguración de la red, y posteriormente se van a explorar las principales propuestas de la literatura, tanto en el dominio de las SG, como en las redes IIoT.

#### 2.2.3.1. Terminología básica

La adopción de métodos basados en AI, en algoritmos heurísticos/metaheurísticos y optimizadores se ha convertido en un pilar para la optimización y la reconfiguración de redes densas y heterogéneas.

Históricamente, el concepto de AI se remonta a tiempos antiguos, cuando mitos y leyendas imaginaban seres artificiales dotados de conciencia; el desarrollo del pensamiento lógico y del razonamiento formal a lo largo de la historia sentó las bases teóricas que condujeron, en la década de 1940, a la invención del ordenador digital programable (Las máquinas inteligentes de Norbert Wiener y John von Neumann [84]), un dispositivo nacido de razonamientos matemáticos abstractos que abrió la posibilidad práctica de construir un “cerebro electrónico”. Figuras claves como Alan Turing [85] plantearon ya en ese periodo preguntas fundamentales sobre si las máquinas podrían pensar; el propio término *inteligencia artificial* fue acuñado en 1956 por John McCarthy en un *workshop* de Dartmouth [86], evento fundacional que agrupó a quienes serían las referencias del campo durante décadas. Aquel optimismo inicial, reforzado por importantes aportaciones públicas en financiación, anticipó que máquinas con capacidades comparables a las humanas aparecerían en una generación; desde entonces, y especialmente desde la segunda mitad del siglo XX, la AI ha evolucionado hasta producir máquinas capaces de aprender y aplicar ese aprendizaje en dominios que hasta entonces se consideraban exclusivamente humanos. A día de hoy la AI se encuentra presente en todos los sectores y dominios de nuestra sociedad, con

especial atención a la llegada de la AI generativa basada en *transformers*, y su “boom” de los modelos Generative pre-trained Transformers (GPTs), que han transformado el plano social, académico e industrial de nuestra sociedad.

En la práctica, la AI se clasifica principalmente a través de tres grandes familias: aprendizaje automático (ML), aprendizaje profundo (DL) y aprendizaje por refuerzo (RL). El ML incluye modelos que extraen patrones a partir de datos para realizar predicciones o decisiones y se subdivide en aprendizaje supervisado (p. ej. regresión lineal, SVM, árboles de decisión, Random Forest, XGBoost), no supervisado (clustering, reducción de dimensiones) y por refuerzo (Q-learning, entre otros). El DL emplea redes neuronales profundas para representar funciones complejas: arquitecturas basadas en la convolución como Convolutional Neural Network (CNN) son útiles para trabajar con señales o imágenes, las Recurrent Neural Networks (RNN) que trabajan de forma secuencial con datos son interesantes para series temporales y, de especial interés para redes, las GNN para datos con estructura topológica. El RL y sus variantes son apropiados cuando la decisión debe optimizarse secuencialmente en entornos estocásticos, por ejemplo para políticas de reconfiguración en tiempo real. A un nivel estructural, las diferencias entre las técnicas de ML y las de DL radican principalmente en el manejo de las características o de los datos. En un enfoque basado en ML, se tendrán que manejar un gran volumen de datos, previo diseño y selección de características, mientras que los enfoques basados en DL suelen ser los propios modelos los que tratan y se auto-ajustan de forma interna.

Muchos problemas se plantean como optimización matemática (LP/MILP, problemas no lineales, MINLP). Estas formulaciones permiten obtener soluciones globalmente óptimas con solvers comerciales [87] (como por ejemplo, CPLEX, Gurobi), pero en instancias grandes son computacionalmente costosas. Para mitigarlo se usan técnicas de descomposición [88] (p. ej. dual decomposition, ADMM) que facilitan aproximaciones distribuidas o híbridas, permitiendo partir el problema en subproblemas manejables y adecuarlos a arquitecturas centralizadas o distribuidas según restricción de latencia y recursos.

Cuando la optimización global de un problema no es viable en un tiempo real o sobre un hardware con recursos limitados, los heurísticos y metaheurísticos cobran un especial protagonismo. Entre ellos se encuentran la búsqueda local [89] (hill-climbing, tabu-search, simulated annealing), algoritmos evolutivos (Genetic Algorithm (GA)), incluso métodos bioinspirados, que se basan en procesos presentes en la naturaleza para optimizar un problema. Por ejemplo, el método PSO se basa en el movimiento natural de una bandada de pájaros o un banco de peces para moverse forma síncrona para evitar peligros, el método Ant Colony Optimization (ACO) se basa en cómo las hormigas empleando sus feromonas son capaces de encontrar el camino más corto entre una fuente de alimento y el hormiguero, incluso podemos ver métodos basados en la reproducción y supervivencia dentro de un arrecife de coral, Coral Reefs Optimization (CRO). Estos métodos son robustos para problemas combinatorios y pueden adaptarse a operación distribuida, aunque requieren ajuste de hiperparámetros y no garantizan optimización global; su ventaja práctica reside en ofrecer soluciones lo suficientemente buenas con tiempos de cómputo acotados. A continuación, se presenta la Figura 2.10 que resume todos los enfoques descritos.

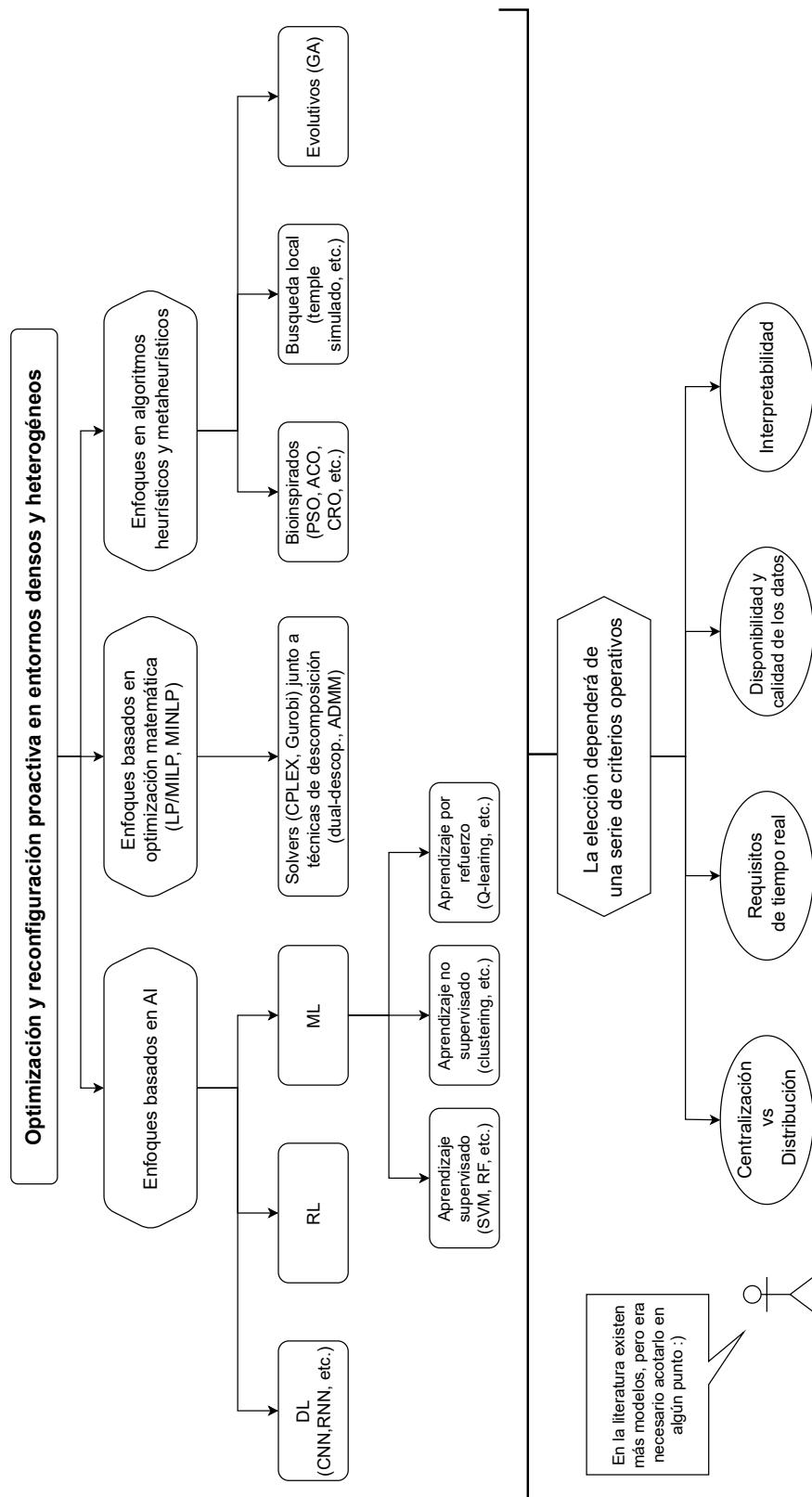


Figura 2.10: Resumen de los principales enfoques para la optimización y reconfiguración proactiva de la red.

La elección entre las familias de técnicas suele pivotar sobre una serie de criterios operativos: centralización contra la distribución, requisitos de tiempo real, disponibilidad y calidad de datos, y necesidad de interpretabilidad. Los *solvers* matemáticos y modelos DL exigen visibilidad global y recursos computacionales, siendo adecuados para un enfoque centralizado que busque optimización global. En cambio, heurísticos ligeros, RL distribuido con inferencia local son preferibles en despliegues con latencia estricta y nodos con recursos limitados (p. ej. IIoT). Además, en entornos críticos (redes eléctricas) la trazabilidad favorece métodos explicables o esquemas híbridos que incluyan mecanismos deterministas de *fallback* para garantizar seguridad de la red. Una tendencia práctica y fructífera es la hibridación, combinar optimización clásica con aprendizaje. Estas aproximaciones permiten conservar garantías estructurales mientras se mejora la eficiencia en tiempo de ejecución y la capacidad de adaptación a entornos dinámicos en aras de optimizar un problema, el cual, en este caso sería la optimización y reconfiguración de la red.

A continuación, se introduce la sub-Sección 2.2.3.2 donde se revisan los trabajos más significativos de optimización y reconfiguración en entornos densos y heterogéneos, como pueden ser las redes de sensores/IIoT, o redes de distribución eléctrica.

### 2.2.3.2. Propuestas de optimización y reconfiguración proactiva de la red

En el marco de los servicios avanzados del MCC, la optimización y la reconfiguración proactiva de redes densas y heterogéneas constituyen un área de investigación activa y muy variopinta. Los trabajos existentes adoptan modelos y objetivos muy diversos (optimización del QoS, minimización del consumo energético, maximización del intercambio de recursos como capacidad de cómputo o energía, mejora de la resiliencia ante fallos, etc.), por lo que la elección de la metodología adecuada depende del caso de uso final y de las restricciones operativas definidas en la Sección 2.2.3.1. Para acotar el análisis en esta Tesis nos centramos principalmente en dos dominios representativos: las redes de sensores/IIoT y las redes inteligentes de distribución eléctrica (SG), ámbitos donde confluyen limitaciones de recursos, tiempo, topologías jerárquicas y requisitos de disponibilidad que condicionan fuertemente el diseño de las soluciones.

Comenzando con las redes inteligentes de distribución eléctrica (SG), la optimización y reconfiguración en estas redes se puede identificar como un problema complejo de optimización discreta que puede abordarse mediante métodos diversos: *solvers* de optimización exacta, algoritmos metaheurísticos y, más recientemente, modelos de ML/DL [90, 91, 92]. La elección del enfoque depende del tamaño y la complejidad de la red, así como de los requisitos operativos (latencia de decisión, disponibilidad de datos, capacidad computacional). En la literatura sobre reconfiguración se distinguen, de forma general, dos líneas principales: por un lado, trabajos que definen funciones objetivo orientadas a optimizar el rendimiento interno de la red (minimización de pérdidas, balance de cargas, reducción de costes); por otro lado, estudios centrados en la mejora de la resiliencia y la tolerancia a fallos mediante estrategias de reconfiguración proactiva de la red y redistribución de generación/consumo.

Entrando en ejemplos representativos, se tiene que volver a mencionar Rodriguez *et al.* [79], que proponen un enfoque distribuido de reconfiguración basado en la adaptación del protocolo Open Shortest Path First (OSPF) para detección de fallos y minimización de pérdidas en distribución. Su solución, implementada como un sistema multiagente en subestaciones secundarias y validada sobre la topología de ejemplo del Institute of Electrical and Electronics Engineers (IEEE) 123 Node Test Feeder, muestra la viabilidad del replanteamiento distribuido en entornos reales; sin embargo, el coste de señalización y la complejidad de coordinación entre agentes son puntos a optimizar. En el ámbito de microredes aisladas (Islanded Microgrids (IMGs)), Hemmatpour *et al.* [93] emplean un Harmony Search (algoritmo metaheurístico basado en la música) de forma adaptativa y con multi-objetivo, para mejorar la estabilidad de tensión y la capacidad de carga en topologías de ejemplo IEEE 33-bus y 69-bus, logrando reducción de pérdidas y mayor intercambio de cargas entre los nodos de la topología (lo cual es clave en IMGs, tender hacia un autoabastecimiento); la principal limitación es la naturaleza heurística del método y su sensibilidad a la parametrización. Por su parte, Sun *et al.* [94] plantean una estrategia de autoreparación para el proceso de aislamiento en microgrids modelando el problema con un optimizador matemático que mejora redistribución de generación y las perdidas de carga; la validación en topología de ejemplo IEEE de 9-bus señala la eficacia del esquema para escenarios reducidos, pero también evidencia la dificultad de escalar a sistemas mayores. En el contexto de transmisión de corriente continua de alto voltaje (del inglés, High-Voltage Direct Current (HVDC)) y parques eólicos, Sanz *et al.* [95] emplean PSO para reconfigurar topologías en mallas de tipo HVDC con el objetivo de minimizar pérdidas, mostrando buenas prestaciones en su dominio específico; no obstante, la generalización a topologías mixtas y la robustez frente a fallos múltiples requieren análisis adicional. En conjunto, estos trabajos optimizan parámetros operativos internos, pero a menudo no abordan explícitamente la gestión de fallos a gran escala, ni la validación con datos reales.

En la vertiente orientada a resiliencia y detección/localización de fallos en las SG: la adopción de técnicas de ML/DL ha crecido notablemente. Hosseinzadeh *et al.* [96] combinan k-Nearest Neighbors (KNN) con Principal Component Analysis (PCA) y Linear Discriminant Analysis (LDA) (enfoque ML supervisado) para clasificación eficiente de fallos, obteniendo precisión y robustez aptas para aplicaciones en tiempo real dentro de una SG. Li *et al.* [97] emplean aprendizaje profundo, en concreto una CNN, sobre datos de tensión para localizar fallos en escenarios de baja observabilidad, mostrando mejoras significativas sobre las topologías de ejemplo del IEEE 39-bus y 68-bus. Trabajos similares, como el de Alhanaf *et al.* [98], que también emplean aprendizaje profundo utilizando Artificial Neural Network (ANN) y CNN para extraer características directamente de señales de potencia y gestionar fallos en tiempo real sobre la red (fue evaluado en la topología IEEE 6-bus). Kaplan *et al.* [99] proponen redes Long Short-Term Memory (LSTM) (aprendizaje profundo, se considera un tipo de RNN) para diagnóstico de red en presencia de fuentes renovables, validadas en modelos Simulink, y demuestran mayor capacidad predictiva frente a técnicas clásicas. Finalmente, Ding *et al.* [100] combinan control distribuido de generadores y reconfiguración topológica con optimización mediante CPLEX para restauración de carga post-fallo, probando su enfoque en sistemas de hasta 615 nodos. Aunque estos trabajos

muestran el potencial de ML/DL para detección y respuesta rápida (en algunos casos también de optimizadores matemáticos), predominan las validaciones de simulación en topologías concretas, por lo que los modelos no se encuentran generalizados; la falta de conjuntos de datos reales y la adaptación al cambio topológico o a la heterogeneidad de los enlaces son retos aún abiertos.

En síntesis, la literatura sobre reconfiguración y resiliencia en SG ofrece una amplia batería de técnicas y enfoques, desde optimizadores exactos y heurísticos hasta soluciones basadas en AI/ML, que aportan soluciones valiosas según el caso de uso. Sin embargo, persisten carencias relevantes para la aplicabilidad en entornos densos y heterogéneos: la escalabilidad temporal de métodos centralizados, la eficiencia de señalización en enfoques distribuidos, la necesidad de modelos agnósticos, es decir, que no se encuentren pre-entrenados únicamente para una topología, y la validación con datos y testbeds reales.

En el ámbito de soluciones aplicadas redes de sensores junto a entornos industriales IIoT, Bonada *et al.* [101] revisan el uso de técnicas de AI y ML para mejorar la monitorización y la optimización de procesos dentro del paradigma Industria 4.0. El trabajo recoge casos reales y proyectos de I+D que muestran el potencial de técnicas predictivas (mantenimiento predictivo, sensores virtuales, algoritmos para predicción de calidad) y anticipa avances en RL, DL para visión y sistemas colaborativos humano-AI. Aunque exhaustivo en propuestas y casos de uso, el estudio es más descriptivo que experimental y no siempre aporta evaluaciones cuantitativas comparativas en entornos heterogéneos. Mezair *et al.* [102] proponen un marco avanzado de DL para diagnóstico de fallos en entornos industriales habilitados con 6G. Su arquitectura combina LSTM, CNN y GNN para integrar datos heterogéneos (imágenes, vídeo, series temporales y grafos) en una salida única de diagnóstico. Además, introducen una estrategia de ramificación y límite para la búsqueda eficiente del espacio de hiperparámetros, mejorando la eficiencia de entrenamiento. Los resultados muestran mejoras respecto a técnicas de referencia en tasa de detección, tiempo de ejecución y consumo energético; no obstante, la validación pragmática en despliegues reales y la reproducibilidad del preprocesado multiformato requieren un desarrollo adicional para su adopción industrial. De forma similar, otro trabajo que emplea aprendizaje profundo es, Aminabadi *et al.* [103], donde presentan un sistema de control totalmente automático compatible con la Industria 4.0. Integran medidas en línea, análisis en tiempo real y control AI apoyado en modelos DL (p. ej. ResNet-18) para evaluar calidad superficial y predecir características. El control se realiza mediante una aproximación heurística, que ajusta los parámetros de la máquina en producción. Los experimentos demuestran control efectivo de calidad, aunque la generalización a procesos multiobjetivo y su robustez frente a variabilidad de materias primas quedan abiertos como líneas futuras.

En conjunto, estos trabajos ilustran el potencial de la AI/ML para mejorar la monitorización, el control y el mantenimiento en entornos IIoT e Industria 4.0: desde modelos DL para diagnóstico de fallos hasta arquitecturas para despliegue a gran escala con un control automático. No obstante, persisten retos comunes: (i) la integración y sincronización de datos heterogéneos en tiempo real, (ii) la eficiencia energética y la latencia en dispositivos con recursos limitados, y (iii) la validación con datos reales y en producción.

### 2.2.3.3. Conclusiones y alineación con los objetivos de la Tesis

El análisis de la literatura en servicios avanzados de optimización y reconfiguración proactiva ha mostrado que, tanto en SG como en redes densas de sensores (IIoT), existen soluciones maduras en términos metodológicos (optimizadores exactos, metaheurísticas, y modelos AI/ML) que aportan mejoras claras en eficiencia, pérdidas de potencia, detección de fallos y rapidez de recuperación. No obstante, al trasladar estas soluciones a entornos densos y heterogéneos emergen limitaciones prácticas: en SG persisten problemas de escalabilidad temporal de los enfoques centralizados, exceso de señalización en soluciones distribuidas, falta de modelos agnósticos (entrenados para una topología concreta) y escasa validación con datos y testbeds reales; en IIoT se repiten retos análogos añadidos a la necesidad de integrar y sincronizar datos heterogéneos en tiempo real, garantizar eficiencia energética y latencias bajas en nodos con recursos limitados, y validar en despliegues productivos.

Estas lagunas condicionan directamente los objetivos de la Tesis. En primer lugar, la necesidad de protocolos y algoritmos *in-band* y de control que funcionen en redes densas y heterogéneas vincula con el primer bloque de objetivos (estudio y extensión del paradigma SDN a IIoT y SG): se requiere diseñar mecanismos de control y etiquetado jerárquico que permitan arranque, encaminamiento y reconfiguración con baja sobrecarga de señalización, mínima dependencia de cambios en el *dataplane* y soporte multi-raíz con coordinación entre controladores. En segundo lugar, los vacíos detectados en la integración de AI/ML (modelos no agnósticos, necesidad de técnicas de inferencia ligera, falta de datos reales) enlazan con la intención de la Tesis de incorporar herramientas de AI/ML como asistente al control: se debe investigar cómo aplicar modelos robustos (p. ej. para decisiones locales topológicas, y que sean adaptables entre topologías, y técnicas de compresión para ejecución en nodos limitados) sin imponer cargas de comunicación o cómputo imposibles de soportar por los dispositivos finales.

Concretamente, para cerrar los *gaps* se plantea en la Tesis una estrategia coordinada: (i) proponer esquemas híbridos de decisión que combinen optimización centralizada (cuando sea factible) con mecanismos locales ligeros para reducir latencias y señalización; (ii) diseñar protocolos de reconfiguración y optimización que empleen etiquetado jerárquico y rutas enraizadas para disminuir estado y mensajes en la red; (iii) integrar mecanismos de seguridad ligeros en la fase de *bootstrapping*/etiquetado; (iv) desarrollar y evaluar modelos AI/ML agnósticos, y (v) aplicar técnicas para ejecutar inferencia eficiente en el borde (*edge*) de la red preservando privacidad y consumo energético. Estas líneas alinean y materializan los objetivos de investigar mecanismos de control adaptativos y la integración de AI/ML como herramienta auxiliar para decisión proactiva. Finalmente, la validación experimental será un pilar fundamental de la Tesis. Se propondrá métricas claras y un plan de evaluación exhaustivo, para que las soluciones propuestas no se encuentren sesgadas. Se intentará emplear datos reales, junto a simulación y a emulación, y experimentos sobre *benchmarks* relevantes que se han encontrado en la literatura (p. ej. feeders IEEE), con la intención de demostrar viabilidad práctica y facilitar la transferencia a escenarios reales.

## 2.3. Casos de uso en entornos densos y heterogéneos

En este último bloque se revisan los casos de uso más relevantes reportados en la literatura para entornos densos y heterogéneos. Estos ejemplos prácticos muestran cómo las tecnologías habilitantes, y en particular, las redes programables y softwarizadas, se aplican en contextos reales y con requisitos operativos muy distintos, siendo paradigmas representativos las SG y las redes de sensores IIoT.

Aunque en las secciones anteriores hemos analizado ambos dominios desde el punto de vista de los servicios básicos y avanzados del MCC (arranque y provisión de canales de control, descubrimiento, gestión y planificación de recursos, optimización y reconfiguración proactiva de la red), aquí extendemos ese análisis para estudiar con mayor profundidad cómo se materializan las soluciones en cada dominio con un enfoque más general. En el caso de las SG nos centraremos en cómo se realiza el encaminamiento de energía, los esquemas de reconfiguración y las exigencias de resiliencia propias de la distribución eléctrica. Para las redes de sensores IIoT examinaremos las arquitecturas inteligentes propuestas, los criterios operativos (consumo energético, latencia, fiabilidad) y las estrategias de despliegue que condicionan la elección de modelos de control y gestión. El objetivo de esta sección es extraer lecciones prácticas y limitaciones de cada dominio que sirvan para orientar las propuestas de la Tesis.

### 2.3.1. Redes inteligentes de distribución eléctrica

La transición energética global hacia sistemas de energía más sostenibles, resilientes y eficientes ha puesto de manifiesto la necesidad de nuevas arquitecturas en la distribución eléctrica. Este cambio de paradigma está impulsado, sobre todo, por la integración masiva de fuentes renovables (solar, eólica) y por la progresiva descentralización en la generación eléctrica. Estas tendencias han incrementado notablemente la complejidad de los sistemas de potencia modernos, especialmente en lo que respecta a los procesos de planificación, operación y control [104].

En este marco, las SG se perfilan como la solución futura para las redes de transmisión y distribución eléctrica [105, 106], pues su objetivo principal es posibilitar un monitoreo integral de la red que permita equilibrar de forma eficiente la producción y el consumo de energía. Los sistemas SG deben ser capaces de reaccionar de forma rápida y predecible, adaptándose a variaciones en la demanda y en la oferta mediante el control de consumos y de dispositivos de almacenamiento. Las aplicaciones que gestionan una SG requieren, por tanto, de una infraestructura de comunicaciones segura, altamente escalable y con disponibilidad continua, capaz de manejar grandes volúmenes de datos en tiempo real para responder de forma consistente a cambios en el estado de la red [107]. No obstante, estas configuraciones introducen retos importantes en términos de coordinación, protección y control del sistema, ya que los flujos de potencia se vuelven más dinámicos e impredecibles. Abordar estos desafíos exige desplegar inteligencia descentralizada y capacidades de respuesta rápida, lo que motiva la adopción de tecnologías avanzadas como los Energy Routers (ERs).

En este contexto, el concepto de Energy Internet (EI) está ganando relevancia [108]. La EI propone un sistema eléctrico altamente digitalizado, automatizado y bidireccional, que opera bajo principios análogos a los de las redes de comunicaciones convencionales. La visión consiste en una infraestructura integrada donde generadores, consumidores y operadores interactúan de forma dinámica y descentralizada. La materialización de esta visión depende en gran medida del desarrollo y la implantación de tecnologías capaces de monitorizar en tiempo real, encaminar inteligentemente y controlar de forma flexible los flujos de energía en la red. Una tecnología práctica habilitadora fundamental para la EI es el ER [109]. Este dispositivo aporta un enfoque novedoso para la gestión de la energía al permitir un control programable y dinámico de los flujos en redes de distribución eléctrica. Basados en electrónica de potencia de alto rendimiento, los ERs permiten no solo regular variables eléctricas clásicas (tensión, corriente, frecuencia), sino también encaminar selectivamente potencia entre nodos del sistema [110]. Conceptualmente, su funcionalidad es análoga a la de los routers en redes de comunicaciones y supone un cambio de paradigma en la forma de gestionar la electricidad en media y baja tensión. La integración de ERs posibilita el desarrollo de una nueva arquitectura de red conocida como Power Router Grid (PRG) (en el contexto de las SG).

En esta configuración distribuida, los ERs actúan como nodos inteligentes que facilitan la interconexión flexible entre activos energéticos heterogéneos, incluyendo generación convencional y renovable, sistemas de almacenamiento y diversidad de cargas finales. La arquitectura resultante permite optimizar dinámicamente los flujos de potencia en función de la demanda, la oferta, los precios de mercado y las restricciones operativas de la red [111]. Además, habilita paradigmas de control innovadores como el *energy packet switching*, que conceptualiza la electricidad como paquetes discretos susceptibles de priorización, enrutamiento y almacenamiento temporal. Las PRGs presentan múltiples beneficios operativos: reducen pérdidas gracias al enrutamiento optimizado y localizado de la energía; mejoran la resiliencia del sistema al ofrecer caminos alternativos de suministro ante contingencias; y facilitan la integración de generación renovable intermitente (fotovoltaica, eólica) mediante respuestas rápidas y localizadas a la variabilidad de generación y consumo [112]. Asimismo, las PRGs constituyen una plataforma técnica robusta para prestar servicios auxiliares (regulación de tensión, soporte de frecuencia y compensación de potencia reactiva) directamente desde nodos distribuidos, disminuyendo la dependencia de infraestructuras centralizadas.

A continuación, se introduce la sub-Sección 2.3.1.1, donde se revisarán las propuestas más atractivas dentro del contexto del EI y las PRGs/SG.

#### 2.3.1.1. Propuestas en el contexto del EI y las PRGs/SG

El desarrollo de las PRGs y la visión más amplia del EI han promovido una creciente línea de trabajo centrado en tecnologías de encaminamiento de energía y en los mecanismos que las habilitan. A continuación se revisan las contribuciones más relevantes, poniendo el foco en las estrategias algorítmicas, las propuestas arquitectónicas y las implementaciones

prácticas que abordan la coordinación de múltiples ERs y la gestión dinámica de recursos energéticos distribuidos.

En el contexto del EI, Huang *et al.* [113] abordan la gestión energética en microredes mediante agentes instalados en cada ER. Estos agentes emplean mecanismos de consenso distribuido basados únicamente en el intercambio de información entre vecinos inmediatos. Presentan dos algoritmos de consenso: uno para regular el coste incremental de cada generador distribuido y otro para estimar el desajuste global de potencia. Los resultados de simulación respaldan la efectividad de la propuesta en escenarios controlados. Sin embargo, asume comunicaciones entre nodos vecinos fiables y oportunas (lo que no siempre se cumple en entornos heterogéneos), además, la convergencia y el coste de señalización en topologías muy densas no se analiza en profundidad. En conjunto, es una buena base distribuida pero le faltan pruebas en escalas reales y medidas frente a adversarios o fallos múltiples. De forma similar, Wang *et al.* realizan un análisis del EI identificando cinco tecnologías habilitadoras clave: predicción energética, modelos de tarificación, interoperabilidad, generadores distribuidos y evaluación de estabilidad. Posteriormente, sintetizan los retos para su despliegue, entre ellos la complejidad, la eficiencia, la fiabilidad y la seguridad [114]. En otra contribución relacionada, Wang *et al.* [115] proponen un diseño de ER junto con un algoritmo de encaminamiento energético basado en teoría de grafos; el prototipo, evaluado en Matlab/Simulink, demuestra la factibilidad del enfoque a nivel conceptual. Sin embargo, las limitaciones son claras, la validación principalmente en Matlab/Simulink aporta poco realismo en cuanto a latencias y heterogeneidad de enlaces, además, no trata aspectos operativos como coordinación multi-raíz. Es decir, es buen marco conceptual pero con supuestos de despliegue optimistas.

Otros trabajos sobre encaminamiento energético, como Jiang *et al.* [116] proponen un algoritmo de encaminamiento heurístico energético basado en el camino de menor coste para unir nodos con cargas de consumo, a fuentes de generación, mediante la selección de rutas eficientes. Comparado con soluciones óptimas, el método logra costes dentro del 18 % del óptimo, reduciendo notablemente la complejidad computacional. Sin embargo, es un enfoque esencialmente estático que no gestiona bien escenarios altamente dinámicos ni objetivos múltiples (p. ej. resiliencia vs pérdidas vs balance local), además, no incorpora criterios multi-raíz ni coordinación entre controladores, y su robustez ante fallos múltiples o cambios rápidos de generación/consumo no se evalúa. Es útil como heurística pero insuficiente como solución completa para entornos heterogéneos y dinámicos. Gayo *et al.* [117] implementan un sistema de encaminamiento local para una SG compuesta por prosumidores heterogéneos conectados a la red principal. Su propuesta garantiza una asignación justa de costes mediante un mercado energético local asentado sobre tecnología blockchain, mostrando viabilidad para micromercados locales. De forma similar, S. Hussain *et al.* [118] plantean un algoritmo de encaminamiento que selecciona caminos con pérdidas energéticas mínimas. Desarrollan además un modelo informativo basado en IEC61850 y emplean la mensajería IEC61850 para la implementación práctica. Esta solución es llamativa, dado que emplean un protocolo de comunicación de facto en las SGs. Por último, Gu *et al.* [119] se centran en la planificación de carga/descarga de baterías en las SGs. Aunque no abordan el encaminamiento de forma directa, sus decisiones impactan en la

## 2.3 CASOS DE USO EN ENTORNOS DENSOS Y HETEROGÉNEOS

---

topología lógica del flujo energético porque las baterías actúan como nodos con capacidad de ofrecer potencia a la red; su trabajo incluye simulación y una prueba de concepto para fomentar el enrutamiento Peer-to-Peer (P2P) entre prosumidores.

La revisión pone de manifiesto avances significativos en técnicas de encaminamiento energético y en arquitecturas para el EI, pero también evidencia limitaciones claras. La mayoría de las soluciones existentes se apoyan en optimizaciones estáticas o centralizadas, lo que restringe su escalabilidad y capacidad de adaptación en entornos dinámicos. Son escasas las propuestas que explotan plenamente topologías multi-raíz con toma de decisiones autónoma a nivel de nodo. Asimismo, falta un consenso estandarizado sobre criterios de selección de rutas que equilibren minimización de pérdidas, balance local y resiliencia. La integración de tecnologías emergentes (por ejemplo, blockchain para validación transaccional o AI para encaminamiento predictivo) está todavía poco explorada. En conclusión, la bibliografía existente aporta una base sólida de técnicas y prototipos, pero la adopción práctica en PRGs y SG heterogéneas exige marcos de encaminamiento más flexibles, distribuidos y en tiempo real, capaces de incorporar decisiones locales (posiblemente asistidas por AI/ML). Estos vacíos motivan las líneas de trabajo propuestas en la Tesis: diseñar esquemas de encaminamiento y etiquetado que sean agnósticos a la heterogeneidad de enlaces, poco invasivos sobre el plano de datos, seguros en la fase de asignación y capaces de coordinarse en entornos multi-raíz y distribuidos.

### 2.3.1.2. Conclusiones y alineación con los objetivos de la Tesis

La revisión de la bibliografía en el ámbito del encaminamiento energético y las arquitecturas para el EI pone de manifiesto avances relevantes, pero también limitaciones claras que condicionan la adopción práctica en entornos densos y heterogéneos. En particular, la mayoría de las propuestas analizadas descansan en optimizaciones estáticas o en soluciones centralizadas, lo que reduce su capacidad de adaptación y escalabilidad frente a escenarios dinámicos; son escasas las aproximaciones que exploten topologías multi-raíz con decisión autónoma a nivel de nodo; no existe consenso sobre criterios de selección de rutas que equilibren simultáneamente minimización de pérdidas, balance local y resiliencia; y la integración de tecnologías emergentes (p. ej. blockchain para validación transaccional o AI para encaminamiento predictivo) está aún poco explorada. En suma, la literatura ofrece una base técnica sólida, pero carece de marcos de encaminamiento distribuidos, ligeros y en tiempo real que puedan integrarse en PRGs y SG heterogéneas sin requerir cambios invasivos en el plano de datos.

Estos vacíos conectan de forma directa con el primer objetivo planteado en la Tesis. El primer bloque de objetivos, profundizar en los mecanismos de control de redes programables y extender el paradigma SDN a IIoT y redes de distribución eléctrica, queda plenamente justificado, dado que los entornos estudiados requieren un mecanismos de control para el intercambio eficiente de la potencia, donde los esquemas de encaminamiento/etiquetado jerárquico se presentan como una buena solución, dado que son agnósticos a la heterogeneidad de enlaces (capacidad/latencia), además de ser capaces de operar en topologías multi-raíz y con coordinación entre controladores distribuidos. Además, estos protocolos

se tienen que diseñar para minimizar la señalización en nodos con recursos limitados. Por tanto, una línea central de la Tesis consistirá en diseñar y evaluar esquemas de etiquetado y encaminamiento jerárquico que atiendan estas propiedades, dentro del marco de las SG.

### 2.3.2. Arquitecturas inteligentes de sensores IoT/IIoT

En los últimos años, el ecosistema IoT ha arraigado nuevos contextos como el IIoT, el cual, ha evolucionado rápidamente, impulsado por la creciente demanda de soluciones industriales inteligentes, eficientes y con capacidad de adaptación en tiempo real, en aras de mejorar la productividad y la seguridad en el tejido industrial de esta sociedad [120, 121].

Esta transformación viene acompañada de nuevos requerimientos técnicos, entre los que destacan la baja latencia, el procesamiento distribuido de datos, y la necesidad de gestionar recursos dinámicamente en entornos cada vez más complejos y heterogéneos. Es por ello que, en este contexto, el *edge/fog computing* [122] se ha consolidado como una tecnología clave para acercar la inteligencia al lugar donde realmente se produce el valor: el origen de los datos. En entornos industriales, los datos generados por sensores, dispositivos o procesos productivos son altamente dinámicos, voluminosos y sensibles al tiempo. Procesarlos en el borde no solo reduce significativamente la latencia y la carga sobre la red troncal, sino que también permite una respuesta contextualizada, localizada y casi inmediata ante eventos críticos o condiciones cambiantes [123]. Este paradigma habilita lo que se conoce como un enfoque basado en datos (conocido como *data-driven*), donde el comportamiento, la toma de decisiones y la orquestación de los servicios no se programan de forma rígida, sino que emergen en función de la información recopilada, procesada y analizada en tiempo real.

En otras palabras, son los datos, y no reglas fijas o configuraciones estáticas, los que guían el funcionamiento interno del sistema. Su integración con arquitecturas softwariadas basadas en microservicios, virtualización ligera y estándares abiertos, permite desplegar funciones avanzadas de manera flexible y escalable, habilitando nuevos casos de uso en el entorno IIoT, como la monitorización en tiempo real o la toma de decisiones autónomas, como por ejemplo la propia reconfiguración de la red con modelos de AI [124]. Además, esta evolución tecnológica se alinea con las visiones a futuro del 6G [125, 126], que promueven arquitecturas distribuidas, cooperativas y centradas en los datos, así como con los principios de los *data spaces* industriales. En este sentido, la inteligencia en el *edge* se perfila como un componente fundamental en las infraestructuras de próxima generación, facilitando la interoperabilidad y la coordinación entre dispositivos, redes y servicios. Sin embargo, a pesar del avance conceptual y normativo, persisten importantes desafíos. Entre ellos, destacan la escalabilidad de las soluciones, los mecanismos de seguridad, y especialmente la falta de implementaciones prácticas que validen estos principios en entornos industriales reales [127]. Como reflejan iniciativas recientes dentro del marco de las especificaciones 3rd Generation Partnership Project (3GPP), a partir del Release 18 [128], existe un esfuerzo por normalizar estas capacidades, pero aún se echa en falta la disponibilidad de arquitecturas integradas y pruebas de concepto que materialicen estas ideas de forma reproducible y abiertas.

A continuación, se introduce la sub-Sección 2.3.2.1, donde se revisarán las propuestas más atractivas e incipientes en el diseño de arquitecturas inteligentes enfocadas en el IIoT.

### 2.3.2.1. Propuestas de arquitecturas inteligentes IIoT

El despliegue de sistemas IIoT ha generado una línea de investigación sostenida sobre arquitecturas y tecnologías habilitadoras orientadas a conseguir soluciones escalables, seguras y adaptables. En esta sección se revisan las contribuciones más relevantes que abordan tanto la coordinación de la gestión y la monitorización de sensores inteligentes como la capacidad de la infraestructura para adaptarse dinámicamente a las demandas de los agentes de la red. Los trabajos analizados van desde propuestas conceptuales y modelos arquitectónicos hasta implementaciones prácticas y casos de uso industriales, e incluyen además enfoques que integran técnicas de AI/ML para tareas como mantenimiento predictivo, balanceo de carga y reconfiguración automática de la topología. El objetivo es identificar diseños reutilizables y las limitaciones actuales, en términos de interoperabilidad, coste computacional y eficiencia energética, que condicionan la adopción de estas arquitecturas en entornos reales.

El *core* de trabajos sobre arquitecturas IIoT muestra un abanico amplio de aproximaciones que van desde diseños centrados en SDN y calidad de servicio determinista, hasta arquitecturas energéticamente eficientes, soluciones de *retrofitting* (reacondicionamiento) industrial y propuestas descentralizadas basadas en blockchain o aprendizaje federado (FL). En la parte más orientada a garantizar temporización y QoS, la propuesta SD-IIoT de Hu *et al.* [129] integra dispositivos industriales, *gateways*, infraestructura de red y la nube combinando WirelessHART, Constrained Application Protocol (CoAP), WebSocket y SDN para ofrecer un esquema CoAP+SDN de QoS dirigido a entornos industriales críticos. Esta integración proporciona elevada flexibilidad y comportamiento determinista en escenarios controlados, pero su dependencia de un plano SDN relativamente centralizado y de protocolos concretos puede penalizar su aplicabilidad en redes extremadamente heterogéneas o con nodos con recursos muy limitados, donde la sobrecarga de control y señalización resulta crítica.

Por su parte, otros enfoques arquitectónicos como el propuesto por Wang *et al.* [130] sacrifican parte de la flexibilidad para maximizar la eficiencia energética: una arquitectura de tres capas (sensado, *gateway* y control) que orquesta ciclos de sueño-actividad mediante asignación centralizada de sensores a *gateways* logra prolongar la vida de redes masivas de sensores. Aunque efectiva desde la perspectiva energética, esta centralización introduce un punto único de decisión que limita la escalabilidad temporal y la resiliencia ante fallos del *gateway* o cambios rápidos en la topología, evidenciando la clásica dicotomía centralizado contra distribuido en IIoT. La experiencia práctica con *retrofitting* industrial, mostrada por Strauß *et al.* [131] en la línea de producción de BMW, complementa estas aproximaciones: instrumentar maquinaria *legacy* con sensores baratos e integrar un *pipeline* incremental de ML (detección semisupervisada → clustering → clasifi-

ficación supervisada) demuestra ser una vía viable para desplegar soluciones industriales reales, aunque plantea la necesidad de etiquetado y mecanismos generales para gestionar la enorme heterogeneidad de dispositivos sin intervención humana constante.

En cuanto a la seguridad y eficiencia en dispositivos restringidos, la propuesta de TD2SecIoT de Dejene *et al.* [132] propone un enfoque basado en criptografía de curva elíptica (del inglés, Elliptic-curve cryptography (ECC)) para ofrecer autenticación mutua y confidencialidad con bajo coste computacional, lo que equilibra seguridad y rendimiento en nodos limitados; la limitación principal es la falta de validación en despliegues heterogéneos a gran escala. En paralelo, arquitecturas abiertas orientadas al usuario, como la de Zhang *et al.* [133], facilitan la extensibilidad y la interacción bidireccional usuario–activos eléctricos (validada en parques eólicos), pero requieren una gobernanza e interoperabilidad robustas para su adopción masiva. La línea de trabajo federada y colaborativa aparece con la propuesta de Fed-IIoT de Taheri *et al.* [134], que usa aprendizaje federado y mecanismos adversarios (del inglés, Generative Adversarial Network (GAN)) para mitigar envenenamiento de datos durante el entrenamiento colaborativo; ofrece privacidad y resistencia frente a ataques de datos, aunque plantea desafíos de orquestación y coste computacional en *gateways* y nodos.

Los trabajos centrados en *gateways* y orquestación muestran asimismo tensiones prácticas: el *gateway* multi-protocolo de Zhang *et al.* [135] garantiza interoperabilidad entre protocolos industriales, procesando los protocolos OPC UA, Modbus, Siemens S7 a mensajes Message Queuing Telemetry Transport (MQTT), con procesamiento asíncrono y cifrado en tres capas, facilitando la integración de equipos *legacy* con la nube; sin embargo, la concentración del procesado de protocolos y la transformación puede introducir cuellos de botella si no se diseña una topología distribuida y tolerante a la sobrecarga introducida. El aprovechamiento de redes móviles avanzadas aparece en Chandra *et al.* [136], que explota capacidades 5G, como son Enhanced Mobile Broadband (eMBB)/Massive Machine Type Communications (mMTC)/Ultra-Reliable Low Latency Communication (URLLC), para manufactura en tiempo real; su fortaleza es la ultrabaja latencia y densidad de dispositivos, pero su despliegue queda condicionado a infraestructuras 5G y coordinaciones multi-operador. En cuanto a soluciones enfocadas en el *edge*, tenemos a SEGA de Ghosh *et al.* [137] demuestra que el procesamiento de inferencia KNN en *gateways* en el *edge*, y el *offload* de decisiones de reconfiguración en el *cloud* son viables con baja latencia y seguridad, aunque su despliegue basado en Docker presenta problemas de escalado en ingestión y almacenamiento de los datos recolectados; frente a esto, plataformas orquestadas con Kubernetes mejoran la escalabilidad a costa de mayor complejidad de despliegue.

### 2.3.2.2. Conclusiones y alineación con los objetivos de la Tesis

Los estudios revisados muestran avances relevantes en arquitecturas inteligentes IIoT y tecnologías habilitadoras para redes softwarizadas (*edge/fog* y *cloud*), pero ponen de manifiesto vacíos que conectan directamente con el segundo bloque de objetivos de la Tesis. En primer lugar, gran parte de las propuestas permanecen en un estadio teórico o cerrado (implementaciones propietarias), con escasas referencias a implementaciones re-

## 2.3 CASOS DE USO EN ENTORNOS DENSOS Y HETEROGÉNEOS

---

producibles sobre hardware industrial o testbeds reales. Esta carencia limita la capacidad de evaluar de forma realista el comportamiento de las plataformas de orquestación, despliegue y monitorización que se proponen en la Tesis, y por tanto justifica la necesidad de desarrollar prototipos abiertos y experimentación en entornos representativos. En segundo lugar, se detecta la ausencia de marcos de orquestación unificados que integren salidas de inferencia en tiempo real para desencadenar reconfiguraciones dinámicas de red, reubicación de servicios y mantenimiento preventivo. Este *gap* es especialmente relevante para el objetivo de proveer herramientas de despliegue, monitorización y gestión que permitan al administrador tomar decisiones automáticas basadas en información operacional continua. La Tesis abordará este punto diseñando mecanismos de control que vinculen la telemetría de los sensores y las predicciones (AI/ML) con mecanismos automáticos de reconfiguración y *placement* de servicios.

Además, las pocas arquitecturas basadas en microservicios que aparecen en la literatura suelen mostrar limitaciones de rendimiento ante despliegues de alta densidad de sensores o cargas intermitentes: ausencia de autoescalado nativo, cuellos de ingestión de datos y latencias inasumibles. Esto repercute en la operatividad y en la capacidad para mantener criterios de QoS en escenarios industriales. Por ello, uno de los objetivos concretos de la Tesis es analizar las implicaciones de rendimiento de las soluciones software (orquestradores, pipelines de inferencia, planos de control) y proponer mecanismos de elasticidad y mitigación de cuellos de botella, valorando tanto soluciones centralizadas como distribuidas. Finalmente, emergen retos transversales de seguridad, privacidad y gobernanza (exposición de modelos, intercambio de datos heterogéneos, validación de decisiones automáticas) que deben integrarse desde el diseño de la infraestructura. En consonancia con el segundo bloque de la Tesis, se plantea investigar mecanismos ligeros de protección (autenticación para despliegue), trazabilidad de decisiones y evaluación de impacto de la reconfiguración automática sobre la seguridad operativa.

En conjunto, los huecos identificados, como la falta de implementaciones reproducibles, carencia de orquestación basada en inferencia en tiempo real, limitaciones de escalado en arquitecturas microservicio y necesidades de seguridad integrada, confirman la pertinencia de los objetivos del segundo bloque de la Tesis. La investigación propuesta se orientará a cerrar dichas brechas mediante: (i) prototipos abiertos y validación experimental en testbeds representativos; (ii) un *framework* de orquestación que conecte telemetría, modelos AI/ML y políticas automáticas de reconfiguración; (iii) técnicas de autoescalado y mitigación de cuellos de botella para entornos de alta densidad; y (iv) mecanismos de autenticación ligeros para la gestión automática de la infraestructura.



## Capítulo 3

# Planteamiento del problema

En este capítulo se sintetizan y consolidan los huecos identificados en el capítulo del Estado del Arte, con el objetivo de transformar las lecciones aprendidas en un planteamiento claro del problema, y marcar una hoja de ruta clara para la Tesis. Para ello, se recogen las conclusiones parciales ya extraídas en las secciones correspondientes: control *in-band* (Sección 2.1.4), arranque y provisión de canales de control en entornos densos y heterogéneos (Sección 2.2.1.3), gestión y planificación de recursos (Sección 2.2.2.3), optimización y reconfiguración proactiva (Sección 2.2.3.3), encaminamiento energético y arquitecturas para EI/PRGs/SG (Sección 2.3.1.2) y arquitecturas inteligentes IIoT (Sección 2.3.2.2).

A partir de esa revisión consolidada se extraerán las lecciones aprendidas, los huecos comunes, se priorizarán los retos con mayor impacto práctico, pudiendo sentar las bases de la Tesis. Finalmente, este capítulo presenta la estrategia de trabajo propuesta, y como se han organizado las contribuciones de la Tesis en los siguientes capítulos.

Las lecciones extraídas de la revisión sistemática del estado del arte muestran un panorama consistente: existen avances metodológicos y prototipos relevantes en control *in-band*, arranque/etiquetado jerárquico, gestión de recursos, optimización proactiva y arquitecturas IIoT/SG, pero la mayoría de estas aportaciones no resuelven de forma conjunta los requerimientos de despliegue en entornos densos y heterogéneos. En todos los dominios examinados emergen huecos comunes que limitan la adopción práctica: falta de soluciones «agnósticas» al *dataplane* y al proveedor, escasez de protocolos *in-band* estandarizados, ausencia de mecanismos de arranque (*bootstrapping*) seguros y ligeros validados a gran escala, carencia de coordinación eficiente entre controladores distribuidos, y validaciones limitadas sobre hardware real o testbeds representativos. Estas carencias son recurrentes y condicionan tanto la viabilidad técnica como la transferencia de resultados a escenarios reales (IIoT, micro-redes, smart grids).

Del análisis por áreas concretas se obtienen conclusiones operativas que permiten priorizar líneas de trabajo. En el ámbito del arranque y provisionamiento de canales de control, el etiquetado jerárquico y las construcciones modo árbol enraizadas emergen como la técnica más prometedora: reducen estado local, permiten encaminamiento con bajo coste y faci-

litan el diseño de rutas de control compactas. No obstante, muchas propuestas actuales (Torii/GA3/eTorii, Amaru, IoTorii, Izzy, etc.) están validadas en topologías homogéneas o en simulación y/o requieren modificaciones del *dataplane*; pocas consideran nativamente heterogeneidad de enlaces, nodos con recursos muy limitados (a excepción de IoTorii) o escenarios multi-raíz con coordinación entre controladores. En gestión y planificación de recursos, la tensión centralización contra distribución es clave. Los esquemas centralizados facilitan optimización global pero fallan en escalabilidad y resiliencia; los distribuidos ganan en tolerancia y reparto de carga, pero sufren mayor complejidad, tiempos de convergencia más grandes y visibilidad parcial de la red. Aquí el etiquetado jerárquico puede actuar como palanca técnica: al simplificar la topología física a la topología lógica, se reduce señalización y habilita decisiones locales eficientes, lo que hace posibles diseños distribuidos más prácticos. En optimización y reconfiguración proactiva (tanto para SG como para redes IIoT), los enfoques existentes (optimizadores exactos, metaheurísticos y modelos AI/ML) demuestran potencial, pero tropiezan con problemas reales: modelos no agnósticos (entrenados para una topología concreta), elevada señalización o coste computacional para nodos con recursos limitados, y validaciones insuficientes con datos reales o testbeds. Finalmente, en IIoT las arquitecturas muestran avances (*edge/fog*, microservicios, federated learning, etc.), pero muchas implementaciones son cerradas o no escalan en despliegues de alta densidad; falta un *framework* de orquestación que conecte inferencia en tiempo real con reconfiguración automática, y es necesario abordar la autoscalabilidad, latencia y seguridad desde el diseño. En el ámbito del encaminamiento energético y las arquitecturas para el EI pone de manifiesto avances relevantes, sin embargo, la mayoría de las propuestas analizadas descansan en optimizaciones estáticas o en soluciones centralizadas, lo que reduce su capacidad de adaptación y escalabilidad frente a escenarios dinámicos; son escasas las aproximaciones que exploten topologías multi-raíz con decisión autónoma a nivel de nodo; no existe consenso sobre criterios de selección de rutas que equilibren simultáneamente minimización de pérdidas, balance local y resiliencia.

A partir de esos *gaps*, la hoja de ruta de la Tesis articula unas líneas de investigación concretas, metodológicamente acotadas y alineadas con los dos bloques de objetivos previamente definidos.

- En primer lugar, se plantea diseñar y formalizar esquemas de etiquetado jerárquico con las siguientes propiedades: (i) «agnosticismo» frente a heterogeneidad de enlace (capacidad/latencia) y a implementaciones de *dataplane legacy*; (ii) mínima o nula necesidad de modificar el hardware/software de reenvío; (iii) soporte para multi-raíz y coordinación eficiente entre controladores distribuidos; (iv) reducción de señalización mediante decisiones locales informadas (posiblemente asistidas por modelos AI/ML compactos); y (v) capacidad de reconfiguración de la red proactivamente (rutas de respaldo). Estas contribuciones atacan directamente el primer bloque de objetivos, proporcionando los bloques de arranque, encaminamiento y reconfiguración de baja sobrecarga necesarios para operar en topologías reales, densas y heterogéneas.
- En segundo lugar, la Tesis investigará estrategias híbridas de gestión y orquestación de recursos que combinen optimización centralizada (cuando la latencia y la

---

capacidad de señalización lo permitan) con decisiones locales ligeras basadas en el etiquetado jerárquico. En la práctica esto implica diseñar protocolos jerárquicos de coordinación, políticas de delegación de decisiones (qué se decide localmente y qué se delega al controlador), y mecanismos de retroalimentación que permitan al plano de control incorporar información sin sobrecargar la red.

- En tercer lugar, la Tesis se plantea diseñar una arquitectura IIoT inteligente, con marcos de orquestación unificados que integren salidas de inferencia (AI/ML) en tiempo real de telemetría de los sensores para desencadenar reconfiguraciones dinámicas de red, reubicación de servicios y mantenimiento preventivo. Además, se debe respetar el prototipado abierto y reproducible de microservicios, y pipelines de inferencia con autoescalado. Otro aspecto importante es la evaluación de cuellos de ingestión de datos y la incorporación de mecanismos ligeros de autenticación y trazabilidad de decisiones para proteger el despliegue de la arquitectura. Estas tareas responden a los *gaps* detectados en las arquitecturas IIoT (ausencia de implementaciones abiertas, carencia de orquestación basada en inferencia en tiempo real, y limitaciones en autoscalado).

Metodológicamente, la Tesis combinará análisis teórico, simulación/emulación y prototipado práctico. El plan experimental incluye:

1. Validaciones por simulación/emulación en topologías heterogéneas y densas, donde, se emplearán topologías aleatorias, densas, e hiperconectivas. Se hará uso del generador de topologías aleatorias conocido como Boston university Representative Internet Topology gEnerator (BRITE) [138].
2. Validación real cuando se tengan las capacidades hardware de llevarlo a cabo.
3. Experimentos en testbeds representativos. En caso de trabajar con las SG, se emplearán benchmarks IEEE para feeders. En el caso de despliegues de arquitecturas IIoT, se empleará el CPD de la universidad para emular la visión continua de *factory/edge/cloud*.

En resumen, la Tesis propone una agenda integradora: avanzar en mecanismos prácticos de control *in-band* y etiquetado jerárquico (soporte multi-raíz), desarrollar estrategias de orquestación híbrida que combinen decisión local y optimización global, integrar AI/ML de forma eficiente y portable para asistencia en la toma de decisiones, y validar todo ello mediante prototipos abiertos y experimentación en escenarios representativos (IIoT, SG y micro-redes). Estas líneas están alineadas de forma directa con los dos grandes bloques de objetivos de la Tesis: (i) profundizar en mecanismos de control para redes programables y su extensión a IIoT y redes de distribución eléctrica; y (ii) estudiar y construir infraestructuras software de despliegue, monitorización, orquestación y seguridad que permitan la toma de decisiones automáticas y robustas en entornos reales y heterogéneos. Estas líneas de trabajo se presentarán a continuación, de forma secuencial y cronológica (en la medida de lo posible), organizando las propuestas/publicaciones realizadas en diferentes capítulos.



## Capítulo 4

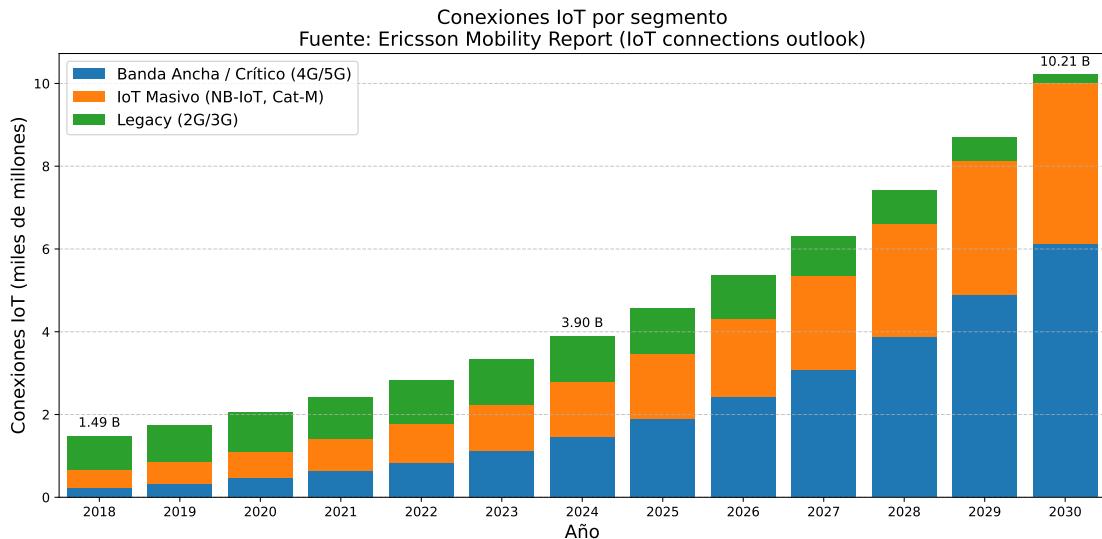
# Propuesta escalable de control In-band para entornos inalámbricos y de baja capacidad

En este capítulo se presenta la primera aportación de la Tesis: una propuesta de control *in-band* escalable para redes SDN en entornos inalámbricos y de baja capacidad, concebida en el marco de los futuros ecosistemas 6G. El origen de esta línea de trabajo se remonta a mi Trabajo Fin de Máster [139], donde se sentaron las primeras bases conceptuales. Posteriormente, se consolidó mediante el desarrollo y publicación de una revisión sistemática sobre los mecanismos de control SDN *in-band* [19], y materializó como primera contribución de la Tesis en forma de ponencia en la conferencia CommNet 2023 [140]. Aunque la evaluación experimental de esta propuesta se planteó principalmente como una prueba de concepto y, por tanto, no resulta exhaustiva, constituye un punto de partida sólido que guía y fundamenta las siguientes contribuciones de la Tesis.

### 4.1. Introducción

Los recientes avances en comunicaciones móviles, junto con la mejora de las capacidades de hardware, han impulsado el crecimiento del IoT al interconectar miles de millones de objetos mediante comunicaciones Machine-to-Machine (M2M) en entornos tanto domésticos como industriales [141]. Puede afirmarse sin lugar a dudas que el IoT forma parte integral del Internet presente y futuro: ha transformado la forma en que interactuamos con el entorno, permitiendo la interconexión autónoma de dispositivos a través de la red y ofreciendo entornos inteligentes y adaptativos que responden a las necesidades de la sociedad. No obstante, el crecimiento exponencial de dispositivos IoT conectados a redes móviles introduce nuevas exigencias en términos de capacidad, rendimiento, latencia y eficiencia de red que deben abordarse (Ver Figura 4.1). La tecnología 5G se propuso y desplegó comercialmente para cubrir muchos de los requisitos de las redes IoT y sus aplicaciones. Sin embargo, con la rápida proliferación de nuevos sensores y la consiguiente expansión de las redes IoT, los requisitos técnicos necesarios para sostener los entornos tradicionales M2M (plenamente autónomos, dinámicos e inteligentes) se han incrementado. Por tanto,

se requiere una tecnología más avanzada para satisfacer las demandas futuras de las redes IoT, y la arquitectura 6G se postula como una solución capaz de afrontar estos nuevos retos [142].



**Figura 4.1:** Estudio del crecimiento de las conexiones de dispositivos IoT desglosadas por segmento [143].

Un aspecto esencial de la nueva generación de redes móviles es la arquitectura de interconexión física que se propondrá. En 5G, la columna vertebral SDN existente (en las redes móviles anteriores) se reutilizó incorporando modificaciones software para acomodar las nuevas especificaciones arquitectónicas mediante flexibilidad y programabilidad [144]. En esta línea, cabe preguntarse si la futura red 6G aprovechará los beneficios de SDN para el procesamiento de datos en su *backbone*. Los informes preliminares sobre el diseño de 6G [125, 145, 146] indican que SDN, junto con tecnologías como P4 y técnicas de AI/ML, se emplearán para definir el plano de procesamiento de datos y mejorar el rendimiento y la orquestación de la infraestructura existente. Además, las redes 6G enfatizarán la necesidad del paradigma MEC, extendiendo el borde de la red hacia los usuarios y construyendo el llamado continuo *edge-to-cloud* (o simplemente *cloud continuum*) [147]. Aunque SDN sigue siendo un pilar del *edge computing*, existen aún desafíos en la integración de IoT y SDN en el borde de la red [148], tales como la adaptación de estándares y recomendaciones SDN a entornos inalámbricos, y la compatibilidad con dispositivos heterogéneos y con recursos restringidos, característicos de las redes IoT (Según se ha analizado en el Capítulo 3).

En el caso específico del plano de control SDN, se consideran dos paradigmas de control: *out-of-band* e *in-band* (Según se estudió en la Sección 2.1.2). Mientras que en el enfoque *out-of-band* cada nodo SDN dispone de un enlace dedicado al controlador (es decir, la información de control circula por una red dedicada), en el modelo *in-band* sólo

ciertos nodos gestionados cuentan con enlace directo al controlador y el resto de dispositivos reutilizan dichos enlaces para reenviar la información de control hacia el controlador SDN (la información de control, al no disponer de una red dedicada, viaja conjuntamente con el tráfico de datos hasta alcanzar el controlador). Aunque no existe un paradigma de control intrínsecamente superior, cada enfoque tiene ventajas e inconvenientes y la elección depende del caso de uso concreto, los dispositivos IoT con recursos restringidos en el extremo de la red se benefician en general del control *in-band*: aunque potencialmente menos seguro [149], sus requisitos computacionales y energéticos suelen ser inferiores. No obstante, según se revisó (Sección 2.1.4), no existe un estándar específico que defina el control *in-band*; y, si bien existen propuestas en la literatura, pocas contemplan entornos inalámbricos y ninguna está diseñada explícitamente para dispositivos con recursos limitados.

En este capítulo se presenta, **win-BOFUSS**, una solución evolucionada del switch Basic OpenFlow User-space Software Switch (BOFUSS) que soporta control *in-band* para escenarios inalámbricos, diseñada de forma escalable y orientada a dispositivos IoT.

## 4.2. Descripción del protocolo

El Estado del Arte ha puesto de manifiesto que el etiquetado jerárquico y las técnicas basadas en árboles enraizados constituyen un marco fértil y consolidado para abordar el arranque y la provisión de canales de control en entornos densos y heterogéneos (Ver Sección 2.2.1.3). Las propuestas revisadas comparten la idea de explotar información codificada localmente en etiquetas para reducir estado y facilitar encaminamiento hacia la raíz o raíces. Es por ello, que nuestro protocolo *in-band* busca y establece múltiples rutas entre un nodo raíz y el resto de los nodos de la topología, creando múltiples árboles conectados. La novedad de este protocolo es que permite al controlador SDN comunicarse mediante canales *in-band* en entornos inalámbricos, lo cual no es común dado que SDN fue diseñado originalmente para escenarios cableados. Existen propuestas enfocadas de etiquetado jerárquico en redes LLNs (IoTorii, de Rojas *et al.* [70]), que al igual que esta solución, están concebidas para requerir pocos recursos computacionales y de memoria en dispositivos restringidos, sin embargo, esta propuesta habilita a los nodos IoT interactuar de forma nativa en entornos SDN. Por lo tanto, resulta particularmente adecuado para redes IoT-SDN. Asimismo, gracias a la funcionalidad *multipath* que ofrece el protocolo, también es posible almacenar diferentes rutas de respaldo que pueden utilizarse en caso de fallos o en escenarios de movilidad dentro del área de cobertura, manteniendo siempre activa la sesión con el controlador.

La definición de este protocolo se apoya en el trabajo de Constantin *et al.* [150], en el que se establecen múltiples rutas entre un nodo raíz y el resto de nodos de la topología en un escenario cableado. Estas múltiples rutas se construyen combinando dos técnicas (según se explicaba en la Sección 2.2.1.1): el etiquetado jerárquico de los dispositivos y una inundación controlada de dichas etiquetas. En primer lugar, se selecciona un nodo raíz de la red (generalmente un nodo con acceso al controlador SDN), y dicho nodo genera un

mensaje inicial con una etiqueta específica que envía a sus nodos directamente conectados. El dispositivo receptor inunda este mensaje a sus vecinos a través de todas las interfaces excepto aquella por la que recibió el mensaje. Además, añade un nuevo elemento a la etiqueta recibida por cada interfaz de salida con el fin de rastrear la ruta recorrida a lo largo de la topología. Cada vecino que recibe el mensaje repetirá este proceso. El proceso de inundación puede detenerse en cualquier nodo considerando parámetros específicos de la etiqueta recibida, tales como la longitud de la etiqueta, el prefijo común de la etiqueta, entre otros. Eventualmente, cada nodo recibirá un conjunto de etiquetas jerárquicas que identifican múltiples rutas para alcanzar el nodo raíz (quien envió la primera etiqueta) desde el nodo actual. Si el nodo raíz es un nodo conectado al controlador SDN, entonces cada nodo podrá establecer comunicación con el controlador a través de cualquiera de esas rutas etiquetadas. Estas etiquetas reciben el nombre de HLMAC, siguiendo la idea del estándar IEEE 802c-2017 [151], que define el uso de direcciones Medium Access Control (MAC) locales.

Nuestro protocolo adapta la idea original a entornos inalámbricos. El reto principal en estos escenarios es que, a diferencia de las redes cableadas, los dispositivos inalámbricos suelen disponer de una única interfaz radio que les permite comunicarse con todos los vecinos dentro de su área de cobertura; por tanto, el mecanismo de etiquetado jerárquico propuesto en [150] no puede aplicarse directamente. Para solventar esta limitación incorporamos conceptos tomados de IoTorii de Rojas *et al.* [70] (una línea de trabajo en la que participé antes del Doctorado) y adaptamos el etiquetado a la naturaleza broadcast del medio inalámbrico. Concretamente, el proceso se basa en identificar previamente los vecinos que se encuentran realmente en rango de cobertura y, sobre esa información local, asignar las etiquetas jerárquicas correspondientes en función de la relación de vecindad (y no del puerto de salida). De este modo cada nodo construye su vista local del entorno y puede generar/propagar etiquetas coherentes para el encaminamiento en topologías inalámbricas.

Por tanto, el protocolo se basa en dos procesos fundamentales que operan de forma complementaria:

- **Reconocimiento de vecindad** (Detallado en la sub-Sección 4.2.1): Mecanismo local y periódico mediante el cual cada nodo detecta y mantiene la lista actualizada de vecinos en rango (tabla de vecinos). Esta información sirve para identificar enlaces activos, eliminar vecinos “muertos” y proporcionar la base para la asignación de identificadores locales que sustituyen al concepto de puerto físico en entornos cableados.
- **Etiquetado jerárquico** (Detallado en la sub-Sección 4.2.2): Proceso de generación y difusión controlada de etiquetas (HLMAC) que permite construir uno o varios árboles enraizados en los nodos con acceso al controlador SDN. El etiquetado utiliza la información de vecindad para propagar rutas jerarquizadas, reducir el estado por nodo y habilitar encaminamiento multipath y protección ante fallos o movilidad.

### 4.2.1. Proceso de reconocimiento de vecindad

El proceso de reconocimiento de vecindad es un proceso ligero y periódico cuya función es mantener, en cada nodo, una tabla de vecinos con entradas de tipo tuplas, donde se asocia cada dirección MAC vecina con un identificador local (ID). Esta tabla se actualiza localmente en cada nodo mediante mensajes *Hello*, que cada nodo difunde periódicamente en su área de cobertura. Por tanto, los objetivos del proceso de reconocimiento de vecindad son los siguientes:

- Detectar vecinos nuevos. Al recibir un mensaje *Hello*, el nodo inspecciona la dirección MAC remitente; si no está presente en la tabla de vecinos, se crea una nueva entrada almacenando la asociación ( $MAC \rightarrow ID$ ). La ID puede generarse como un contador incremental (sufijo) o mediante otro esquema determinista según la implementación.
- Detectar vecinos “muertos”. Cada entrada en la tabla incorpora un temporizador (*timeout*) que se refresca al recibir *Hello* periódicos del vecino correspondiente. Si el temporizador expira, la entrada se elimina y se invalidan las rutas que dependieran de dicho vecino, asumiendo una ausencia por fallo o movilidad.
- Parametrización y control de señalización. El periodo de emisión de mensajes tipo *Hello* y los umbrales de *timeout* son parámetros configurables que permiten adaptar el coste de señalización a la densidad y movilidad del despliegue (p. ej., intervalos más largos en redes muy densas o donde la energía es limitada).
- Escalabilidad local. El mecanismo está diseñado para que la carga por nodo dependa principalmente de su grado de conectividad (número de vecinos). Esto facilita dimensionar recursos (memoria, CPU) y ajustar parámetros en dispositivos con capacidades limitadas.

Desde el punto de vista de coste, el mecanismo es muy simple: por cada ronda de descubrimiento, cada nodo emite un *Hello* y recibe tantos *Hello* como grado local de conectividad tenga. Esta caracterización ayuda a dimensionar el periodo de *Hello* y el tiempo de expiración (*timeout*) en función de la densidad y movilidad del despliegue.

Para ilustrar este proceso, se presenta la Figura 4.2. Según se puede apreciar en el inferior de la Figura, se ha exemplificado una topología inalámbrica de ejemplo, donde los enlaces representados indican que ambos nodos se encuentran en radio de cobertura. Tomando la Figura 4.2 como ejemplo, podemos observar cómo se llenarían las tablas de vecinos una vez el proceso hubiera convergido. En el caso del nodo *A*, sólo detecta al nodo *B* y almacenará la entrada ( $MAC_B$ ,  $ID = 1$ ). El nodo *B* detecta a  $\{A, C, D\}$  y asigna tres IDs locales  $\{1, 2, 3\}$  respectivamente; *C* registra a  $\{B, F\}$  con dos IDs, y así de forma análoga. Nodos con alta conectividad (p. ej. *B* o *D*) soportan mayor carga de recepción y mantienen más entradas, lo que debe tenerse en cuenta al dimensionar recursos en dispositivos con capacidades limitadas.

Es por ello que, el proceso de reconocimiento de vecindad, apoyado en los mensajes *Hello* y la tabla de vecinos proporcionan una base local, eficiente y fácilmente parametrizable

para habilitar el etiquetado jerárquico y la construcción de árboles en entornos inalámbricos y móviles, controlando explícitamente el intercambio de señalización en función de la densidad y la movilidad de la red.

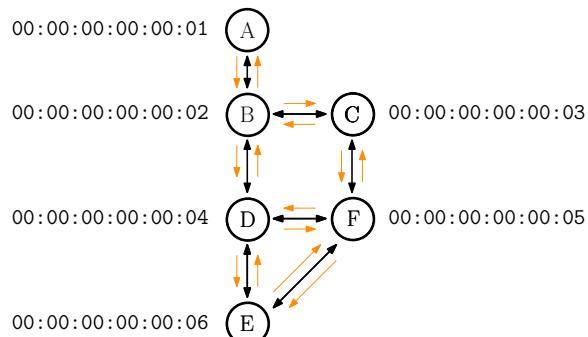
Tabla de vecinos – Nodo A			Tabla de vecinos – Nodo D		
		Sufijo			Sufijo
$MAC_{NodoB}$	00:00:00:00:00:02	1	$MAC_{NodoB}$	00:00:00:00:00:02	1
$MAC_{NodoF}$			$MAC_{NodoF}$	00:00:00:00:00:05	2
$MAC_{NodoE}$			$MAC_{NodoE}$	00:00:00:00:00:06	3

Tabla de vecinos – Nodo B			Tabla de vecinos – Nodo E		
		Sufijo			Sufijo
$MAC_{NodoA}$	00:00:00:00:00:01	1	$MAC_{NodoD}$	00:00:00:00:00:04	1
$MAC_{NodoC}$	00:00:00:00:00:03	2	$MAC_{NodoF}$	00:00:00:00:00:05	2
$MAC_{NodoD}$	00:00:00:00:00:04	3			

Tabla de vecinos – Nodo C			Tabla de vecinos – Nodo F		
		Sufijo			Sufijo
$MAC_{NodoB}$	00:00:00:00:00:02	1	$MAC_{NodoC}$	00:00:00:00:00:03	1
$MAC_{NodoF}$	00:00:00:00:00:05	2	$MAC_{NodoD}$	00:00:00:00:00:04	2



**Figura 4.2:** Proceso de reconocimiento de vecindad empleando las tablas locales de vecinos.

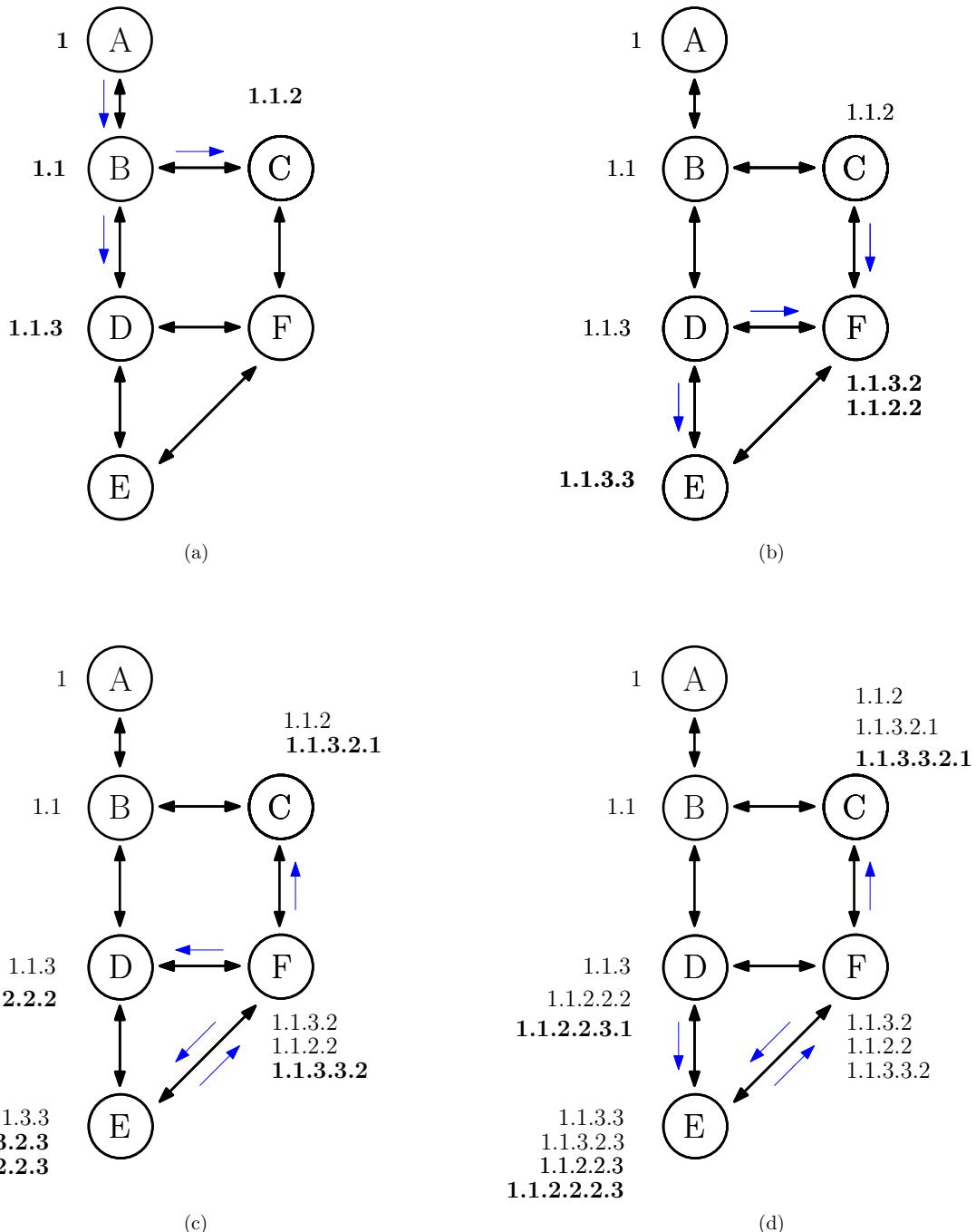
#### 4.2.2. Proceso de etiquetado jerárquico

Una vez se ha explicado el proceso de reconocimiento de vecindad en la topología inalámbrica, se va a detallar el proceso de etiquetado jerárquico del protocolo. Para ilustrar y aclarar el proceso de construcción de la topología lógica a partir de la topología física, la Figura 4.3 muestra el proceso de inundación de las HLMAC en una topología de ejemplo de seis nodos, donde el nodo *A* actúa como nodo raíz (el que tendrá acceso con el controlador SDN). El procedimiento arranca después de haberse intercambiado al menos una ronda de paquetes *Hello*, es decir, cuando todos los nodos conocen a sus vecinos y han poblado sus tablas locales de vecinos con la asociación dirección ( $MAC \rightarrow ID$ ). Las áreas de cobertura y las relaciones de vecindad se representan con una flecha doble negra (por ejemplo, *B* tiene por vecinos a *A*, *C* y *D*). La figura se divide en cuatro subfiguras que representan pasos sucesivos del procedimiento.

La Fig. 4.3(a) ilustra el primer paso del proceso: el nodo *A* envía la primer HLMAC, que es recibida por el nodo *B*. La HLMAC inicial enviada por *A* está compuesta por el ID de *A* y por el ID local que *A* asignó a *B*, representado como 1.1. Concretamente, *A* sabe por el intercambio previo de los mensajes *Hello* que sólo tiene un vecino (*B*) y le asigna el ID 1. Cuando *B* recibe 1.1, consulta su tabla local de vecinos y comprueba que tiene tres vecinos [*A,C,D*] con ID locales [1,2,3]. En ese contexto, *A* se considera el nodo padre (que fue quien envió el HLMAC inicial) y *C*, *D* son nodos hijos. A continuación, *B* genera dos nuevas HLMAC, 1.1.2 y 1.1.3, y las envía a *C* y *D*, respectivamente; pero no se envía HLMAC al nodo padre. Es importante subrayar que no se inundan todos los nodos con HLMAC para evitar bucles: sólo se retransmiten aquellas HLMAC que no sean extendidas de una HLMAC previamente almacenada como hija. Por ejemplo, si *B* ya ha guardado 1.1 y en una iteración posterior recibe una HLMAC del tipo 1.1.x..., la descartará inmediatamente porque proviene de uno de sus hijos y por tanto representaría un bucle.

La Fig. 4.3(b) muestra el segundo paso: las HLMACs enviadas por *B* (1.1.2 y 1.1.3) son recibidos por los nodos *C* y *D*. El nodo *C* verifica que tiene dos vecinos [*B,F*] con IDs [1,2] y, por tanto, sólo genera y reenvía la HLMAC 1.1.2.2 hacia su hijo *F*. En el caso del nodo *D*, con vecinos [*B,F,E*] e IDs [1,2,3] (donde *F* y *E* son hijos), produce 1.1.3.2 y 1.1.3.3 y las reenvía a *E* y *F*. El proceso se repite en *E* y *F*, tal y como se muestra en la Fig. 4.3(c), donde las nuevas HLMACs aprendidas por cada nodo se resaltan en negrita. La subfigura inferior derecha (Fig. 4.3(d)) recoge el conjunto final de HLMACs almacenadas en todos los nodos de la topología.

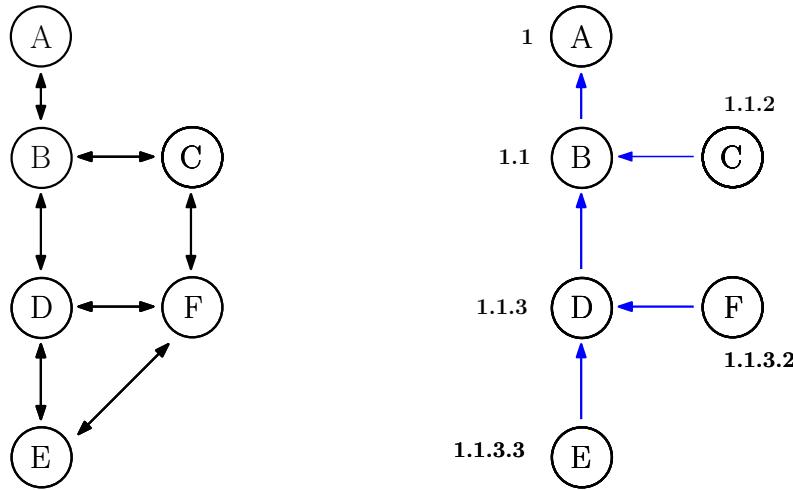
Al finalizar el procedimiento, cada nodo habrá almacenado múltiples HLMACs, es decir, varias rutas posibles hacia la raíz. Por ejemplo, *B* sólo tendrá un HLMAC (una única ruta hacia la raíz), mientras que *C*, *D* y *F* dispondrán de tres HLMACs (tres rutas hacia el nodo raíz) y *E* de cuatro HLMACs, teniendo más opciones para alcanzar al nodo que da acceso al controlador SDN. Además, el número total de HLMACs por nodo puede limitarse a un valor prefijado, de modo que cada nodo mantenga como máximo un número de rutas, lo que representa una tensión entre cantidad de información de estado por nodo contra la capacidad de resiliencia con rutas de respaldo a nivel de nodo.



**Figura 4.3:** Proceso de difusión de etiquetas Hierarchical Local MAC (HLMAC) en la topología de ejemplo.

La elección de que ruta utilizar depende de criterios del caso de uso final, y de las características propias de la topología, pudiendo optar por el menor número de saltos, menor latencia, máxima redundancia, entre otros. En este caso, al ser una prueba de

concepto, no se ha explorado en profundidad el impacto de los criterios, y se ha aplicado la elección de la HLMAC con menor latencia, obteniendo la siguiente topología lógica según se indica en la Figura 4.4. Si consideramos todas las  $HLMAC_{active}$  de los nodos en la topología, podemos definir una topología lógica que contiene todos los nodos de la topología física pero solo un subconjunto de los enlaces presentes en dicha topología física. La formulación del proceso de establecimiento la topología lógica se puede encontrar descrito en la Sección 2.2.1.1. Es importante resaltar que este proceso de selección de rutas y asignación de HLMAC tiene un impacto significativo en la eficiencia y utilización de los enlaces en la topología. Aquellos enlaces que no sean utilizados, quedan en desuso, lo que podría liberar recursos y mejorar la capacidad de la red para adaptarse a cambios en la topología.



**Figura 4.4:** Establecimiento de la topología lógica.

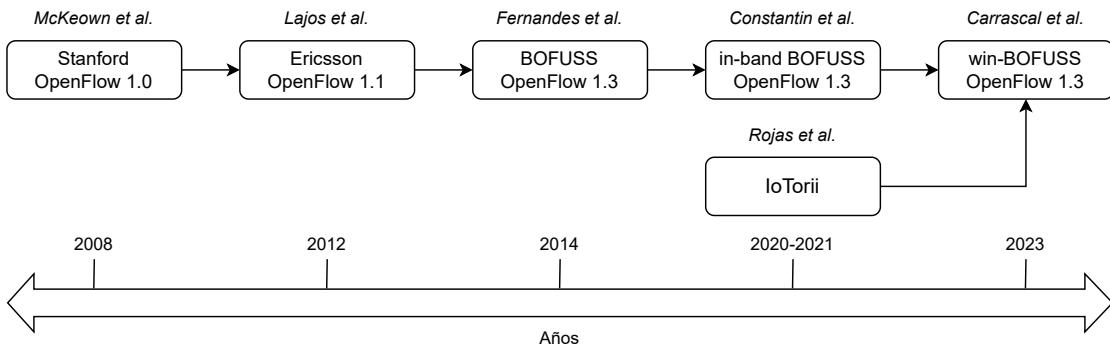
La principal ventaja de este diseño *in-band* es su simplicidad, en el caso más sencillo, el protocolo puede operar con una única HLMAC asignada por nodo, es decir, una sola entrada de enrutamiento, y, aun así, ofrecer conectividad con el controlador. Además, el número de HLMACs que requiere cada dispositivo es independiente del tamaño total de la red, además de configurable, lo que reduce drásticamente la memoria necesaria en dispositivos IoT con recursos limitados.

En cuanto al número de mensajes intercambiados, cada nodo sólo necesita enviar un mensaje *Hello* periódico y tantas transmisiones de asignación de HLMACs como caminos *multipath* se configuren; si sólo se mantiene una HLMAC, bastan únicamente dos mensajes periódicos en total. Estos intercambios están basados en eventos, y su dinámica es comparable a la difusión de un mensaje desde el nodo raíz hacia el resto de la red, proceso que suele ser rápido en comparación con protocolos de estado de enlace. Por último, la frecuencia de estas actualizaciones es configurable en función de la movilidad: escenarios con alta movilidad requieren períodos cortos, mientras que entornos estáticos pueden emplear períodos más largos.

### 4.3. Implementación de wireless In-Band Basic OpenFlow Software Switch (win-BOFUSS)

La implementación del protocolo se ha realizado sobre la plataforma *Basic OpenFlow Software Switch* (BOFUSS). Esta elección no ha sido arbitraria, ya que, en comparación con otros software-switches ampliamente utilizados, como OVS, BOFUSS ofrece un data-path cuya implementación resulta más sencilla de desarrollar y depurar, al ejecutarse en espacio de usuario y no en espacio de *kernel*.

Los orígenes de BOFUSS se remontan a 2008, cuando en la Universidad de Stanford se desarrolló la primera implementación de referencia del protocolo OpenFlow, denominada *The Stanford Reference OpenFlow Switch* (McKeown *et al.* [20]). Dicho desarrollo constituyó un producto mínimo viable, concebido con el objetivo de demostrar el funcionamiento del protocolo y facilitar el proceso de estandarización de OpenFlow 1.0. Posteriormente, el proyecto fue retomado por los laboratorios de Ericsson, donde se amplió su funcionalidad para soportar OpenFlow 1.1 (Lajos *et al.* [152]). Años más tarde, el trabajo fue continuado por el investigador Eder Fernandes en el marco de su TFM y su tesis doctoral en Brasil [153], consolidando lo que actualmente conocemos como BOFUSS, con soporte para OpenFlow 1.3 (Fernandes *et al.* [154]). Posteriormente, este software switch fue extendido por mi compañero de laboratorio, Constantin, quien implementó capacidades de comunicación *in-band* en entornos cableados (Constantin *et al.* [150]). A partir de esta última versión, y teniendo en cuenta las lecciones aprendidas de IoTorii (Rojas *et al.* [70]), se desarrolló *win-BOFUSS*, una implementación del software switch con soporte *in-band* en redes inalámbricas de baja capacidad. En la Figura 4.5 se muestra la evolución histórica del desarrollo del switch, desde su origen en 2008 con la propuesta del protocolo OpenFlow, hasta la actual implementación de *win-BOFUSS* (Carrascal *et al.* [140]).

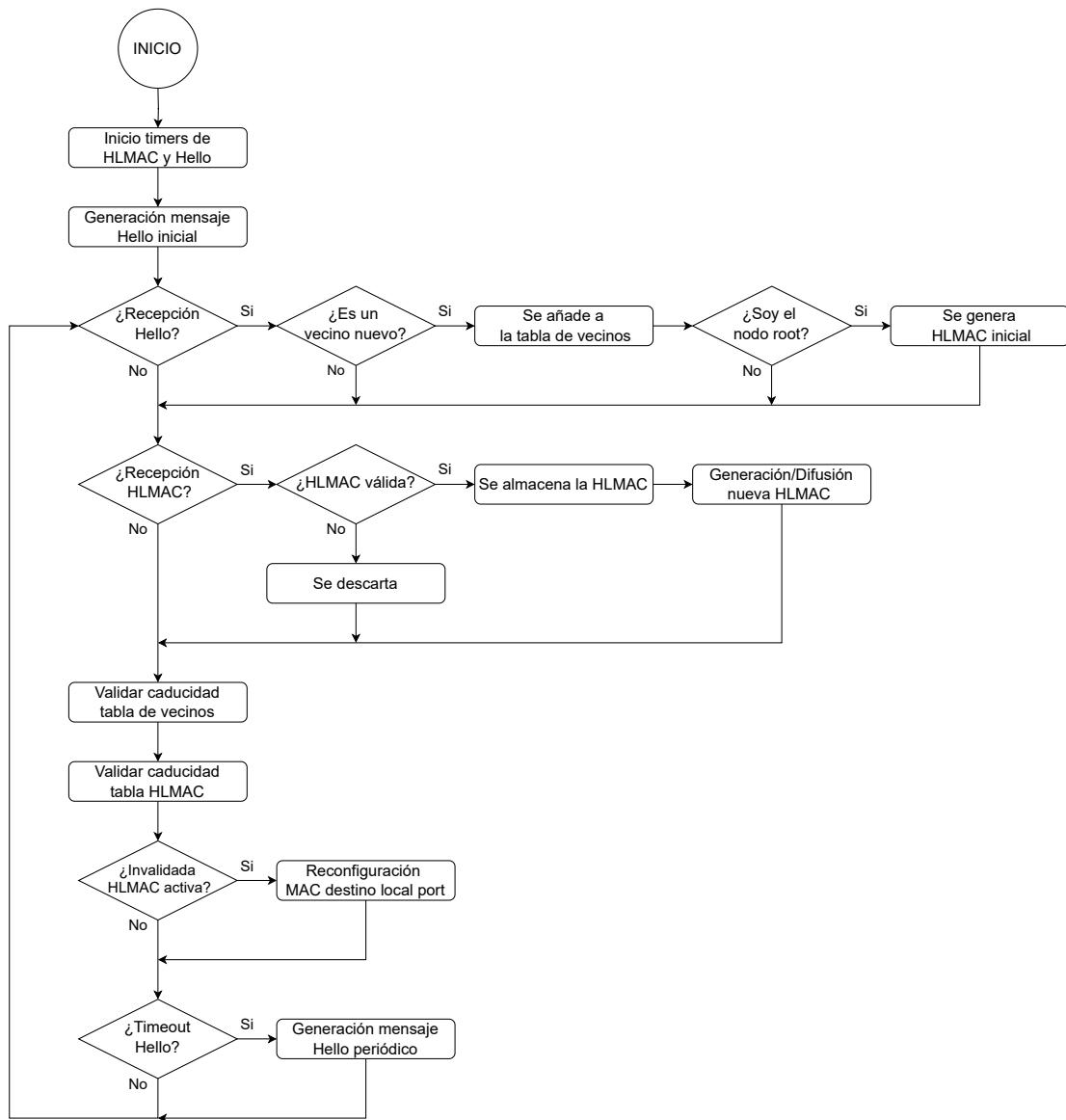


**Figura 4.5:** Evolución cronológica del desarrollo del software switch BOFUSS.

En la Figura 4.6 se muestra el diagrama de flujo que recoge la implementación de la lógica del protocolo. Por defecto, BOFUSS no arranca con la funcionalidad *in-band* habilitada, por lo que al invocar el proceso en espacio de usuario, donde, entre otras cosas, se especifican las interfaces a gestionar, es preciso activar explícitamente la lógica *in-band* mediante el parámetro `-inband`. Tras la inicialización del protocolo, el primer paso consiste

#### 4.3 IMPLEMENTACIÓN DE WIRELESS IN-BAND BASIC OPENFLOW SOFTWARE SWITCH (WIN-BOFUSS)

en configurar los temporizadores: los asociados al envío periódico de mensajes *Hello* y los vinculados a las entradas de las tablas de vecinos y de HLMAC. A continuación, cada nodo utiliza su interfaz inalámbrica para emitir el primer mensaje *Hello* y así descubrir qué dispositivos se encuentran dentro de su área de cobertura, estableciendo, de ese modo, las relaciones de vecindad iniciales. Completada la fase de inicialización, el sistema entra en el bucle principal de la lógica del protocolo, que gestiona el refresco de temporizadores, la recepción y procesamiento de *Hello*s y HLMACs, y las acciones de mantenimiento de tablas y rutas.



**Figura 4.6:** Diagrama de flujo de la implementación sobre el software switch BOFUSS.

En el bucle principal se gestionan de forma *event-driven* los siguientes sucesos: recepción de un mensaje *Hello*, recepción de una HLMAC, expiración de entradas en la tabla de vecinos, expiración de entradas en la tabla de HLMAC, verificación de la ruta principal ( posible reconfiguración) y vencimiento del temporizador periódico de envío de *Hello*. Al recibir un *Hello* se comprueba si el emisor es un vecino nuevo; en ese caso se inserta la entrada correspondiente en la tabla de vecinos y se inicia su temporizador. Si el emisor es el nodo raíz (*root*), además se genera el HLMAC inicial que pone en marcha la asignación jerárquica de etiquetas en la topología. Al recibir una HLMAC, se valida que no represente un bucle; si es válida se almacena en la tabla de HLMAC y, acorde con el procedimiento descrito en la Sección 4.2.2, se difunde a los hijos correspondientes. Posteriormente, se evalúan los criterios para activar la dirección HLMAC preferente y se actualizan las rutas. Finalmente, en cada iteración se comprueba la caducidad de las entradas de las tablas (que se refrescan con la llegada de mensajes coincidentes) y se actúa en consecuencia: eliminar vecinos o HLMAC caducados, o forzar una reconfiguración si la ruta principal ha dejado de ser válida.

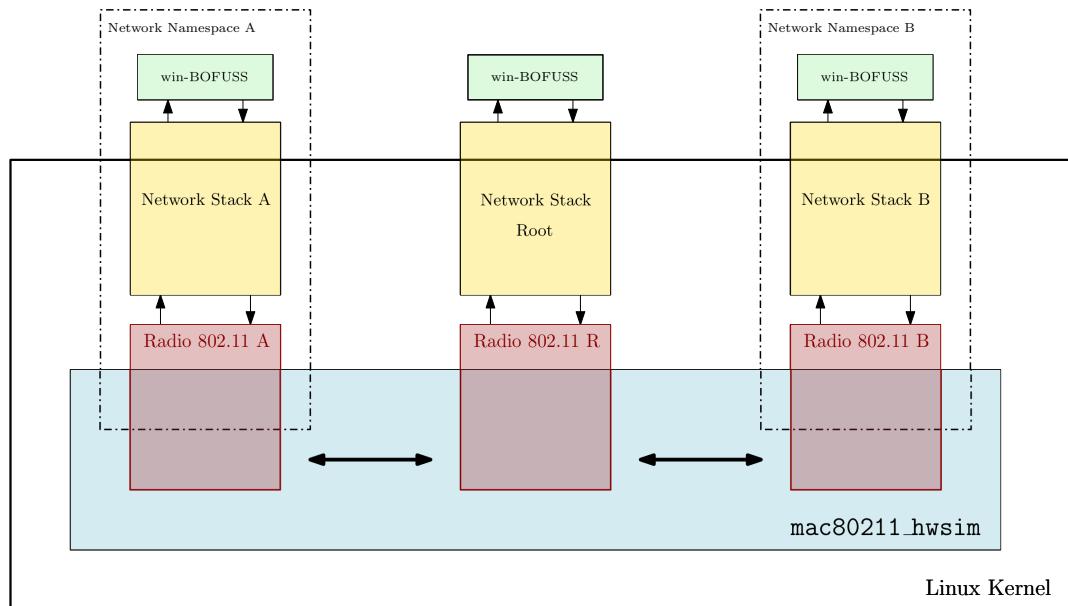
Cuando una ruta almacenada en la tabla HLMAC deja de renovarse, es decir, el nodo correspondiente al siguiente salto deja de enviar mensajes *Hello*, la entrada se invalida y el mecanismo elige la siguiente ruta disponible en la misma tabla. Dado que la interfaz física no cambia (misma interfaz inalámbrica), el puerto local permanece, pero sí varía el *next-hop*: es necesario actualizar la dirección MAC de destino asociada a la IP del siguiente salto en la pila del sistema para que los paquetes salientes alcancen al nuevo vecino. En sistemas Linux esta actualización se puede realizar mediante Netlink, concretamente enviando un mensaje del tipo RTM\_NEIGH para modificar la caché ARP. El procedimiento general es el siguiente: abrir un socket Netlink hacia el kernel, construir el mensaje RTM\_NEIGH con los atributos adecuados (IP objetivo y nueva dirección MAC de enlace), enviarlo y gestionar la respuesta/confirmación del *kernel*. Tras esta operación, los paquetes destinados a la IP en cuestión deberán resolverse hacia la nueva MAC. En la implementación actual se ha observado una limitación práctica: la actualización de la entrada ARP no siempre es suficiente para mantener transparente una conexión TCP ya establecida con el controlador, por lo que, como solución robusta y sencilla, el prototipo vuelve a establecer la sesión TCP tras actualizar la información de *next-hop*. En trabajos futuros se explorará una integración más fina (p. ej. manipulación de tablas de encaminamiento/vecindad sin interrumpir sockets existentes o el uso de opciones de *kernel* que permitan refrescar la resolución de enlace de forma totalmente transparente).

Por último, se verifica el vencimiento del temporizador periódico de *Hello*; si ha caducado, se emite un nuevo mensaje *Hello* que refresca el estado en las tablas de los vecinos. La resiliencia del protocolo depende en gran medida de su capacidad para detectar y recuperarse con rapidez ante fallos y cambios topológicos. Ajustando adecuadamente el periodo de los *Hello* y los tiempos de caducidad de las entradas, el protocolo puede identificar con rapidez enlaces caídos, nodos inalcanzables o la aparición de nuevos vecinos, reconstruir la tabla de vecinos y comutar a rutas alternativas cuando proceda. Este mecanismo reduce el impacto de las interrupciones, acelera la convergencia de la red y optimiza el uso de los recursos disponibles.

## 4.4. Evaluación

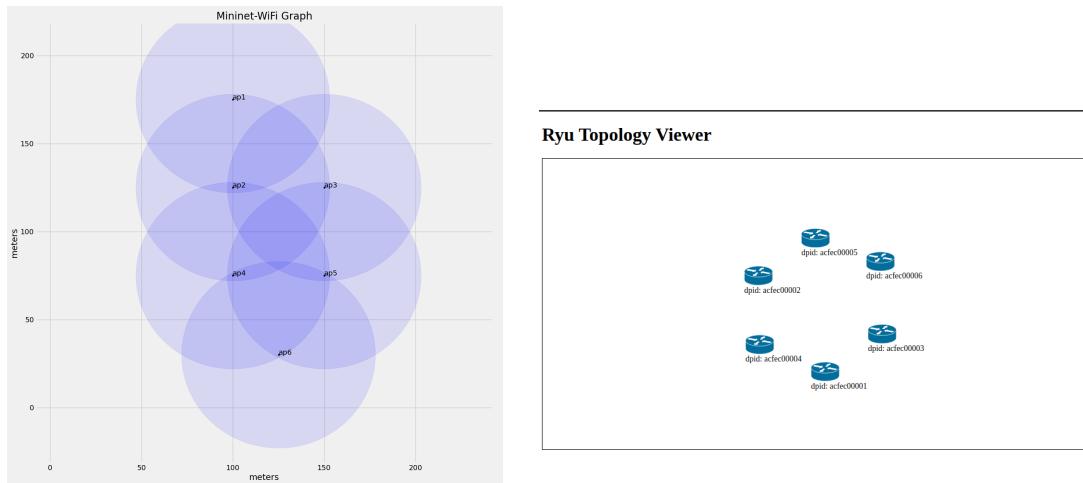
El banco de pruebas elegido para llevar a cabo la prueba de concepto del protocolo es un entorno de emulación que utiliza, como biblioteca base, el software switch implementado, `win-BOFUSS`. No obstante, BOFUSS no incorpora soporte inalámbrico nativo, por lo que se ha recurrido al módulo `mac80211_hwsim` para emular el medio inalámbrico. En este escenario, cada instancia de `win-BOFUSS` combinada con `mac80211_hwsim` actúa como un dispositivo inalámbrico IoT con una tarjeta Wi-Fi virtual, permitiendo emular escenarios inalámbricos en una única máquina física. La Figura 4.7 ilustra la combinación de estas herramientas en una máquina. Como puede observarse, cada dispositivo IoT se emula encapsulándolo en una *Network Namespace* para aislarlo (desde un punto de vista de red) del resto y forzar su comunicación mediante `mac80211_hwsim`. Este módulo define los radios Wi-Fi de cada tarjeta virtual y solo intercambia información entre nodos que se encuentran dentro de su área de cobertura.

Las características del medio radio (por ejemplo, limitaciones de transmisión por nodo) se configuran mediante el Traffic Controller (TC) de Linux. La información generada atraviesa las capas hasta llegar a la instancia de `win-BOFUSS`, que implementa la lógica del protocolo propuesto. Las topologías de los escenarios se construyen mediante un conjunto de shell-scripts que definen el *Network Namespace* aislado de cada dispositivo, su posición en un plano y su área de cobertura, de forma análoga a emuladores conocidos como Mininet o Mininet-WiFi [155], lo que facilita la definición y despliegue de los escenarios y la integración con herramientas auxiliares (por ejemplo, trazado y representación de topologías).



**Figura 4.7:** Entorno de validación mediante emulación empleando el módulo del *kernel* `mac80211_hwsim`.

Para la prueba de concepto de nuestro protocolo de control SDN *in-band*, denominado **win-BOFUSS**, se eligió la topología de la Figura 4.3 con el fin de validar el comportamiento descrito en la Sección 4.2. La Figura 4.8 muestra dicha topología empleando las herramientas de visualización: a la izquierda el visualizador de grafos de Mininet-WiFi (Ver Figura 4.8a) y a la derecha el visor de topología del controlador Ryu (Ver Figura 4.8b). Para comprobar el funcionamiento *in-band* de la propuesta, el controlador SDN Ryu se lanza en la *Network Namespace* raíz, en la cual solo se encontrará corriendo uno de los nodos de la topología, por lo que el resto tendrá que valerse del mecanismo *in-band* implementado para alcanzar al controlador. De esta forma, ejecutando la herramienta de descubrimiento de la topología de Ryu, si el funcionamiento de la implementación de mecanismo *in-band* es correcto, el controlador sería capaz de descubrir todos los nodos de la topología.



(a) Topología representada por la herramienta gráfica Mininet-WiFi. (b) Topología representada por la herramienta de descubrimiento topológico de Ryu.

**Figura 4.8:** Topología de ejemplo elegida para el banco de pruebas.

Como puede observarse en la Figura 4.8b, aunque solo un dispositivo mantiene una conexión directa al controlador, este descubre correctamente el resto de los dispositivos, lo que indica que las rutas *in-band* se han establecido con éxito y que el diálogo OpenFlow puede iniciarse. Hay que tener en cuenta que el visor de topología de Ryu no representa las interconexiones inalámbricas entre nodos, y además no diferencia el origen de los mensajes OpenFlow de cada dispositivo cuando todos ellos llegan a través de la misma sesión (la que mantiene el nodo raíz con el controlador). Por este motivo, en la visualización de Ryu los nodos aparecen aislados y no se muestran los enlaces inalámbricos explícitamente. En cualquier caso, la topología de ejemplo queda correctamente desplegada: todos los nodos son detectados por el controlador y las rutas *in-band* necesarias se han configurado.

Para analizar y verificar con mayor detalle el comportamiento del protocolo, se examinan las tablas generadas en cada dispositivo de la red. En primer lugar, se marca el nodo *A*

como raíz y se lanza el protocolo. Por un lado, se recopila la asociación ( $MAC \rightarrow ID$ ) creada en la tabla de vecinos por cada nodo tras el intercambio de mensajes *Hello*. Por otro lado, se registran las HLMAC almacenadas en cada nodo tras el proceso de inundación de etiquetas. Estos resultados se muestran en la Figura 4.9: la Figura 4.9a contiene el resultado del proceso de intercambio de *Hello*, y la Figura 4.9b ilustra las rutas HLMAC almacenadas en cada nodo. Como puede apreciarse, el análisis teórico realizado en la Sección 4.2 coincide con los resultados experimentales. Por ejemplo, en la Figura 4.9a, *A* ha detectado un único vecino al que ha asignado el ID 1, que corresponde al nodo *B*, tal y como en el ejemplo teórico. De igual modo, los nodos *B*, *D* y *F* presentan tres vecinos, mientras que *C* y *E* muestran dos. Respecto a las rutas HLMAC almacenadas, la Figura 4.9b reproduce los mismos resultados que el análisis teórico: por ejemplo, *D* ha almacenado las rutas *1.1.3*, *1.1.2.2.2* y *1.1.2.2.3.1*, lo que confirma que el protocolo funciona correctamente en el escenario emulado.

Nodo	Vecino 1	Vecino 2	Vecino 3
<i>A</i>	<i>B</i>	-	-
<i>B</i>	<i>A</i>	<i>C</i>	<i>D</i>
<i>C</i>	<i>A</i>	<i>B</i>	-
<i>D</i>	<i>A</i>	<i>B</i>	<i>E</i>
<i>E</i>	<i>A</i>	<i>B</i>	-
<i>F</i>	<i>A</i>	<i>B</i>	-

Nodo	Ruta 1	Ruta 2	Ruta 3
<i>A</i>	-	-	-
<i>B</i>	-	-	-
<i>C</i>	-	-	-
<i>D</i>	<i>1.1.3</i>	<i>1.1.2.2.2</i>	<i>1.1.2.2.3.1</i>
<i>E</i>	<i>1.1.3.3</i>	<i>1.1.3.2.3</i>	<i>1.1.2.2.3</i>
<i>F</i>	<i>1.1.3.2</i>	<i>0 1.1.2.2</i>	<i>0 1.1.3.3.2</i>

Figura 4.9: Prueba funcional de la implementación en la topología básica de ejemplo.

## 4.5. Conclusiones

En este trabajo hemos definido e implementado un protocolo de control *in-band* específicamente diseñado para su aplicación en entornos SDN inalámbricos, prestando especial atención a los dispositivos IoT con recursos restringidos en el borde de la red. La implementación se ha integrado con éxito en el switch de referencia BOFUSS, acuñándolo como *win-BOFUSS*, y se ha probado conjuntamente con la plataforma controladora Ryu. El protocolo requiere un número muy reducido de mensajes de señalización y mantiene compactas las entradas en las tablas de enrutamiento, lo que lo hace especialmente apropiado para dispositivos con capacidades de cómputo, memoria y energía limitadas. Además, ofrece mecanismos intrínsecos de resiliencia: se generan rutas de respaldo que permiten mantener la sesión de control y garantizar continuidad de servicio frente a fallos o escenarios de movilidad. En conjunto, esta aportación constituye una base sólida para futuros desarrollos orientados al despliegue del *cloud continuum* y al uso del etiquetado jerárquico para establecer encaminamiento autónomo.



# Capítulo 5

## DEN2NE

Una vez sentadas las bases del etiquetado jerárquico como mecanismo habilitador de exploración y creación de canales de control *in-band* en redes inalámbricas-SDN, con la siguiente propuesta se quiso ampliar el alcance, poniendo el foco en las redes densas, heterogéneas y distribuidas.

En este capítulo se presenta Distributed ENergy ENvironments and NEtworks (DEN2NE), un algoritmo escalable para la distribución y reasignación automática de recursos en entornos distribuidos, basado también en el etiquetado jerárquico. El origen del nombre hace referencia al Pokémon tipo eléctrico/hada, llamado Dedenne, el cual, no es capaz de generar electricidad por si mismo, por lo que utiliza su cola para absorber la electricidad de los hogares y recargarse. Esta idea surgió a la par que se empezó a gestar este trabajo, el cual originalmente, iba a ser una colaboración con el Departamento de Electrónica de la UAH, enfocado en el desarrollo de soluciones de encaminamiento inteligente en redes de distribución eléctrica.

Finalmente, la colaboración dio lugar a otro trabajo que se detallará más adelante en el Capítulo 7. Sin embargo, esta propuesta siguió adelante manteniendo el nombre de DEN2NE, dando lugar a una solución con un marco más general y aplicable a distintos dominios de redes densas y heterogéneas, convirtiéndose en una de las principales contribuciones de esta Tesis (Carrascal *et al.* [156]).

### 5.1. Introducción

En la era contemporánea, la globalización y los avances tecnológicos han propiciado una profunda interconexión a nivel mundial en diversos aspectos, tales como la comunicación, la logística, la provisión de servicios y la explotación de recursos naturales. Todos estos avances, englobados bajo el concepto de Internet del Todo (del inglés, Internet of Everything (IoE)) [157], se caracterizan por la presencia de redes densas y nodos heterogéneos, cada uno de los cuales busca intercambiar recursos en función de sus necesidades y capacidades específicas.

Según se pudo analizar en profundidad en la Sección 2.2.2.2, la gestión y la planificación de recursos en entornos densos y heterogéneos ha dado lugar a una familia diversa de soluciones, las cuales se diferenciarán en función del dominio de aplicación.

En lo que respecta a la gestión energética, la implementación de las redes eléctricas inteligentes [79, 80] (SG) ha permitido optimizar el aprovechamiento de la energía generada localmente, favoreciendo la autocompensación y mejorando la eficiencia en su distribución. De forma análoga, las redes de gas natural [158] han tenido que incorporar mecanismos avanzados de control de flujo [159] para interconectar de manera más eficiente a productores y consumidores. En el sector servicios, tanto la logística [82, 81] como las aplicaciones de *food delivery* [160, 161] han evidenciado la necesidad de optimizar las rutas de reparto con el fin de reducir costes y tiempos de entrega. Por su parte, en el ámbito de las comunicaciones, la expansión del IoT y la próxima generación de las redes 6G han intensificado la interconexión de dispositivos [126], apoyándose en tecnologías habilitadoras como el SDN, que proporcionan una gestión centralizada y flexible de la red. Dentro del ecosistema 6G, el paradigma de *fog/edge computing* introduce un nuevo escenario en el que el recurso a intercambiar es la capacidad de cómputo [74], lo que requiere de algoritmos y estrategias distribuidas que permitan la cooperación eficiente entre nodos. Sin embargo, la creciente densidad y heterogeneidad de dichos nodos plantea un desafío considerable, pues las diferencias en capacidades y requisitos dificultan la gestión óptima de los recursos. Tal y como se expuso en el Capítulo 3, abordar esta complejidad exige recurrir a algoritmos de optimización y a sistemas de gestión capaces de facilitar la toma de decisiones en la asignación de recursos, garantizando así un intercambio justo, beneficioso y equilibrado entre los diferentes actores que conforman estas redes interconectadas.

Es por ello, que se presenta DEN2NE, un algoritmo novedoso para el reparto y la reasignación automática de recursos en entornos distribuidos. Aunque inicialmente se pensó para distribuir energía entre prosumidores en redes inteligentes, DEN2NE puede ser aplicado para cualquier entorno distribuido donde los nodos tengan que intercambiar algún recurso o colaborar en alguna tarea con objetivos comunes. Un ejemplo típico son las redes de *fog computing*, donde los dispositivos en el *edge* suelen tener limitaciones de memoria o energía, por lo que la redistribución de tareas es fundamental y debe realizarse en el momento oportuno.

La principal contribución de DEN2NE radica en la capacidad del algoritmo de descubrir los nodos y los recursos asociados en redes densas de múltiples saltos, esbozando posteriormente un esquema de distribución eficiente. Todo ello se logra de manera efectiva y escalable, teniendo en cuenta la topología de la red. Más específicamente, y en comparación con el estado del arte (analizado en la Sección 2.2.2.2), esta contribución puede desglosarse en los siguientes puntos:

- Descubrimiento de topologías malladas mediante un procedimiento generalizado de etiquetado jerárquico, lo suficientemente versátil como para aplicarse a cualquier tipo de red.
- Identificación de una o varias rutas desde cada nodo de la red hasta el sumidero de

la topología (nodo *root*), lo que permite establecer caminos de respaldo en caso de fallo de nodos.

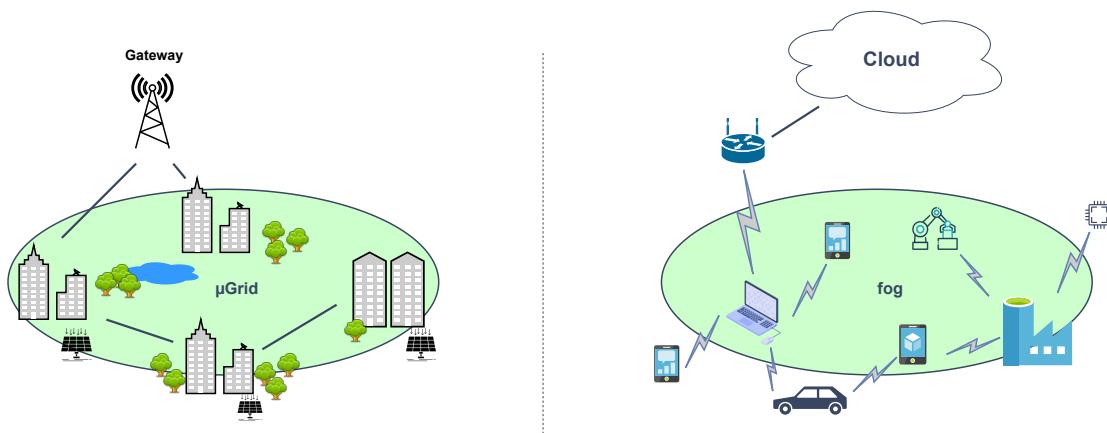
- Desarrollo de un procedimiento de asignación de recursos adaptado a redes densas de múltiples saltos, garantizando al mismo tiempo la escalabilidad (baja complejidad y tiempo de convergencia reducido).
- Diseño de seis criterios diferenciados que permiten adaptar el algoritmo a distintos tipos de escenarios en red, en función del recurso que se desee balancear.

A partir de estas contribuciones, cabe destacar la versatilidad, la resiliencia y la escalabilidad que caracterizan a DEN2NE, considerando además entornos de multi-salto, lo cual representa un enfoque particularmente novedoso en el área.

## 5.2. Definición del algoritmo

Para explicar los objetivos y principios de diseño de nuestro algoritmo, resulta necesario en primer lugar contextualizar su motivación a través de una red compuesta por múltiples nodos que son heterogéneos y/o con recursos limitados, en la que una o varias tareas deben completarse en un tiempo determinado. Cada una de estas tareas no puede ser realizada por un único nodo de manera individual. Por tanto, en este tipo de redes, los nodos deben necesariamente delegar y cooperar para completar las tareas de forma conjunta; al mismo tiempo, deben considerar las necesidades de los demás con el fin de que la redistribución resulte lo suficientemente justa.

La Figura 5.1 ilustra dos ejemplos de estas redes. En el lado izquierdo se representa una *microgrid*. Los sistemas de *microgrid* son redes energéticas formadas por un conjunto de prosumidores, nodos que pueden actuar tanto como productores como consumidores de electricidad, normalmente porque disponen de una fuente renovable instalada (por ejemplo, paneles solares).



**Figura 5.1:** Dos ejemplos de redes que aprovecharían DEN2NE para equilibrar los recursos.

Estos sistemas necesitan redistribuir la producción excedente entre otros consumidores, y dicha distribución energética debe realizarse lo más cerca posible de la fuente para minimizar las pérdidas por transmisión. En la parte derecha se representa una red de *fog computing*, compuesta por diversos dispositivos IoT, como ordenadores portátiles, teléfonos móviles, automóviles o robots industriales. En este último escenario, un nodo IoT podría necesitar ejecutar una tarea, y descargarla (hacer un *offload*) entre otros dispositivos cercanos con recursos de cómputo disponibles que podrían acelerar su finalización. Como puede observarse, ambas implementaciones requieren algoritmos análogos para compartir una o varias acciones/tareas que deben completarse (por ejemplo, producción de energía, cómputo de una tarea, etc.). Además, ambos ejemplifican una porción de una red mucho mayor, lo que implica que disponen de una conexión (p. ej., un *gateway*) con el resto de esa red global. Por ejemplo, una *microgrid* podría usar el *gateway* para enviar o recibir energía del resto del despliegue de la red inteligente, y lo mismo ocurre en el entorno *fog*, que podría delegar la resolución de la tarea en la nube en el peor de los casos (es decir, cuando todos los nodos están saturados y no pueden procesarla localmente).

En consecuencia, el objetivo principal de nuestro algoritmo es redistribuir de forma eficiente recursos entre cualquier tipo de nodos conectados. Para ello, cualquier nodo que actúe como fuente puede delegar un recurso a cualquier nodo destino. En este sentido, la formulación del problema se asemeja a un problema de encaminamiento (según se estudió en la Sección 2.2.2.3), con una diferencia: en el encaminamiento hay una información que debe entregarse desde un origen concreto a un destino concreto y solo es necesario calcular la ruta; en este caso, aunque existe un origen y una información definida, los destinos potenciales pueden ser múltiples, y deben decidirse conjuntamente con la ruta que seguirá la información/recurso a distribuir. Dicho de otro modo, un recurso puede compartirse entre uno o varios nodos destino, y esa decisión debe calcularse por el algoritmo, junto con las rutas para alcanzarlos.

Para cumplir el objetivo anterior, el diseño del DEN2NE se sustenta en tres principios fundamentales:

1. **Tiempo de rápido convergencia:** la decisión sobre el reparto de recursos debe computarse con rapidez; es decir, el algoritmo debería presentar un tiempo de convergencia bajo. Además, dicho tiempo no debería verse penalizado de forma drástica por la cantidad de datos/recursos disponibles o por el número de nodos de la red; idealmente crecerá de manera cercana a lineal con el tamaño de la red ( $\mathcal{O}(n)$ ). Dado que, tiempos de convergencia cortos facilitan actualizaciones ágiles en escenarios de alta movilidad.
2. **Baja complejidad:** dado que los nodos que ejecutarán el algoritmo suelen ser dispositivos con recursos limitados (memoria, batería, CPU), el método debe exigir poca capacidad de cómputo y almacenamiento, e implicar un número reducido de mensajes de control intercambiados.
3. **Versatilidad:** además de escalable, el algoritmo debe ser fácil de adaptar a distintos entornos prácticos (redes cableadas e inalámbricas, SG, entornos IIoT, enfoques

centralizados o distribuidos, etc.). En concreto, su diseño debe evitar supuestos demasiado restrictivos sobre la infraestructura, para permitir su reutilización en escenarios variados.

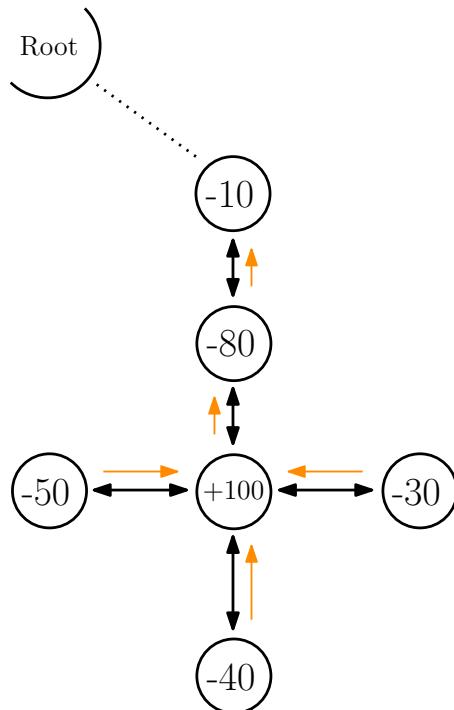
Sobre la base de estos tres principios, se tomó como punto de partida el protocolo de encaminamiento IoTorii de Rojas *et al.* [70] (Analizado en la Sección 2.2.1.2), ya que cumple con requisitos de convergencia rápida y bajo consumo de recursos. Aunque IoTorii está concebido como protocolo de encaminamiento, su naturaleza distribuida facilita transformarlo a una versión centralizada si fuera necesario. No obstante, dado que IoTorii selecciona rutas para pares origen–destino concretos, ha de adaptarse para satisfacer los objetivos del DEN2NE, que no asume un conjunto de destinos predefinidos de partida. En particular, el DEN2NE se diseña para localizar los vecinos más cercanos a los que se puede delegar una tarea o recurso. En este escenario se asume la existencia, de al menos, un nodo con acceso a recursos «ilimitados», que actúe como sumidero de recursos de la red; a dicho nodo lo denominamos *gateway* o nodo raíz (*root*). Los *gateway* pueden asumir la porción restante de la tarea o recurso cuando el *offloading* local no sea factible. En las siguientes sub-Secciones se describirá en detalle el algoritmo DEN2NE.

### 5.2.1. Terminología del algoritmo

Una vez presentada la motivación del algoritmo y antes de profundizar en su descripción, definiremos algunos términos que se emplearán posteriormente. Para mayor claridad, nos centraremos en la Figura 5.2, como ejemplo de escenario de aplicación. En dicho escenario, contamos con seis nodos (representados como círculos) conectados a un nodo raíz (*root*). Los nodos están conectados bidireccionalmente (ya sea de forma cableada o inalámbrica), como indican las flechas negras bidireccionales más oscuras, y sus números simbolizan un «recurso» que debe ser distribuido o balanceado. Más concretamente, dicho recurso puede representar una **oferta** o una **demand**a (por ejemplo, producción/consumo eléctrico o capacidades computacionales disponibles o requeridas). En el caso de la Figura 5.2, los valores negativos expresan una demanda, mientras que los positivos expresan una oferta. En este sentido, por ejemplo, una demanda de -80 se compensaría con una oferta equivalente de +80 o, alternativamente, con una combinación de ofertas que sumaran +80. Por lo tanto, el objetivo principal de nuestro algoritmo es compensar todos los recursos en la red, es decir, que todos los nodos tiendan hacia el valor cero (un uso óptimo), mientras que el remanente (ya sea exceso de demanda u oferta) se redirija hacia el nodo raíz.

Antes de describir las tres fases del algoritmo DEN2NE (en las Secciones 5.2.2, 5.2.3, y 5.2.4, respectivamente), reflexionemos sobre el ejemplo de la Figura 5.2 para entender por qué se definieron tres fases. Considerando los principios de diseño mencionados anteriormente, calcular todas las posibles combinaciones de reparto no es factible en un tiempo razonable. Por otro lado, la información local podría reducir los tiempos de convergencia y la complejidad, pero tampoco sería suficiente para decidir de manera óptima. Como ejemplo de una mala decisión, el nodo con una demanda de -80 solo conoce a sus dos vecinos: uno que ofrece (+100) y otro que demanda (-10), y podría pensar que el nodo

más adecuado para cubrir su demanda (-80) es el vecino inferior (+100) (en la Figura 5.2). Sin embargo, esta decisión local resulta ser incorrecta, ya que existen otros nodos con demandas adicionales (-50, -40 y -30), que también requieren de esa oferta y se encuentran mucho más alejados del nodo raíz. De hecho, la solución de distribución óptima se ilustra con flechas naranjas claras (que representan la dirección desde el nodo demandante hacia el nodo con oferta). Esta distribución implicaría que todos los nodos quedaran balanceados en cero y que la demanda restante de -110 (suma de -50 -40 -30 +100 -80 -10) se desviara hacia el nodo raíz, el cual, sería capaz de cubrirla.



**Figura 5.2:** Ejemplo de equilibrio de recursos en una red con seis nodos y un *gateway* (o nodo raíz).

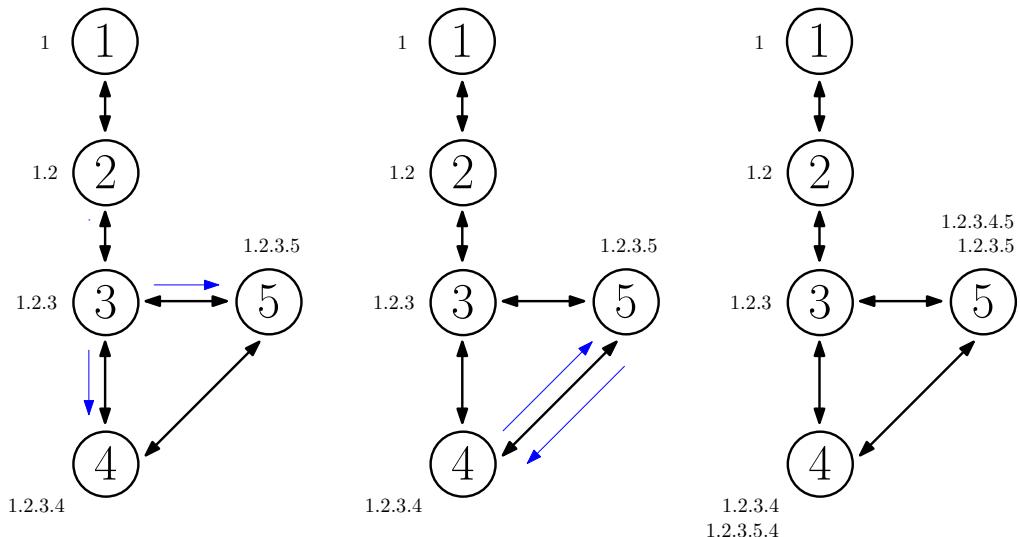
Consecuentemente, el algoritmo DEN2NE calcula la distribución de recursos considerando información de todos los nodos atravesados, en lugar de limitarse únicamente a la información local de los vecinos. Además, prioriza la distribución en los nodos más alejados del nodo raíz, arrastrando el remanente (oferta o demanda) hacia la raíz para que pueda ser compensado allí por el resto de la red (por ejemplo, la nube o la *smartgrid*).

En las subsecciones siguientes se describen en detalle las tres fases que componen el algoritmo. La primera fase está dedicada a explorar la topología y agregar información de los nodos recorridos mediante etiquetado jerárquico; la segunda selecciona, entre todas las etiquetas recibidas, el conjunto de información que parece más cercano a la solución óptima; y la tercera ejecuta el balance de recursos en función de la decisión tomada en la fase segunda. Más concretamente:

- En la **Fase 1** (Subsección 5.2.2) se explica el etiquetado jerárquico para calcular rutas desde cada nodo hasta el nodo raíz. Esta fase se inspira en los trabajos relacionados sobre encaminamiento escalable y resiliente examinados en la Sección 2.2.1.2.
- En la **Fase 2** (Subsección 5.2.3), una vez que se han generado todas las rutas posibles hacia la raíz, se selecciona, en cada nodo, una de ellas siguiendo un criterio concreto. Este criterio proporciona la versatilidad prevista como principio de diseño: proponemos seis criterios para que el más adecuado se aplique según el escenario, aunque en la práctica cualquier otro criterio nuevo podría emplearse si fuese necesario.
- En la **Fase 3** (Subsección 5.2.4), tras la selección de la ruta activa ( $ID_{activa}$ ) en la fase anterior, se realiza el balanceo de recursos en base a la lista de nodos representada por las etiquetas activas. Esta fase tiene en cuenta además las condiciones reales de la red, por ejemplo, si los enlaces presentan pérdidas o capacidades limitadas.

### 5.2.2. Fase 1 - Etiquetado jerárquico

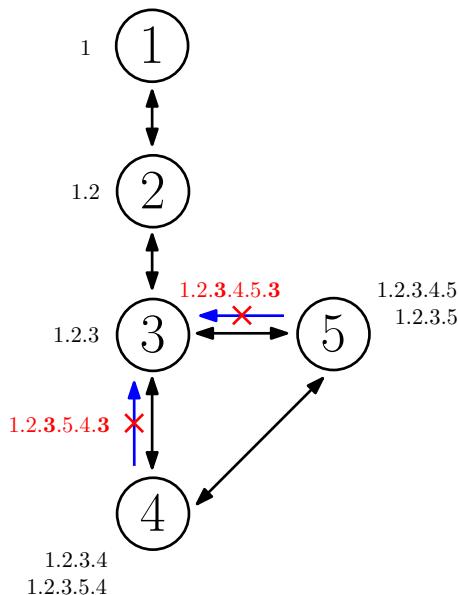
El algoritmo DEN2NE comienza localizando las posiciones relativas de cada nodo en la red. Para ello, se realiza un etiquetado jerárquico a partir de cada nodo raíz presente en el despliegue. Como algoritmo centralizado, cada nodo dispone de un identificador numérico único (ID) que les identifica en la red, y obtienen durante el proceso una etiqueta jerárquica que les indicará su posición en la topología, y una ruta para alcanzar el sumidero de la misma. Este mecanismo de etiquetado jerárquico comparte ciertos aspectos con el descrito en el Capítulo 4, sin embargo, en vez de emplear un etiquetado basado en sufijos auto-asignados en procesos de reconocimiento de vecindad, se emplearán identificadores únicos, aligerando el proceso de exploración de la topología. La Figura 5.3 presenta el esquema de etiquetado diseñado.



**Figura 5.3:** Procedimiento de etiquetado jerárquico en DEN2NE.

El procedimiento parte en el nodo raíz, que en la Figura 5.3 tiene ID 1 y un vecino conectado con ID 2. Para correlacionarlos jerárquicamente, el algoritmo asigna la etiqueta 1 al raíz y concatena el ID del vecino a la suya propia, de modo que el nodo 2 recibe la etiqueta 1.2. A continuación, el nodo con ID 3 obtendrá la etiqueta 1.2.3, y los nodos con IDs 4 y 5 recibirán las etiquetas 1.2.3.4 y 1.2.3.5, respectivamente. Además, conforme progresó la asignación, el nodo con ID 4 también podrá registrar la etiqueta 1.2.3.5.4 y el nodo con ID 5 la etiqueta 1.2.3.4.5, según correspondan las rutas alternativas. Este procedimiento de etiquetado continúa hasta que se explora toda la red y cada nodo dispone, al menos, de una etiqueta jerárquica. En la práctica, cada etiqueta representa una ruta para alcanzar el nodo raíz, por lo que, nodos con múltiples rutas tendrán capacidad de balancear sus recursos por rutas alternativas.

Para evitar bucles durante la fase de etiquetado jerárquico, el algoritmo no asigna a un nodo etiquetas que hereden de sí mismo. Esto se puede prever fácilmente ya que las etiquetas concatenan los IDs de los nodos recorridos, por lo que ninguna etiqueta debe contener un mismo ID repetido; de lo contrario, indicaría la existencia de un bucle en la ruta. Por ejemplo, la Figura 5.4 ilustra gráficamente esta restricción y muestra el siguiente paso respecto a lo representado en la Figura 5.3. Cuando el nodo con ID 4 recibe la etiqueta 1.2.3.5.4 del nodo con ID 5, podría intentar propagar la etiqueta 1.2.3.5.4.3 hacia el nodo con ID 3, pero esta acción no se realiza porque la etiqueta contiene el ID 3 en dos ocasiones, lo que representaría un bucle. Una vez descartada dicha etiqueta, no se propagarán más etiquetas que la deriven. De manera análoga, si el nodo con ID 5 generara 1.2.3.4.5.3 para el nodo 3, esta también sería rechazada por el mismo motivo. En este ejemplo concreto, tras estos pasos descritos, no se asignan más etiquetas y el procedimiento del etiquetado jerárquico habría concluido en este punto.



**Figura 5.4:** Mecanismo de evitación de bucles durante el etiquetado jerárquico.

Como se indicó anteriormente, cada etiqueta asignada representa una ruta hacia la raíz y cada nodo obtiene una o varias etiquetas (es decir, múltiples rutas hacia el nodo raíz). Por tanto, esta primera fase del algoritmo puede calcular, en principio, todas las rutas posibles desde cualquier nodo de la red hasta la raíz. Esto puede conllevar miles o incluso millones de rutas, según la conectividad de la topología. Por ello, desde el punto de vista de la implementación, la fase de etiquetado jerárquico suele limitarse a aprender únicamente un subconjunto de etiquetas y descartar el resto siguiendo ciertos parámetros, como la disjunción de rutas o un número mínimo de saltos [45]. El Algoritmo 1 resume todo el procedimiento de la Fase 1, el cual, toma como entrada el grafo de la red y el nodo desde el que se inicia el etiquetado jerárquico (es decir, el nodo raíz) y devuelve el grafo con la asignación de etiquetas correspondiente. El algoritmo recorre los nodos partiendo de los vecinos del nodo raíz. Un nodo puede ser procesado más de una vez, hasta un máximo de etiquetas asignadas (IDs\_MAX) o hasta que se detecte un bucle (*check\_loop()*). Por este motivo, podemos afirmar que la complejidad de la fase de etiquetado es  $\mathcal{O}(n)$ , ya que crece de forma lineal con el número de nodos  $n$ , cada uno visitado como máximo IDs\_MAX veces; dado que IDs\_MAX se considera una constante de diseño, la complejidad asintótica se mantiene en  $\mathcal{O}(n)$  independientemente de la forma o tamaño de la topología.

---

**Algoritmo 1:** Proceso de etiquetado jerárquico

---

```

input :  $G(\mathcal{N}, \mathcal{L})$ ,  $root$ 
output:  $G(\mathcal{N}, \mathcal{L})$ 

1 init  $nodes\_to\_attend$ ;
2  $\mathcal{N}(root).push(ID(None, root))$ ;
3  $nodes\_to\_attend.push(\mathcal{N}(root))$ ;

4 while  $length(nodes\_to\_attend) > \emptyset$  do
5    $curr\_node = nodes\_to\_attend[0]$ ;
6   for  $j \leftarrow length(curr\_node.ids)$  do
7     if  $not curr\_node.ids[j].used$  then
8       for  $neighbor \leftarrow curr\_node.neighbors$  do
9         if  $check\_loop(curr\_node.ids[j], neighbor)$  then
10          | pass;
11        else if  $neighbor.ids \geq IDs\_MAX$  then
12          | pass;
13        else
14          |  $\mathcal{N}(neighbor).push(ID(curr\_node.ids[j], neighbor))$ ;
15          |  $nodes\_to\_attend.push(\mathcal{N}(neighbor))$ ;
16        curr_node.ids[j].used = true;
17      pop  $nodes\_to\_attend[0]$ ;

18 return  $G(\mathcal{N}, \mathcal{L})$ 

```

---

### 5.2.3. Fase 2 - Selección de las mejores etiquetas o identificadores jerárquicos

Como se ha presentado anteriormente, los nodos raíz o *gateways* son aquellos conectados a recursos externos (energía, capacidad de cómputo, etc.) que pueden considerarse prácticamente «ilimitados». Por este motivo, al repartir una tarea o recurso, los nodos más alejados del nodo raíz tienen menor probabilidad de disponer de recursos locales, mientras que los nodos próximos a la raíz pueden delegar con facilidad en el *gateway* para completar la operación. La primera fase del algoritmo aporta precisamente la localización relativa de los nodos respecto a la raíz/raíces, lo que permite priorizar de forma natural qué nodos deben considerarse preferentes a la hora de decidir dónde delegar una tarea; en suma, simplifica y ordena el proceso de toma de decisiones.

Concretamente, una vez finalizado el etiquetado jerárquico, cada nodo dispone de una o varias etiquetas que representan rutas hacia la raíz, dado que cada etiqueta codifica la lista de nodos a atravesar para alcanzarla. Si un nodo cuenta con una única etiqueta, existe un único camino conocido hacia la raíz; si dispone de varias, hace falta aplicar criterios de selección para escoger la ruta más adecuada. Dado que las rutas hacia la raíz se construyen de forma coordinada entre nodos, ese criterio de selección debe aplicarse respetando un cierto orden y coordinación entre los participantes.

La segunda fase del algoritmo, por tanto, decide dónde distribuir los recursos aplicando criterios sobre las etiquetas o identificadores jerárquicos obtenidos en la primera fase. Como referencia, el algoritmo DEN2NE incorpora seis criterios distintos (basados en la literatura) para puntuar cada ruta disponible: cada criterio calcula un coste asociado a una etiqueta concreta (denotado  $Cost_{ID}$ ). A continuación, se selecciona la ruta activa, el identificador jerárquico activo  $ID_{activa}$ , como la que minimiza dicho coste, tal y como se expresa en la Ecuación 5.1.

$$\langle ID_{activa} \rangle = \min(Cost_{ID}) \quad (5.1)$$

Los seis criterios se describen en las secciones siguientes. Es importante remarcar que pueden definirse e incorporar criterios adicionales al algoritmo: la segunda fase únicamente exige la especificación de una función de coste para cada criterio, por lo que la elección concreta no limita el funcionamiento del método y, de hecho, dota de versatilidad a la propuesta. Asimismo, algunos criterios resultan aplicables de forma genérica a múltiples casos de uso, mientras que otros son específicos de dominios concretos (p. ej. SG, *fog computing*, etc.). Por estas razones, los criterios se presentan de forma concisa y directa a modo de referencia; en la práctica, DEN2NE es potencialmente agnóstico respecto al criterio seleccionado.

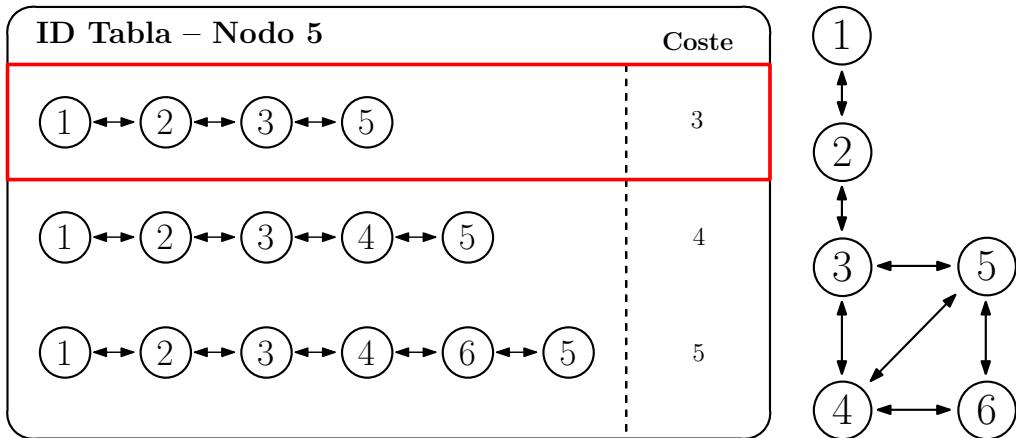
#### 5.2.3.1. Criterio 1 - Número de saltos

Este primer criterio evalúa el número de saltos necesarios para alcanzar el nodo raíz, de modo que en cada nodo se selecciona el identificador jerárquico con menor número de

saltos. Se basa en el clásico criterio de camino más corto para encaminamiento en redes de comunicaciones, denominado *widest-shortest path*, Ma *et al.* [162]. Por tanto, la función de coste para este criterio puede expresarse como en la Ecuación 5.2.

$$Cost_{ID} = \text{length}(ID) - 1 \quad (5.2)$$

Para comprenderlo, la Figura 5.5 muestra un ejemplo en el que el nodo 5 ha adquirido tres identificadores jerárquicos y se representan las tres rutas potenciales hacia la raíz (nodo 1). En este escenario, el coste se calcula como la longitud del identificador (donde la longitud es el número de cifras en la etiqueta jerárquica) menos uno; por ejemplo, para el identificador 1.2.3.5 el coste es  $3 = 4 - 1$ , dado que dicho identificador tiene 4 cifras. Una vez calculado el coste para cada identificador jerárquico, el identificador seleccionado en el ejemplo es 1.2.3.5 conforme a la Ecuación 5.1, pues presenta el menor coste, lo que implica menos saltos para alcanzar la raíz. En caso de fallo o indisponibilidad de esa ruta principal, cualquiera de las rutas restantes podría emplearse, ordenadas por coste.



**Figura 5.5:** Selección de ID basada en el criterio del número de saltos.

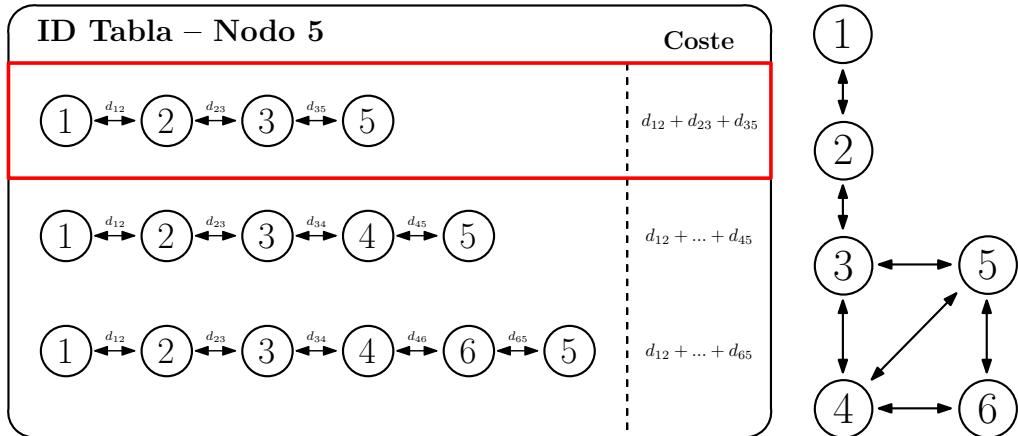
### 5.2.3.2. Criterio 2 - Distancia

El segundo criterio incorpora el concepto de distancia entre nodos. Dicha distancia puede ser física (por ejemplo, en despliegues de *microgrids*, donde la distancia física es conocida y constante porque los nodos no son móviles) o lógica (por ejemplo, un coste de enlace basado en el rendimiento o la latencia en un despliegue de red). Al igual que el criterio anterior, se inspira en técnicas clásicas de encaminamiento en redes de comunicaciones y recibe el nombre de *shortest-distance path*, Ma *et al.* [162]. En este caso, el coste de cada identificador jerárquico se calcula como la suma de las distancias de los enlaces recorridos, tal como se expresa en la Ecuación 5.3 (una “distancia” puede equivaler también al inverso del *throughput* del enlace u otra métrica relacionada).

$$Cost_{ID} = \sum_{i=0}^{N_{saltos}-1} d_i \quad (5.3)$$

La Figura 5.6 muestra el mismo ejemplo anterior aplicado con este segundo criterio. En dicho escenario, el identificador 1.2.3.5 resulta seleccionado si se cumple la condición indicada en la Ecuación 5.4; no obstante, dependiendo de los valores reales de las distancias, podría seleccionarse cualquier otro identificador aunque la ruta tenga más saltos hasta la raíz.

$$\langle ID_1 = ID_{activa} \rangle \Leftrightarrow \begin{cases} [d_{35} < (d_{34} + d_{45})] \\ \quad \wedge \\ [d_{35} < (d_{34} + d_{46} + d_{65})] \end{cases} \quad (5.4)$$



**Figura 5.6:** Selección de ID basada en el criterio de distancia.

### 5.2.3.3. Criterio 3 - Pérdidas de enlace

El tercer criterio está directamente relacionado con el caso de uso de redes inteligentes de distribución eléctrica (inspirado en la topología IEEE 123-node test feeder [163]) en el que puede aplicarse este algoritmo, y se vincula asimismo con el criterio anterior sobre distancia. Concretamente, se persigue la minimización de las pérdidas en los enlaces debidos a la transmisión. Para distinguirlo del criterio 2, conviene subrayar que el criterio 2 suele centrarse en características del enlace o su capacidad (p. ej. rendimiento o distancia), mientras que el criterio 3 atiende al efecto de degradación del enlace (p. ej. pérdidas de potencia). No obstante, ambos criterios están estrechamente relacionados y, de hecho, su formulación es conceptualmente similar, intercambiando distancias por pérdidas, como se muestra en la Ecuación 5.5.

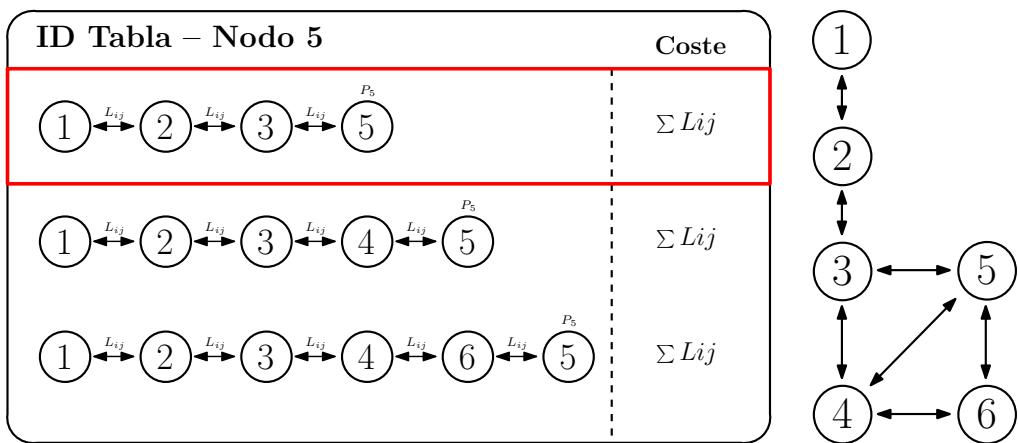
$$Cost_{ID} = \sum_{ij}^{N_{ID}} L_{ij} = \sum_{ij}^{N_{ID}} \left( \frac{\alpha_{R_{ij}} \times d_{ij}}{V_d^2} \times P_{in}^2 \right) \quad (5.5)$$

En el caso específico de las redes eléctricas, las pérdidas se calculan a partir de modelos de pérdidas en líneas de transmisión que consideran parámetros tales como: tensión ( $V_d$ ), distancia física ( $d_{ij}$ ), coeficiente de reflexión ( $\alpha_{R_{ij}}$ ) y potencia de entrada ( $P_{in}$ ), tal y como recoge la Ecuación 5.6 (incorporada en la Ecuación 5.5). Como puede apreciarse, la distancia física es uno de los parámetros que afectan al coste, pero no el único. Por ello, como hemos mencionado antes, los criterios 2 y 3 están relacionados pero no necesariamente producen el mismo resultado, pues intervienen factores adicionales. En otros escenarios, las “pérdidas” podrían interpretarse de forma análoga (p. ej. fiabilidad de enlace en un entorno inalámbrico para computación).

$$L_{ij} = \frac{\alpha_{R_{ij}} \times d_{ij}}{V_d^2} \times P_{in}^2 \quad (5.6)$$

En el ejemplo concreto de la Figura 5.7, la ruta hacia el nodo raíz seleccionada como  $ID_{activa}$  es la primera (1.2.3.5), que en este caso coincide con la escogida por el criterio anterior. Siempre y cuando, la condición indicada en la Ecuación 5.7 se cumpla.

$$\langle ID_1 = ID_{active} \rangle \Leftrightarrow \begin{cases} [\sum_{\substack{1 \leq i \leq 3 \\ 2 \leq j \leq 5 \\ j \neq 4}} L_{ij} < \sum_{\substack{1 \leq i \leq 4 \\ 2 \leq j \leq 5}} L_{ij}] \\ \wedge \\ [\sum_{\substack{1 \leq i \leq 3 \\ 2 \leq j \leq 5 \\ j \neq 4}} L_{ij} < \sum_{\substack{1 \leq i \leq 5 \\ 2 \leq j \leq 6}} L_{ij}] \end{cases} \quad (5.7)$$



**Figura 5.7:** Selección de ID basada en el criterio de pérdidas de enlace.

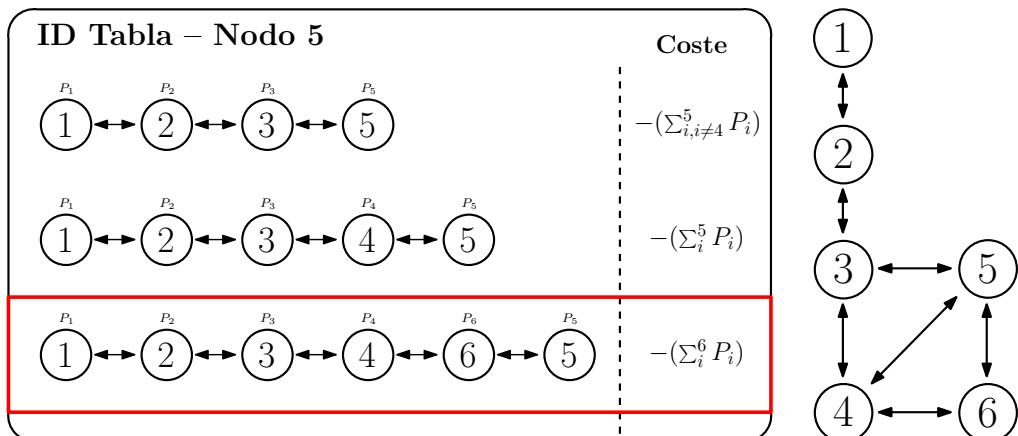
### 5.2.3.4. Criterio 4 - Balance de potencia

El cuarto criterio evalúa las rutas con más recursos disponibles en su recorrido, y las selecciona preferentemente frente a otras. Dado que cada nodo puede tener una oferta de recursos (valor positivo) o una demanda (valor negativo), la suma de todos los valores a lo largo de la ruta representará un coste total de la misma, tal y como se define en la Ecuación 5.8. Este coste se calcula cambiando el signo de la suma de todas las potencias en el trayecto, ya que nuestro objetivo es minimizarlo. Por lo tanto, si existen muchas ofertas de recursos, la suma será un valor positivo elevado y, al invertir su signo, se traducirá en un valor muy negativo (es decir, un bajo coste). Es importante destacar que el nombre de este criterio proviene del uso de las redes eléctricas inteligentes, en el que los recursos son la potencia eléctrica. Sin embargo, aunque en la fórmula se considere la oferta y la demanda de potencia, otros parámetros, como la capacidad de cómputo en entornos de Collaborative Edge Computing (CEC), podrían aplicarse de manera análoga en este criterio. De hecho, en el campo del enrutamiento clásico, Ma *et al.* [162] lo denominan *shortest-widest path*.

$$Cost_{ID} = -\left(\sum_i^{N_{ID}} P_i\right) \quad (5.8)$$

Una vez más, la selección de  $ID_{activa}$  se basará en el valor mínimo de  $Cost_{ID}$ . En el caso de nuestro ejemplo, como se observa en la Figura 5.8, el ID seleccionado es 1.2.3.4.6.5, el cual debería cumplir con la condición de la Ecuación 5.9.

$$\langle ID_3 = ID_{activa} \rangle \Leftrightarrow \begin{cases} [\sum_i^6 P_i > \sum_i^5 P_i] \\ \wedge \\ [\sum_i^6 P_i > \sum_{i,i \neq 4}^5 P_i] \end{cases} \quad (5.9)$$



**Figura 5.8:** Selección de ID basada en el criterio de balance de potencia.

### 5.2.3.5. Criterio 5 - Balance de potencia con pérdidas

El quinto criterio combina los criterios 3 y 4 [163, 162], ya que busca maximizar los recursos disponibles en la ruta, al mismo tiempo que limita las pérdidas potenciales que una ruta específica (especialmente si es más larga que otras), pueda conllevar. Una vez más, en el caso de las redes eléctricas inteligentes, esto se traduce en la oferta y la demanda de potencia, restando las pérdidas en la ruta (que siguen la Ecuación 5.6), tal y como se representa en la Ecuación 5.10.

$$Cost_{ID} = -\left(\sum_i^{N_{ID}} P_i - \sum_{ij}^{N_{ID}-1} L_{ij}\right) \quad (5.10)$$

Cuando este criterio se aplica al ejemplo de la Figura 5.9, la ruta seleccionada es 1.2.3.4.5, que en realidad representa una decisión intermedia entre el criterio 3 (pérdidas en los enlaces) y el criterio 4 (balance de potencia). En consecuencia, debe cumplirse la condición de la Ecuación 5.11 para que el  $ID_{activa}$  corresponda al mostrado en la Figura 5.9.

$$\langle ID_2 = ID_{activa} \rangle \Leftrightarrow \begin{cases} [(\sum_i^6 P_i - \sum L_{ij}) > (\sum_i^5 P_i - \sum L_{ij})] \\ \wedge \\ [(\sum_i^6 P_i - \sum L_{ij}) > (\sum_{i,i \neq 4}^5 P_i - \sum L_{ij})] \end{cases} \quad (5.11)$$

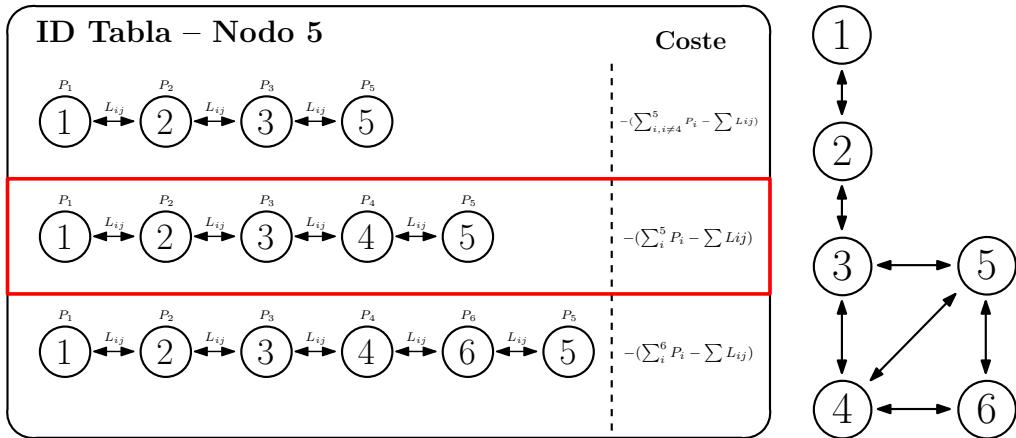


Figura 5.9: Selección de ID basada en el criterio de balance de potencia con pérdidas.

### 5.2.3.6. Criterio 6 - Balance de potencia ponderado

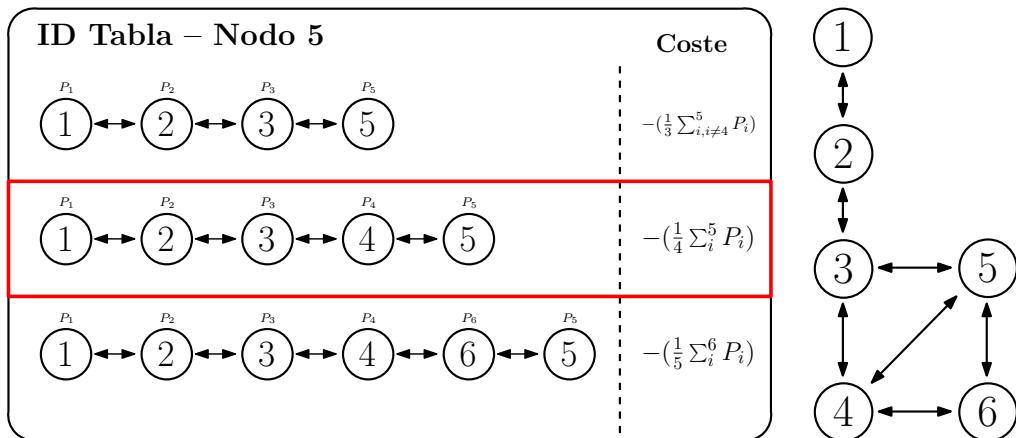
El sexto y último criterio representa una implementación alternativa, mediante ponderaciones, de las ideas presentadas en los dos criterios anteriores [163, 162]. Mientras que

el criterio 5 penaliza la ruta en función de la longitud y otros parámetros de pérdidas, el criterio 6 normaliza la capacidad de potencia de la ruta en función del número de nodos atravesados. Este criterio pone de manifiesto directamente el efecto negativo que puede tener el criterio 4, que busca el mejor balance de potencia, ya que eventualmente podría favorecer rutas más largas hacia la raíz. Por lo tanto, el criterio 5 reajusta este cálculo considerando también las pérdidas potenciales, y el criterio 6 proporciona un balance de potencia ponderado, de modo que se tenga en cuenta tanto la capacidad como el número de saltos en la ruta. Este criterio puede resultar particularmente útil cuando no se dispone de una representación clara de las pérdidas de los enlaces. La fórmula resultante se representa en la Ecuación 5.12.

$$Cost_{ID} = -\left(\frac{1}{M} \sum_i^{N_{ID}} P_i\right), \quad M = \frac{1}{(length(ID) - 1)} \quad (5.12)$$

Considerando el ejemplo de la Figura 5.10, el  $ID_{activa}$  corresponde a la ruta 1.2.3.4.5, que tiene un menor número de nodos que la seleccionada en el criterio 4, como era de esperar, pero presenta la mejor relación  $\frac{recursos}{saltos}$  entre todas las rutas. Esta ruta cumple la condición representada en la Ecuación 5.13.

$$\langle ID_2 = ID_{activa} \rangle \Leftrightarrow \begin{cases} [\frac{1}{4} \sum_i^5 P_i > \frac{1}{3} \sum_{i \neq 4}^5 P_i] \\ \wedge \\ [\frac{1}{4} \sum_i^5 P_i > \frac{1}{5} \sum_i^6 P_i] \end{cases} \quad (5.13)$$



**Figura 5.10:** Selección de ID basada en el criterio de balance de potencia ponderado.

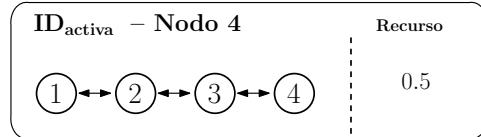
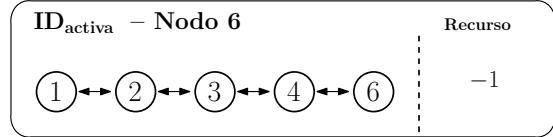
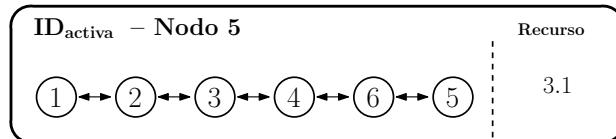
### 5.2.4. Fase 3 - Balance global

Una vez que las etiquetas se asignan a lo largo de toda la topología, se aplica un criterio determinado y cada nodo tiene un  $ID_{activa}$ , puede comenzar el procedimiento para equilibrar los recursos disponibles. El objetivo del algoritmo es alcanzar un estado de balance global trasladando recursos desde los nodos más alejados, hacia la raíz. Este movimiento se implementa de manera local, pero siguiendo un orden específico basado en el  $ID_{activa}$  de cada nodo, lo que finalmente produce un equilibrio general de los recursos para todos los nodos de la topología.

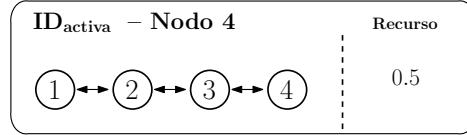
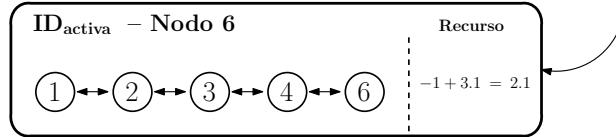
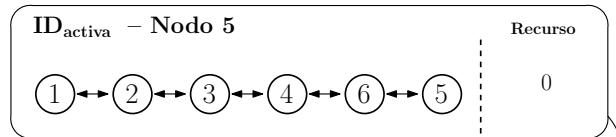
Para proceder, dado que el algoritmo es centralizado, es posible recopilar el  $ID_{activa}$  de cada nodo y ordenarlos en función de su longitud. Cuanto más largo sea el  $ID_{activa}$ , más alejado estará el nodo, desde un punto de vista lógico, de la raíz<sup>1</sup>. Por lo tanto, el algoritmo comienza equilibrando la carga desde el  $ID_{activa}$  más largo, que está asociado a un nodo específico de la red. En caso de que dos o más  $ID_{activa}$  tengan la misma longitud, se seleccionará uno de ellos de manera aleatoria. Una vez que los nodos se ordenan en función de su  $ID_{activa}$ , el algoritmo visita cada nodo una sola vez para equilibrar el recurso con su nodo predecesor en la lista ordenada representada por el  $ID_{activa}$ . Considerando la terminología explicada en la sección 5.2.1, las demandas se expresarán como un valor negativo y las ofertas como un valor positivo, siendo el objetivo dejar a cada nodo en un estado neutro (en este caso, sin demanda ni oferta implica un valor cero) y transferir la demanda u oferta restante al siguiente vecino de la lista. Este proceso se repite para cada nodo hasta llegar al último nodo de la lista (que corresponde al nodo con el  $ID_{activa}$  más corto, es decir, el nodo raíz). En consecuencia, la raíz obtiene la oferta o demanda restante de toda la red, es decir, el balance global. Posteriormente, el nodo raíz, como nodo pasarela hacia el núcleo de la red, puede decidir por sí mismo qué hacer con dicha oferta o demanda sobrante.

Para ejemplificar el procedimiento descrito, se presenta la Figura 5.11a, la cual, representa la topología de la red de ejemplo utilizada para explicar el algoritmo hasta el momento, en la cual el  $ID_{activa}$  más largo es el asociado al nodo 5 (1.2.3.4.6.5). En consecuencia, la Figura 5.11a muestra la lista ordenada, donde el primer nodo a comprobar es el nodo 5, mientras que el último es el nodo 1. Para simplificar, en la figura solo se representan los  $ID_{activa}$  de los tres primeros nodos que serán visitados (5, 6 y 4). En este escenario, el nodo 5 tiene una oferta de 3.1, el nodo 6 tiene una demanda de  $-1$ , y el nodo 4 una oferta de 0.5. Como el algoritmo comienza en el nodo 5, su estado se fijará en cero, y su oferta se transferirá al siguiente vecino en la lista, el nodo 6, que ahora actualizará sus recursos a  $-1 + 3.1 = 2.1$ , tal como se muestra en la Figura 5.11b. Posteriormente, el siguiente nodo a procesar será el nodo 6 (1.2.3.4.6), que también se fijará en cero y su oferta o demanda se transferirá al siguiente nodo, de modo que el nodo 4 pase a tener  $0.5 + 2.1 = 2.6$ , como se ilustra en la Figura 5.11c. El algoritmo DEN2NE continúa procesando todos los nodos hasta llegar finalmente al nodo raíz, de manera que todos los nodos quedan equilibrados y el nodo raíz obtiene un valor global de oferta/demanda de la red.

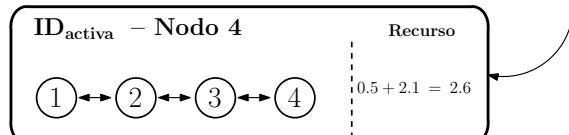
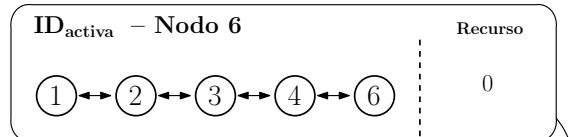
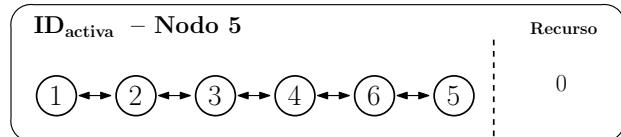
<sup>1</sup>Es importante destacar el adverbio *lógicamente* en este contexto, ya que el nodo podría estar físicamente más cerca, pero el  $ID_{activa}$  seleccionado puede proporcionar una vista lógica diferente.



(a) Paso 1.



(b) Paso 2.



(c) Paso 3.

**Figura 5.11:** Ejemplo del procedimiento de balance global.

El procedimiento anterior se resume en el Algoritmo 2. Como se puede observar, las entradas del algoritmo son: (1) el grafo  $G(\mathcal{N}, \mathcal{L})$ , que es el conjunto de nodos de la topología ( $\mathcal{N}$ ) y enlaces ( $\mathcal{L}$ ), y (2) el *scenario\_type*, que puede ser uno de los siguientes cuatro (definidos únicamente como referencia para representar y, posteriormente, evaluar, distintos casos de uso):

- **Escenario ideal:** Representa un escenario en el que no existe pérdida de recursos durante la transmisión cuando el recurso ( $r_i$ ) se envía de un nodo  $i$  a otro  $j$  ( $i, j \in \mathcal{N} \forall i \neq j$ ) a través de un enlace sin pérdidas, es decir,  $L_{ij} = 0$ . En consecuencia, para cualquier transmisión de recursos  $i \rightarrow j$ , en la que el estado inicial es  $\{i = r_i, j = r_j\}$ , los recursos resultantes tras la transmisión son  $\{i' = 0, j' = r_i + r_j\}$ .
- **Escenario con pérdidas:** Implica la existencia de pérdidas durante la transmisión de recursos, es decir,  $L \neq \emptyset$ . En consecuencia, para cualquier transmisión de recursos  $i \rightarrow j$ , en la que el estado inicial es  $\{i = r_i, j = r_j\}$ , los recursos resultantes tras la transmisión son  $\{i' = 0, j' = r_i + r_j - L_{ij}\}$ . El modelo de pérdidas que se empleará está descrito en [163], utilizando las pérdidas de transmisión definidas en la Ecuación 5.6.
- **Escenario con capacidad de enlace restringida:** Hasta ahora, los enlaces no tenían ninguna limitación particular al transmitir un recurso, pero si el enlace está restringido a ciertos valores de transmisión, el recurso no se reenviará oportunamente y el excedente será descartado (al menos, de forma lógica para el algoritmo, incluso si el exceso de recurso permanece en su origen). Más específicamente, si la capacidad del enlace es  $C_{ij}$ , con  $C \neq \emptyset$ , y  $r_i \geq C_{ij}$ , entonces la transmisión de recursos asociada  $i \rightarrow j$  con estado inicial  $\{i = r_i, j = r_j\}$  resultará en el estado final  $\{i' = 0, j' = C_{ij} + r_j\}$ . El modelo de las capacidades de los enlaces empleados está descrito en [163], donde se especifica que la capacidad máxima de cada enlace es  $C_{max} = I_{max} \times V_d$ .
- **Escenario con pérdidas y capacidad de enlace restringida:** Este último tipo combina los dos anteriores, es decir,  $L \neq \emptyset$  y  $C \neq \emptyset$ . En consecuencia, para cualquier transmisión de recursos  $i \rightarrow j$ , en la que el estado inicial es  $\{i = r_i, j = r_j\}$ , los recursos resultantes tras la transmisión son  $\{i' = 0, j' = r_i + r_j - L_{ij}\}$  en caso de que  $r_i \leq C_{ij}$ , y  $\{i' = 0, j' = C_{ij} + r_j - L_{ij}\}$  en caso de que  $r_i \geq C_{ij}$ .

Con respecto a las salidas del Algoritmo 2, *total\_balance* proporciona el balance global de la topología (que corresponde a la carga final en la raíz, ya que esta permanece como el último nodo y conserva la diferencia de todas las transferencias de recursos; de ahí que  $total\_balance = root.load$ ), mientras que *abs\_flux* proporciona un valor absoluto de los intercambios de recursos a lo largo de la red (para cualquier intercambio se considera su valor absoluto, ya que el signo negativo o positivo solo indica la dirección, pero el movimiento de recursos ocurre en cualquier caso).

Por lo tanto, *total\_balance* expresa si la red presenta una demanda o un exceso global de recursos, mientras que *abs\_flux* representa el cantidad de recursos distribuidos.

**Algoritmo 2:** Proceso de balance global

---

```
input :  $G(\mathcal{N}, \mathcal{L})$ , scenario_type
output: total_balance, abs_flux

1 init global_ids, abs_flux;
2 sort global_ids;
3 while length(global_ids) > 1 do
4     origin =  $\mathcal{N}(\text{global\_ids}[0])$ ;
5     destination = get next hop from origin;
6     if origin.load > 0 then
7         setLinkDirection  $\mathcal{L}(\text{origin}, \text{destination})$  to down;
8     else
9         setLinkDirection  $\mathcal{L}(\text{origin}, \text{destination})$  to up;
10    switch scenario_type do
11        case ideal do
12            destination.load = destination.load + origin.load;
13            abs_flux = abs_flux + |origin.load|;
14        case lossy do
15            destination.load = destination.load + origin.load - losses;
16            abs_flux = abs_flux + |origin.load - losses|;
17        case capacity do
18            if origin.load  $\geq$  capacity then
19                destination.load = destination.load + capacity;
20                abs_flux = abs_flux + |capacity|;
21            else
22                destination.load = destination.load + origin.load;
23                abs_flux = abs_flux + |origin.load|;
24        otherwise do
25            if origin.load  $\geq$  capacity then
26                destination.load = destination.load + capacity - losses;
27                abs_flux = abs_flux + |capacity - losses|;
28            else
29                destination.load = destination.load + origin.load - losses;
30                abs_flux = abs_flux + |origin.load - losses|;
31        origin.load =  $\emptyset$ ;
32        pop global_ids[0];
33 return root.load, abs_flux
```

---

Concretamente, *total\_balance* podría ser igual o cercano a cero, lo cual significa que la red no tiene una demanda específica de recursos, pero *abs\_flux* podría indicar un

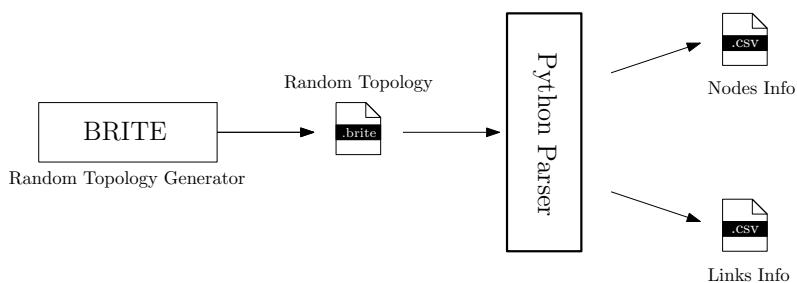
valor elevado, lo que evidencia que, para alcanzar un balance global, se requiere un alto número de intercambios. De hecho, *total\_balance* simplemente ilustra la naturaleza del escenario (es decir, cantidad de ofertas frente a demandas), mientras que *abs\_flux* constituye realmente un indicador de la eficacia del algoritmo, ya que cuanto menor sea su valor, menor será el coste para balancear óptimamente la carga de recursos en la topología.

Como se puede observar, el Algoritmo 2 es lineal (complejidad  $\mathcal{O}(n)$ ), ya que los nodos de la red son visitados una sola vez considerando la etiqueta o identificador seleccionado en la segunda fase del algoritmo, lo cual queda representado por el único bucle (*while*) del mismo. Finalmente, cabe destacar que el algoritmo DEN2NE se ejecuta en base a una instantánea específica de la red. Si la red se actualiza debido a la movilidad de nodos, o a la modificación de ofertas y demandas, el algoritmo puede ejecutarse de nuevo, en cualquier momento, para redistribuir los nuevos recursos.

### 5.3. Entorno de evaluación

La implementación y configuración de la evaluación están basadas en Python 3.8<sup>2</sup>. Más específicamente, la evaluación se divide en dos etapas:

1. Un generador de topologías aleatorias para probar exhaustivamente DEN2NE, que aprovecha el entorno BRITE [138], como se muestra en la Figura 5.12 (el cual traduce los archivos `*.brite` en archivos `*.csv` para los nodos y enlaces de la topología, utilizados como entrada en nuestro algoritmo).
2. La ejecución del algoritmo, desarrollado como un script centralizado, que importa todas las topologías generadas, una por una, utilizando semillas y diferentes parámetros de DEN2NE para su comparación.

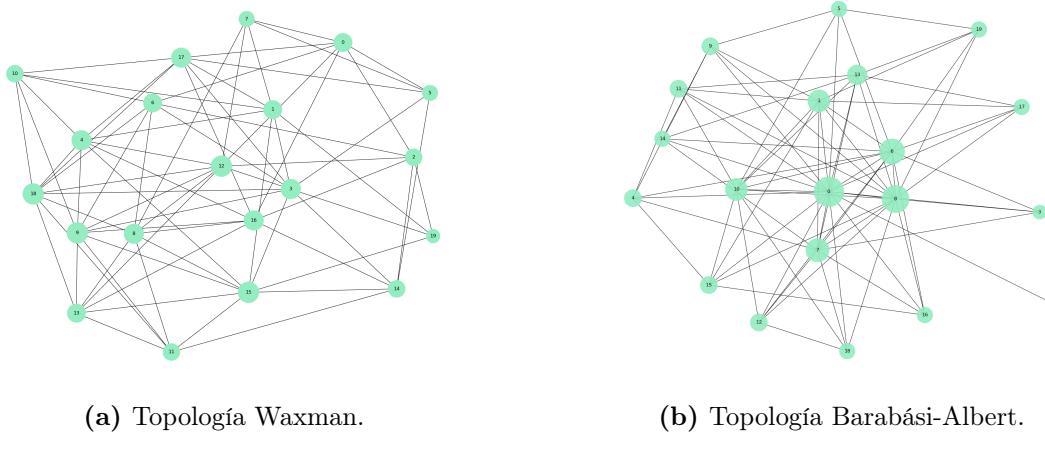


**Figura 5.12:** Generador de topologías para DEN2NE basado en BRITE.

Nuestro objetivo es proporcionar un análisis exhaustivo del algoritmo DEN2NE, abarcando la mayor cantidad posible de tipos de escenarios. De hecho, los grafos proporcionados por BRITE se basan en topologías aleatorias Waxman [164] y Barabási-Albert [165], las cuales se emplean para modelar una amplia gama de sistemas de interconexión, desde

<sup>2</sup><https://peps.python.org/pep-0569/>

redes de computación, a redes de comunicaciones, y hasta redes sociales. La red aleatoria Waxman se fundamenta en un modelo probabilístico para la generación aleatoria de una red, mientras que la red Barabási-Albert se enmarca dentro de la categoría de redes *scale-free* (también conocidas como redes *hub-and-spoke*), en las cuales ciertos nodos presentan un número de conexiones significativamente mayor que el resto. Una vista simplificada de ambos modelos se muestra en la Figura 5.13.



**Figura 5.13:** Topologías aleatorias empleadas en el estudio de DEN2NE.

En lo que respecta a los parámetros de generación de topologías, estos se resumen en la Tabla 5.1, la cual incluye: tipo de topología (uno de los dos tipos mencionados anteriormente), número de nodos (de 10 a 200 nodos en intervalos de 10 nodos), grado de conectividad (número promedio de enlaces por nodo, de 4 a 6, en intervalos de 2 enlaces) y una semilla de topología aleatoria (para obtener 10 topologías aleatorias de cada tipo). De acuerdo con estos parámetros, el número total de topologías evaluadas fue de 1200, tal y como se indica en la Ecuación 5.14.

Atributos	Valores
Tipo de topología	["waxman", "barabasi-albert"]
Número de nodos	[10:10:200]
Grado de conectividad	[4:2:6]
Semillas de las topologías aleatorias	[1:1:10]

**Tabla 5.1:** Parámetros de generación de las topologías aleatorias.

$$\begin{aligned} \langle N_{topos} \rangle &= N_{tipo} \times N_{nodos} \times N_{grado} \times N_{semilla} \\ &= 2 \times 20 \times 3 \times 10 = 1200 \text{ topos} \end{aligned} \tag{5.14}$$

La evaluación realizada tiene como objetivo comparar el comportamiento de los 6

criterios descritos en la Sección 5.2.3 en cuatro tipos de escenarios de red (ideal, con pérdidas, con capacidad de enlace restringida y con capacidad de enlace restringida y pérdidas). Adicionalmente, el banco de pruebas debía asignar una oferta o demanda inicial de recursos a cada nodo de la topología. Esta asignación de recursos fue agnóstica (no se asocia ninguna unidad o métrica en particular) y podía estar limitada a un valor total determinado (fijado en 250, asumiendo al menos 1 unidad de recurso por nodo en promedio en las topologías más grandes) o no estarlo (sin un valor máximo establecido). La limitación en la asignación de recursos se definió con el fin de depurar la implementación y el comportamiento del algoritmo en los escenarios más complejos. En cualquier caso, tanto en la versión limitada como en la no limitada, la carga asignada a cada nodo siguió una distribución uniforme, como se ilustra en la Figura 5.14.



**Figura 5.14:** Función de densidad de probabilidad para la generación de cargas aleatorias en cada topología.

Finalmente, para validar nuestros resultados, cada escenario se repitió 10 veces utilizando diferentes semillas para la generación aleatoria de recursos. Estas semillas solo modificaban la ubicación de la raíz y la asignación de recursos, mientras que el resto del escenario se mantenía igual, con el fin de demostrar exhaustivamente la validez de los resultados obtenidos. Todos estos parámetros se resumen en la Tabla 5.2. Por lo tanto, el número total de simulaciones de nuestro estudio es de 576000, tal y como se indica en la Ecuación 5.15.

$$\begin{aligned}
 \langle N_{\text{simulaciones}} \rangle &= N_{\text{topos}} \times N_{\text{criterios}} \times N_{\text{escenarios}} \times N_{LR} \times N_{\text{semilla}} \\
 &= 1200 \times 6 \times 4 \times 2 \times 10 \\
 &= 576000 \text{ simulaciones}
 \end{aligned} \tag{5.15}$$

Atributo	Valores
Criterios	[1:1:6]
Tipo de escenario	[1:1:4]
Limitación de recursos	[0, 1]
Semilla de ejecución aleatoria	[1:1:10]

**Tabla 5.2:** Parámetros para la ejecución de pruebas.

## 5.4. Resultados y evaluación del algoritmo

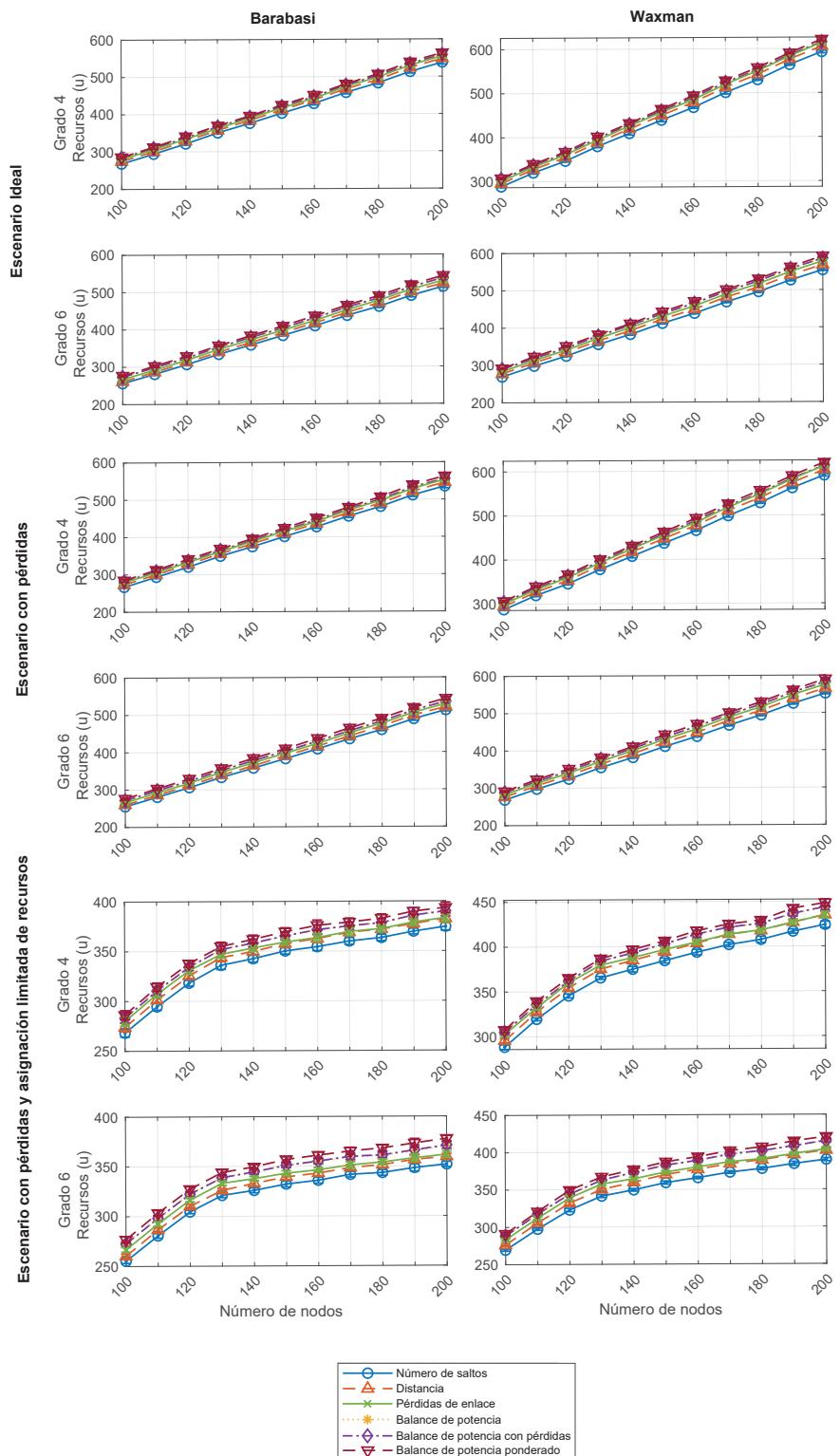
Considerando la implementación y el entorno de evaluación presentados en la Sección 5.3, en esta sección se exponen los resultados obtenidos a partir de las simulaciones descritas. Para evaluar el comportamiento de DEN2NE, se midieron tres métricas diferentes: la primera en relación con el balance de recursos (que corresponden a las salidas del algoritmo DEN2NE, tal y como se describió previamente en el Algoritmo 2), y las otras dos en relación con el tiempo de ejecución, tanto para la asignación de etiquetas como para el balance de recursos, ya que nuestro objetivo era evaluar qué tan eficiente es DEN2NE equilibrando la carga y qué tan rápido realiza dicha tarea. Por este motivo, las siguientes tres secciones examinan estos parámetros en detalle, considerando el tamaño, tipo y conectividad de la red, el tipo de escenario y los seis criterios definidos en la Sección 5.2.3. Todas las métricas se han obtenido mediante múltiples repeticiones, tal y como se indicó en la Sección 5.3, e incluyen su respectiva desviación estándar, aunque esta resulta despreciable en la mayoría de los gráficos. Aunque se han llevado a cabo simulaciones en todos los casos posibles, con el fin de facilitar la interpretación de los resultados, la representación gráfica se ha restringido únicamente a los más representativos.

Es importante destacar que la validez de los resultados presentados en esta sección está limitada a la implementación realizada en la Sección 5.3. En la práctica, si el escenario incluyera un controlador centralizado de software, podríamos obtener resultados muy similares. No obstante, por ejemplo, deberían añadirse los tiempos de comunicación entre el controlador y los demás nodos de la red, los cuales dependerían de condiciones completamente ajenas al algoritmo en sí.

### 5.4.1. Flujo de balance de recursos

El flujo de balance de recursos representa la cantidad de recursos transferidos entre todos los nodos de la topología. Esta métrica se ha calculado sumando todos los intercambios de recursos entre pares de nodos en valor absoluto, y fue presentada previamente en el Algoritmo 2 como la variable *abs\_flux*. Esta métrica indica el coste de equilibrar los recursos: cuanto mayor sea su valor, más costoso resulta dicho balance.

## 5.4 RESULTADOS Y EVALUACIÓN DEL ALGORITMO



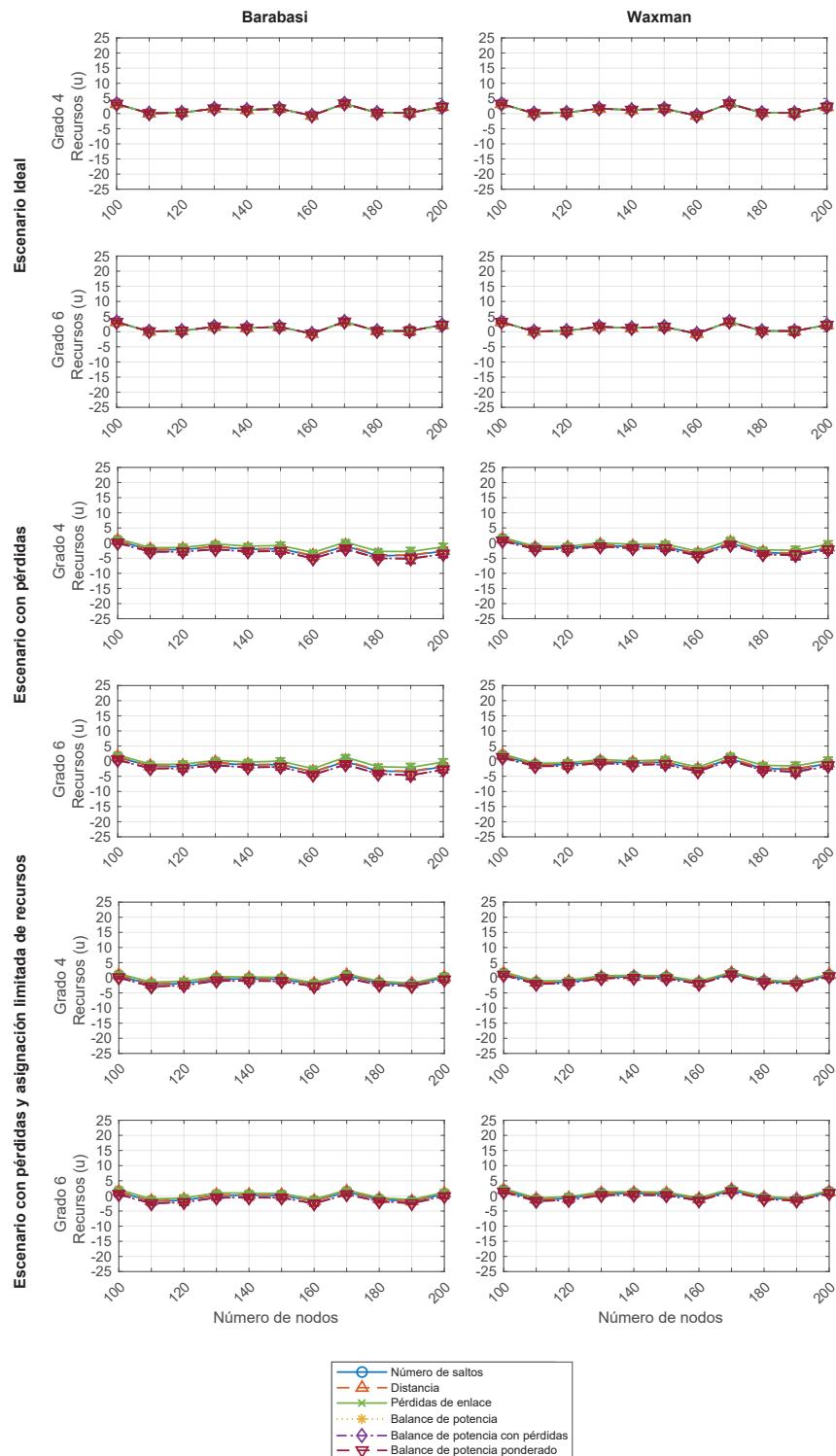
**Figura 5.15:** Flujo de balance de recursos (*abs\_flux*) - Tres escenarios y dos grados de red.

La Figura 5.15 ilustra el flujo de balance de recursos para redes con grados de conectividad de 4 a 6, basadas en topologías de Barabási y Waxman, con tamaños comprendidos entre 100 y 200 nodos, y evaluadas con los seis criterios definidos en tres tipos de escenarios: ideal, con pérdidas y con pérdidas bajo limitación de asignación de recursos, respectivamente. Como puede observarse, el flujo crece de manera lineal con el tamaño de la topología, lo que indica que el algoritmo presenta un comportamiento escalable ( $\mathcal{O}(n)$ ), sin estar fuertemente condicionado por el número de nodos involucrados. Únicamente en el último escenario, representado por los cuatro gráficos situados en la parte inferior de la Figura 5.15, se aprecia un crecimiento de tipo logarítmico, aunque este efecto se debe a la limitación impuesta en la asignación de recursos.

Adicionalmente, considerando los seis criterios, parece que el Criterio 1 (número de saltos) ofrece mejores resultados (menor flujo) que el resto, especialmente en topologías Waxman y cuando la conectividad de los nodos aumenta; mientras que el Criterio 6 (balance de potencia ponderado) se presenta como la peor opción, independientemente del escenario (incluso en los casos con pérdidas o con limitaciones de recursos). La discrepancia entre estos criterios puede explicarse por el hecho de que aquellos orientados a minimizar la distancia o el número de saltos hacia el nodo raíz favorecen trayectorias más cortas, mientras que los que consideran la cantidad de recursos a lo largo del camino hacia la raíz tienden a seleccionar trayectorias más largas. Por esta razón, en el Criterio 1 y el Criterio 2 (número de saltos y distancia, respectivamente), el flujo medio de recursos es menor en comparación con el Criterio 3 y el Criterio 6 (balance de potencia y balance de potencia ponderado, respectivamente), donde las trayectorias tienden a ser más largas en promedio, resultando en un mayor flujo de recursos.

De la misma forma, la Figura 5.16 ilustra el balance total de recursos para redes con grados de conectividad de 4 a 6, basadas en topologías de Barabási y Waxman, con tamaños comprendidos entre 100 y 200 nodos, y evaluadas con los seis criterios definidos en tres tipos de escenarios: ideal, con pérdidas y con pérdidas bajo limitación de asignación de recursos, respectivamente. Como puede observarse, el balance total de recursos tiende a 0 en todos los casos, lo cual resulta coherente si se considera que la asignación de recursos sigue la función previamente mostrada en la Figura 5.14, cuyo valor medio es 0. Además, en el escenario ideal todos los criterios producen el mismo resultado, y únicamente en los escenarios con pérdidas se observa cierta variabilidad, debido a que los caminos hacia la raíz (y, por tanto, las pérdidas) difieren. Esto indica que el algoritmo es capaz de compensar todos los recursos en la red, es decir, que todos los nodos tiendan hacia el valor cero (un uso óptimo), poniendo de manifiesto una distribución eficiente, dado que en pocos casos el nodo que actúa como pasarela (root) tendrá que demandar o verter excedentes en niveles superiores de la red.

## 5.4 RESULTADOS Y EVALUACIÓN DEL ALGORITMO



**Figura 5.16:** Balance total de recursos (*total\_balance*) - Tres escenarios y dos grados de red.

### 5.4.2. Tiempo de convergencia de la asignación de etiquetas

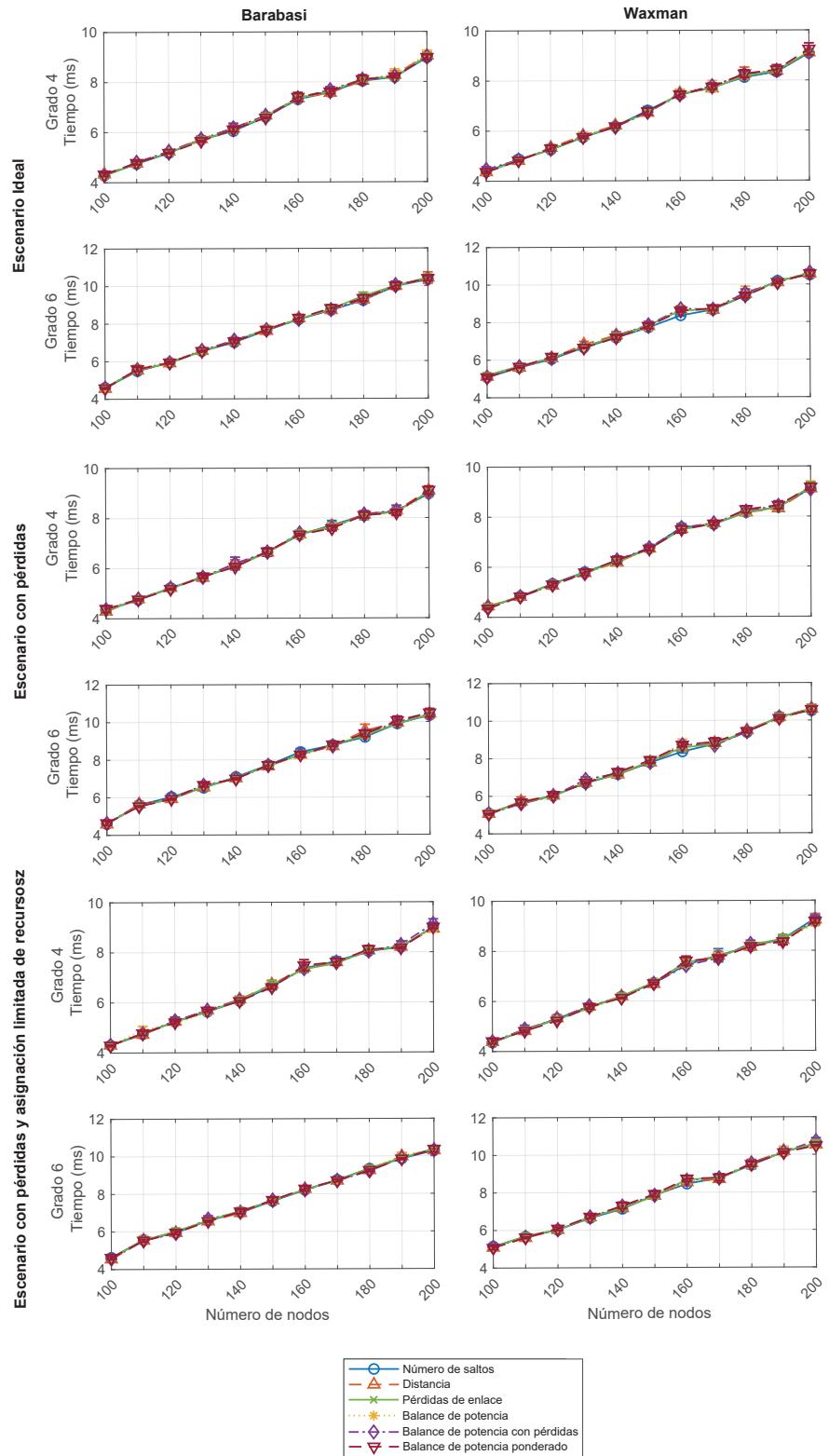
El tiempo de convergencia en la asignación de etiquetas representa el tiempo requerido por el algoritmo para asignar todas las etiquetas jerárquicas desde la raíz hasta el resto de los nodos de la topología. La Figura 5.17 muestra el tiempo de convergencia de la asignación de etiquetas para redes con grados de conectividad de 4 a 6, basadas en topologías de Barabási y Waxman, con tamaños comprendidos entre 100 y 200 nodos, y evaluadas con los seis criterios definidos en tres tipos de escenarios: ideal, con pérdidas y con pérdidas bajo limitación en la asignación de recursos, respectivamente.

Como puede observarse, este tiempo de convergencia crece linealmente con el tamaño de la topología, sin verse afectado exponencialmente por el grado de conectividad u otros parámetros. Este hecho refleja una buena escalabilidad potencial del algoritmo, ya que, incluso en los peores casos, el tiempo de convergencia no supera los 15 ms. Además, los seis criterios presentan comportamientos muy similares, sin diferencias significativas. Esto se debe a que el tiempo de convergencia en la asignación de etiquetas depende estrictamente de la primera fase del algoritmo, la cual es independiente del criterio aplicado posteriormente. En caso de existir pequeñas variaciones, estas podrían atribuirse a la naturaleza aleatoria de las pruebas realizadas.

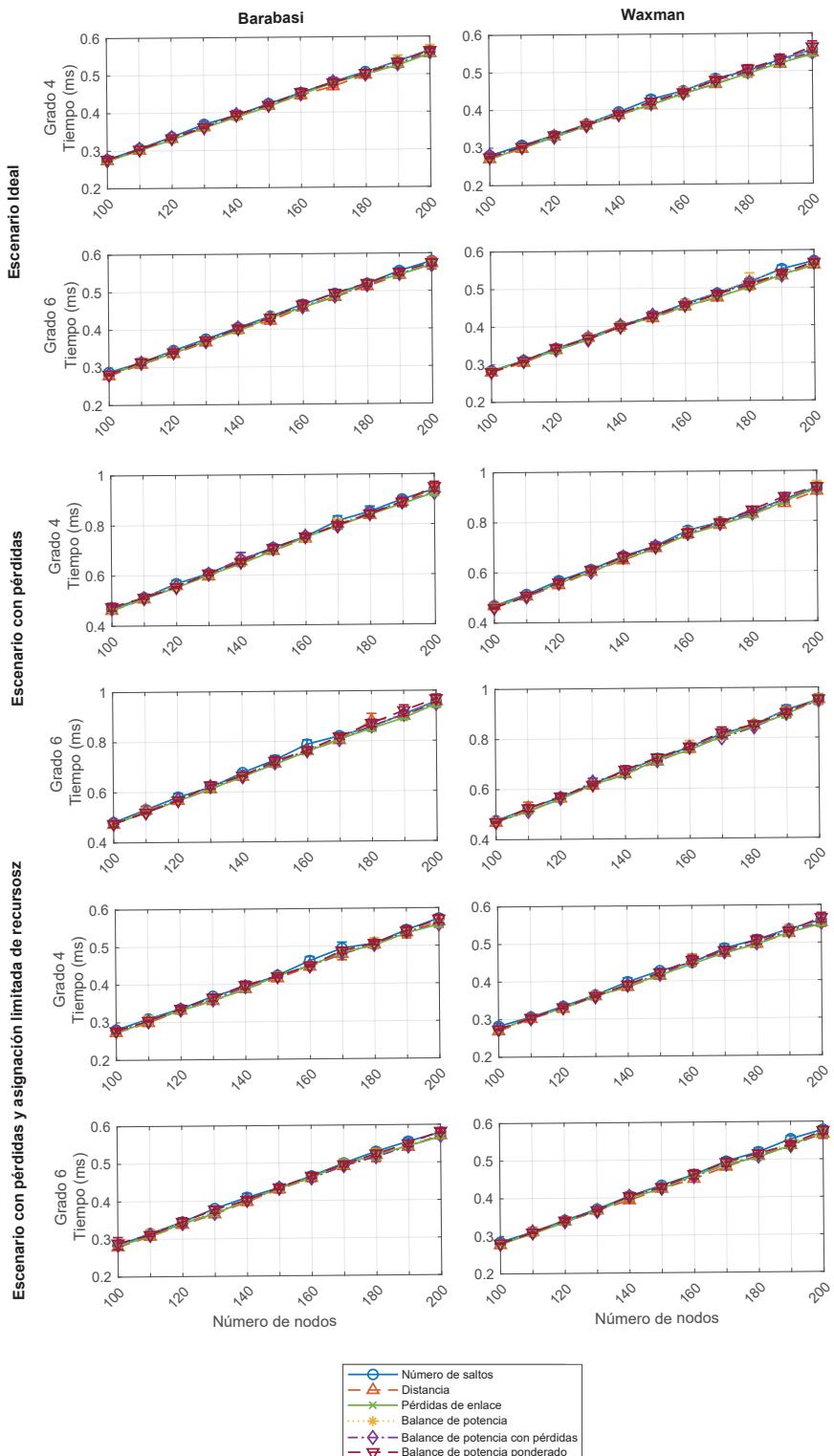
### 5.4.3. Tiempo de convergencia del balanceo de recursos

El tiempo de convergencia en el balanceo de recursos representa el tiempo requerido por el algoritmo para realizar todos los intercambios de recursos hasta que la topología quede balanceada con respecto a la raíz. De igual forma, la Figura 5.18 muestra el tiempo de convergencia del balanceo de recursos para redes con grados de conectividad de 4 a 6, basadas en topologías de Barabási y Waxman, con tamaños comprendidos entre 100 y 200 nodos, y evaluadas con los seis criterios definidos en tres tipos de escenarios: ideal, con pérdidas y con pérdidas bajo limitación en la asignación de recursos, respectivamente. Como puede observarse, y de manera similar al tiempo de convergencia de la asignación de etiquetas, el tiempo de convergencia del balanceo de recursos crece linealmente con el tamaño de la topología, sin verse afectado de forma exponencial por el grado de conectividad u otros parámetros. Este hecho refleja una buena escalabilidad potencial del algoritmo, ya que, incluso en los peores casos, el tiempo de convergencia no supera 1 ms. En consecuencia, el tiempo total de ejecución del algoritmo DEN2NE fue inferior a 20 ms en todos los escenarios evaluados.

Además, al igual que en la métrica de tiempo anterior, los seis criterios presentan comportamientos muy similares. En este caso, el tiempo de convergencia del balanceo de recursos sí depende de los criterios (a diferencia de la asignación de etiquetas), aunque la propia definición del algoritmo tiene un mayor impacto en este tiempo que el criterio seleccionado. Este aspecto resulta particularmente relevante, ya que los criterios se definen para favorecer una mejor asignación de recursos en distintos escenarios. Por lo tanto, estos resultados demuestran que nuestro algoritmo puede ajustarse y adaptarse a casos de uso específicos sin experimentar variaciones significativas en el tiempo de ejecución.



**Figura 5.17:** Tiempo de convergencia de la asignación de etiquetas - Tres escenarios y dos grados de red.



**Figura 5.18:** Tiempo de convergencia del balance de recursos - Tres escenarios y dos grados de red.

## 5.5. Conclusiones

En este trabajo se ha presentado DEN2NE, un algoritmo para la gestión de recursos en redes densas multi-hop que destaca por su elevada escalabilidad y bajos tiempos de convergencia.

El algoritmo descubre los diferentes nodos de la topología y asigna a cada uno de ellos una o varias etiquetas jerárquicas, que representan múltiples rutas hacia el nodo raíz o *gateway*. Posteriormente, redistribuye la carga de trabajo desde los nodos situados en la parte inferior de la jerarquía hasta los niveles superiores. Esta reorganización de recursos puede llevarse a cabo siguiendo distintos criterios: en este trabajo se han evaluado seis como ejemplos representativos, aunque el algoritmo no está limitado a ellos, pudiendo adaptarse fácilmente a otros criterios en función del escenario de aplicación. La evaluación se ha realizado sobre un amplio rango de topologías, con hasta 200 nodos, mostrando tiempos de convergencia reducidos (inferiores a 20 ms) y con un crecimiento lineal. Esta característica resulta especialmente beneficiosa en escenarios con alta movilidad y/o que requieren una mayor fiabilidad. Además, el procedimiento de asignación de etiquetas se ejecuta en solo dos pasos, y cada nodo únicamente necesita almacenar tantas etiquetas como caminos existan hacia el nodo raíz, independientemente del tamaño de la red.

Cabe destacar que con DEN2NE se sientan las bases para el desarrollo de mecanismos de control, esquemas de encaminamiento basados en etiquetado jerárquico y estrategias de gestión y planificación de recursos en entornos densos y heterogéneos (por ejemplo, SG, logística, *edge/fog computing*, entre otros). DEN2NE demuestra que es posible combinar asignación jerárquica de rutas, selección de rutas mediante criterios adaptables y una fase de balanceo eficiente con complejidad y requerimientos reducidos, propiedades que resultan críticas en nodos con capacidad limitada. Encajando estas características, directamente con los objetivos de la Tesis.



## Capítulo 6

# Propuesta de reconfiguración y predicción de errores en redes de distribución eléctrica

Partiendo de DEN2NE (Capítulo 5), que establece las bases para mecanismos de control, esquemas de encaminamiento mediante etiquetado jerárquico y estrategias de gestión/planificación de recursos en entornos densos y heterogéneos, este capítulo explora su aplicación a la optimización y reconfiguración proactiva de redes de distribución eléctrica (SG). La combinación de un mecanismo eficiente de descubrimiento/etiquetado con modelos de ML/DL permite abordar dos retos clave (Analizado en la Sección 2.2.3.2): (i) detectar y predecir anomalías o fallos antes de que afecten al servicio, y (ii) coordinar reconfiguraciones automáticas y seguras que minimicen pérdidas y aumenten la resiliencia del sistema.

En este capítulo se presenta una propuesta novedosa que integra técnicas de aprendizaje automático (ML), y aprendizaje profundo (DL) para la predicción temprana de fallos en red con algoritmos de reconfiguración basados en el etiquetado jerárquico propuesto con DEN2NE. El sistema propuesto opera en tres capas: (a) adquisición y agregación de telemetría distribuida (sensores, medidores y nodos etiquetados), (b) motores predictivos de ML/DL que estiman la localización de fallos a corto plazo en la SG, y (c) un módulo de decisión que traduce las predicciones en posibles acciones de reconfiguración.

Este trabajo nació a raíz de DEN2NE (Carrascal *et al.* [156]), y se materializó como uno de los primeros TFM que pude dirigir en la universidad (Bartolomé *et al.* [166]), dando lugar a una propuesta (Carrascal *et al.* [167]) que pone de manifiesto la versatilidad y capacidad de DEN2NE para operar en diferentes dominios y casos de uso finales.

### 6.1. Introducción

Las SG representan la solución futura para las redes de transmisión y distribución eléctrica [105, 106], cuyo objetivo principal es alcanzar una monitorización completa de

la red que permita un mejor balance energético entre productores y consumidores. Uno de los aspectos más relevantes de las SG es la reconfiguración de la distribución de energía dentro de la propia red. Esto resulta crucial porque permite reencaminar la potencia en caso de fallo en alguno de los enlaces de la red, o simplemente optimizar la distribución según un criterio determinado. Numerosas contribuciones existentes vistas en la Sección 2.2.3.2 inciden especialmente en este último caso, en el que la reconfiguración mediante optimización reduce pérdidas por propagación o sobrecarga de línea, o actúa sobre otros parámetros eléctricos. De forma general, la mayoría de los algoritmos de reconfiguración aplicados a SG que se centran en la resiliencia ante fallos buscan proporcionar rutas de respaldo o caminos alternativos para llevar a cabo la distribución. No obstante, las estrategias propuestas suelen ser siempre reactivas frente al fallo, ya sea de manera centralizada o distribuida.

Por ello, en este trabajo nos proponemos generar una estrategia de reconfiguración preventiva, y por tanto proactiva, frente a fallos de red, aprovechando conjuntos de datos reales de SG y técnicas de ML y DL. Como se ha indicado, el enfoque propuesto descansa sobre DEN2NE [156]. Partimos de dicha base para desarrollar un mecanismo de predicción de fallos que clasifica rutas alternativas obtenidas en la etapa de etiquetado jerárquico suministrado por DEN2NE, con el fin de identificar de forma anticipada aquellas con mayor probabilidad de provocar errores en la red. Existen múltiples criterios en las SG que pueden dar lugar a fallos en la red, en nuestro caso, analizaremos cuando una línea de la red excede la capacidad física de la misma. En consecuencia, tras el análisis del estado del arte (Sección 2.2.3.3), las principales aportaciones de nuestro trabajo son las siguientes:

1. Una solución de predicción de fallos específicamente orientada a la reconfiguración de SG, entrenada y validada con conjuntos de datos reales tal y como se detalla en la Sección 6.2.
2. Generalizable a topologías de cualquier tipo, forma o tamaño, evaluada con el generador de topologías aleatorias BRITE y parámetros realistas, empleando parámetros inspirados en el modelo IEEE 123 Node Test Feeder [163]. A partir de estas validaciones, hemos estudiado la optimización de varios modelos de ML y DL para DEN2NE mediante selección de características y evaluación del modelo predictivo más adecuado para cada escenario.

Este último punto es de especial interés, dado que, en la literatura revisada, la mayoría de propuestas que emplean AI/ML suelen tender a generar modelos con un único propósito y un único tipo de red, no siendo extrapolables a otros tipos/casuísticas de SG.

## 6.2. Búsqueda y análisis de fuentes de datos

Para evaluar de manera integral las diferentes técnicas de AI aplicadas a la predicción de fallos en escenarios de reconfiguración de SGs, resulta esencial disponer de un conjunto de datos de alta calidad. Tanto la calidad como la cantidad de datos son factores determinantes para definir un modelo consistente. En el contexto energético, nuestra experiencia

demuestra que existe una menor disponibilidad de conjuntos de datos. La protección de la privacidad de los usuarios en la medición y análisis de su comportamiento energético reduce el número de datasets públicos disponibles, ya que estos dependen principalmente de las compañías eléctricas [168].

No obstante, la búsqueda de la optimización en la distribución de la energía y la reducción del consumo de los usuarios, junto con otras motivaciones medioambientales, ha favorecido la creación de conjuntos de datos públicos en los últimos años. La Tabla 6.1 muestra varios datasets residenciales analizados, junto con detalles sobre su implementación, tales como la localización, el número de viviendas, el periodo de despliegue, la frecuencia de muestreo y los parámetros eléctricos medidos. Como parte de nuestro análisis, y con el objetivo de evaluar y seleccionar el dataset más adecuado entre los disponibles en la Tabla 6.1, se han considerado los siguientes requisitos:

- **Cantidad:** Es fundamental revisar la cantidad de datos disponibles en cada dataset, asegurando que incluyan mediciones de un número amplio de viviendas y que abarquen periodos prolongados de tiempo para analizar de manera efectiva los patrones de consumo energético.
- **Calidad:** La calidad de los datos depende de la resolución de las mediciones. Datasets como *REDD* y *BLUED* ofrecen frecuencias de muestreo elevadas, lo que permite una mejor desagregación del consumo energético y un análisis más representativo de los patrones energéticos.
- **Localización:** La localización de los datos es relevante, ya que influye en las diferencias de voltaje entre países. Por ejemplo, datasets de Estados Unidos, como *BLUED*, operan a menos de 120V, mientras que conjuntos europeos como *ECO* trabajan hasta 230V.
- **Parametrización:** Las muestras incluyen tensión (V), corriente (I) y potencia (tanto la activa (P), la reactiva (Q), y la aparente (S)), cada una asociada a una marca temporal y un identificador correspondiente a la vivienda. Algunos datasets, como *HUE* y *SustDataED*, también incorporan datos ambientales, lo cual resulta relevante para comprender el impacto de las condiciones climáticas con la generación de potencia renovable.
- **Generación Fotovoltaica:** En el contexto de las SGs, es esencial elegir conjuntos de datos que incluyan información sobre generación de energía renovable, como *Smart\** y *SustDataED*.

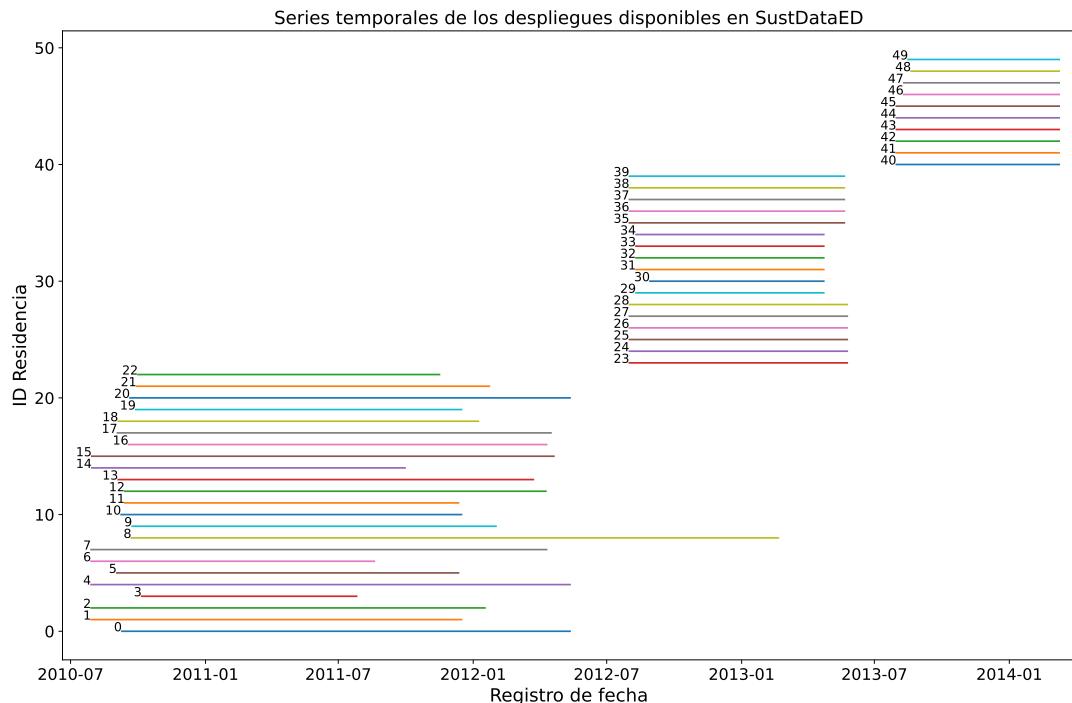
Entre los datasets analizados, se seleccionó especialmente *SustDataED*, debido al elevado número de usuarios con perfiles tanto de consumo, como de producción a lo largo de un periodo extenso de tiempo. Asimismo, resulta clave que las muestras presenten una buena resolución y que ofrezcan información en tiempo real sobre las condiciones climáticas de la ubicación en la que se encuentran las viviendas. En las siguientes secciones, se examina en detalle este conjunto de datos y se describen los principales criterios de diseño aplicados para generar el dataset final, elaborado a partir de *SustDataED* y complementado con la herramienta PVWatts, que finalmente se utilizó en nuestra evaluación.

Nombre	Localización	Residencias	Periodo (días)	Resolución	Parámetros
<i>AMPds2</i> [169]	Vancouver (Canadá)	1	730	60s	I, V, P, S, F, pf
<i>BLUED</i> [170]	Pittsburgh (EE. UU.)	1	8	83.33μs	I, V, eventos switch
<i>ECO</i> [171]	Suiza	6	244	1s	P
<i>GREEND</i> [172]	Italia y Austria	9	310	1s	P
<i>HUE</i> [173]	Columbia Británica (Canadá)	28	60	1s	P
<i>iAWE</i> [174]	Nueva Delhi (India)	1	73	1s	V, I, P, S
<i>REDD</i> [175]	Boston (EE. UU.)	6	11.9	66.66μs	I, V, P
<i>Smart*</i> [176]	Massachusetts (EE. UU.)	3	90	60s	P, S, V, I
<i>SustDataED</i> [177]	Madeira (Portugal)	50	1144	60s	I, V, P, Q, S
<i>UK-DALE</i> [178]	Reino Unido	5	499	62.5μs	P, estados de switch

Tabla 6.1: Comparación de datasets sobre el consumo público de energía eléctrica.

### 6.2.1. Análisis del dataset - SustDataED

El dataset *SustDataED* [177] fue creado como parte del proyecto de investigación SINAIS, cuyo objetivo es proporcionar retroalimentación ecológica a los usuarios para fomentar un comportamiento energético sostenible y un mayor uso de fuentes de energía renovable. Este dataset abarca cinco años de datos de consumo y producción energética de 50 hogares en la ciudad de Funchal (Madeira, Portugal), divididos en tres despliegues diferentes. Esto implica que no existen mediciones de los 50 hogares de forma simultánea a lo largo de los cinco años del proyecto, sino agrupadas en tres conjuntos separados. La Figura 6.1 muestra los tres despliegues mencionados. Dado que trabajamos con mediciones de consumo y generación que están parcialmente correlacionadas con condiciones meteorológicas y temporales, seleccionaremos el periodo de tiempo con el mayor número de hogares desplegados simultáneamente. En nuestro caso, se elige el primer despliegue, en el cual se dispone de mediciones de hasta 23 hogares de manera simultánea.



**Figura 6.1:** Series temporales de los diferentes despliegues del dataset *SustDataED*.

Los datos disponibles en *SustDataED* son bastante variados. La Tabla 6.2 muestra los diferentes tipos de mediciones incluidos en el dataset. Sin embargo, no todas ellas son relevantes para este estudio. Por ejemplo, un alto nivel de granularidad, como las mediciones de eventos de potencia o de eventos de usuario, no es necesario para nuestro análisis. Como se observa en la Tabla 6.2, las mediciones relacionadas con la producción de energía presentan cierta incertidumbre. A primera vista, *SustDataED* resulta adecuado porque ofrece mediciones reales de producción fotovoltaica. No obstante, estos datos se reportan como valores agregados y corresponden a un sistema fotovoltaico que abastece

a todos los hogares del despliegue, sin especificar información sobre las dimensiones del sistema. Además, no está claro si el tamaño del sistema fotovoltaico aumenta con el tiempo, lo que dificulta el análisis a nivel individual de cada hogar.

Tipo de medida	Utilidad
Mediciones del consumo energético	✓
Mediciones de la producción de energía	?
Mediciones demográficas	✓
Mediciones de las condiciones ambientales y climáticas	✓
Mediciones de eventos de usuario	✗
Mediciones de eventos de potencia	✗

**Tabla 6.2:** Tipos de mediciones en el dataset *SustDataED*.

### 6.2.2. Deficiencias del dataset - SustDataED

Dado que la literatura sobre *SustDataED* no ofrecía claridad en cuanto a las mediciones de producción de energía fotovoltaica a nivel individual de cada hogar, se buscó una alternativa para conformar un nuevo conjunto de datos de producción. Por ello, se propuso llevar a cabo simulaciones de producción fotovoltaica en la ciudad de Funchal, con las características y dimensiones de un sistema fotovoltaico convencional, para así tener una estimación coherente de la generación a nivel de un hogar. Para realizar esta simulación de datos de producción, se eligieron dos herramientas con el fin de comparar y garantizar que los resultados obtenidos fueran consistentes:

- **Global Solar Atlas [179]:** Esta plataforma está financiada por el Programa de Asistencia para la Gestión del Sector Energético (ESMAP), con el objetivo de mapear los recursos de energía renovable a nivel mundial, proporcionando acceso tanto a datos promediados a largo plazo como a datos en tiempo real para cualquier ubicación del planeta.
- **PVWatts [180]:** Esta plataforma es proporcionada por el Laboratorio Nacional de Energías Renovables (NREL), que forma parte del Departamento de Energía de los Estados Unidos.

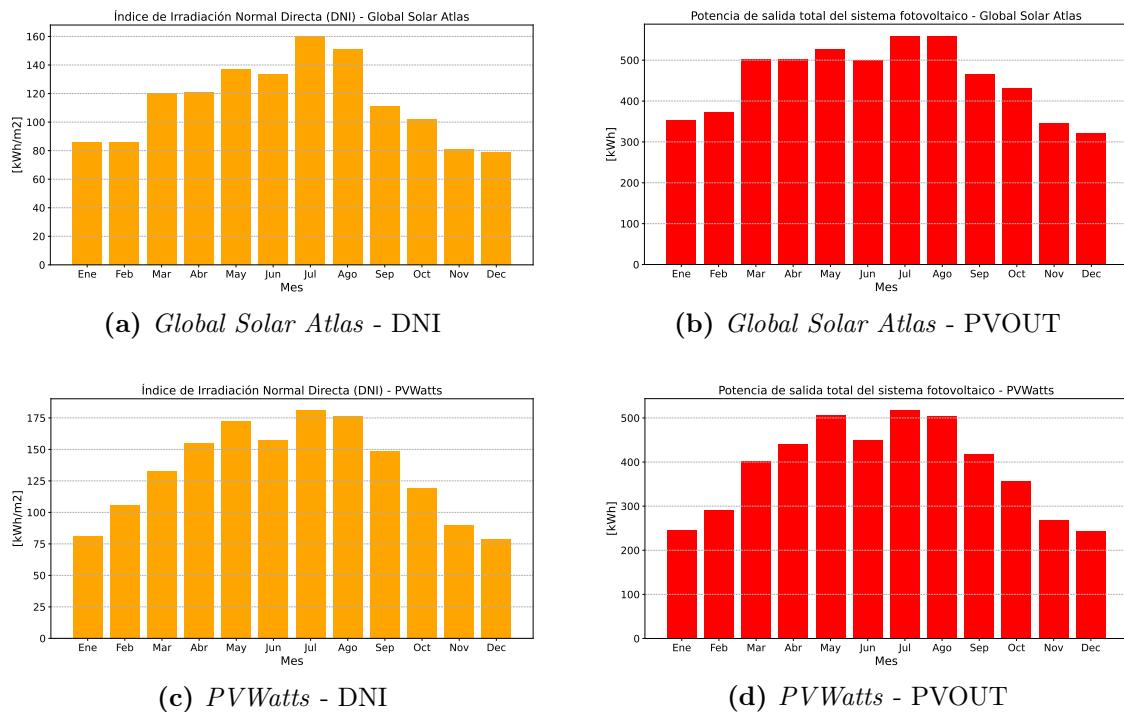
Para cada herramienta, la ubicación se fijó en Funchal (Portugal), dado que el análisis debe corresponder a los datos recogidos en el conjunto *SustDataED*. Una vez establecida la localización, se configuró el sistema fotovoltaico para cada hogar y se extrajeron las capas de datos relevantes, como el Direct Normal Irradiation (DNI) ( $\frac{kWh}{m^2}$ ) y el Photovoltaic Power Potential (PVOUT) ( $\frac{kWh}{kW_p}$ ), que son parámetros clave para estimar la producción de los sistemas fotovoltaicos.

Además, los factores climáticos desempeñan un papel crucial en la verificación de la consistencia de la energía generada. Según las clasificaciones climáticas de Köppen y Trewartha,

## 6.2 BÚSQUEDA Y ANÁLISIS DE FUENTES DE DATOS

Funchal presenta un clima mediterráneo con influencias oceánicas, o bien un clima templado con veranos secos. Al estar situada en una zona subtropical, la ciudad experimenta variaciones diarias mínimas de temperatura, lo que se traduce en condiciones relativamente estables a lo largo del año.

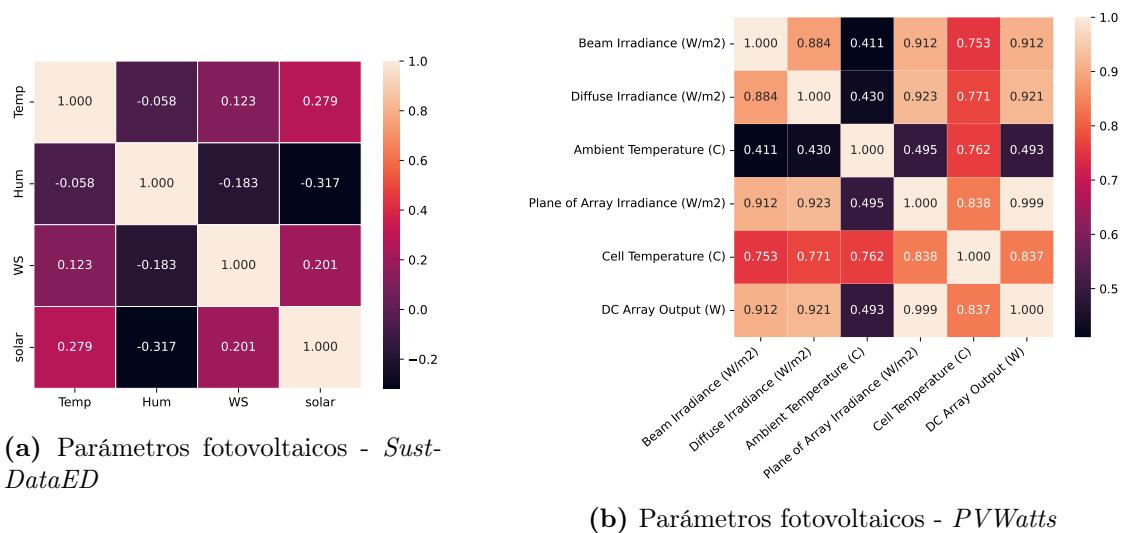
Tras configurar ambos sistemas, se alinearon las fechas correspondientes a las series temporales de *SustDataED* para recopilar los datos de producción fotovoltaica. En ambas herramientas, tal como se muestra en la Figura 6.2, se llevó a cabo un análisis comparativo de los mismos parámetros, DNI y PVOUT, durante los mismos períodos de tiempo. Los resultados, ilustrados en la Figura 6.2, resultaron ser prácticamente idénticos. Una vez verificada esta consistencia entre las dos herramientas para la simulación de valores de producción fotovoltaica, se seleccionó *PVWatts* debido a su capacidad de exportar datos de forma más sencilla, lo que facilitó la integración con los datos de *SustDataED* para la construcción del dataset final.



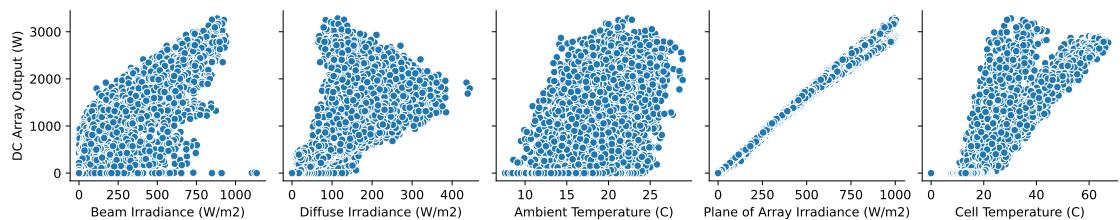
**Figura 6.2:** Comparación entre *Global Solar Atlas* y *PVWatts*, valores mensuales totales de producción de energía fotovoltaica (PVOUT) frente a la irradiación directa normal (DNI).

Además, se realizó un análisis de correlación exhaustivo sobre los parámetros de generación de energía fotovoltaica tanto del conjunto de datos *SustDataED* como de *PVWatts*, con el objetivo de evaluar de forma concluyente la fiabilidad de los datos de producción energética. Al examinar la correlación entre las variables climáticas y la generación de energía en ambos conjuntos de datos, tal como se muestra en la Figura 6.3, emergen patrones diferenciados.

Para *SustDataED* (Figura 6.3 (a)), es evidente que la temperatura presenta una correlación débil con la potencia generada, arrojando un coeficiente de tan solo 0.279. Esta falta de correlación significativa se repite en otras variables, lo que dificulta el análisis de la producción de energía en relación con las condiciones climáticas. En contraste, para el conjunto de datos *PVWatts* (Figura 6.3 (b)), la mayoría de los parámetros muestran un alto grado de correlación, cercano a la unidad. Por ejemplo, la correlación entre la irradiancia del plano del array, que es la cantidad total de radiación solar que incide sobre la superficie de un panel solar, y la generación de potencia es prácticamente ideal, con un coeficiente de 0.999, como se observa claramente en la gráfica.



**Figura 6.3:** Comparación mediante matrices de correlación de los parámetros de producción fotovoltaica.



**Figura 6.4:** Patrones de correlación entre los parámetros de producción en *PVWatts*.

Como se ilustra en la Figura 6.4, el diagrama de dispersión correspondiente a la correlación de los parámetros de producción de *PVWatts*, revela que los parámetros más significativos para la generación de energía fotovoltaica son: la irradiancia del plano del array del panel solar y la temperatura de las celdas solares. Dadas las limitaciones de los datos de producción de *SustDataED*, especialmente en términos de precisión y gra-

nularidad, se decidió utilizar los datos de producción simulados mediante la herramienta *PVWatts*. Esta elección se sustenta en la consistencia mostrada por los datos de *PVWatts* al compararlos con los resultados del *Global Solar Atlas*, y la capacidad que tiene la herramienta para exportar datos de simulación para la integración de los datos de generación fotovoltaica en el dataset final. En la siguiente sección, se presentará el dataset final, sus características clave, y la organización de los ficheros resultantes.

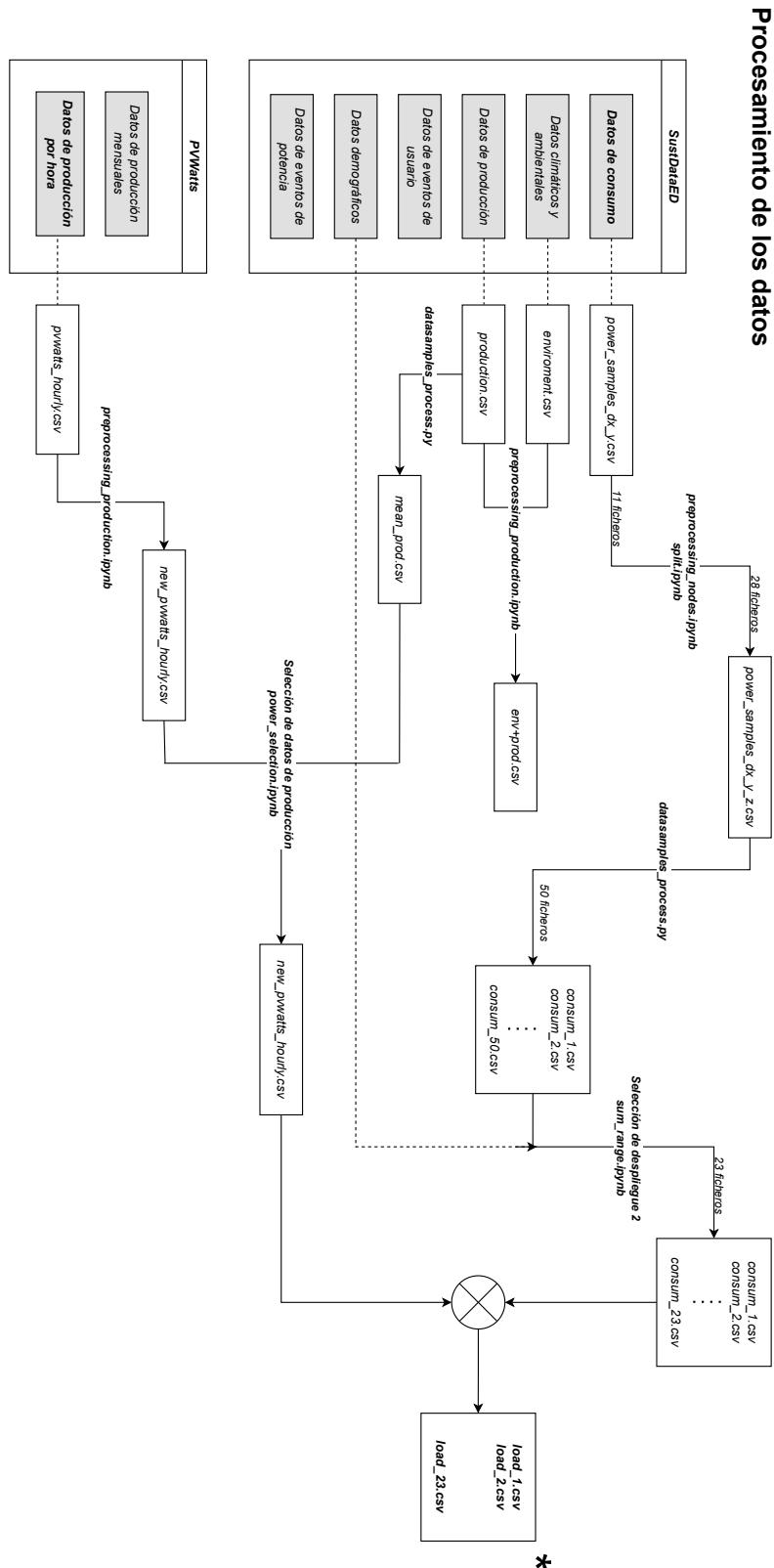
### 6.2.3. Dataset final

Tras identificar y abordar las limitaciones de *SustDataED* mediante la incorporación de la herramienta *PVWatts* para los datos de generación fotovoltaica, se procesó el conjunto completo de datos con el fin de crear el dataset final. El flujo de procesamiento de datos se muestra en la Figura 6.5. Como se observa, existen dos ramas principales: la primera procede del conjunto *SustDataED* e incluye todos los datos de consumo de los hogares, mientras que la segunda corresponde a los datos de generación fotovoltaica. De los 50 hogares, solo se seleccionaron 23 pertenecientes al primer despliegue, como se indicó previamente. Posteriormente, se generaron 23 perfiles de generación fotovoltaica, conformando finalmente 23 archivos de cargas netas reales, que representan la suma de consumo y generación. Los campos finales incluidos en el dataset final se detallan en la Tabla 6.3.

Campo	Descripción	Unidades
<i>timestamp</i>	Instante temporal de medida	datetime
<i>datetime</i>	Fecha del valor promedio	datetime
<i>H</i>	Hora del valor promedio	-
<i>iid</i>	Identificador de vivienda	-
<i>Diffuse Irradiance</i>	Índice de radiación difusa (DIF)	W/m <sup>2</sup>
<i>Plane of Array Irradiance</i>	Índice de radiación en el plano del array (POA)	W/m <sup>2</sup>
<i>Ambient Temperature</i>	Temperatura ambiente	C
<i>Cell Temperature</i>	Temperatura de las células solares	C
<i>DC Array Output</i>	Potencia de salida DC del array	W
<i>AC System Output</i>	Potencia de salida AC del sistema	W
<i>Pavg</i>	Potencia consumida	W
<i>Dif</i>	Carga neta calculada	W

**Tabla 6.3:** Características principales del dataset final (*KeyMonData*).

Este nuevo conjunto de datos requirió un procesamiento adicional para limitar tanto su tamaño como la serie temporal, de modo que se ajustara al despliegue especificado. Además, para hacerlo más manejable, los datos fueron sintetizados. Concretamente, *SustDataED* ofrece medidas con una resolución a nivel de minuto; sin embargo, trabajar con muestras minuto a minuto resulta innecesario, dado que las variaciones de consumo son mínimas. Por ello, se redujo la resolución a muestras horarias. Así, si el primer despliegue se restringe a un año de duración, se obtienen 8760 intervalos temporales ( $24\text{ h} \times 365\text{ días}$ ) por cada hogar, para un total de 23 hogares.



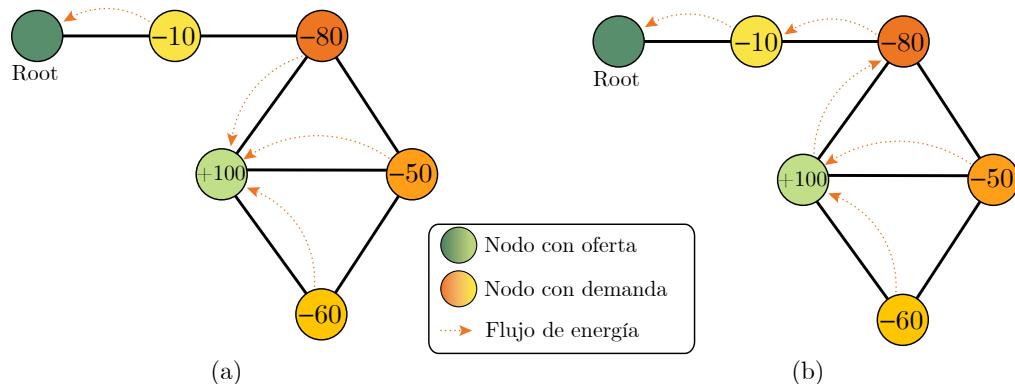
**Figura 6.5:** Esquema de procesamiento de datos para la generación del dataset final.

En cuanto a la convención de nombres de archivos, cada fichero fue etiquetado como *load\_x.csv*, donde *x* corresponde al identificador del hogar para las cargas calculadas. En consecuencia, se generaron un total de 23 archivos de cargas (*load\_x.csv*), cada uno con 8760 intervalos temporales. Tras el procesamiento, limpieza y síntesis, el nuevo dataset fue denominado *KeyMonData* y se ha puesto a disposición pública para su uso por parte de cualquier interesado [181].

### 6.3. Algoritmo de reconfiguración y propuesta de extensión

El algoritmo de reconfiguración para las SG, sobre el cual se construirán los modelos de ML y DL, es DEN2NE [156]. Como se ha explicado anteriormente, ver Capítulo 5, DEN2NE es un algoritmo diseñado para el descubrimiento de rutas en redes densas y heterogéneas, permitiendo automatizar el proceso de encaminamiento desde los nodos hoja hasta el nodo raíz de la topología, al tiempo que gestiona de forma eficiente el uso compartido de recursos. En el contexto específico de una SG, el nodo raíz se define como aquel que dispone de conexión directa con la red de distribución eléctrica, mientras que los nodos “hoja” representan el resto de nodos de la SG.

Para explicar con mayor detalle el proceso de reconfiguración seguido por DEN2NE, la Figura 6.6 ilustra un escenario de redistribución de carga dentro de una SG, similar al ejemplo de la topología IEEE 5-bus. En verde se muestran los nodos con superávit de carga, que puede ser ofrecida a otros nodos de la red. En naranja se destacan los nodos que demandan energía de sus vecinos o, en última instancia, del nodo raíz. Sin embargo, calcular todas las posibles soluciones de redistribución de carga puede ser computacionalmente complejo en un tiempo razonable. Por ejemplo, la Figura 6.6 (a) muestra una redistribución subóptima de recursos, que ocurre cuando los vecinos toman decisiones locales, lo que lleva a que la mayoría de solicitudes se dirijan al nodo con un superávit de +100 (insuficiente para cubrir toda la demanda). En cambio, la Figura 6.6 (b) representa un escenario de distribución óptima, que requiere un enfoque más sofisticado para la distribución eficiente de recursos.



**Figura 6.6:** Ejemplo de redistribución de carga en una SG.

Para abordar este problema en las SG, DEN2NE propone una solución en tres fases. La primera fase, mostrada en la Figura 6.7, consiste en explorar todas las rutas posibles desde el nodo raíz hasta cada nodo de la topología (mecanismo similar al visto en la Sección 5.2.2). Esta exploración, como se ilustra en la figura, se realiza mediante el uso de etiquetas. El proceso comienza en el nodo raíz, que genera la primera etiqueta (1) y la transmite a todos sus vecinos inmediatos, en este caso al nodo 2. El nodo 2 recibe la etiqueta, añade su identificador de nodo, la almacena en su tabla de aprendizaje y, a continuación, la transmite de manera similar a todos sus vecinos contiguos. De esta forma, el nodo 3 recibe la etiqueta 1.2, lo que indica que se encuentra a dos saltos del nodo raíz, y repite el proceso asignando la etiqueta 1.2.3 a sus vecinos, los nodos 4 y 5. Estos, a su vez, realizan el mismo procedimiento, intercambiando las etiquetas 1.2.3.4 y 1.2.3.5, que les permiten aprender una ruta alternativa hacia el nodo raíz. Esto se debe a que ahora disponen de una ruta directa a través del nodo 3, así como de una ruta más larga a través del nodo que les asignó la nueva etiqueta. Finalmente, los nodos 4 y 5 intentarán transmitir las últimas etiquetas aprendidas (1.2.3.4 y 1.2.3.5) de regreso al nodo 3. Sin embargo, la lógica de DEN2NE incorpora un mecanismo de prevención de bucles: si la etiqueta ofrecida forma parte de alguna de las etiquetas previamente aprendidas en la tabla de aprendizaje, será descartada. Por esta razón, etiquetas como 1.2.3.5.4 y 1.2.3.4.5 son descartadas por el nodo 3, tal y como se muestra en la Figura 6.7.

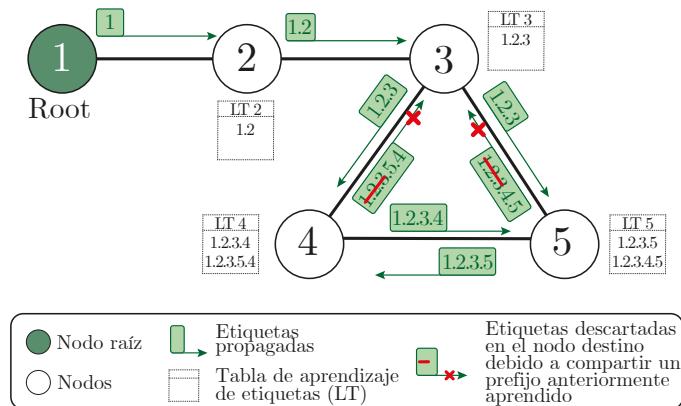


Figura 6.7: Funcionamiento de la primera fase del algoritmo DEN2NE.

Una vez finalizado el proceso de etiquetado en toda la red, se considera completa la primera fase del algoritmo. La segunda fase de DEN2NE consiste en seleccionar la mejor etiqueta por nodo para la redistribución de carga, basándose en los criterios definidos en el propio algoritmo (Ver Sección 5.2.3). En este ejemplo, los nodos con una sola etiqueta no necesitan aplicar esta fase. Sin embargo, los nodos 4 y 5 deberán elegir qué etiqueta utilizarán, ya que esta determinará posteriormente cómo redistribuirán la carga dentro de la SG. La tercera fase del algoritmo se centra en la redistribución de cargas entre los nodos (Ver Sección 5.2.4). Esta fase parte de la suposición de que todos los nodos han seleccionado una etiqueta, es decir, una ruta para llegar al nodo raíz. Como resultado, se formará una topología lógica sobre la topología física de la SG, lo que puede dejar ciertos

enlaces sin utilizar. Esto permite una distribución óptima de la energía dentro de la SG, como se ilustra en la Figura 6.6 (b). En resumen, DEN2NE primero explora toda la topología de la red (asignando etiquetas durante el proceso), luego define el orden a seguir para la distribución de recursos (este orden está representado por la etiqueta seleccionada) y, finalmente, reconfigura la red en función de dicho orden.

En este trabajo, se evalúan varios modelos de ML y DL para apoyar al algoritmo DEN2NE durante su segunda fase. Más concretamente, estos modelos proporcionarán información adicional para mejorar la selección de etiquetas (y, por ende, el orden de distribución de recursos), ya que serán elegidas considerando la predicción de posibles fallos que puedan surgir en la SG, lo que, en última instancia, mejorará el proceso de reconfiguración en su conjunto. Es importante destacar que, aunque evaluamos estos modelos exclusivamente con DEN2NE, tanto nuestro conjunto de datos como nuestro análisis de técnicas de IA también podrían servir como base para analizar otros algoritmos de reconfiguración en redes SG. En definitiva, las técnicas examinadas en nuestro estudio ofrecen información complementaria sobre la predicción de fallos, y queda a discreción del mecanismo de reconfiguración específico incorporar dicha información para mejorar su procedimiento.

## 6.4. Evaluación

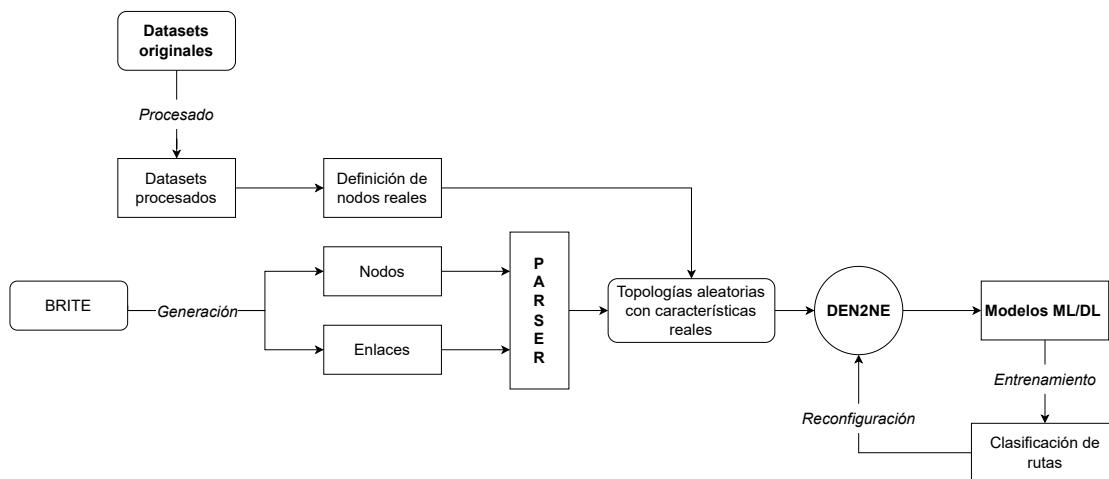
Considerando el objetivo de este trabajo, que busca detectar y predecir errores, o fallos más específicamente, es necesario definir primero un criterio de error para etiquetar los resultados del algoritmo. Dado que las simulaciones están configuradas con un escenario real que contempla pérdidas y limitaciones de capacidad en los enlaces, se decide establecer la condición de fallo en función de la presencia de un exceso de capacidad en un intercambio de energía entre dos nodos de la red. Estos errores no se introducen de forma arbitraria, sino que se simula el balance de potencia en base a la configuración establecida de la SG. En los escenarios donde ocurren pérdidas por exceso de capacidad, se indica que, para una entrada determinada del dataset, bajo esas condiciones específicas, se producirá un fallo en la red. Posteriormente, nuestro objetivo es clasificar si ocurrirá un fallo en función de condiciones y configuraciones predefinidas. En este paso, se decide aplicar una configuración de enlaces basada en el IEEE 123 Node Test Feeder [163] para garantizar que el proceso de asignación de capacidades a los enlaces sea realista. Para simplificar, en la Tabla 6.4 se definen tres tipos de enlaces.

$R$ (ohm/km)	$I$ max (A)	Sección ( $\text{mm}^2$ )
0.272	185	70
0.78	100	25
1.91	53	10

**Tabla 6.4:** Configuración de enlaces basada en la topología IEEE 123 Node Test Feeder.

Una vez que la definición de error está clara, en esta sección se evaluarán diversas técnicas de ML y DL en términos de predicción de error. En particular, se definen Random

Forest (RF) y Support Vector Machine (SVM) como métodos de ML, y ANN como un método de DL, en función de sus características estructurales. Nuestra evaluación sigue el flujo de trabajo ilustrado en la Figura 6.8, en el cual, primero se genera un conjunto exhaustivo de topologías aleatorias haciendo uso de BRITE, y luego se les añaden características reales, como consumo y generación por nodo, condiciones climáticas (empleando el dataset final *KeyMonData*). El siguiente paso es la configuración del entorno de simulación utilizando DEN2NE, y finalmente se realiza un análisis detallado de los distintos modelos de ML y DL, cada uno de los cuales, una vez entrenados, serán capaces de predecir si una ruta obtenida por DEN2NE dará lugar a un error, exponiendo la capacidad de reconfigurar de forma proactiva la red.



**Figura 6.8:** Definición de la *pipeline* para detectar y predecir errores durante el proceso de distribución de energía.

El proceso de generación aleatoria de topologías desempeña un papel fundamental en la validación y prueba de los modelos de ML y DL. La creación de escenarios de red diversos establece una base sólida para la detección y predicción precisa de fallos en redes SG reales. Para ello, de igual forma que en el Capítulo 5, se empleó el generador de topologías aleatorias BRITE. Un aspecto clave de este trabajo es la automatización de la generación de topologías, ya que resulta esencial para llevar a cabo simulaciones a gran escala y posibilitar una evaluación exhaustiva de los modelos de ML y DL en la detección de errores. Mediante el uso de scripts personalizados, se automatiza la generación de topologías con BRITE, obteniendo un total de 180 topologías que posteriormente se utilizan en diferentes ejecuciones de simulación dentro de DEN2NE. El número total de topologías generadas se determina como el producto de varios parámetros configurados:

- **Modelos de topología:** dos modelos, Router Waxman y Router Barabasi-Albert.
- **Número de nodos:** se generan topologías con 100, 150 y 200 nodos.
- **Niveles de conectividad:** se especifican tres grados distintos de conectividad,

considerando que cada topología presenta un grado de 2m (debido a los enlaces bidireccionales).

- **Semillas aleatorias:** se aplican 10 archivos de semillas diferentes para garantizar la diversidad en las topologías generadas.

Una vez generadas, las topologías se importan al marco de trabajo DEN2NE, donde el algoritmo ejecuta simulaciones para modelar la distribución de energía entre los nodos. Dichas simulaciones incorporan perfiles de carga reales previamente extraídos y procesados en la Sección 6.2.3, lo que permite reproducir procesos de distribución de energía en entornos realistas de SG. Dado que el objetivo es detectar transacciones energéticas entre nodos que superen la capacidad de sus enlaces, se define un escenario de enlaces limitados con pérdidas. Además, en el caso de BRITE, se aplican 5 archivos de semillas distintos a DEN2NE, lo que permite obtener simulaciones diferentes a partir de la misma topología de entrada.

Considerando los parámetros configurados y las 8760 filas de datos reales disponibles (correspondientes a un año de mediciones horarias), el algoritmo podría ejecutar potencialmente hasta 47,304,000 simulaciones únicas. Sin embargo, dado que resulta ineficiente e impracticable ejecutar todas ellas, es necesario seleccionar cuidadosamente qué pruebas realizar en DEN2NE. En este trabajo se han elegido 12 instantes específicos de tiempo, cada uno representando una hora concreta en un día de cada mes. Para simplificar el proceso, y dado que el periodo de datos comienza el 28 de noviembre de 2010, se selecciona el día 28 de cada mes. La hora escogida para la extracción de perfiles de carga es las 11:00 de la mañana, ya que en ese intervalo suele registrarse la mayor producción promedio de energía. Esta estrategia de selección aumenta la probabilidad de encontrar intercambios energéticos que excedan la capacidad de los enlaces, generando un mayor número de fallos en el proceso de distribución. Dicho enfoque resulta crucial para entrenar de manera efectiva los modelos de ML y DL.

#### 6.4.1. Técnicas de Machine Learning

Para detectar y predecir fallos en el proceso de distribución energética se realiza una clasificación binaria de las instancias del conjunto de datos empleando dos técnicas supervisadas de ML: RF y SVM. El RF combina múltiples árboles de decisión y obtiene la predicción final por votación mayoritaria; cada árbol divide el espacio de características siguiendo criterios como la entropía o el índice de Gini, lo que le aporta robustez frente al ruido y capacidad de modelar interacciones no lineales entre variables. Por su parte, el SVM (Support Vector Machine) busca el hiperplano óptimo que maximice la separación entre las dos clases; su comportamiento frente a no linealidades se ajusta mediante el tipo de kernel y el parámetro  $\gamma$ , mientras que el parámetro de regularización  $C$  controla el equilibrio entre margen y errores de clasificación.

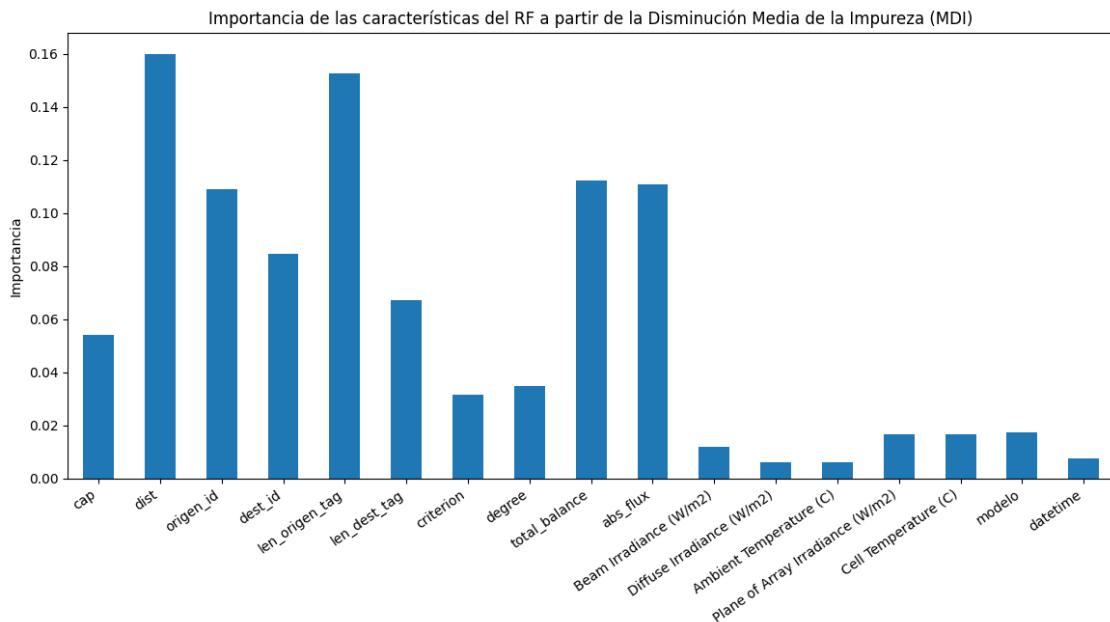
El desarrollo práctico de ambos modelos sigue una secuencia de pasos estándar para garantizar modelos bien entrenados y comparables: limpieza y normalización de los datos (especialmente relevante para SVM), selección/ingeniería de características, particionado

en conjuntos de entrenamiento y evaluación (p. ej. *train/validation/test*), tratamiento de desequilibrios de clase (p. ej. ponderación de clases o técnicas como SMOTE), búsqueda de hiperparámetros mediante *grid* o *random search* y validación cruzada estratificada para estimar rendimiento. Finalmente, la implementación y experimentación se realiza con la biblioteca *scikit-learn*, aprovechando sus utilidades para ajuste de modelos, validación cruzada y métricas, lo que facilita la reproducibilidad y comparación sistemática entre las configuraciones.

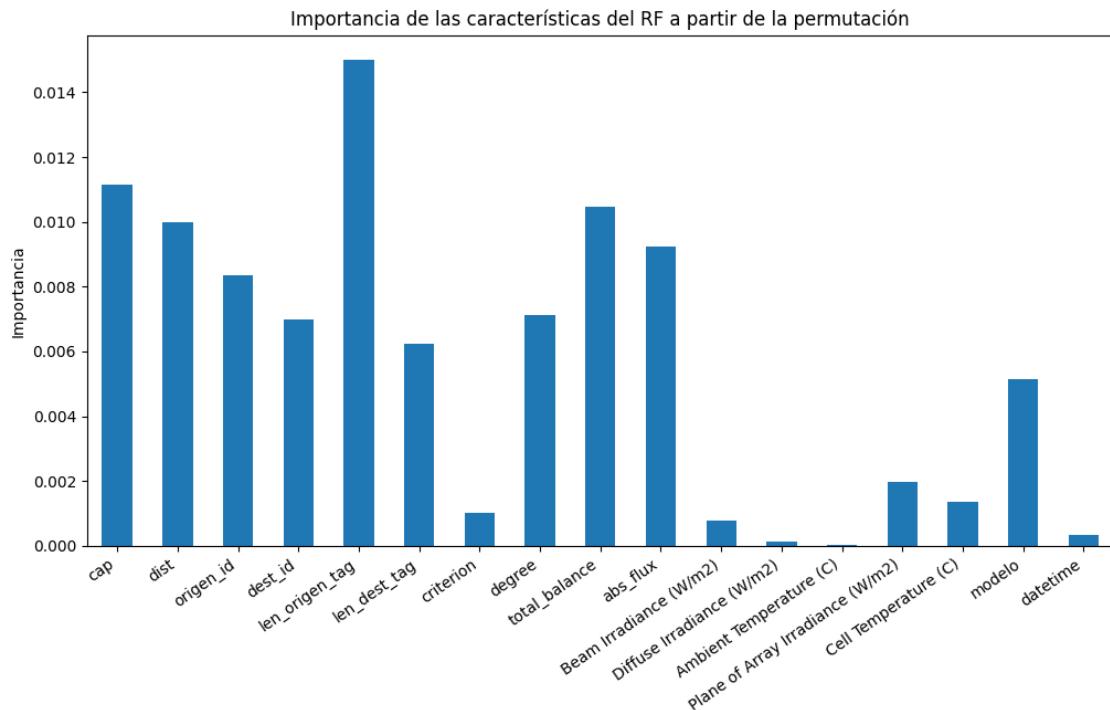
#### 6.4.1.1. Puntuación de características

En este primer paso se diseña un modelo por defecto para cada técnica de ML, con el fin de identificar qué características del conjunto de datos resultan más relevantes en el proceso de clasificación. Para la técnica de RF, se entrena un modelo utilizando 10 estimadores y el criterio de entropía. La importancia de las características del conjunto de datos se estima a través del método *feature\_importances\_* proporcionado por el clasificador. Este cálculo se basa en la media y la desviación estándar de la reducción de impureza que ocurre en cada árbol. En la Figura 6.9 se muestran los valores relativos de importancia asignados a cada característica, cuya suma total es igual a 1. Se observa un impacto significativo en la distancia entre nodos y en los parámetros derivados de DEN2NE (*total\_balance* y *abs\_flux*), que corresponden respectivamente a la carga del nodo root tras el balanceo energético y al flujo total de recursos intercambiados en el proceso de distribución. No obstante, este método introduce un sesgo hacia aquellas características con alta cardinalidad, es decir, con un gran número de valores únicos. Dicho sesgo se debe a que estas características generan nodos de división más profundos en los árboles, al existir más opciones para particionar el conjunto de datos; por ello, tienen mayor probabilidad de recibir puntuaciones más altas de importancia. Teniendo esto en cuenta, se emplea el método de importancia por permutación como análisis comparativo. En este segundo método, los valores de cada característica se permutan aleatoriamente de manera individual, evaluando el rendimiento del clasificador en cada iteración. Así, una característica que provoca una disminución significativa en la precisión del modelo al ser permutada recibe una puntuación más alta en importancia. En este caso, como se muestra en la Figura 6.10, los parámetros derivados de DEN2NE (*total\_balance* y *abs\_flux*) mantienen puntuaciones elevadas de importancia; sin embargo, ahora también la longitud de las etiquetas de los nodos y la capacidad de los enlaces presentan una relevancia destacada.

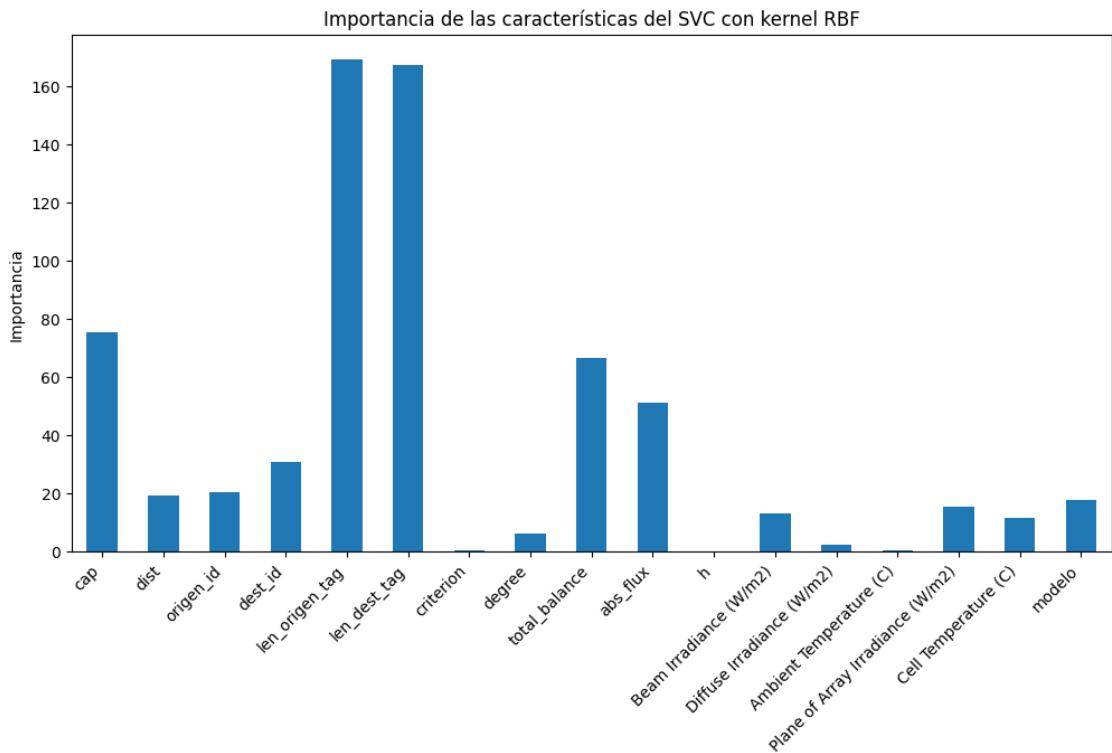
Para la técnica de SVM, se entrena un modelo por defecto con un kernel de Función de Base Radial (RBF), adecuado para conjuntos de datos no linealmente separables. En consecuencia, la importancia de las características del conjunto de datos se evalúa a través de los métodos *support\_vectors\_* y *dual\_coef\_* proporcionados por la clase. Tal como se ilustra en la Figura 6.10, los resultados indican que la longitud de las etiquetas de origen y destino ejerce una influencia significativa en el conjunto de datos. De manera análoga a lo realizado con RF, se aplica el método de permutación, obteniéndose resultados similares. Se observa que la longitud de las etiquetas, la capacidad de los enlaces y el flujo energético total (*abs\_flux*) se destacan en la clasificación.



**Figura 6.9:** Puntuación de la importancia de las características para el RF utilizando el método *feature\_importances\_*.



**Figura 6.10:** Puntuación de la importancia de las características para el RF utilizando el método *permutation\_importance*.



**Figura 6.11:** Puntuación de importancia de las características para el SVM utilizando los métodos *support\_vectors\_* y *dual\_coef\_*.

#### 6.4.1.2. Optimización de hiperparámetros

Para encontrar la mejor combinación de hiperparámetros en cada modelo se emplea el método *Grid Search*. Este método realiza una búsqueda exhaustiva dentro de una cuadrícula de hiperparámetros con el fin de identificar la combinación que ofrece la mayor precisión del modelo.

En el caso de la técnica RF, el análisis se centró en dos hiperparámetros principales: el número de estimadores o árboles y el criterio de medida de impureza (basado en la entropía o en Gini). La ejecución del método determinó, a partir de los métodos *best\_score\_* y *best\_params\_*, la mejor combinación de hiperparámetros, alcanzando una precisión del 99.18% con un clasificador de 100 estimadores empleando el criterio de entropía (Tabla 6.5). Para un análisis más detallado, se utilizó el atributo *cv\_results\_*, que proporcionó información completa sobre la búsqueda. Se observa que, aunque las variaciones en la precisión entre las distintas combinaciones de parámetros fueron mínimas, los tiempos de ajuste mostraron diferencias significativas. Tal como se presenta en la Tabla 6.6, la duración del entrenamiento aumentó de forma proporcional al número de estimadores configurados en el modelo. Por lo tanto, dado que la precisión no mejoró de manera significativa al superar los 25 estimadores, se determinó que el clasificador con 25 estimadores y criterio de entropía constituye el modelo más óptimo.

Número de estimadores	Criterio	
	Entropía	Gini
10	99.07	99.02
25	99.16	99.14
50	99.16	99.15
75	99.17	99.17
100	99.18	99.16

**Tabla 6.5:** Evolución de la Precisión (%) aplicando la búsqueda (*Grid Search*) sobre el RF.

Número de estimadores	Criterio	
	Entropía	Gini
10	147	146
25	373	382
50	747	761
75	1072	1002
100	1439	987

**Tabla 6.6:** Evolución del Tiempo (s) de entrenamiento del RF en función del número de estimadores.

Para modelar una SVM óptima se estudiaron principalmente dos hiperparámetros: el tipo de núcleo (*kernel*) y el parámetro de regularización  $C$  (véase Tablas 6.7a y 6.7b). El parámetro  $\gamma$  fue omitido debido al escalado previo de los datos, que eliminó su influencia. En este caso, a diferencia de la técnica RF, la búsqueda de la mejor combinación de hiperparámetros requirió un mayor tiempo de ejecución, alcanzando aproximadamente las 350 horas. Los resultados de este proceso mostraron que el modelo óptimo correspondía a una SVM con núcleo RBF y un parámetro de regularización  $C$  igual a 1, logrando una precisión del 97.78 % y requiriendo, además, un tiempo de entrenamiento relativamente bajo en comparación con otras combinaciones de hiperparámetros.

**Tabla 6.7:** Resultados extraídos del método *cv\_results\_* de la búsqueda *Grid Search* en la SVM.(a) Precisión (%) (*mean\_test\_score*)

<i>Kernel / C</i>	<i>poly</i>	<i>rbf</i>	<i>sigmoid</i>
0.25	97.65	97.66	95.96
0.5	97.65	97.71	95.94
0.75	97.65	97.75	95.82
1	97.65	97.78	95.81

(b) Tiempo (s) (*mean\_fit\_time*)

<i>Kernel / C</i>	<i>poly</i>	<i>rbf</i>	<i>sigmoid</i>
0.25	21030	13876	14659
0.5	28807	10277	18695
0.75	42183	9328	12235
1	47460	9446	12869

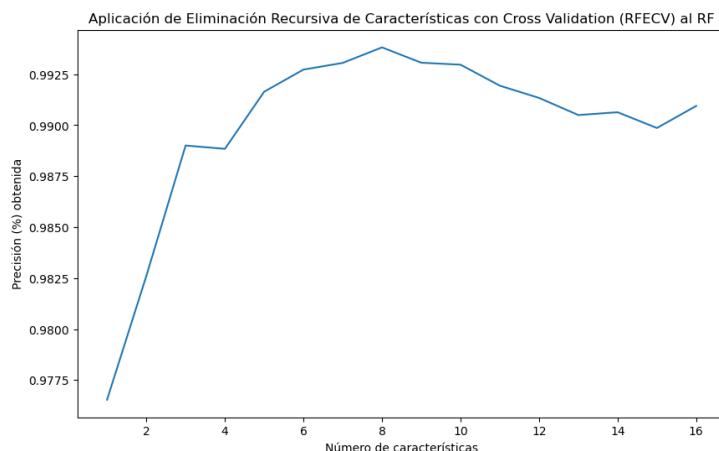
El análisis del atributo *cv\_results\_* reveló que el núcleo *sigmoid* ofreció el peor desempeño, mientras que el núcleo polinómico (*poly*) presentó un aumento del tiempo de

entrenamiento a medida que se incrementaba el valor de  $C$ , lo que sugiere que un valor elevado de este parámetro no resulta adecuado para dicho núcleo.

#### 6.4.1.3. Reducción de dimensionalidad

Tras analizar las mejores combinaciones de hiperparámetros para cada técnica de ML, se estudia la aplicación de algunas metodologías de reducción de dimensionalidad sobre el dataset. El objetivo es eliminar información irrelevante o redundante para lograr un entrenamiento de modelos más eficiente. Se probaron dos técnicas diferentes con el fin de comparar posteriormente el rendimiento de los modelos RF y SVM en la sección de resultados.

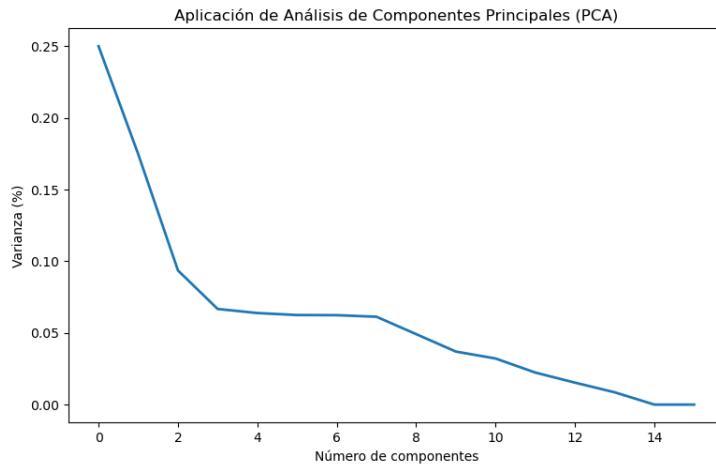
El primero de ellos es Recursive Feature Elimination with Cross Validation (RFECV). Este método descarta de manera iterativa las características menos influyentes hasta que el rendimiento del modelo deja de mejorar significativamente. Con una validación cruzada de 5 *folds*, puede observarse en la Figura 6.12 que la máxima precisión para el modelo óptimo de RF se alcanzó con ocho características. Además, se confirmó que las características identificadas (*cap*, *dist*, *origen\_id*, *dest\_id*, *len\_origen\_tag*, *len\_dest\_tag*, *total\_balance*, *abs\_flux*) coincidían con aquellas que obtuvieron las puntuaciones más altas en la Sección 6.4.1.1. Sin embargo, en el caso de SVM, la clase del clasificador no contaba con los atributos necesarios para implementar RFECV (*coef\_*, *feature\_importances\_*), por lo que esta técnica solo se aplicó a RF.



**Figura 6.12:** Análisis de la precisión del modelo RF basado en el número de características utilizadas con el método RFECV.

El segundo método que se implementó fue PCA, el cual, reduce la dimensionalidad del conjunto de datos utilizando la técnica Singular Value Decomposition (SVD). Su funcionamiento se basa en encontrar las direcciones principales de variación ( $k$  vectores) en el conjunto de datos para construir una matriz de proyección, que establece un nuevo espacio de características de dimensión  $k$ . Para definir el número óptimo de componentes,

se empleó el método del codo. La Figura 6.13 muestra que la varianza se estabiliza en tres componentes, por lo que, con fines comparativos, se estudió el rendimiento de RF y SVM para dos y cuatro componentes. El tercer método es la selección de características univariada, el cual, utiliza el método *SelectKBest()* que requiere un escalado previo y la especificación de un número fijo de características con las que trabajar. En este caso, se seleccionaron valores de  $k$  iguales a cinco y ocho características.



**Figura 6.13:** Análisis de varianza basado en el número de componentes utilizados en el PCA.

#### 6.4.2. Técnicas de Deep Learning

El desarrollo de las técnicas de DL se dividió en el diseño óptimo y el entrenamiento de dos modelos de ANN, empleando dos bibliotecas diferentes: *Scikit-Learn* y el módulo *Keras* de *TensorFlow*. Las ANNs, y en particular las Multilayer Perceptrons (MLPs), se componen de múltiples capas de nodos y conexiones que imitan la estructura neuronal del cerebro humano. En cuanto a su desarrollo, cabe destacar que se omitieron las etapas de puntuación de características y de reducción de dimensionalidad aplicadas a los modelos de ML en la Sección 6.4.1. Esto se debe a que las ANNs son capaces de analizar internamente y ponderar la relevancia de las características dentro de la red, por lo que dichos pasos no resultan necesarios en este caso.

##### 6.4.2.1. Optimización de hiperparámetros

En esta etapa se lleva a cabo el ajuste de hiperparámetros con el fin de lograr un rendimiento óptimo de una ANN. Para ello, se aplica el método Grid Search al modelo MLP proporcionado por la biblioteca *Scikit-Learn*. El estudio se centra en la configuración de las capas ocultas y el número de neuronas por capa (*hidden\_layer\_sizes*), así como en otros hiperparámetros como la función de activación (*activation*) y el algoritmo de optimización (*solver*). Para evitar el sobreajuste durante la ejecución de Grid Search, se establece un máximo de 100 iteraciones junto con la técnica de *early stopping*, que se activa si no se producen mejoras significativas en 10 iteraciones consecutivas.

Se ha estimado que el proceso completo de búsqueda requiere aproximadamente 2.34 horas en un servidor con 32 procesadores. En la Tabla 6.8 se observa que el algoritmo de optimización Stochastic Gradient Descent supera, en general, al algoritmo *Adam*. En cuanto a los tiempos de entrenamiento, la Tabla 6.9 refleja el impacto tanto de la estructura de capas como del número de neuronas configuradas en la red. De manera similar, las dimensiones también influyen en las precisiones alcanzadas, registrándose algunos casos de sobreajuste que derivaron en menores tasas de clasificación.

<i>Solver</i>	<i>adam</i>		<i>sgd</i>	
	<i>relu</i>	<i>tanh</i>	<i>relu</i>	<i>tanh</i>
(5,)	90.17	97.61	97.66	97.54
(8,)	92.03	89.73	97.65	97.65
(10,)	87.00	89.90	97.48	89.76
(50,)	86.89	89.82	89.53	97.65
(100,)	90.95	89.71	84.56	89.72
(5, 5)	89.80	89.72	97.66	97.65
(8, 8)	88.49	89.72	89.89	97.65
(10, 10)	81.68	89.70	90.30	97.65
(50, 50)	81.91	82.04	97.66	89.73

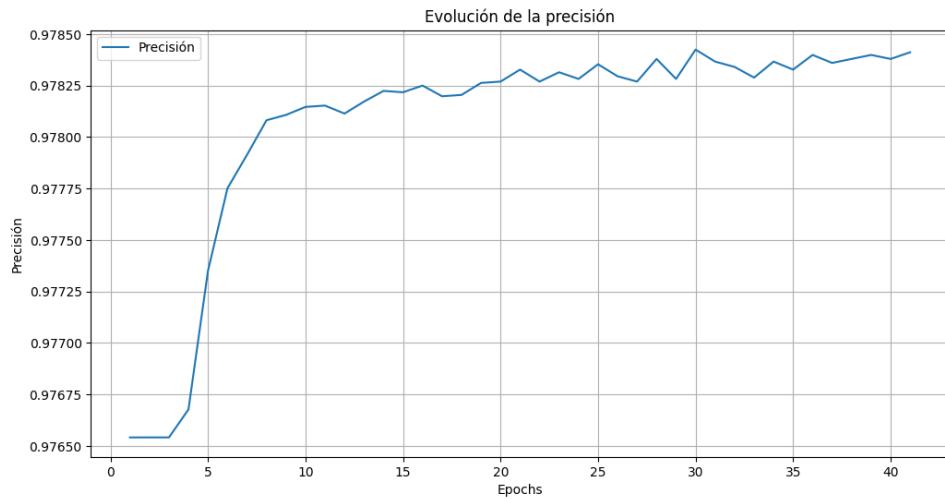
**Tabla 6.8:** Precisión (%) (*mean\_test\_score*) extraída del método *cv\_results\_* del MLP.

<i>Solver</i>	<i>adam</i>		<i>sgd</i>	
	<i>relu</i>	<i>tanh</i>	<i>relu</i>	<i>tanh</i>
(5,)	88	37	63	35
(8,)	96	43	35	36
(10,)	94	40	56	43
(50,)	157	168	143	70
(100,)	236	223	170	131
(5, 5)	136	50	52	44
(8, 8)	159	54	69	46
(10, 10)	149	55	54	48
(50, 50)	333	319	166	155

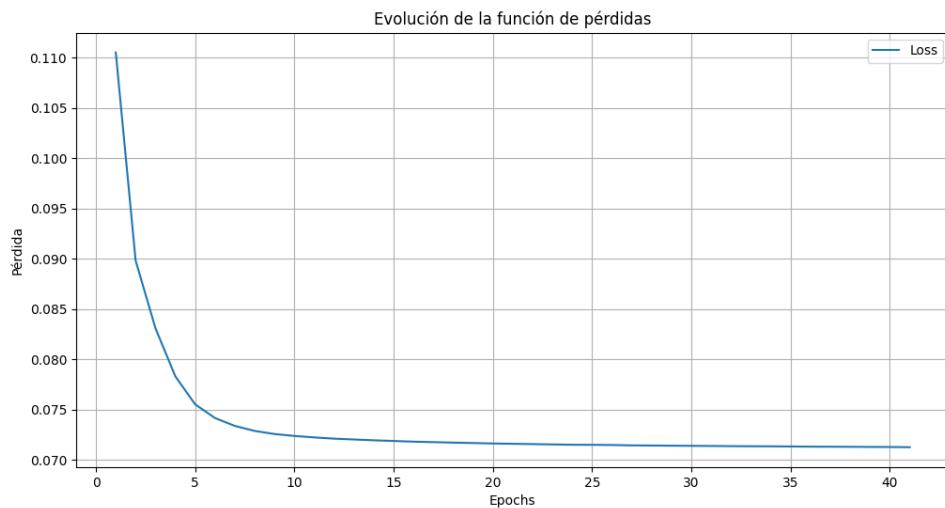
**Tabla 6.9:** Tiempo (s) (*mean\_fit\_time*) extraída del método *cv\_results\_* del MLP.

Por tanto, el análisis de resultados determina que la configuración óptima del MLP está formada por dos capas ocultas de 5 neuronas cada una (estructura (5,5)), función de activación Rectifier Lineal Unit (ReLU) y algoritmo de optimización SGD (solver). Las Figuras 6.14 y 6.15 muestran, respectivamente, la evolución de la precisión y de la función de pérdidas durante el entrenamiento. El modelo óptimo converge en la iteración (*epoch*)

41, alcanzando una precisión del 97.66 % y un valor de la función de pérdida de 0.0712.



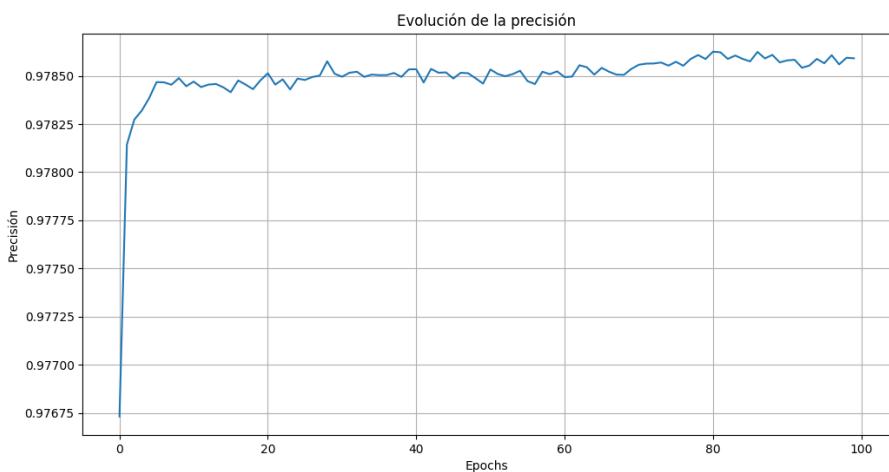
**Figura 6.14:** Evolución de la precisión del ANN de *Scikit-Learn*.



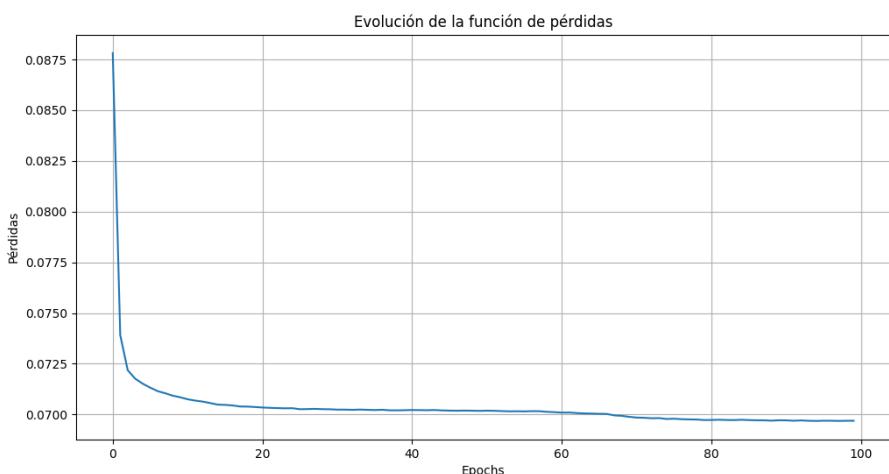
**Figura 6.15:** Evolución de la función de pérdidas para la ANN de *Scikit-Learn*.

Tras identificar la configuración de hiperparámetros óptima para el MLP mediante *Grid Search* sobre la implementación de *Scikit-Learn*, dicha configuración se replicó en un modelo ANN equivalente implementado con el módulo *Keras* de *TensorFlow*. El objetivo de esta réplica es comprobar la consistencia de los resultados y verificar que dos implementaciones distintas de redes neuronales con la misma arquitectura e hiperparámetros alcanzan comportamientos comparables en términos de convergencia, precisión y función de pérdidas.

En las Figuras 6.16 y 6.17 se muestra la evolución de la precisión y de la función de pérdidas durante el entrenamiento del ANN en *Keras*. A diferencia del MLP de *Scikit-Learn*, que alcanzó convergencia en la iteración 41 bajo la estrategia de *early stopping*, la implementación en *Keras* completó las 100 iteraciones programadas; aun así, los resultados finales son muy similares: la pérdida en la última iteración es aproximadamente 0.068 y la precisión alcanzada ronda el 97.87 %. Estas cifras corroboran la reproducibilidad del diseño elegido y apuntan a una estabilidad del modelo frente a diferencias internas entre librerías (por ejemplo, implementaciones del optimizador, inicialización de pesos o manejo del *early stopping*). En conjunto, los experimentos muestran que la configuración propuesta para la red neuronal es robusta y proporciona un rendimiento elevado y consistente para la tarea de predicción de fallos en reconfiguración de SGs.



**Figura 6.16:** Evolución de la precisión para el ANN implementado en *Keras*.



**Figura 6.17:** Evolución de la función de pérdidas para el ANN implementado en *Keras*.

## 6.5. Resultados y discusión de los modelos entrenados

Los resultados recogidos en la Tabla 6.10 ofrecen una visión global del rendimiento de los modelos estudiados. Para validar los modelos de ML se han empleado dos métodos complementarios: la matriz de confusión (y sus métricas derivadas) y la validación cruzada *K-Fold*. En el caso de las técnicas de DL, además de la matriz de confusión, se han usado las métricas que proveen *Scikit-Learn* y *Keras* (pérdidas y precisión) durante el entrenamiento y la validación.

A continuación se presentan los resultados y su discusión para cada familia de modelos; primero se exponen los resultados obtenidos y, a continuación, se interpretan sus implicaciones prácticas y limitaciones. Al final se discute brevemente la generalización potencial a otras topologías y algoritmos de reconfiguración.

### 6.5.1. Random Forest

La matriz de confusión proporciona métricas esenciales para evaluar el rendimiento del modelo RF. Si nos fijamos únicamente en la *Accuracy*, los resultados parecen buenos en las cinco pruebas realizadas con distintas técnicas de reducción de dimensionalidad aplicadas durante el entrenamiento del RF. No obstante, al analizar las métricas de *Precision* y *Recall* surgen diferencias relevantes. Por ejemplo, la aplicación de PCA con dos componentes obtiene una *Accuracy* elevada (97.55 %), pero la *Precision* es muy baja (12.52 %), lo que indica que el porcentaje de errores efectivamente detectados respecto al total de errores del conjunto es extremadamente reducido (0.67 %). En consecuencia, el *F1 Score* también resulta muy bajo (1.27 %), ya que combina *Precision* y *Recall*.

Analizando en conjunto todas las métricas ofrecidas por la matriz de confusión, se concluye que la técnica de reducción de dimensionalidad que mejor funciona para entrenar el RF es RFECV. Este método presenta el menor coste en predicción de errores porque reduce tanto la probabilidad de falsas alarmas como la de no detectar errores reales, alcanzando altas tasas de *Precision* (94.28 %) y *Recall* (81.05 %). Como segunda mejor opción para la detección y predicción de errores se sitúa el uso de todas las características del conjunto de datos, sin aplicar reducción alguna.

Los resultados obtenidos mediante la validación cruzada *K-Fold* muestran las mismas tendencias en *accuracy* que las derivadas de la matriz de confusión, lo que confirma la robustez y buena generalización del enfoque con RFECV en nuestro conjunto de datos.

### 6.5.2. Support Vector Machine

Para evaluar el rendimiento del modelo SVM conviene subrayar el elevado coste computacional y temporal requerido para su entrenamiento: en nuestro caso fueron necesarias aproximadamente 325 horas para ejecutar y validar las 5 pruebas. En la matriz de confusión se observa de inmediato que, en ambos casos en los que se aplicó PCA, aparecen valores indefinidos para las métricas de *Precision* y *F1 Score*; esto se debe a que el mo-

delo SVM no realiza predicciones de la clase de interés (es decir, no predice ejemplares positivos), por lo que dichas métricas no pueden calcularse.

Aunque el porcentaje de *Accuracy* resulte alto (97.61 %), el modelo no está realizando una clasificación útil para nuestro objetivo, y el coste asociado a errores (falsos negativos/falsos positivos según el caso) es elevado. Respecto a la *Precision*, los mejores resultados se obtienen al no aplicar ninguna técnica de reducción de dimensionalidad (89.32 %); en cambio, con la selección univariante de características (*SelectKBest*) la eficiencia disminuye al reducir el número de características (por ejemplo, 65.25 % con menos variables). Por otra parte, la métrica *Recall* presenta valores bajos en todas las pruebas, lo que indica que la proporción de fallos correctamente identificados es muy pequeña.

Con base en este análisis se concluye que el SVM no resulta suficientemente eficiente para el objetivo de detección y predicción de fallos en nuestro contexto. La validación por *K-Fold* arroja valores de *accuracy* similares a los de la matriz de confusión (como ocurría con el RF), pero las limitaciones comentadas: alto coste computacional; baja capacidad para identificar la clase positiva; y sensibilidad a la reducción de dimensionalidad, hacen que descartemos el uso de SVM para detección de errores en SG. Estos resultados aportan una justificación sólida para priorizar modelos alternativos más adecuados a la naturaleza del problema.

### 6.5.3. ANNs

Dado que en el desarrollo de las técnicas de DL no se aplicaron pasos dedicados de puntuación de características ni de reducción de dimensionalidad, la evaluación se centra en la comparación de los resultados obtenidos por las dos ANN desarrolladas. Como se indicó en la Sección 6.4.2, el rendimiento de ambos ANN es muy similar: las precisiones alcanzadas son prácticamente idénticas (97.84 %) y las demás métricas derivadas de la matriz de confusión muestran diferencias mínimas entre ambos modelos. Se observa un *Recall* ligeramente superior en la ANN de *Scikit-Learn* (14.98 %), si bien ambos modelos evidencian una eficacia reducida en la detección de fallos reales. Por su parte, la ANN implementada con *Keras* presenta una *Precision* mayor (72.17 %), lo que se traduce en una menor probabilidad de falsas alarmas. Además, se aplicaron los métodos de evaluación provistos por *Scikit-Learn* (`score()`) y por *Keras* (`evaluate()`) a ambos modelos: `score()` devuelve la precisión, mientras que `evaluate()` aporta también el valor de la función de pérdida. Los resultados siguen siendo comparables entre ambas implementaciones y, si se prioriza la detección de errores reales (*recall*), la ANN de *Keras* resulta la opción preferible por su menor tasa de falsas alarmas relativa.

Por último, cabe destacar que algunos métodos ML más sencillos (por ejemplo, RF con RFECV) alcanzan un *Recall* mayor que las ANN aquí evaluadas. Este comportamiento puede deberse, en parte, a la eficacia de las técnicas de reducción de dimensionalidad aplicadas en los modelos ML y a la forma distinta en que las ANN realizan internamente la ponderación y selección de características.

Tabla 6.10: Resumen de los resultados obtenidos del desarrollo de modelos ML y DL.

		Matriz de confusión			K-Fold Validación cruzada		
		Accuracy	Precision	Recall	F1 Score	Accuracy	Standard Deviation / Loss
<b>Random Forest</b> [25 estimadores, criterio de entropía]	<i>Por defecto</i>	99.29	94.40	74.27	83.16	99.30	0.02
	<i>RFFCV</i>	99.44	94.28	81.05	87.17	99.47	0.02
	<i>kbest (n=5)</i>	97.79	65.97	12.55	21.08	97.80	0.02
	<i>kbest (n=8)</i>	97.74	53.95	27.37	36.32	97.79	0.01
	<i>PCA (n=2)</i>	97.55	12.52	00.67	01.27	97.35	0.01
	<i>PCA (n=4)</i>	98.06	81.66	22.60	35.41	98.04	0.00
<b>SVM</b> [C=1, kernel=rbf]	<i>Por defecto</i>	97.23	89.32	06.95	12.83	97.79	0.01
	<i>kbest (n=5)</i>	97.11	65.25	12.29	20.73	97.80	0.01
	<i>kbest (n=8)</i>	97.05	71.19	09.30	16.46	97.79	0.01
	<i>PCA (n=2)</i>	97.61	-	0	-	97.76	0.00
	<i>PCA (n=4)</i>	97.61	-	0	-	97.76	0.00
	<i>sklearn</i>	97.84	69.03	14.98	24.66	97.83	*_-
<b>ANN</b>	<i>[(5, 5), relu, sgd]</i>	97.84	72.17	13.49	22.74	97.85	**0.0690

Nota: \* *Score()* / \*\* *Evaluate()*

#### 6.5.4. Generalización a otras topologías y algoritmos de reconfiguración

Estos resultados ponen de manifiesto la eficacia de las técnicas de ML y DL para mejorar la fiabilidad de las SG y ofrecen orientación práctica para el desarrollo futuro de sistemas inteligentes de gestión de red. Conviene aclarar que los modelos desarrollados no están pensados como soluciones a medida para una topología fija y concreta; por el contrario, se han optimizado empleando topologías aleatorias que varían de forma sistemática en número de nodos, grado de conectividad y modelo probabilístico (Waxman / Barabási-Albert), aplicando parámetros inspirados en el IEEE 123 Node Test Feeder para dotar de realismo y diversidad al conjunto de pruebas. Es cierto que el modelo no buscará el máximo rendimiento en una topología particular, sino que persigue un rendimiento medio óptimo sobre un amplio abanico de topologías, adoptando así una estrategia de compromiso que generaliza bien frente a condiciones de red heterogéneas.

En coherencia con lo anterior, sería deseable en trabajos futuros estudiar de forma más profunda la adaptación de modelos específicos a topologías particulares o la comparación frente a otros algoritmos de reconfiguración (incluyendo enfoques puramente centralizados, puramente distribuidos o híbridos). Dichas investigaciones permitirían cuantificar hasta qué punto la personalización de modelos mejora el rendimiento frente a la solución generalista aquí propuesta. En cualquier caso, este estudio inicial constituye una base sólida para análisis posteriores y demuestra la capacidad de las técnicas propuestas para potenciar el algoritmo DEN2NE, que era el objetivo principal de esta aportación.

### 6.6. Conclusiones

En este trabajo hemos propuesto un enfoque novedoso para la predicción de fallos en SG mediante la aplicación de modelos de ML y DL como apoyo al proceso de reconfiguración proactiva de la red. Partiendo del algoritmo DEN2NE, abordamos las complejidades de la redistribución de carga en SG densas y heterogéneas aplicando técnicas avanzadas de ML/DL para mejorar la toma de decisiones durante la fase de reconfiguración de la red. Mediante el uso de conjuntos de datos reales como *SustDataED* e integrando datos simulados de generación fotovoltaica obtenidos con herramientas como *PVWatts*, este trabajo demuestra la viabilidad de emplear modelos impulsados por AI para predecir fallos potenciales y aumentar la resiliencia del sistema.

La metodología propuesta identificó de forma efectiva rutas óptimas para el balance de carga minimizando errores en la distribución energética. Entre los modelos evaluados, el clasificador Random Forest optimizado con RFECV para la reducción de dimensionalidad mostró un comportamiento superior y consistente: alcanzó niveles elevados de Precision (94.28 %) y Recall (81.05 %), garantizando una detección de errores equilibrada con una tasa reducida de falsas alarmas frente a otras alternativas. Las pruebas y validaciones extensivas en entornos simulados confirman que esta configuración (RF+RFECV) mejora la robustez y adaptabilidad de la SG, gestionando de forma efectiva la incertidumbre asociada a las fuentes de energía renovables y los cambios dinámicos en las redes de distribución eléctrica.

# Capítulo 7

## BLOSTE

Tras presentar DEN2NE (Capítulo 5) y exponer una propuesta que extiende DEN2NE hacia un servicio de optimización y reconfiguración proactiva para SGs (Capítulo 6), se avanzó en una colaboración con el Departamento de Electrónica de la UAH para adaptar y versionar DEN2NE específicamente al dominio de las redes inteligentes de distribución en el marco del EI. Esta colaboración permitió un ajuste fino (*fine-tuning*) de los criterios de optimización y la incorporación de conocimiento experto en electrónica de potencia para hacer la solución más adecuada al entorno real de distribución eléctrica.

De esta colaboración nace Balanced Logical Overlay for Smart Topology Energy-routing (BLOSTE), un marco reconfigurable que aprovecha topologías malladas configurables como soporte para una distribución de energía más eficiente y resiliente. BLOSTE crea una superposición lógica dinámica sobre la infraestructura física de la red, habilitando a cada nodo para calcular múltiples rutas hacia los nodos raíz y seleccionar de forma autónoma la ruta óptima según cuatro criterios de optimización parametrizables. Este esquema facilita el equilibrio energético local, promueve la autosuficiencia entre prosumidores vecinos y reduce la dependencia del suministro central, con los beneficios esperados de menor pérdida energética, menor desequilibrio y postergación de costosas ampliaciones de infraestructura.

El capítulo describe el diseño conceptual de BLOSTE, los criterios de selección y reconfiguración, las implicaciones prácticas reales y los resultados experimentales conseguidos sobre topologías del IEEE ampliamente conocidas en el mundo de las SG. Finalmente, parte de este trabajo ha dado lugar a una publicación enviada a una revista de alto impacto (Carrascal *et al.* [182]).

### 7.1. Introducción

Como se ha podido estudiar en el Estado del Arte (Sección 2.3.1), la transición energética global está tendiendo hacia sistemas eléctricos más sostenibles, resilientes y eficientes que exigen replantear las arquitecturas de distribución de energía. El incentivo de la integración masiva de generadores renovables con la Directiva 2018/2001/UE sobre energías renovables [13], materializada en España a través del Real Decreto 244/2019 [14], y la des-

centralización de la producción han incrementado la complejidad operativa y de control de las redes de distribución, complicando tareas de planificación, protección y coordinación [104].

En respuesta a estos cambios, las topologías radiales tradicionales están dando paso a configuraciones más malladas. Las redes malladas ofrecen mayor flexibilidad operativa al habilitar múltiples trayectorias para el flujo de potencia, lo que reduce la dependencia de rutas únicas, mejora la continuidad del suministro y facilita la integración de recursos distribuidos (Distributed Energy Resources (DERs)). Sin embargo, este beneficio conlleva retos: los flujos de potencia se vuelven más dinámicos y menos predecibles, aumentando las exigencias de control, la complejidad de la protección y la necesidad de respuestas locales rápidas. Para afrontar estos retos son necesarias arquitecturas con inteligencia distribuida y capacidad de reconfiguración en tiempo real, lo que motiva la adopción de elementos como los ERs [109] y paradigmas tipo EI [108].

Una tecnología habilitadora fundamental para el EI es el ER [109]. Estos dispositivos, basados en electrónica de potencia de alto rendimiento, permiten un control programable y dinámico de los flujos en redes de distribución: regulan variables eléctricas (tensión, corriente, frecuencia) y permiten el enrutamiento selectivo de potencia entre nodos [110]. Su funcionamiento es análogo al de los routers en redes de datos y supone un cambio de paradigma en la gestión de la electricidad a nivel de distribución. La integración de ERs habilita la arquitectura conocida como PRG, en la que los ERs actúan como nodos inteligentes que interconectan activos heterogéneos (generación convencional y renovable, almacenamiento, cargas), permitiendo la optimización dinámica del flujo de energía según demanda, oferta, precios de mercado y restricciones operativas [111].

A pesar del progreso en la literatura sobre encaminamiento energético y arquitecturas para el EI, persisten limitaciones relevantes: (i) muchas soluciones son estáticas o centralizadas, lo que restringe su escalabilidad y adaptabilidad; (ii) hay escasez de enfoques que exploten topologías multi-raíz o malladas con decisiones autónomas a nivel de nodo; (iii) falta consenso sobre criterios de selección de rutas que equilibren pérdidas, balance local y resiliencia; y (iv) la integración de tecnologías emergentes (p. ej., blockchain para validación transaccional o AI para encaminamiento predictivo) está aún poco explorada. Estos vacíos subrayan la necesidad de marcos de encaminamiento flexibles, distribuidos y en tiempo real capaces de operar en entornos heterogéneos y dinámicos.

En este contexto presentamos BLOSTE, un marco de encaminamiento dinámico diseñado para abordar los retos de las redes de distribución de próxima generación. BLOSTE construye una superposición lógica sobre la infraestructura física mallada mediante procedimientos de etiquetado jerárquico que permiten a cada nodo aprender múltiples rutas hacia uno o varios nodos raíz. A diferencia de enfoques centralizados, BLOSTE facilita la toma de decisiones autónoma en el propio nodo basada en información local y en criterios de encaminamiento configurables. Su diseño prioriza la resiliencia y la eficiencia: los nodos pueden seleccionar rutas según objetivos operativos distintos (minimizar saltos, pérdidas, o equilibrar potencia local), y el sistema incorpora mecanismos de refresco y estrategias de

## 7.2 ENFOQUE JERÁRQUICO DE REDISTRIBUCIÓN DE ENERGÍA BASADO EN ÁRBOLES

---

respaldo para adaptarse a degradaciones de enlace o cambios topológicos. Las principales contribuciones de este capítulo son:

- **Formación dinámica de topologías lógicas.** Método para generar, desde uno o varios nodos raíz, topologías lógicas jerarquizadas sobre una red física mallada mediante etiquetado, facilitando el aprendizaje de múltiples rutas por nodo.
- **Criterios de encaminamiento configurables.** Implementación de cuatro criterios ajustables (mínimos saltos, minimización de pérdidas, balance local a cero y balance local ponderado por pérdidas) que cada nodo puede emplear de forma autónoma para seleccionar su ruta activa.
- **Resiliencia operativa.** Mecanismos periódicos de refresco y políticas de commutación que permiten recomputar rutas y mantener servicio ante degradaciones o fallos, sin depender exclusivamente de control central.
- **Algoritmo eficiente de balance de potencia.** Procedimiento iterativo de reparto de carga que converge hacia estados globalmente equilibrados minimizando pérdidas de transmisión.
- **Evaluación exhaustiva.** Validación experimental sobre las topologías IEEE 123-node y 34-node test feeder, mediante un amplio conjunto de simulaciones (escenarios ideales, con pérdidas y con pérdidas+capacidad limitada), que demuestra la capacidad de BLOSTE para mejorar tiempos de convergencia, balance energético y adaptabilidad, destacando especialmente el criterio de balance local a cero.

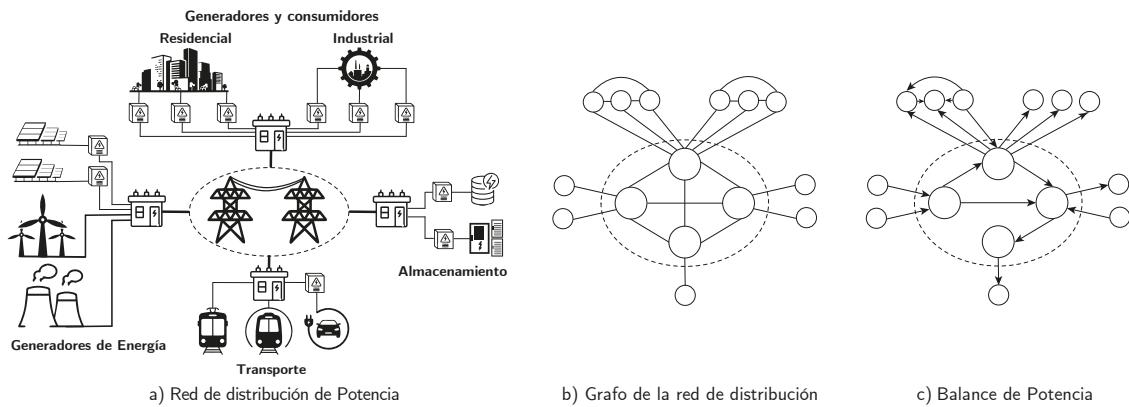
En las secciones siguientes se presenta con detalle la arquitectura de BLOSTE, los algoritmos de etiquetado y selección de rutas, el procedimiento de balance energético y los resultados de la evaluación, incluyendo un análisis computacional frente a estrategias alternativas y una discusión sobre su aplicabilidad práctica en PRGs.

## 7.2. Enfoque jerárquico de redistribución de energía basado en árboles

Como se ha destacado previamente, las redes de distribución eléctrica tienden hacia un modelo más hiperconectado o en malla para mejorar su rendimiento y eficiencia y aprovechar mejor la generación distribuida asociada a las energías renovables [183]. Este cambio hacia arquitecturas malladas ha sido respaldado por diversos estudios que subrayan las ventajas de la operación en malla para facilitar la integración de generadores renovables variables [184, 185]. Por ejemplo, algunos autores destacan ya el potencial de los sistemas de distribución eléctrica mallados para acoger generación renovable variable a gran escala [185]. Asimismo, se han propuesto arquitecturas en malla basadas en LoRa para la automatización de redes rurales de distribución eléctrica, demostrando la aplicabilidad de las estructuras malladas más allá de entornos urbanos [186]. La motivación detrás de la transición hacia una red eléctrica hiperconectada radica en su capacidad para gestionar de forma eficaz la generación descentralizada procedente de fuentes renovables, optimizar

la distribución de la energía y aumentar la resiliencia del sistema en su conjunto [183]. Modelar la red de distribución como una estructura mallada se alinea con este propósito, ya que permite flujos de energía más dinámicos y eficientes [184]. Al conceptualizar la red mallada como un grafo, es posible visualizar sus interconexiones y los distintos caminos de flujo, lo que facilita la comprensión de su dinámica operativa [187].

Para ilustrar este proceso, introducimos la Figura 7.1, que muestra cómo puede representarse una red de distribución de energía hiperconectada como un grafo. El proceso comienza con la caracterización de la red de distribución: identificación de fuentes de generación, consumidores y las líneas de transmisión que los conectan (Figura 7.1a). Esta información se emplea para construir un grafo en el que generadores y consumidores se representan como nodos y las líneas eléctricas como enlaces (Figura 7.1b). Además, la naturaleza heterogénea de los agentes, generación, industria, transporte, residencial, almacenamiento, etc., implica perfiles de consumo variados que cambian dinámicamente según las condiciones ambientales (clima, horarios laborales, festivos, etc.), dando lugar a una red de potencia con múltiples fuentes y consumos variables en el tiempo, donde es necesario optimizar el equilibrio de las cargas (Figura 7.1c).

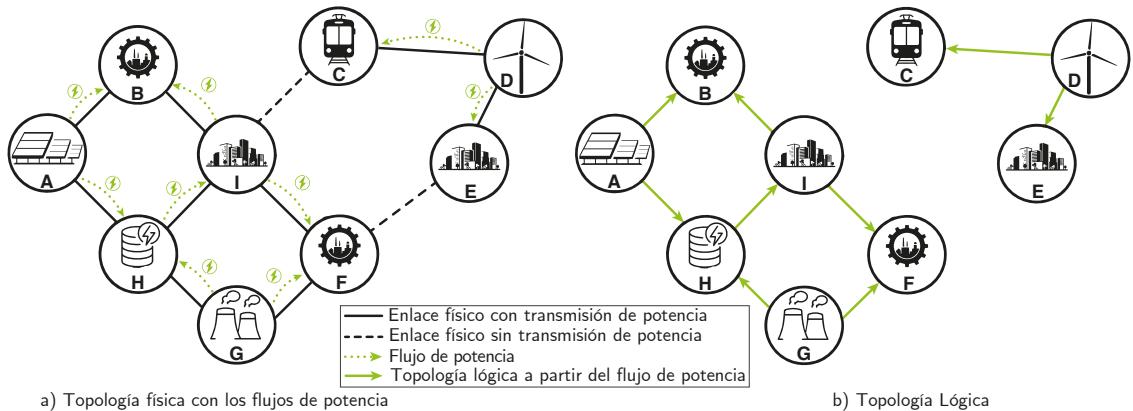


**Figura 7.1:** Ejemplo de distribución de potencia.

### 7.2.1. Construcción de la topología lógica

Como hemos visto a lo largo de la Tesis, la modelización de la topología lógica puede expresarse en forma matemática: dado un grafo  $G = (\mathcal{N}, \mathcal{L})$ , que representa la topología completa, donde  $\mathcal{N}$  es un conjunto de  $N$  nodos y  $\mathcal{L}$  un conjunto de  $L$  enlaces tal que  $\mathcal{L} \subseteq \{\{i, j\} | i, j \in \mathcal{N} \forall i \neq j\}$ . En función del balance de potencia de la red de distribución eléctrica en un instante dado, puede definirse un subgrafo  $G'$  a partir de  $G$  que toma parte de los elementos dependiendo de la oferta y la demanda de energía en los distintos nodos, denotado como  $G' = (\mathcal{N}', \mathcal{L}')$ , donde  $\mathcal{N}' \subseteq \mathcal{N}$  y  $\mathcal{L}' \subseteq \mathcal{L}$ . De esta manera, se pueden obtener diferentes topologías lógicas en distintos momentos en función de las condiciones energéticas, pudiendo optimizar de esta forma la topología final, prescindiendo de enlaces innecesarios para la distribución eléctrica.

## 7.2 ENFOQUE JERÁRQUICO DE REDISTRIBUCIÓN DE ENERGÍA BASADO EN ÁRBOLES



**Figura 7.2:** Establecimiento de la topología lógica.

La Figura 7.2 proporciona un ejemplo de cómo puede derivarse una topología lógica a partir de un grafo físico. Se divide en dos secciones: la sección izquierda (Figura 7.2a) ilustra la topología de la red de distribución física, grafo  $G$ , mientras que la sección derecha (Figura 7.2b) muestra la topología lógica  $G'$  derivada del grafo físico  $G$ , considerando las demandas y excedentes de potencia de la red de distribución. Más específicamente, este ejemplo contiene una topología con 9 nodos identificados de manera única, etiquetados de la  $A$  a la  $I$ , los cuales están conectados mediante líneas de transmisión eléctrica. Las líneas de transmisión se representan como líneas continuas o discontinuas, indicando si están transportando energía de forma activa o si se encuentran inactivas, respectivamente. Por ejemplo, el enlace que conecta los nodos  $I$  y  $C$  no transporta energía, por lo que se representa con una línea discontinua, mientras que el enlace entre  $C$  y  $D$  sí transporta energía y, por tanto, se representa con una línea continua.

Los nodos de la red se categorizan según su función como nodos demandantes, con excedentes o mixtos, mientras que el flujo de energía entre nodos se representa mediante una flecha verde punteada. Por ejemplo, el nodo  $A$  representa un generador solar, por lo que es un nodo con excedentes que suministra energía a los nodos  $B$  (un nodo demandante), un parque industrial que demanda energía, y  $H$ , un nodo de almacenamiento que tanto demanda como suministra energía a otros nodos (un nodo mixto). El resto de los nodos contribuyen de manera análoga al balance energético de la red según su rol.

Toda la información anteriormente descrita se encuentra en la Figura 7.2a, a partir de la cual pueden derivarse múltiples topologías lógicas dependiendo del balance energético de la red y de las estrategias de distribución adoptadas. Las dos topologías lógicas resultantes de este ejemplo se ilustran en la Figura 7.2b. Éstas representan dos islas energéticas aisladas que optimizan el balance de energía, garantizando que la demanda sea cubierta y minimizando las pérdidas debidas a inserción y reflexión en enlaces inactivos de la red de distribución. Como se explicó previamente, esta topología lógica constituye la representación de la red en un instante concreto. En consecuencia, dicha topología lógica no es estática, es decir, cambia en el tiempo en función de las ofertas y demandadas de la red.

Por lo tanto, existe una clara necesidad de un mecanismo específico que permita calcular y gestionar topologías lógicas de manera dinámica, con el fin de habilitar una distribución eficiente de la energía a lo largo de la red bajo condiciones variables. Para abordar esta necesidad, en este trabajo se introduce un mecanismo que explora y caracteriza sistemáticamente la red de distribución eléctrica mediante un procedimiento de etiquetado jerárquico basándose en las lecciones aprendidas de DEN2NE. Este enfoque sienta las bases para la construcción de topologías lógicas adaptadas a objetivos de optimización concretos en SG, tales como minimizar las pérdidas de energía o maximizar el balance energético. Los detalles del proceso de exploración y etiquetado, que constituyen el núcleo de nuestra propuesta, se describen en las siguientes secciones.

#### 7.2.1.1. Procedimiento de etiquetado jerárquico basado en un árbol

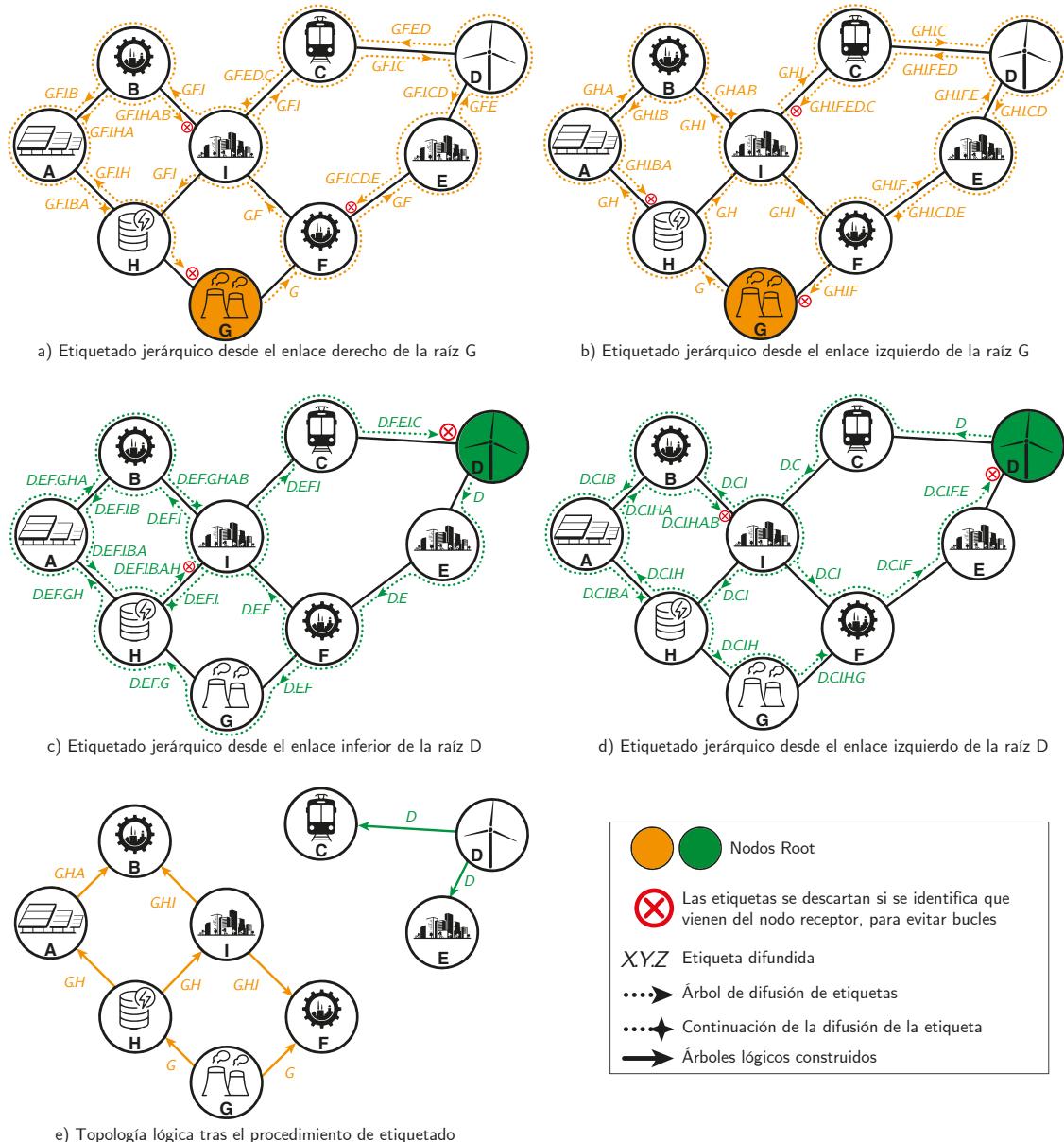
Esta sección describe en detalle un procedimiento de distribución de etiquetas para construir un árbol jerárquico a partir de un nodo seleccionado de una topología, denominado *nodo raíz*. En teoría de grafos (Ver Sección 2.2.1.1), un árbol es un tipo especial de grafo conexo que es acíclico. Esto significa que no contiene ciclos y que cada par de nodos en el grafo está conectado por un único camino. Además, el nodo raíz es el nodo superior del cual se derivan el resto. Cada nodo en el árbol tiene exactamente un nodo padre y cero o múltiples nodos hijos, mientras que el nodo raíz no tiene padre y solo posee nodos hijos. Esto establece una jerarquía que estructura el árbol en múltiples niveles.

Para construir una topología lógica a partir de una topología física, nuestra propuesta establece un procedimiento de etiquetado jerárquico desde un nodo (el nodo raíz) con el objetivo de establecer relaciones estructurales lógicas entre los nodos de la topología que permitan satisfacer la demanda y la producción de energía. Este nodo raíz, desde el cual comienza el procedimiento de etiquetado jerárquico, puede ser un punto de generación, transformación o distribución de energía, entre otros, dependiendo de las necesidades de la red. Cada nodo de la red debe contar con un identificador único (ID), ya que este ID se difunde de manera controlada a través de la red para recopilar información y establecer la jerarquía de la topología. Este ID puede ser un número, una letra o una cadena alfanumérica, siempre que sea único para cada nodo.

Para explicar mejor el procedimiento, el proceso de difusión de etiquetas se ha ilustrado en la Figura 7.3 utilizando la topología de ejemplo de la Figura 7.2, correspondiente a una red de distribución eléctrica de 9 nodos. El ID de cada nodo corresponde a su etiqueta ( $A, B, \dots, I$ ). En este ejemplo, el nodo  $G$  fue seleccionado como nodo raíz, al ser un punto de generación de energía. El procedimiento comienza difundiendo la etiqueta  $G$  desde el nodo  $G$  a través de todos sus enlaces. La Figura 7.3a ilustra el procedimiento de difusión de etiquetas desde el enlace derecho del nodo  $G$ , mientras que la Figura 7.3b muestra el procedimiento desde el enlace izquierdo del mismo nodo. Es en este momento cuando empiezan a crearse las relaciones jerárquicas en la topología. Centrándonos en la Figura 7.3a, la etiqueta  $G$  llega al nodo  $F$ . Este nodo ocupa el segundo nivel en la jerarquía, siendo el nodo  $G$  su nodo padre. A continuación, el nodo  $F$  crea una nueva etiqueta resultante

## 7.2 ENFOQUE JERÁRQUICO DE REDISTRIBUCIÓN DE ENERGÍA BASADO EN ÁRBOLES

de la concatenación de la etiqueta recibida más su propio ID ( $G.F$ ). El nodo  $F$  difunde esta nueva etiqueta a través de todos sus enlaces, llegando a los nodos  $E$  e  $I$ , los cuales repiten el proceso (definiendo su nivel en la jerarquía, generando una nueva etiqueta con su propio ID y difundiéndola).

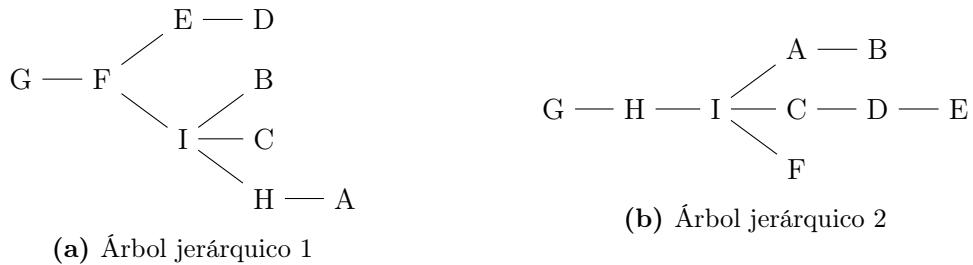


**Figura 7.3:** Procedimiento de etiquetado jerárquico.

El proceso de difusión de etiquetas finaliza cuando toda la red ha sido explorada, asegurando que cada nodo recibe al menos una etiqueta jerárquica. En la práctica, cada etiqueta representa una ruta hacia el nodo raíz. Para evitar bucles durante la fase de

etiquetado jerárquico, el algoritmo impide asignar etiquetas que contengan identificadores previamente recorridos. Esta precaución es sencilla de aplicar, ya que las etiquetas se generan concatenando los IDs de los nodos visitados, prohibiendo así la inclusión de IDs repetidos, lo cual indicaría un bucle. Por ejemplo, en la Figura 7.3a, el nodo *I* descarta y finaliza el proceso de difusión de la etiqueta *G.F.I.H.A.B*, dado que previamente había creado y difundido la etiqueta *G.F.I*, que comparte el prefijo *G.F.I*. Este mecanismo evita la formación de bucles y garantiza la construcción de un árbol jerárquico desde el nodo raíz. Para simplificar la representación de la figura, no se han incluido todos los procesos de difusión de etiquetas, los cuales se han señalado en el diagrama con una estrella de cuatro puntas.

El proceso de difusión de etiquetas desde el enlace izquierdo del nodo raíz *G*, mostrado en la Figura 7.3b, se desarrolla de manera similar. Este proceso genera otro subconjunto de rutas que forman un nuevo árbol complementario al anterior. El resultado final se resume en la Figura 7.4, que muestra los dos árboles jerárquicos derivados del nodo raíz *G*, seleccionando la ruta más rápida. En este sentido, cada nodo almacena dos etiquetas de diferentes árboles derivados del mismo nodo raíz. De este modo, cada nodo dispone de dos rutas alternativas para alcanzar el nodo raíz, que pueden seleccionarse en una fase posterior para construir la topología lógica de acuerdo con distintos criterios generales, como disyunción de rutas o número mínimo de saltos [45, 70], o con criterios específicos de redes de distribución eléctrica, como el balance de potencia [188] o las pérdidas energéticas [189].

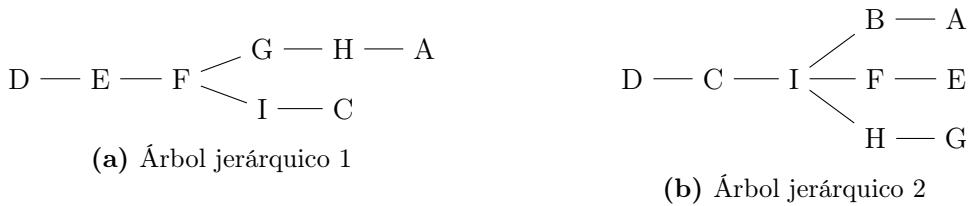


**Figura 7.4:** Árboles jerárquicos tras el proceso de difusión del etiquetado desde el nodo raíz *G*

### 7.2.1.2. Procedimiento de etiquetado jerárquico basado en múltiples árboles

Las microrredes típicamente cuentan con múltiples puntos de acceso a la red de distribución eléctrica. Estos puntos de acceso permiten la reconfiguración de la red mediante el control de interruptores para optimizar métricas de operación, manteniendo al mismo tiempo una operación segura y estable [190, 191]. Es por ello, que en este escenario puede ser visto como un grafo con varios nodos que conectan la microrred a la red de distribución eléctrica, designados como raíces. Cada raíz debería tener uno o más árboles jerárquicos para distribuir la energía a los nodos según la oferta o demanda de la microrred. Por lo tanto, es posible construir múltiples árboles jerárquicos a partir de varios nodos raíz.

El mecanismo presentado en este trabajo permite ejecutar múltiples procedimientos de difusión de etiquetas desde varios nodos raíz simultáneamente, proporcionando múltiples caracterizaciones de la topología de la red desde diferentes nodos raíz al mismo tiempo. Las Figuras 7.3c y 7.3d muestran el procedimiento de difusión de etiquetas desde un nuevo nodo raíz  $D$ . Este proceso podría ejecutarse al mismo tiempo que el procedimiento de difusión de etiquetas desde el nodo  $G$  descrito anteriormente. Ambos procesos son independientes y no interfieren entre sí. Dado que la etiqueta raíz difundida desde cada nodo raíz es diferente, los procedimientos derivados de cada uno son independientes y pueden coexistir en el tiempo. A partir de este nuevo nodo raíz, se obtienen dos árboles jerárquicos adicionales, seleccionando la ruta más rápida, como se muestra en la Figura 7.5.



**Figura 7.5:** Árboles jerárquicos tras el proceso de difusión del etiquetado desde el nodo raíz  $D$

Una vez que todas las etiquetas de ambos nodos raíz han sido difundidas, se dispone de varias caracterizaciones de la topología física. La topología lógica puede construirse entonces utilizando uno o más de los criterios definidos en la Sección 7.2.2. En el ejemplo ilustrado en la Figura 7.3e, se crean dos “islas” de energía, una que involucra los nodos  $D, C, E$  y otra que involucra al resto de los nodos. Sin embargo, dependiendo de los criterios de optimización seleccionados, la topología resultante podría ser diferente.

### 7.2.2. Criterios para construir la topología lógica

Como se introdujo anteriormente, una vez concluido el procedimiento de etiquetado, todos los nodos pueden utilizar cualquiera de las etiquetas asignadas para alcanzar el nodo raíz, ya que las etiquetas delinean implícitamente la secuencia de nodos que deben recorrerse para llegar hasta la raíz. Cuando un nodo está asociado únicamente a una etiqueta, existe un único camino hacia la raíz. Sin embargo, si existen múltiples etiquetas, se deben emplear criterios específicos para seleccionar la ruta más óptima hacia la raíz. En consecuencia, la segunda fase del algoritmo determina la distribución de potencia aplicando criterios predefinidos de acuerdo con el orden establecido por las etiquetas o los identificadores jerárquicos<sup>1</sup> configurados durante la primera fase.

Cada etiqueta  $ID$  se define formalmente como una secuencia ordenada de  $k + 1$  nodos que representan un camino válido desde un nodo dado  $n_k \in \mathcal{N}'$  hasta el nodo raíz  $n_0 \in \mathcal{N}'$ , atravesando la topología lógica  $G' = (\mathcal{N}', \mathcal{L}')$ . Es decir,

<sup>1</sup>Nótese que, a partir de este punto, ID representa identificadores jerárquicos o etiquetas, es decir, aquellos derivados del proceso de etiquetado previo (no los IDs únicos de los nodos).

$$\begin{aligned} ID &= \{n_0, n_1, \dots, n_{k-1}, n_k\}, \\ \text{con } (n_{l-1}, n_l) &\in \mathcal{L}', \forall l \in \{1, \dots, k\}. \end{aligned} \tag{7.1}$$

En esta definición, el índice 0 siempre corresponde al nodo raíz, y el índice  $k$  al nodo de origen (es decir, el nodo desde el cual se evalúa el ID). Esta notación facilita la evaluación consistente de cada ruta y la aplicación de criterios que dependen del número de saltos o de las características de los enlaces recorridos.

Como referencia, nuestro algoritmo ofrece cuatro criterios distintos para seleccionar la ruta óptima para la distribución de energía hacia la raíz. Cada criterio considera diversos aspectos topológicos y características de despliegue para calcular un coste para cada ruta disponible, denotado como el coste asociado a cada identificador jerárquico ( $Cost\_ID$ ).

Posteriormente, el algoritmo identifica la ruta óptima, la etiqueta activa o el identificador jerárquico activo ( $ID\_activa$ ) en función del coste mínimo, tal como se especifica en la Ecuación 5.1. Los cuatro criterios se describen en las secciones siguientes; no obstante, es posible definir criterios adicionales e incorporarlos al algoritmo, dado que esta segunda fase únicamente requiere la definición de una función de coste para cada criterio, sin estar limitada a una implementación específica. Por lo tanto, el criterio para la selección de  $ID_{active}$  se mantiene constante para todos los criterios presentados; lo que varía es el método utilizado para calcular el coste asociado a cada ruta según el criterio aplicado.

### 7.2.2.1. Número de saltos

En este primer criterio (*Hops*) se considera el número de saltos necesarios para alcanzar el nodo raíz. Este criterio coincide con el descrito en DEN2NE (ver Sección 5.2.3), por lo que aquí se presenta de manera resumida utilizando la notación adoptada en esta propuesta. Cada identificador jerárquico se define como una secuencia  $ID = \{n_0, n_1, \dots, n_{k-1}, n_k\}$ , donde  $n_0$  corresponde al nodo raíz y  $n_k$  al nodo de origen. El número de saltos se determina por la cantidad de enlaces recorridos a lo largo del camino desde el nodo de origen hasta la raíz, lo que equivale al número de nodos en la secuencia menos uno. Por lo tanto, se selecciona la etiqueta que minimice el número de saltos hacia el nodo raíz. El coste asociado a este criterio puede expresarse de la siguiente manera:

$$Cost_{ID} = \text{length}(ID) - 1 = k \tag{7.2}$$

donde  $\text{length}(ID)$  indica el número de nodos en la secuencia (es decir,  $k + 1$ ), y el coste resultante  $k$  representa el número total de saltos desde el nodo de origen  $n_k$  hasta el nodo raíz  $n_0$ .

### 7.2.2.2. Bajas pérdidas de enlace

Este segundo criterio (*Low-Link Losses*) establece una relación entre las características físicas de las líneas de transmisión y la distancia lógica al nodo raíz. Su objetivo es minimizar las pérdidas totales de propagación que afectan a la potencia incidente en

cada línea, considerando también el número de saltos a lo largo del camino. La lógica detrás de este enfoque consiste en optimizar la distribución de energía favoreciendo tanto rutas físicamente eficientes como aquellas que, desde un punto de vista lógico, acercan la carga al nodo raíz, promoviendo así la agregación hacia el sumidero central. En las redes inteligentes, las pérdidas ( $L_{n_{l-1}, n_l}$ ) se calculan utilizando modelos de pérdida de líneas de transmisión que incorporan parámetros intrínsecos como el voltaje de operación ( $V_d$ ), la distancia física del enlace ( $d_{n_{l-1}, n_l}$ ), el coeficiente de reflexión ( $\alpha_{R_{n_{l-1}, n_l}}$ ) y la potencia incidente en la línea ( $P_{in}$ ), tal como se expresa en la Ecuación 7.3. Es evidente que la distancia física constituye un factor clave que influye en el coste total; no obstante, otros parámetros intrínsecos de la línea de transmisión, como los coeficientes de reflexión y el voltaje de operación, también desempeñan un papel relevante.

$$L_{n_{l-1}, n_l} = \frac{\alpha_{R_{n_{l-1}, n_l}} \times d_{n_{l-1}, n_l}}{V_d^2} \times P_{in}^2 \quad (7.3)$$

Adicionalmente, se considera la distancia lógica al nodo raíz, incentivando, en cierta medida, que la inercia de la carga se dirija hacia el sumidero principal de la topología, es decir, el nodo raíz. Al analizar la Ecuación 7.4, se observa que se introducen dos parámetros de configuración,  $\alpha$  y  $\beta$ , para regular la importancia relativa de las pérdidas de propagación dentro de la función de coste, así como de la distancia lógica al nodo raíz. Por defecto, ambos parámetros se establecen en 0.5, garantizando un equilibrio igualitario entre estos dos factores. No obstante, es posible explorar una optimización adicional de estos valores dependiendo del tipo específico y los requisitos de las redes en las que se implemente este criterio. El número de saltos a lo largo del camino es  $k$ , y las pérdidas de propagación se calculan para cada enlace  $(n_{l-1}, n_l) \in \mathcal{L}'$  utilizando la expresión 7.3.

$$Cost_{ID} = \alpha \cdot k + \beta \cdot \sum_{l=k}^1 \left( \frac{\alpha_{R_{n_{l-1}, n_l}} \cdot d_{n_{l-1}, n_l}}{V_d^2} \cdot P_{in}^2 \right) \quad (7.4)$$

### 7.2.2.3. Balance de potencia local a cero

Este tercer criterio (*Power2Zero*) se centra en optimizar el equilibrio de potencia local entre nodos vecinos, con el objetivo de minimizar o incluso eliminar el intercambio neto de potencia entre ellos. Este concepto resulta fundamental en el contexto de las DER, ya que fomenta la autosuficiencia local: los nodos vecinos procuran cubrir colectivamente su demanda sin depender del nodo raíz ni de la red principal. Al lograr esta compensación local, el sistema reduce significativamente la necesidad de transportar energía a través de la red primaria, lo que a su vez disminuye las pérdidas de transmisión y alivia la presión sobre la infraestructura en su conjunto. Este enfoque no solo mejora la eficiencia energética, sino que también se alinea con el objetivo estratégico de la generación distribuida: satisfacer la demanda local mediante recursos locales. En consecuencia, contribuye a diferir o incluso evitar costosas expansiones de la red y fortalece la sostenibilidad y escalabilidad del sistema eléctrico.

Para abordar este criterio, tal como se ilustra en la Ecuación 7.5, cada nodo evalúa los estados de potencia actuales de sus vecinos y determina qué configuración aproxima más el balance a cero. De manera análoga al criterio anterior, se introducen los parámetros  $\alpha$  y  $\beta$  (ambos fijados inicialmente en 0.5), junto con el componente de distancia lógica al nodo raíz de la topología. Este planteamiento permite ponderar de manera configurable cada componente de la función de coste, posibilitando tanto la optimización del equilibrio de potencia en mínimos locales como la priorización de la distancia lógica hacia el nodo de acceso a la red. De este modo, se fomenta un flujo de potencia inherente hacia el punto de conexión de la red de distribución.

$$Cost_{ID} = \alpha \cdot k + \beta(|P_{k-1} + P_k|) \quad (7.5)$$

Al igual que en los criterios anteriores, esta función de coste se aplica a todas las secuencias  $ID$  disponibles asociadas al nodo de origen ( $n_k$ ). Ello permite identificar al vecino inmediato ( $n_{k-1}$ ) cuyo equilibrio de potencia se vea más favorecido, es decir, más próximo a cero, incorporando al mismo tiempo un componente de ponderación configurable que orienta gradualmente las cargas hacia el nodo raíz de la topología.

#### 7.2.2.4. Balance de potencia local a cero con pérdidas

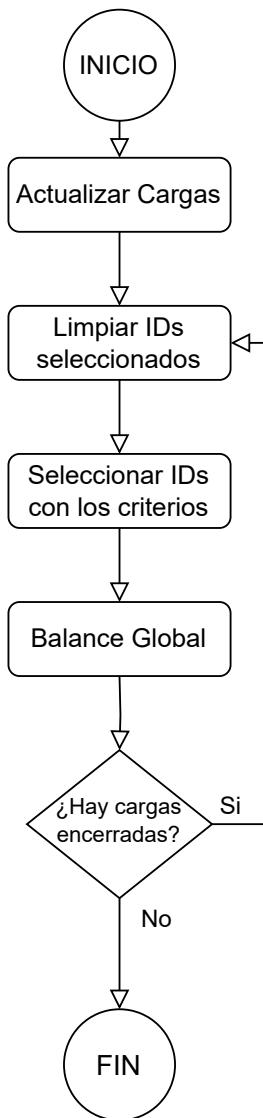
El cuarto criterio (*Power2Zero + Losses*) busca integrar los parámetros evaluados previamente, a saber: el balance de potencia local y las pérdidas derivadas del proceso de compensación. Tal como se ilustra en la Ecuación 7.6, este criterio evalúa cada secuencia jerárquica  $ID = \{n_0, n_1, \dots, n_k\}$ , considerando los estados de potencia de los dos últimos nodos de la secuencia,  $n_k$  (nodo de origen) y  $n_{k-1}$  (vecino a un salto), e incorporando además la pérdida de transmisión  $L_{n_{k-1}n_k}$  asociada al último enlace. El coste resultante se define como:

$$Cost_{ID} = |P_{n_k} + P_{n_{k-1}} - L_{n_{k-1}n_k}| \quad (7.6)$$

Esta formulación combina de manera efectiva los componentes clave de los criterios presentados en las Secciones 7.2.2.3 y 7.2.2.2, al evaluar conjuntamente la compensación local y la eficiencia en la transmisión sobre el tramo final de la ruta. De este modo, se fomenta un intercambio de potencia local que no solo favorece la autosuficiencia, sino que también incorpora la conciencia de pérdidas, mejorando la robustez y la eficiencia global del sistema.

#### 7.2.3. Balanceo utilizando la topología lógica

Una vez que las etiquetas han sido propagadas a lo largo de la topología y se ha aplicado el criterio de selección, cada nodo obtiene un  $ID_{activa}$ , lo que permite establecer la topología lógica y dar inicio al procedimiento de balanceo global de potencia. Este proceso posibilita la redistribución de las cargas, comenzando en los nodos más periféricos de la red y avanzando progresivamente hacia el nodo raíz, mientras se favorece de manera simultánea el balanceo localizado entre nodos adyacentes.



**Figura 7.6:** Balanceo de potencia mediante topología lógica: un enfoque iterativo.

Se ha concebido un enfoque iterativo para llevar a cabo un balanceo sistemático de las cargas de potencia a lo largo de la topología, tal como se ilustra en la Figura 7.6. En cada iteración, tras la actualización de la configuración de cargas, cada nodo debe determinar una nueva  $ID_{activa}$ , desactivando el identificador previamente asignado. Esta reevaluación resulta fundamental para identificar dinámicamente la ruta óptima en función de la distribución de carga vigente. Sin embargo, dado el carácter estático de la topología física, no es necesario realizar una reexploración de las rutas.

Una vez que las  $ID_{activa}$  han sido reasignadas a nivel de nodo y se ha activado el identificador óptimo, el proceso de balanceo de potencia prosigue conforme a la topología lógica.

De esta manera, se garantiza un equilibrio local de potencia, canalizando cualquier demanda o excedente residual hacia el nodo raíz. El proceso iterativo continúa hasta que no exista demanda ni excedente dentro de la topología (es decir, hasta que no queden cargas internas por compensar), lo que señala la resolución completa de los desbalances, con la excepción del nodo raíz, que refleja el balance global correspondiente a cada iteración.

En cada iteración del proceso de balanceo global se registran dos métricas fundamentales: el balance total de potencia de la topología (*total\_balance*) y la magnitud absoluta del flujo de potencia en el sistema (*abs\_flux*). La suma acumulada de estas métricas a lo largo de todas las iteraciones ofrece una medida integral tanto del flujo absoluto de potencia como del balance total alcanzado en la topología. El procedimiento de balanceo global de potencia se repite cada vez que se detecta una nueva configuración de cargas, es decir, en cada intervalo de actualización (*delta\_time*) de las cargas. Los resultados obtenidos y sus implicaciones se presentan en la Sección 7.3. Conviene destacar que este enfoque iterativo se ha empleado únicamente con fines de validación conceptual de la propuesta. No obstante, es previsible que en implementaciones prácticas pueda adoptar variantes, en función del contexto específico de despliegue, como se analizará en la Sección 7.2.5, dedicada a las implicaciones prácticas.

#### 7.2.4. Resiliencia y tasa de refresco

La construcción de la topología lógica basada en etiquetas proporciona de manera inherente resiliencia frente a eventos disruptivos o fallos en la red. Esta resiliencia surge de la disponibilidad de múltiples etiquetas por nodo, siempre que exista más de un camino hacia el nodo raíz, lo que permite la reconfiguración dinámica de las rutas activas en respuesta a degradaciones de enlace, pérdidas de conectividad o fallos parciales del sistema. En este contexto, cuando un nodo detecta la pérdida de un enlace con su nodo ascendente en la topología lógica, o cuando los parámetros de calidad del enlace (como las pérdidas de propagación o el balance de potencia) se degradan por debajo de umbrales predefinidos, puede activar una reevaluación de sus etiquetas alternativas. Al reaplicar los criterios de selección detallados en la Sección 7.2.2, el nodo puede escoger una nueva *ID<sub>activa</sub>* que represente un camino más adecuado hacia el nodo raíz bajo las condiciones actuales de la red. Este mecanismo automático de conmutación de rutas permite una adaptabilidad continua sin necesidad de rediseñar o redistribuir completamente la topología lógica, preservando así la continuidad del servicio y la estabilidad de la red.

Adicionalmente, con el fin de garantizar un comportamiento de red óptimo y actualizado, se ha definido una tasa de refresco periódica para evaluar, a intervalos regulares, la validez de la ruta activa seleccionada (*ID<sub>activa</sub>*). Esta tasa de refresco está parametrizada mediante un temporizador *T<sub>refresh</sub>*, que desencadena la reevaluación de los costes asociados a todas las etiquetas disponibles. En función de la dinámica del sistema (por ejemplo, variaciones en la demanda de potencia o la aparición de nuevas fuentes), este temporizador puede configurarse con valores más agresivos (para entornos altamente dinámicos) o más conservadores (para escenarios estables), equilibrando de este modo la capacidad de respuesta del sistema con el coste computacional del recálculo de rutas.

Cabe señalar que este mecanismo de refresco puede coexistir con un esquema reactivo, en el cual eventos externos, como la desconexión de nodos o la aparición de sobrecargas, también pueden activar una reevaluación inmediata de rutas sin esperar a la expiración del temporizador  $T_{refresh}$ . En conjunto, estos dos mecanismos, tanto proactivo como reactivo, garantizan una operación robusta, eficiente y adaptable del sistema tanto ante condiciones previstas como imprevistas.

### 7.2.5. Implicaciones prácticas

La implementación de la arquitectura propuesta, basada en el etiquetado jerárquico y en topologías lógicas multi-raíz, presenta implicaciones directas y significativas para su despliegue en sistemas eléctricos reales. Para materializar estas técnicas en entornos físicos, resulta esencial la integración de ERs. Estos dispositivos constituyen un pilar fundamental dentro del paradigma de la EI, al permitir no solo la conversión y el intercambio de energía, sino también el encaminamiento inteligente y la toma de decisiones autónoma en cada nodo de la red [192]. Los ERs deben ser capaces de ejecutar algoritmos distribuidos de encaminamiento energético, interpretar etiquetas jerárquicas y evaluar múltiples trayectorias en función de criterios dinámicos como las pérdidas de transmisión, el balance de potencia local o la distancia topológica.

En un sistema real, la presencia de múltiples nodos raíz, tales como puntos de interconexión con la red principal o centros de generación renovable, exigiría la construcción simultánea de diversas superposiciones lógicas. Esta capacidad permitiría a los servicios de operación seleccionar en tiempo real la ruta más adecuada para satisfacer la demanda o evacuar la generación, teniendo en cuenta el estado actual y las condiciones previstas de la red. De este modo, se logra una mayor flexibilidad operativa, ya que el sistema puede adaptarse dinámicamente a cambios en la carga, la generación, la topología o la aparición de fallos sin requerir un control centralizado.

Asimismo, la posibilidad de operar sobre múltiples árboles mediante criterios configurables ofrece una ventaja decisiva en términos de eficiencia energética y resiliencia. Permite al sistema minimizar las pérdidas, aliviar la congestión de los enlaces y mejorar la utilización de los recursos energéticos locales. La lógica de decisión distribuida propuesta en este trabajo es coherente con las tendencias actuales en eficiencia energética, que buscan maximizar el uso de la energía local, reducir los intercambios con la red principal y fomentar la autosuficiencia energética entre nodos vecinos.

En última instancia, el despliegue práctico de esta arquitectura requiere un ecosistema tecnológico alineado con los principios de la EI, en el cual los ERs funcionen no solo como interfaces físicas de los componentes de generación, almacenamiento y consumo, sino también como agentes inteligentes capaces de gestionar de manera autónoma los flujos energéticos. Tal como se enfatiza en [193], este enfoque representa un paso fundamental hacia la transición desde redes eléctricas tradicionales hacia infraestructuras energéticas distribuidas, adaptativas y sostenibles, en consonancia con los objetivos de largo plazo del

paradigma de la EI. En última instancia, será responsabilidad del operador de red que implemente dicho paradigma determinar cómo configurar los criterios de selección, cuáles aplicar en cada nodo y cuáles priorizar en distintos momentos del día, de acuerdo con las necesidades específicas de la red. Además, el operador deberá garantizar la coherencia entre los criterios aplicados en todo el sistema para evitar conflictos y asegurar una toma de decisiones consistente.

### 7.3. Evaluación

Para evaluar nuestro algoritmo, en una primera instancia se empleó la topología de referencia IEEE 123-Node Test Feeder. No obstante, tras el proceso de revisión por pares, se consideró necesario ampliar el alcance de la evaluación incorporando la topología IEEE 34-bus, así como realizar un análisis de la complejidad computacional asociado a las fases de etiquetado jerárquico y balanceo iterativo. Este último análisis resulta fundamental para comprender cómo escalan los bloques principales del algoritmo bajo diferentes tamaños de topologías. Por tanto, las evaluaciones desarrolladas en este trabajo se estructuran de la siguiente manera:

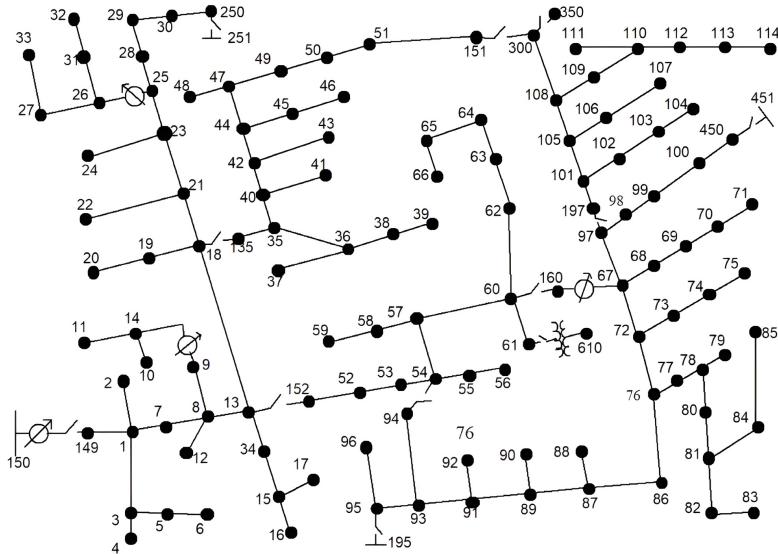
- **Evaluación sobre la topología IEEE 123-Node Test Feeder.** Se analiza el comportamiento del algoritmo fijando un nodo específico como raíz, así como aplicando el procedimiento de manera sucesiva considerando a cada nodo de la topología como nodo raíz. Esta aproximación permite obtener una visión promediada del comportamiento del algoritmo.
- **Evaluación sobre la topología IEEE 34-bus.** Se examina la capacidad del algoritmo para operar en una red de distribución con características distintas, ampliando así la validación de su aplicabilidad.
- **Estudio de la complejidad computacional.** Se realiza un análisis detallado de los dos bloques principales del algoritmo: etiquetado jerárquico; y balanceo iterativo, con el fin de caracterizar su escalabilidad y coste computacional.

#### 7.3.1. Evaluación sobre la topología IEEE 123-Node Test Feeder

El IEEE 123-Node Test Feeder es un sistema teórico de distribución eléctrica ampliamente utilizado para evaluar el desarrollo de estrategias, tecnologías o algoritmos de gestión en redes de potencia [163].

Este sistema, desarrollado por el IEEE, está compuesto por un conjunto de nodos, o puntos de conexión, interconectados dentro de una red típica de distribución en baja tensión. Tal como se ilustra en la Figura 7.7, la red de prueba incorpora un total de 11 interruptores, lo que permite la reconfiguración de la red en función de las necesidades operativas. Detalles adicionales sobre las características de las líneas y de las cargas pueden encontrarse en [163]. Considerando que los interruptores pueden encontrarse en estado abierto o cerrado, existen 2048 ( $2^{11}$ ) posibles configuraciones de conmutación, excluyendo la direccionalidad de cada enlace.

Para modelar los enlaces de la red, se han adoptado las configuraciones descritas en [163]. Estas configuraciones pueden clasificarse en tres tipos de enlace distintos (véase la Tabla 6.4), en función de la resistencia ( $\frac{\text{ohm}}{\text{km}}$ ), la sección transversal ( $\text{mm}^2$ ) y la capacidad máxima de corriente ( $A$ ). Las configuraciones recogidas en la Tabla 6.4 se aplican a cada enlace de la topología IEEE 123-Node Test Feeder, escalando las pérdidas a un 8% y triplicando la capacidad cuando se manejan valores de carga trifásicos. Este ajuste sigue las especificaciones de [163], donde a cada enlace se le asigna una configuración predefinida.

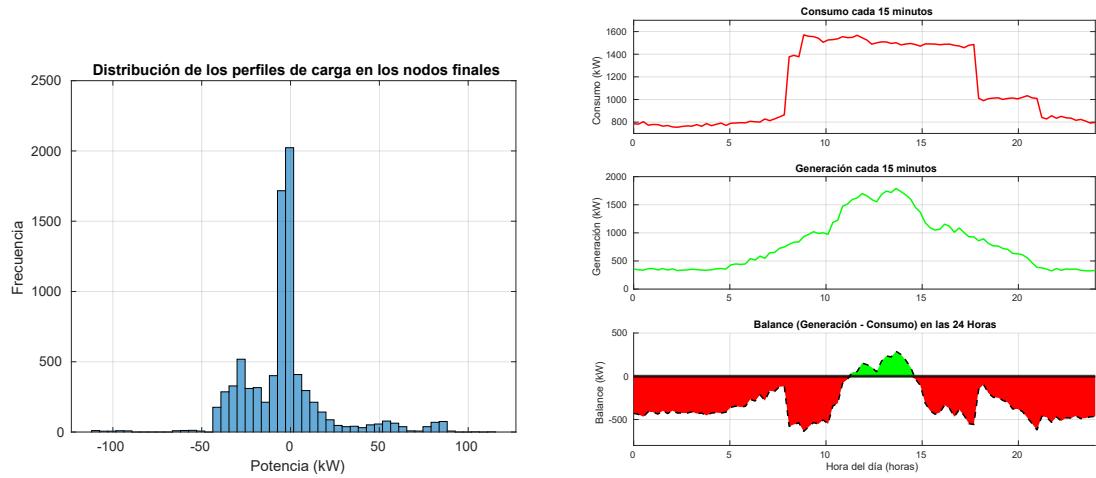


**Figura 7.7:** Topología IEEE 123-Node Test Feeder [163].

Con respecto a la configuración de cargas de la topología, se emplearon las cargas especificadas en los perfiles de nodo detallados en [163]. De acuerdo con la definición de la topología, existen dos tipos de nodos: nodos finales, que en cada instante pueden generar o demandar potencia, y nodos virtuales, que actúan como puntos de interconexión dentro de la topología pero que no generan ni demandan potencia en el estado inicial. Para evaluar el rendimiento de nuestro algoritmo, se han considerado 95 instantes de tiempo distintos, en los cuales cada nodo final presenta una potencia instantánea diferente, siempre coherente con el perfil definido en la descripción de la topología. Con el fin de ofrecer una representación más clara de los distintos perfiles de carga, la Figura 7.8a muestra el histograma global de los perfiles de carga de todos los nodos de la topología. Asimismo, la Figura 7.8b ilustra la generación, el consumo y el balance de potencia a lo largo del conjunto completo de nodos de la topología.

Como puede observarse en la Figura 7.8a, el perfil medio de potencia de los nodos finales tiende a concentrarse en valores negativos, lo cual implica que los resultados de los cálculos de balance de potencia realizados dentro de la topología serán, con alta proba-

bilidad, negativos. En otras palabras, la red tenderá, en promedio, a demandar potencia de la red de distribución. Tal como se ilustra en la Figura 7.8b, la topología presenta predominantemente una configuración de cargas globales negativas a lo largo del día. Por tanto, será de importancia crítica evaluar el desempeño de los criterios con respecto a la variación neta en los balances de potencia durante los momentos del día en los que se produzcan picos de generación.

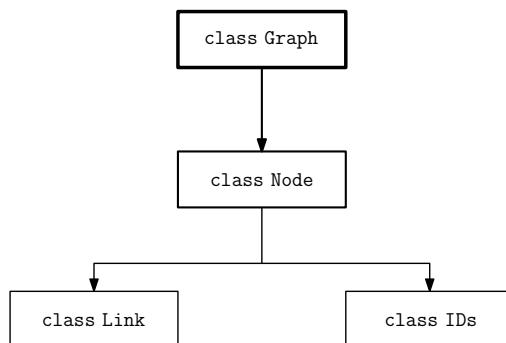


(a) Histograma de la distribución de los perfiles de carga de los nodos finales.

(b) Perfiles generales de consumo, generación y balance durante el día.

**Figura 7.8:** Caracterización de la configuración de carga utilizada en la Topología IEEE 123-Node Test Feeder.

Para la evaluación de nuestro algoritmo, la red IEEE 123-Node Test Feeder ha sido simulada en Python. Se seleccionó Python por su versatilidad, flexibilidad y su amplia adopción en la comunidad científica. Se evitó deliberadamente el uso de librerías externas, basando toda la implementación únicamente en las librerías estándar de Python, con el fin de minimizar la complejidad y reducir posibles errores derivados de dependencias entre paquetes externos. La simulación de la topología se implementó empleando Object-Oriented Programming (OOP), estableciendo una jerarquía de clases (véase la Figura 7.9).



**Figura 7.9:** Jerarquía de clases para simular la Topología IEEE 123-Node Test Feeder.

Dicha jerarquía incluye una clase de grafo, que contiene una lista de objetos nodo. Cada objeto nodo, a su vez, incorpora objetos enlace para representar sus relaciones con los nodos vecinos. Adicionalmente, los objetos nodo contienen objetos ID destinados a almacenar las etiquetas del algoritmo, permitiendo así identificar la posición de cada nodo dentro del o de los árboles.

La evaluación del algoritmo se ejecutó en 96 instantes temporales distintos, en los cuales los nodos finales de la topología partieron de diferentes condiciones de carga, ya fuera en régimen de consumo o de generación, siguiendo el perfil medio de potencia representado en la Figura 7.8b. Las simulaciones se realizaron en un servidor de alto rendimiento con un Ubuntu 22.04, equipado con un procesador Intel(R) Core(TM) i9 y 32 GB de memoria RAM. En cada simulación, para cada instante temporal, se evaluaron los criterios descritos en la Sección 7.2.2, con el fin de comparar el comportamiento de cada uno de ellos y observar cómo optimizan cada parámetro en función de sus respectivas funciones objetivo. Asimismo, para cada simulación temporal y para cada evaluación de criterio, la topología fue analizada bajo tres escenarios diferentes, descritos a continuación.

- **Tipo ideal:** Representa un escenario en el cual no existen pérdidas de transmisión de recursos cuando el recurso ( $r_i$ ) es enviado desde un nodo  $i$  hacia otro nodo  $j$  ( $i, j \in \mathcal{N}$  y  $i \neq j$ ) a través de un enlace, es decir,  $L_{ij} = 0$ . En consecuencia, para cualquier transmisión de recurso  $i \rightarrow j$ , en la cual el estado inicial es  $\{i = r_i, j = r_j\}$ , los recursos resultantes tras la transmisión son  $\{i' = 0, j' = r_i + r_j\}$ .
- **Tipo con pérdidas:** Implica la existencia de pérdidas durante la transmisión de recursos, es decir,  $L \neq \emptyset$ . En este caso, para cualquier transmisión de recurso  $i \rightarrow j$ , en la cual el estado inicial es  $\{i = r_i, j = r_j\}$ , los recursos resultantes tras la transmisión son  $\{i' = 0, j' = r_i + r_j - L_{ij}\}$ .
- **Tipo con pérdidas y restricciones de capacidad:** Hasta este punto, los enlaces no presentan limitaciones particulares para transmitir potencia, pero si el enlace se restringe a ciertos valores de transmisión, la potencia no será reenviada a tiempo y el excedente será descartado (al menos, de forma lógica para el algoritmo, incluso si el exceso de recurso permanece en su origen). Más específicamente, si la capacidad del enlace es  $C_{ij}$ , con  $C \neq \emptyset$ , y  $r_i \geq C_{ij}$ , entonces la transmisión asociada  $i \rightarrow j$  con estado inicial  $\{i = r_i, j = r_j\}$  resultará en el estado final  $\{i' = 0, j' = C_{ij} + r_j\}$ . Este tipo final considera tanto las pérdidas como las restricciones de capacidad, es decir,  $L \neq \emptyset$  y  $C \neq \emptyset$ . En consecuencia, para cualquier transmisión de recurso  $i \rightarrow j$ , en la cual el estado inicial es  $\{i = r_i, j = r_j\}$ , los recursos resultantes tras la transmisión son  $\{i' = 0, j' = r_i + r_j - L_{ij}\}$  en caso de que  $r_i \leq C_{ij}$ , y  $\{i' = 0, j' = C_{ij} + r_j - L_{ij}\}$  en caso de que  $r_i \geq C_{ij}$ .

La justificación de considerar estos tres escenarios distintos es obtener una comprensión más profunda del desempeño real de los criterios diseñados para optimizar o, en su caso, minimizar las pérdidas de potencia a lo largo de las líneas de la topología. Este

enfoque permite analizar en detalle cómo se comportan los criterios a medida que se introducen condiciones cada vez más realistas en la simulación de la topología IEEE 123-Node Test Feeder. Por tanto, considerando todos los instantes temporales simulados, todos los criterios de evaluación y los diferentes escenarios bajo estudio, fijando además el nodo raíz en el nodo 150, tal como se especifica en la descripción de la topología de ejemplo, es posible determinar el número total de simulaciones a realizar. Esto, a su vez, permite obtener una valoración en promedio del comportamiento del algoritmo (cf. Ecuación 7.7).

$$\begin{aligned}\langle N_{\text{simulaciones\_base}} \rangle &= N_{\text{deltas}} \times N_{\text{criterios}} \times N_{\text{escenarios}} \\ &= 96 \times 4 \times 3 \\ &= 1152 \text{ simulaciones}\end{aligned}\tag{7.7}$$

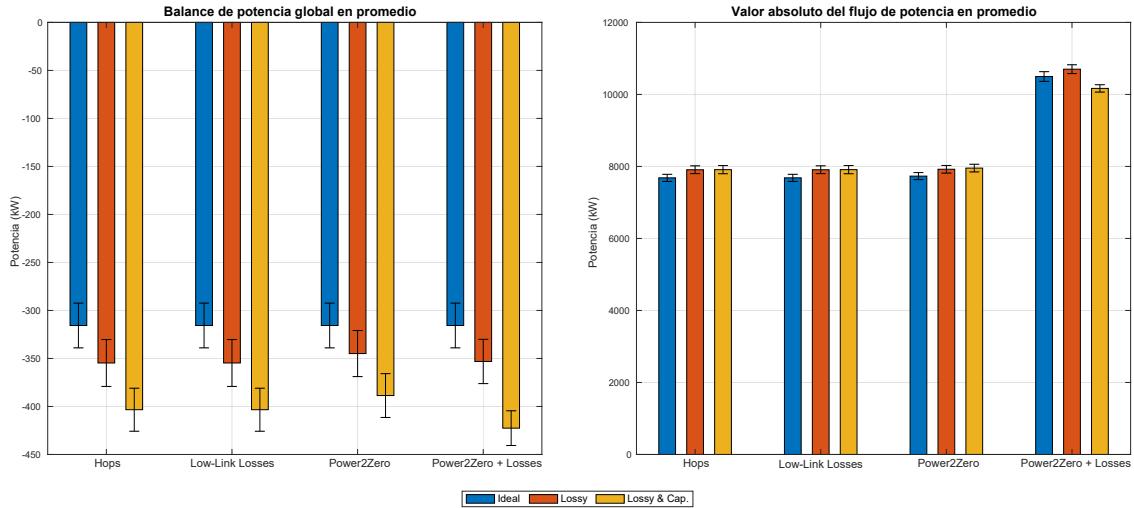
Estos resultados se denominan resultados base, en los cuales se analiza el comportamiento de los criterios descritos a lo largo de diferentes instantes temporales, bajo diversas configuraciones de carga y en tres escenarios distintos: ideal, con pérdidas y con pérdidas y restricciones de capacidad. Este análisis se realiza para el caso específico en el que el nodo raíz se fija en el nodo 150.

Para mitigar posibles sesgos introducidos por la fijación del nodo raíz, se llevó a cabo una evaluación adicional en paralelo. Este experimento, denominado resultados *all roots*, sigue la misma metodología que los resultados base, pero introduce una variación adicional: el nodo raíz de la topología no está fijado. En su lugar, cada nodo de la topología se considera como un posible nodo raíz, y los resultados se promedian considerando todas las configuraciones de nodo raíz. En consecuencia, es posible determinar el número total de simulaciones realizadas, lo que permite un enfoque doblemente promediado, tanto a lo largo de los instantes temporales, como entre los nodos raíz, proporcionando así una evaluación más completa y menos sesgada del desempeño del algoritmo (cf. Ecuación 7.8).

$$\begin{aligned}\langle N_{\text{simulaciones\_all\_roots}} \rangle &= \langle N_{\text{simulaciones\_base}} \rangle \times N_{\text{nodos}} \\ &= 1152 \times 129 \\ &= 148608 \text{ simulaciones}\end{aligned}\tag{7.8}$$

### 7.3.1.1. Resultados base

Iniciamos el análisis con los resultados base, en los que el nodo raíz se ha fijado en el nodo 150. Durante la evaluación del algoritmo, se examinarán diversos parámetros con el objetivo de obtener una comprensión detallada del comportamiento de cada criterio propuesto. El primer parámetro a evaluar es el balance global de potencia, definido idealmente como la suma de todas las configuraciones de carga de todos los nodos de la topología. Sin embargo, al considerar los escenarios que incluyen pérdidas, así como pérdidas combinadas con restricciones de capacidad, este parámetro puede variar, dado que las rutas seleccionadas para el encaminamiento de potencia desde cada nodo se convierten en un factor determinante.

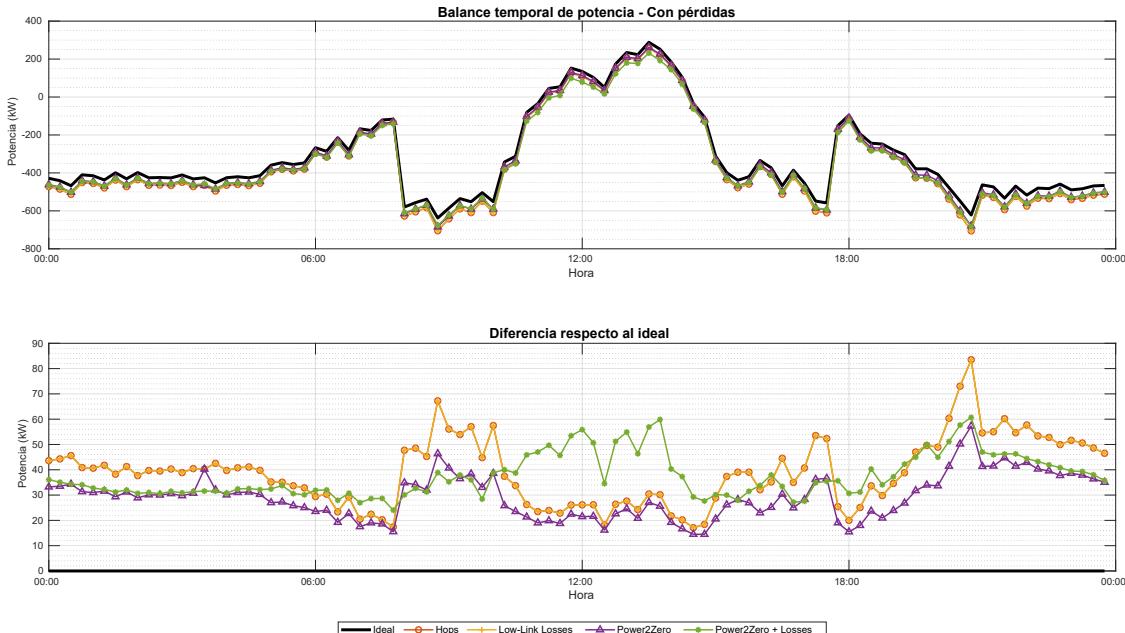


**Figura 7.10:** Balance de potencia global en promedio y valor absoluto del flujo de potencia - Resultados base.

La Figura 7.10 muestra los resultados del balance de potencia considerando los tres escenarios previamente mencionados (Ideal, Con Pérdidas y Con Pérdidas y Restricciones de Capacidad), así como los cuatro criterios de ejemplo descritos en la Sección 7.2.2 (*Hops*, *Low-Link Losses*, *Power2Zero* y *Power2Zero + Losses*, respectivamente). Al analizar la Figura 7.10, se observa que, en el escenario ideal, el resultado promedio entre todos los criterios se mantiene constante. Esto era de esperar, dado que no existen pérdidas de transmisión en los enlaces ni limitaciones de capacidad. Como se explicó anteriormente, el resultado en este escenario corresponde a la suma de todas las configuraciones de carga de los nodos de la topología, independientemente de las rutas de encaminamiento de potencia seleccionadas.

En contraste, al considerar el escenario que incluye tanto pérdidas como restricciones de capacidad, comienzan a apreciarse diferencias significativas. Por ejemplo, el criterio basado en saltos y el criterio de bajas pérdidas en enlaces presentan un comportamiento muy similar. Esta similitud se explica porque, en promedio, las rutas que tienden a seleccionar son independientes de la configuración específica de carga en un instante de tiempo determinado, y se basan principalmente en características intrínsecas de la ruta, como el número de saltos o las pérdidas acumuladas a lo largo del camino. Entre todos los criterios, el de balance de potencia local a cero (*power2zero*) muestra el mejor desempeño. Como se discutió previamente, este criterio busca, en última instancia, la ruta que, en un solo paso, acerca el balance local de potencia lo más posible a cero. Por ello, incluso en un modelo influenciado por pérdidas, este enfoque optimiza indirectamente el balance global de potencia al fomentar la auto-compensación local entre los nodos. Sin embargo, el criterio *power2zero* con pérdidas presenta un desempeño menor, ya que su función de coste, que originalmente hereda del criterio estándar *power2zero*, se ve afectada negativamente por la presencia de pérdidas de potencia.

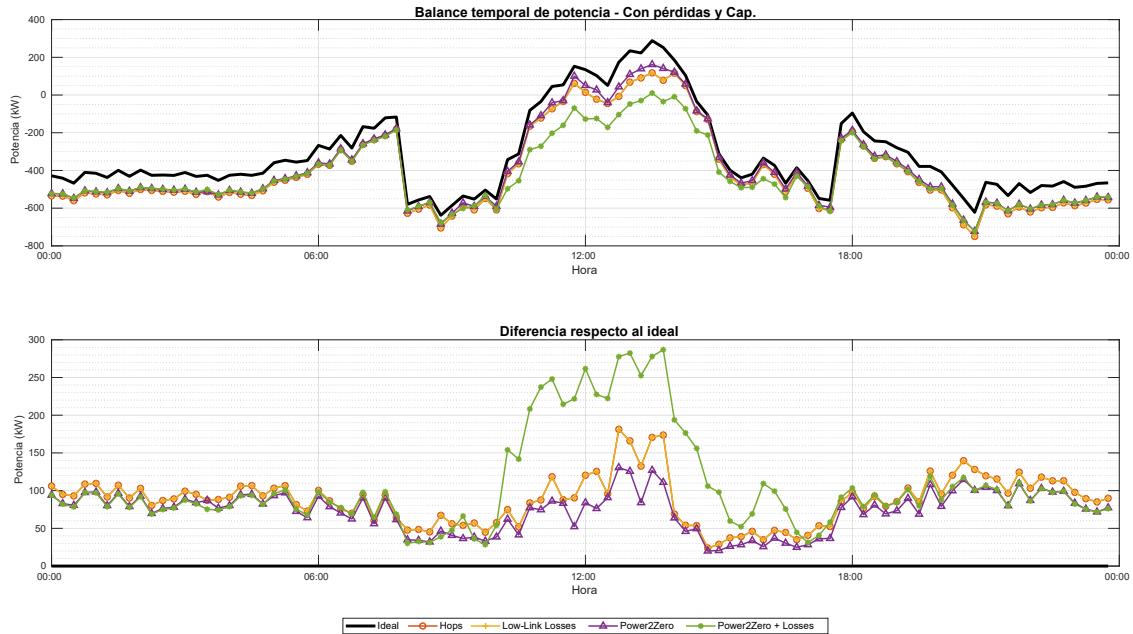
Asimismo, en la Figura 7.10 se analiza el valor absoluto del flujo de potencia, el cual refleja la cantidad total de energía transferida a lo largo de la topología. En promedio, los tres primeros criterios muestran la misma transferencia absoluta de potencia. No obstante, el criterio *Power2Zero* con pérdidas exhibe un mayor volumen de potencia transferida. Este comportamiento se debe a que, en términos relativos, dicho criterio requiere un mayor número de iteraciones para converger, como se ilustrará más adelante en la Figura 7.13.



**Figura 7.11:** Balance temporal de la potencia global en el escenario con pérdidas - Resultados base.

Continuando con las Figuras 7.11 y 7.12, se analiza la evolución temporal del balance global de potencia en los escenarios que consideran pérdidas, y tanto pérdidas como restricciones de capacidad, respectivamente. Estas figuras representan temporalmente el balance global de potencia en el nodo raíz en cada intervalo delta, es decir, cada 15 minutos. Por lo tanto, puede observarse una analogía con el comportamiento en el entorno real, como se ilustró previamente en la Figura 7.8b.

Por ello, las figuras muestran las desviaciones de estos balances respecto al escenario base, correspondiente al caso ideal. Esta representación fue necesaria debido a la homogeneidad de los resultados, con el fin de enfatizar de manera más clara las diferencias entre ellos. De ambas figuras se desprende que un criterio se posiciona como el más efectivo para adaptarse a las configuraciones dinámicas de carga de la topología: el criterio *Power2Zero*. Como se discutió previamente, al priorizar ajustes locales del balance de potencia hacia cero, este criterio minimiza eficazmente las pérdidas y mitiga la degradación del balance global de potencia. Esto ocurre porque el algoritmo fomenta que los nodos se auto-compensen localmente, reduciendo la necesidad de suministro externo de energía.

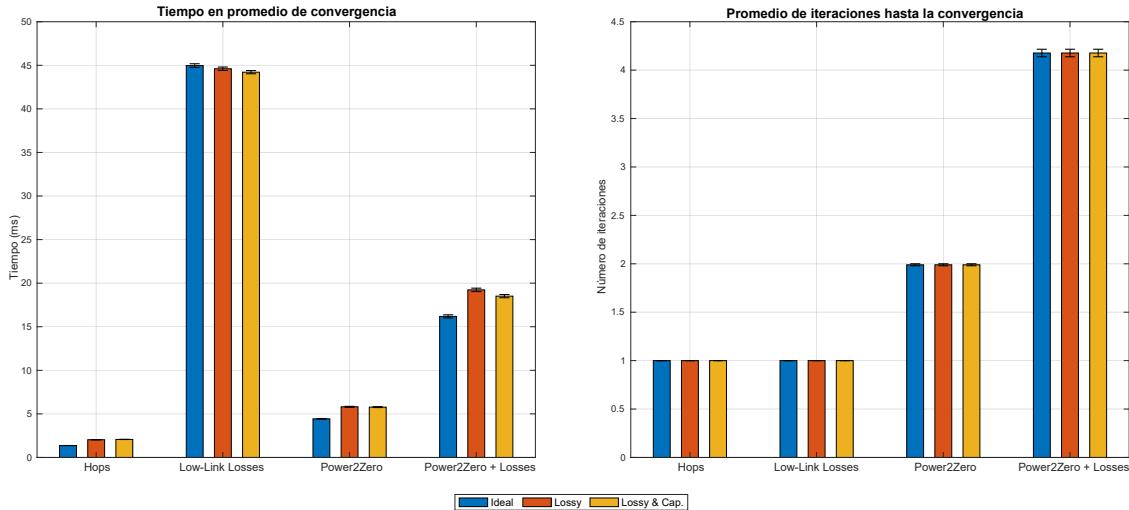


**Figura 7.12:** Balance temporal de la potencia global en el escenario con pérdidas y capacidades - Resultados base.

En contraste, el criterio basado en saltos y el criterio de bajas pérdidas en enlaces presentan un comportamiento más homogéneo. Esto se debe directamente a sus respectivas funciones de coste, que priorizan la selección de rutas en función de características intrínsecas de la topología, más que de las variaciones dinámicas en las configuraciones de carga. Como resultado, estos criterios permanecen en gran medida indiferentes a las fluctuaciones temporales de demanda y generación dentro de la topología. Cabe destacar que, como se observa en la Figura 7.11, el criterio *Power2Zero + Losses* representa un compromiso entre los criterios basados en saltos y bajas pérdidas en enlaces. Su comportamiento es más moderado en comparación con estos dos criterios, aunque no alcanza completamente el nivel de optimización logrado por *Power2Zero*.

Los últimos parámetros analizados en los resultados base son el tiempo total de convergencia y el número de iteraciones requeridas por cada criterio para alcanzar la convergencia en el enfoque iterativo descrito en la Subsección 7.2.3. Se define convergencia como el estado en el que no persisten desequilibrios de potencia residuales dentro de la topología. Como se ilustra en la Figura 7.13, el criterio con el menor tiempo de convergencia es el criterio basado en saltos. Este resultado es esperable, dado que su función de coste se basa únicamente en contabilizar la longitud de las etiquetas difundidas. En consecuencia, converge en promedio en una sola iteración. Un patrón similar se observa para el criterio de bajas pérdidas en enlaces, que sigue un enfoque análogo al del criterio basado en saltos. Sin embargo, a pesar de requerir en promedio únicamente una iteración, el criterio de bajas pérdidas en enlaces presenta el mayor tiempo total de convergencia. Esto se debe a la mayor complejidad computacional asociada a su función de coste, que evalúa todas las

características intrínsecas de toda la ruta, en lugar de apoyarse en un cálculo a un solo salto.

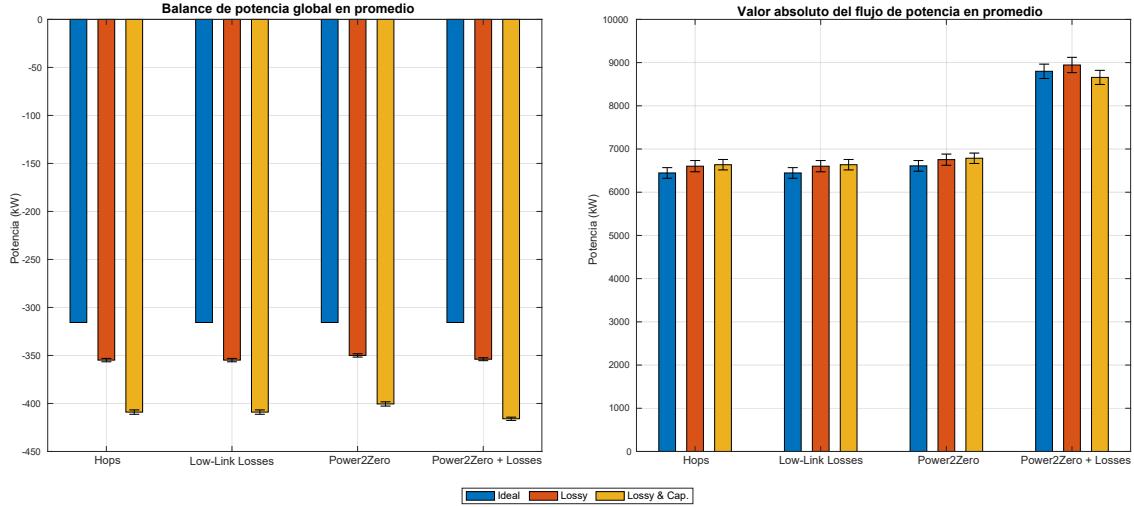


**Figura 7.13:** Promedio de tiempo/iteraciones hasta la convergencia - Resultados base.

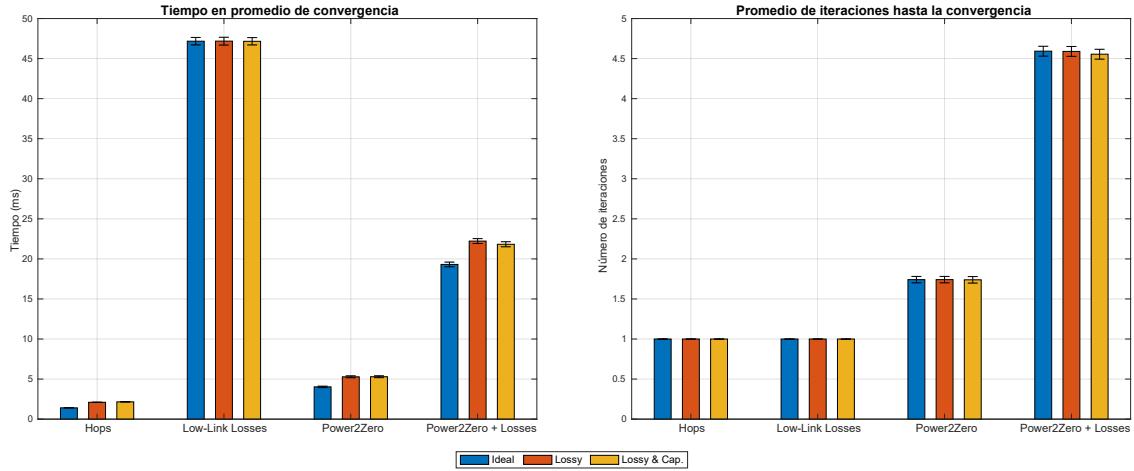
Al comparar estos resultados con los dos criterios *Power2Zero*, surge una observación clave: a pesar de requerir, en promedio, el doble o incluso más del doble de iteraciones que los criterios basados en saltos y bajas pérdidas, el tiempo total de convergencia de *Power2Zero* es significativamente menor, alcanzando incluso la mitad del tiempo de convergencia del criterio de bajas pérdidas. Esta eficiencia se debe a que la función de coste de *Power2Zero* se calcula a nivel local de un solo salto, reduciendo la carga computacional por iteración. Por lo tanto, en términos de tiempo de convergencia, el criterio simple *Power2Zero* se destaca como la opción más efectiva. Aunque requiere ligeramente más iteraciones que el criterio basado en saltos, su desempeño superior en el balance de potencia, tanto en escenarios con pérdidas como en escenarios con pérdidas y restricciones de capacidad, demostrando que es lo suficientemente rápido y dinámicamente adaptable para las necesidades específicas de esta topología y sus configuraciones de carga.

### 7.3.1.2. Resultados *All Roots*

En el escenario con todos los nodos como posibles raíces, denominado como resultados *All Roots*, se evalúan nuevamente los mismos parámetros analizados en los resultados base, no solo como un promedio temporal sobre los 96 intervalos de tiempo, sino también como un promedio agregado de todas las simulaciones en las que distintos nodos actúan como raíz. Como se muestra en las Figuras 7.14 y 7.15, los resultados relativos al balance global de potencia, al flujo absoluto de potencia, al tiempo de convergencia y al número de iteraciones presentan una notable homogeneidad. Esta consistencia valida la robustez de los criterios evaluados, al demostrar que su desempeño se mantiene en gran medida independiente de la elección del nodo raíz. Además, confirma que el sesgo observado en los resultados base es mínimo.



**Figura 7.14:** Balance de potencia global en promedio y valor absoluto del flujo de potencia - Resultados *All Roots*.



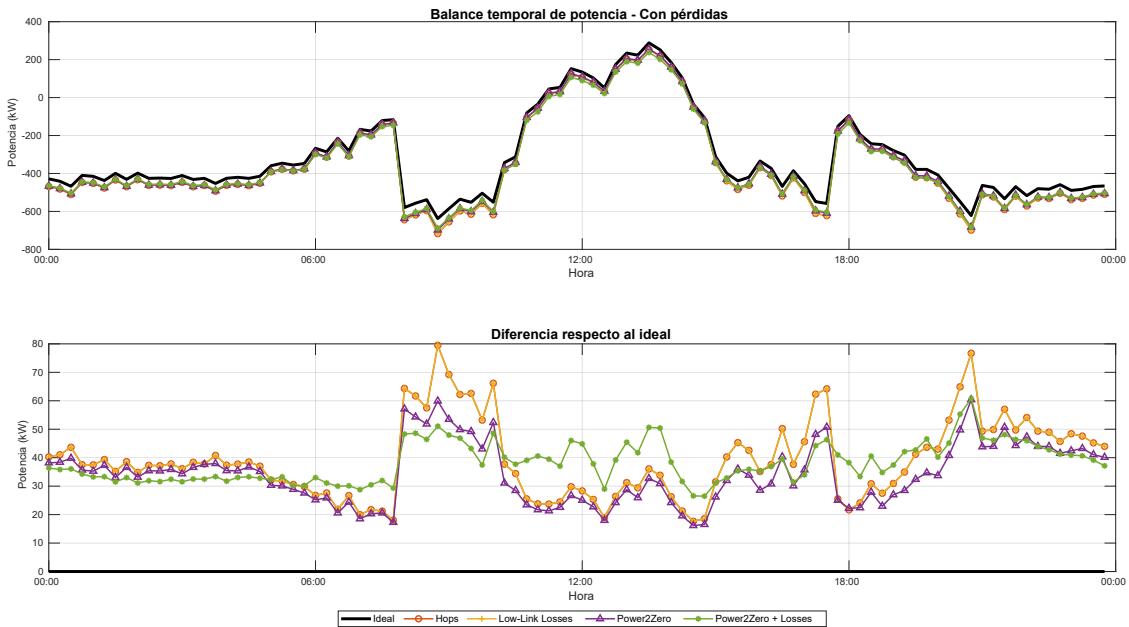
**Figura 7.15:** Promedio de tiempo/iteraciones hasta la convergencia - Resultados *All Roots*.

Al incrementar el número de simulaciones en diversos escenarios, el intervalo de confianza se ha reducido aún más, reforzando las conclusiones obtenidas en el experimento anterior. En particular, el criterio con mejor desempeño en términos de balance global continúa siendo *Power2Zero*; el criterio que desplaza la mayor cantidad de potencia absoluta es *Power2Zero + Losses*; y el criterio que converge más rápidamente es el basado en saltos, seguido muy de cerca por *Power2Zero*. Estos resultados consolidan a *Power2Zero* como el criterio más efectivo de forma global, al ofrecer un equilibrio óptimo entre rendimiento, velocidad de convergencia y adaptabilidad.

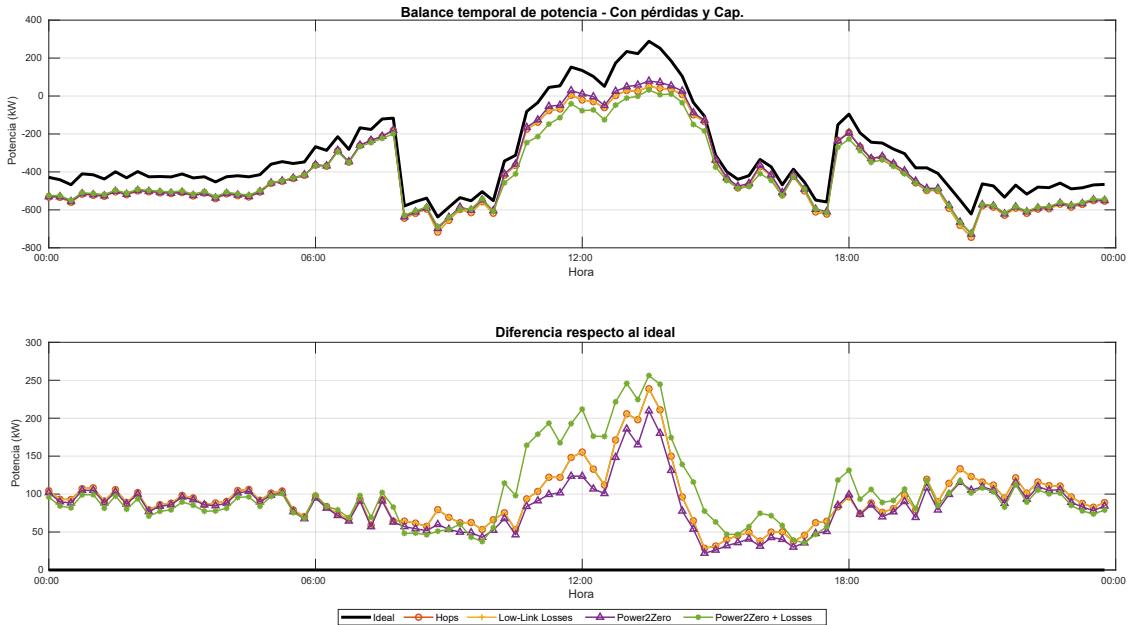
Las Figuras 7.16 y 7.17 muestran la evolución temporal del balance global de potencia, promediado sobre todas las simulaciones con diferentes nodos raíz en la topología.

## CAPÍTULO 7. BLOSTE

---



**Figura 7.16:** Balance temporal de la potencia global en el escenario con pérdidas - Resultados *All Roots*.



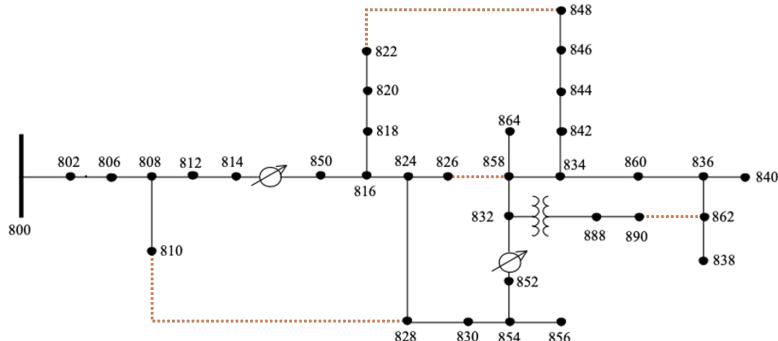
**Figura 7.17:** Balance temporal de la potencia global en el escenario con pérdidas y capacidades - Resultados *All Roots*.

Como era de esperar, los resultados aparecen más suavizados, confirmando las tendencias observadas previamente. El criterio *Power2Zero* continúa mostrando el mejor rendimiento, evidenciando una mayor capacidad de adaptación frente a configuraciones

dinámicas de carga. En contraste, los criterios *Hops* y *Low-Link Losses* exhiben un comportamiento más estático, al mantenerse en gran medida inalterados ante las variaciones en la dinámica de distribución de potencia. El criterio *Power2Zero + Losses* se presenta como una solución intermedia, ofreciendo un rendimiento estable y, en ciertos casos, superando incluso a otros enfoques durante los períodos de máxima demanda. Por ejemplo, como se ilustra en la Figura 7.16, se observa que el criterio *Power2Zero*, durante los picos de generación de las 08:00 y las 17:30, logra adaptarse mejor a las nuevas dinámicas de la red, alcanzando un balance de potencia superior (4.32 %) en comparación con los criterios basados en número de saltos y en pérdidas. En la misma ventana temporal, el criterio *Power2Zero + Losses* muestra un comportamiento más estático que *Power2Zero* en solitario, llegando incluso a rendir por debajo de los criterios basados en saltos y pérdidas entre las 10:30 y las 15:00. No obstante, este comportamiento más conservador contribuye a una respuesta más uniforme frente a los picos de demanda o generación, tal como se observa antes y después del período principal de generación (08:00–17:30). Estos resultados refuerzan la validación de *Power2Zero* como el criterio más robusto y adaptable para optimizar el balance de potencia en condiciones de red dinámicas.

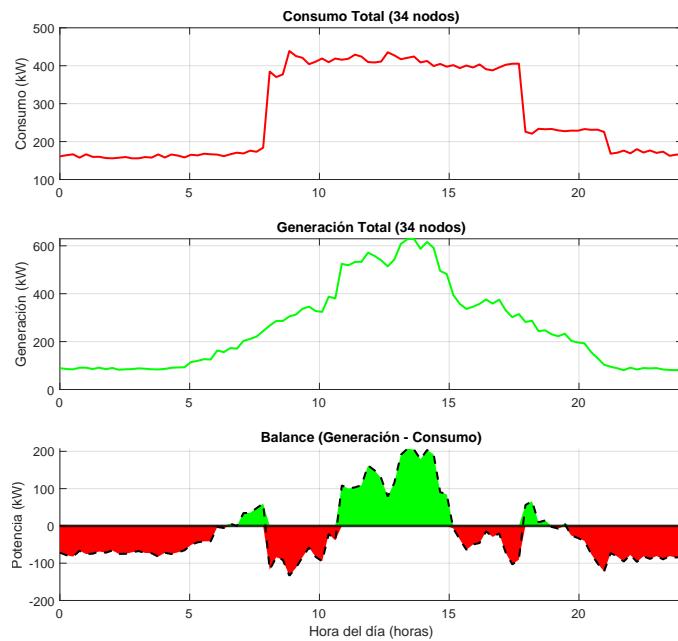
### 7.3.2. Evaluación sobre la topología IEEE 34-Node Test Feeder

En aras de evaluar la capacidad de operación del algoritmo en redes de distribución eléctrica diferentes, se ha elegido la topología IEEE 34-Node Test Feeder. La topología IEEE 34-bus [163] constituye un modelo de referencia ampliamente utilizado en la evaluación de sistemas de distribución eléctrica. Corresponde a un alimentador real, asimétrico y ligeramente cargado, ubicado en Arizona y operando a una tensión nominal de 24,9 kV. Entre sus características principales destacan la presencia de ramales de gran longitud, líneas tanto monofásicas como trifásicas, reguladores de tensión en línea, así como una breve sección operando a 4,16 kV. La Figura 7.18 muestra la disposición de la topología IEEE 34-bus. Como puede observarse, se trata de una estructura prácticamente lineal, con un grado de mallado (*mesh*) muy limitado. Con el fin de permitir una evaluación más representativa de las diferencias entre los distintos criterios analizados, ha sido necesario incorporar enlaces adicionales, representados en color naranja en la Figura 7.18.



**Figura 7.18:** Topología IEEE 34-Node Test Feeder.

Con respecto a la configuración de cargas de la topología, se emplearon también los perfiles de nodo especificados en [163]. Con el objetivo de garantizar la homogeneidad de las condiciones iniciales de la evaluación llevada a cabo en la topología IEEE 123-bus, se seleccionó de manera aleatoria perfiles de cargas a los 34 nodos de la red, de los existentes en la topología IEEE 123-bus. Dicho selección fue implementada con una *seed* predefinida, lo que permite asegurar la reproducibilidad tanto de las cargas como de los resultados obtenidos. Tras evaluar múltiples configuraciones, se seleccionó la semilla  $seed = 1998$ . El perfil de balance resultante se muestra en la Figura 7.19, donde puede observarse la presencia de un pico de generación en torno al mediodía, en concordancia con el patrón de cargas previamente utilizado en la topología IEEE 123-bus.

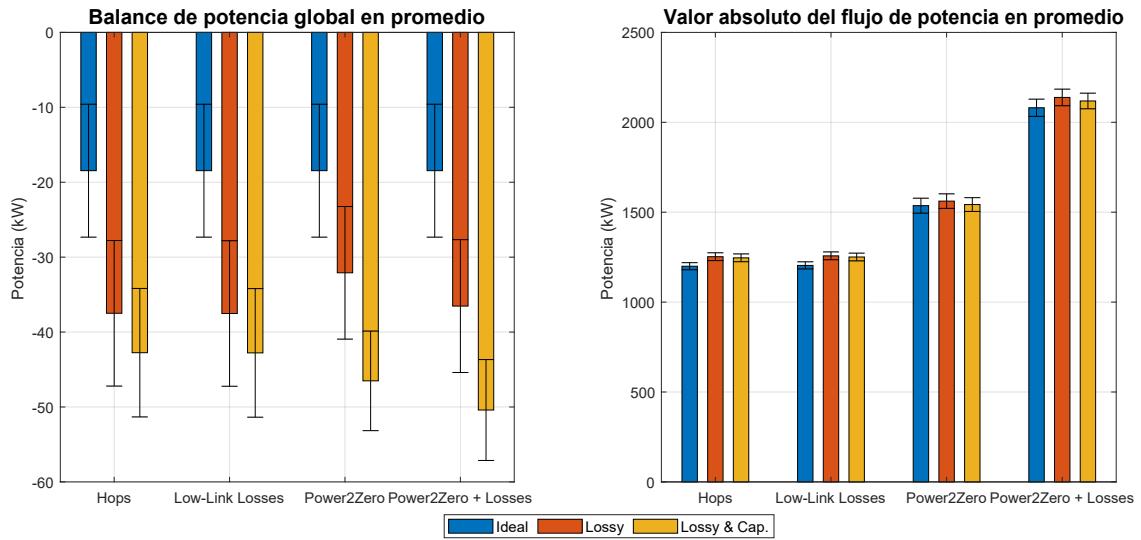


**Figura 7.19:** Perfiles generales de cargas a lo largo del día empleados con la topología IEEE 34-bus.

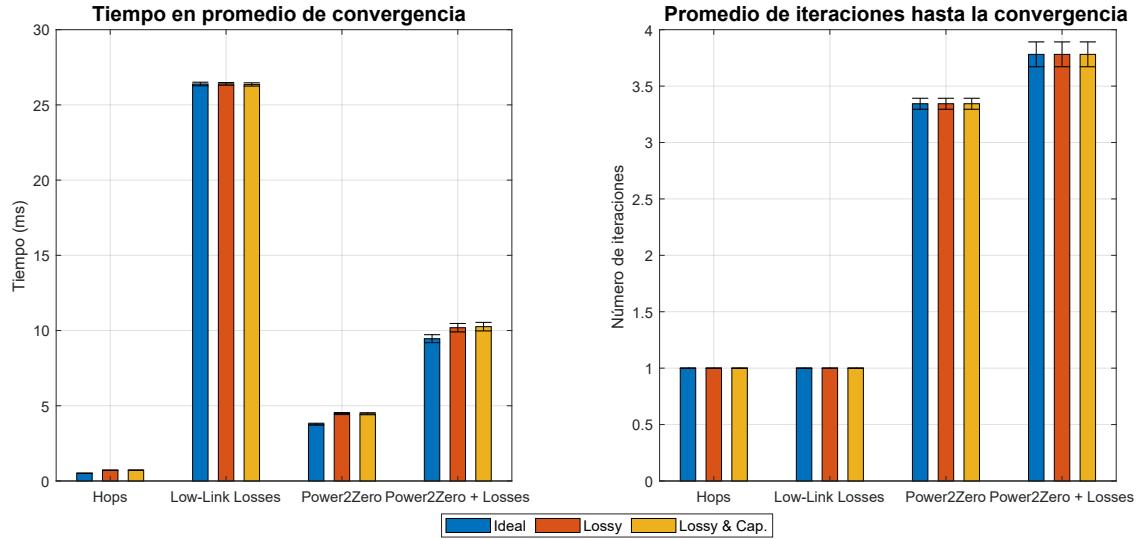
Para el modelado de los enlaces de la red se adoptaron igualmente las configuraciones descritas en [163], adaptando la distancia física de las líneas a los valores típicos empleados en la topología IEEE 123-bus. De este modo, se garantiza que tanto las características intrínsecas de las líneas como las pérdidas asociadas resulten comparables en el caso de la topología IEEE 34-bus. Las configuraciones empleadas se recogen en la Tabla 6.4. La simulación de esta topología se llevó a cabo utilizando la misma jerarquía de clases representada en la Figura 7.9, así como el mismo entorno de ejecución: un servidor con *Ubuntu 22.04*, equipado con un procesador Intel(R) Core(TM) i9 y 32 GB de memoria RAM. Asimismo, se consideraron los tres escenarios previamente descritos (ideal, con pérdidas, y con pérdidas y restricciones de capacidad en las líneas) junto con los cuatro criterios definidos anteriormente. Como nodo raíz de la topología se seleccionó el nodo 800, obteniéndose las simulaciones únicas que se detallan en la Ecuación 7.9.

$$\begin{aligned}
 \langle N_{\text{simulaciones\_ieee34}} \rangle &= N_{\text{deltas}} \times N_{\text{criterios}} \times N_{\text{escenarios}} \times N_{\text{root}} \\
 &= 96 \times 4 \times 3 \times 1 \\
 &= 1152 \text{ simulaciones}
 \end{aligned} \tag{7.9}$$

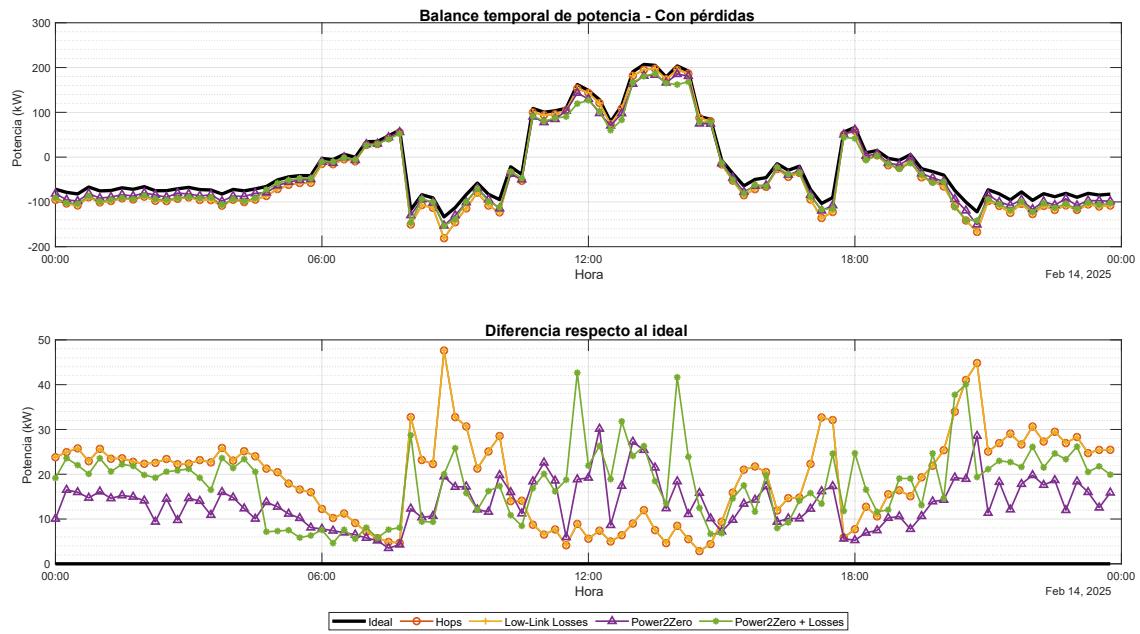
En las Figuras 7.20 y 7.21 se presentan tanto el balance y el flujo medio de potencia, como el tiempo medio de convergencia y el número medio de iteraciones por criterio. Como se aprecia en la Figura 7.20, los resultados de balance y flujo son similares a los obtenidos previamente con la topología de referencia IEEE 123-bus. En este caso, el criterio *Power2Zero* vuelve a destacar como el de mejor desempeño promedio en el escenario con pérdidas. Sin embargo, al considerar el escenario con pérdidas y restricciones de capacidad en los enlaces, se observa una degradación de este criterio frente a *Hops* y *Low-Link Losses*. Esta diferencia se explica porque, a diferencia de la topología IEEE 123-bus, que presenta un mayor grado de mallado (*mesh*) y, por tanto, más alternativas de encaminamiento sin alcanzar los umbrales de capacidad, en la topología de 34 nodos las posibilidades de flujo se ven más limitadas. Como consecuencia, en promedio se registran mayores pérdidas, dado que el criterio *Power2Zero*, al basarse en la auto-compensación local entre nodos vecinos, carece de una visión global de la ruta completa. Otro aspecto a destacar es el aumento de los intervalos de confianza. Esto se debe a que el número de muestras por valor medio se ha reducido de 123 a 34 nodos, lo que incrementa la dispersión y la desviación relativa de los resultados. En cuanto a los tiempos de convergencia, representados en la Figura 7.21, se observa un comportamiento análogo al obtenido con la topología IEEE 123-bus. No obstante, al tratarse de una red con un menor número de nodos, el tiempo medio de convergencia resulta inferior, alcanzando máximos en torno a los 25 milisegundos, frente a los valores promedio de hasta 45 milisegundos registrados en la topología de 123 nodos.



**Figura 7.20:** Balance de potencia global en promedio y valor absoluto del flujo de potencia - Topología IEEE 34-Node Test Feeder.



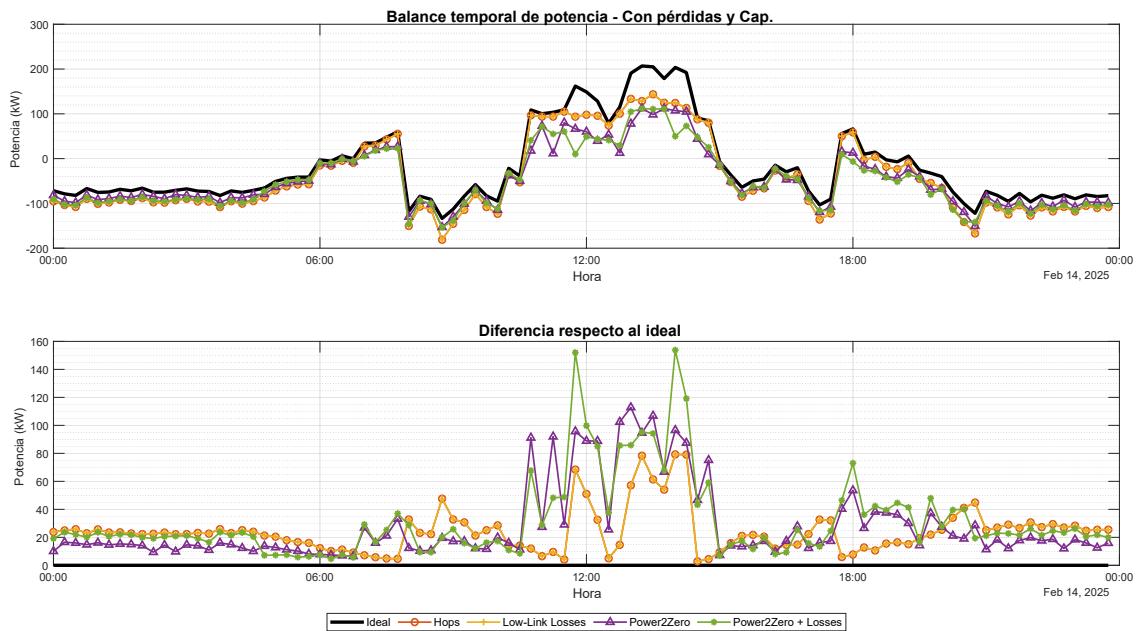
**Figura 7.21:** Promedio de tiempo/iteraciones hasta la convergencia - Topología IEEE 34-Node Test Feeder.



**Figura 7.22:** Balance temporal de la potencia global en el escenario con pérdidas - Topología IEEE 34-Node Test Feeder.

En las Figuras 7.22 y 7.23 se muestra la evolución temporal del balance global de potencia a lo largo de un día completo. Los resultados presentan una fuerte similitud con los obtenidos en la topología de referencia IEEE 123-bus. Tal como se aprecia, los criterios *Hops* y *Low-Link Losses* exhiben un comportamiento marcadamente homogéneo, al basarse en características intrínsecas de la topología. En contraste, los criterios *Po-*

*wer2Zero* y *Power2Zero + Losses* demuestran una mayor capacidad para optimizar tanto el balance como las pérdidas asociadas, al favorecer la compensación local de demandas y excedentes. En la mayoría de los casos, el criterio *Power2Zero* se confirma como la opción más eficaz. No obstante, en el escenario con pérdidas y restricciones de capacidad en los enlaces se observa una excepción relevante: al existir un menor grado de alternativas para encaminar la potencia, los criterios *Hops* y *Low-Link Losses* alcanzan un mejor desempeño, particularmente durante el periodo valle (08:00–17:30), coincidente con los máximos de generación en la red.



**Figura 7.23:** Balance temporal de la potencia global en el escenario con pérdidas y capacidades - Topología IEEE 34-Node Test Feeder.

En conjunto, estos resultados evidencian una clara consistencia tanto en el comportamiento de las topologías evaluadas como en el desempeño de los criterios, reforzando así la validez de las conclusiones obtenidas en los distintos escenarios analizados.

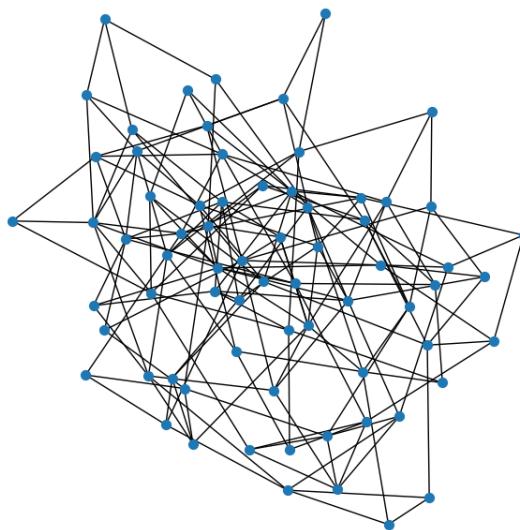
### 7.3.3. Estudio de la complejidad computacional

En esta sección se presenta un análisis de la complejidad computacional del algoritmo. En primer lugar, en la Sección 7.3.3.1 se estudia la complejidad asociada al proceso de etiquetado y exploración de la red, mientras que en la Sección 7.3.3.2 se aborda la complejidad derivada del proceso iterativo y de la aplicabilidad de los criterios diseñados.

Para llevar a cabo este estudio, se generaron topologías aleatorias empleando la librería NetworkX<sup>2</sup> de Python, reproduciendo características propias de las topologías de referencia del IEEE, tales como distancias físicas típicas y configuraciones de enlaces. En lo referente

<sup>2</sup><https://networkx.org/>

a las cargas, estas fueron generadas aleatoriamente en el rango de  $[-4, 4]$  kW a lo largo del día, dado que la configuración de cargas es indiferente para el estudio de la complejidad computacional, pero son necesarias para estudiar el funcionamiento computacional de los criterios. Se consideraron topologías con un número de nodos creciente desde 10 hasta 2500, en incrementos de 10 ( $10:10:2500$ ), almacenando todas las instancias para garantizar la reproducibilidad de los experimentos. La Figura 7.24 ilustra un ejemplo de una topología aleatoria generada con 70 nodos. Como puede observarse, las topologías generadas presentan un carácter marcadamente *mesh*, lo que permite poner de manifiesto las ventajas del algoritmo propuesto.



**Figura 7.24:** Ejemplo de topología aleatoria generada de 70 nodos para el estudio de la complejidad computacional.

#### 7.3.3.1. Complejidad computacional del mecanismo de etiquetado del algoritmo

Para evaluar la complejidad computacional del proceso de etiquetado (`spread_ids`) de nuestro algoritmo, descrito en la Sección 7.2.1.1, se ha evaluado frente a tres construcciones de árbol alternativas: Minimum Spanning Tree (MST), PSO y GA. Para llevar a cabo la evaluación, se siguió el siguiente protocolo:

1. Para cada topología aleatoria de  $n$  nodos ( $n = 10, 20, \dots, 2500$ ):
  - Se cargaron sus ficheros topológicos generados anteriormente con NetworkX.
  - Se eligieron 10 nodos raíz diferentes, al azar, de tal forma:  $(G = (\mathcal{N}, \mathcal{L}), \text{root}_i \in \mathcal{N}, i = 1, \dots, 10)$ .
  - Para cada nodo raíz,  $\text{root}_i$ :
    - Se midió el tiempo de `spread_ids`,  $T_{\text{SI}}$ .

- Se midió el tiempo de construcción de un árbol MST (usando el algoritmo de Prim [194], implementado en NetworkX),  $T_{\text{MST}}$ .
  - Se midió el tiempo de construcción vía PSO discreto,  $T_{\text{PSO}}$ .
  - Se midió el tiempo de construcción vía algoritmo genético,  $T_{\text{GA}}$ .
2. Los resultados temporales ( $\{\text{Run}, \text{Root}, T_{\text{SI}}, T_{\text{MST}}, T_{\text{PSO}}, T_{\text{GA}}\}$ ) se almacenaron para su posterior análisis.

En conjunto, la ejecución completa de las pruebas requirió aproximadamente 50 horas, a continuación, se detallan las implementaciones alternativas, así como los resultados obtenidos.

**7.3.3.1.1. MST (Prim)** El árbol de descubrimiento mínimo se obtiene con el algoritmo de Prim en un grafo no dirigido  $G = (\mathcal{N}, \mathcal{L})$  ponderado por distancia física de los enlaces  $w_{uv}$ . Por tanto:

$$Tree_{\text{MST}} = \arg \min_{\substack{T \subseteq \mathcal{L} \\ |T|=|\mathcal{N}|-1}} \sum_{(u,v) \in T} w_{uv}. \quad (7.10)$$

Para orientar el árbol generado hacia el nodo raíz root, aplicamos un recorrido Breadth-First Search (BFS) que genera aristas dirigidas ( $u \rightarrow v$ ) salientes desde la raíz:

$$\text{orient\_tree}(T, \text{root}) = \{(u, v) \in T \mid u \text{ es antecesor de } v \text{ en BFS desde root}\}. \quad (7.11)$$

La complejidad teórica del proceso, se puede estimar como  $O(|\mathcal{L}| \log |\mathcal{N}|)$  para Prim, más  $O(|\mathcal{N}| + |\mathcal{L}|)$  para BFS de orientación. Hay que tener en cuenta que el árbol generado con esta implementación alternativa es una solución que pertenece al subconjunto de arboles generados con nuestro método de `spread_ids`.

**7.3.3.1.2. PSO discreto** Se diseña un enjambre de  $P$  partículas (En nuestro caso, con 20 partículas), cada una representando un conjunto de  $|\mathcal{N}| - 1$  aristas válidas  $\{e_k\} \subset \mathcal{L}$ . Cuya función de *fitness* se define como la suma de las distancias físicas de los enlaces:

$$f(\{e_k\}) = \sum_k w_{e_k}. \quad (7.12)$$

El algoritmo itera  $\ell = 1, \dots, L$ , siendo  $L$  el número de iteraciones totales configurable, y fijado en este caso a 15 iteraciones. Por cada iteración:

1. Cada partícula  $p$  del enjambre  $P$ :
  - Realiza un cruce de su árbol actual ( $p.\text{tree}$ ), con el mejor árbol de todo el enjambre  $P$  ( $gbest.\text{tree}$ ), de tal forma:  $\text{new} \leftarrow p.\text{tree} \cap gbest.\text{tree} \cup \text{relleno aleatorio}$ .
  - Se evalúa el nuevo árbol:  $\text{fitness}_p = f(\text{new})$ .
  - Actualización de ( $gbest.\text{tree}$ ) en caso de que haya mejora.

Por tanto, tendremos  $P$  partículas, donde cada partícula  $p$  representará un árbol. Después de las  $L$  iteraciones, se orienta `gbest.tree` igual que en el método de MST, teniendo una complejidad teórica aproximada:  $O(L P |\mathcal{N}|)$ .

$$Tree_{PSO} = \text{orient\_tree}(gbest.tree, \text{root}). \quad (7.13)$$

**7.3.3.1.3. Algoritmo Genético (GA)** Operamos con una población inicial de  $P$  árboles aleatorios, fijada en este caso a 20. Con cada generación  $g$ , de  $G$  totales fijadas también a 15, se llevan a cabo los siguientes pasos:

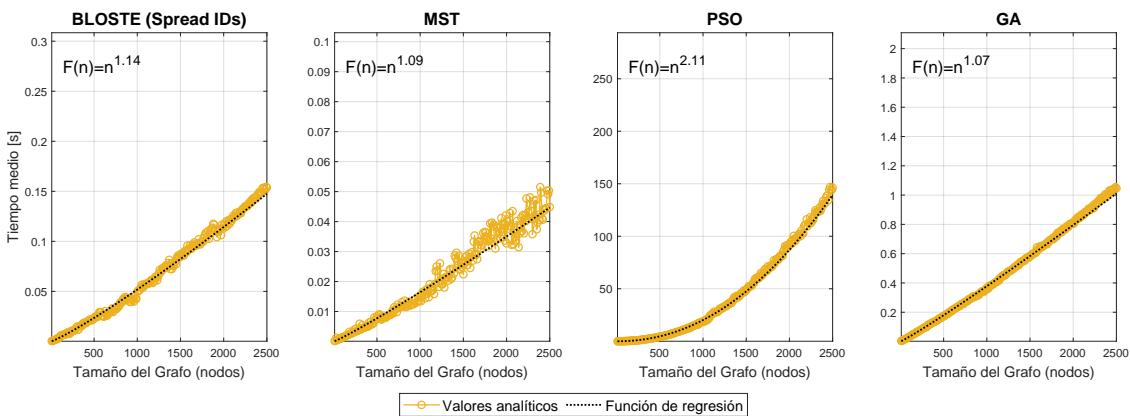
- **Selección:** ordenar por la función de *fitness* (la misma que en el PSO)  $f(\cdot)$  y seleccionar los  $P/2$  mejores árboles.
- **Cruce:** de cada par de árboles  $(T_u, T_v)$  se forma la unión de enlaces  $U = T_u \cup T_v$ , y se extrae un MST de  $U$  con pesos aleatorios para garantizar  $|\mathcal{N}| - 1$  enlaces válidos.
- **Mutación:** con probabilidad  $\mu$  se sustituye aleatoriamente un enlace.

Tras  $G$  generaciones, se toma el mejor el mejor árbol, y se orienta hacia la raíz:

$$Tree_{GA} = \text{orient\_tree}(\arg \min_{T \in P} f(T), \text{root}). \quad (7.14)$$

Teniendo una complejidad teórica aproximada de:  $O(G P |\mathcal{L}| \log |\mathcal{N}|)$ , por el MST interno en cada cruce.

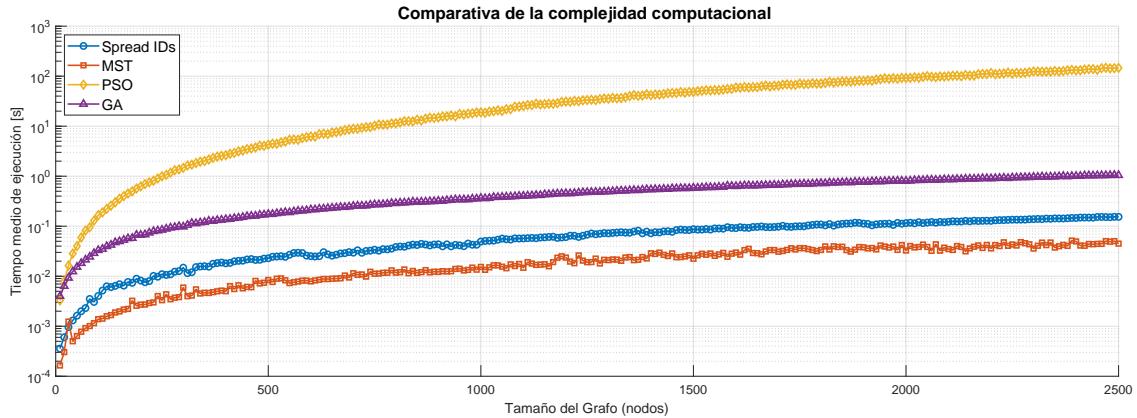
**7.3.3.1.4. Resultados** Tras ejecutar la evaluación descrita, se obtuvieron los resultados mostrados en las Figuras 7.25 y 7.26, que recogen el análisis de la complejidad computacional de la fase de etiquetado para las cuatro implementaciones consideradas: BLOSTE (`spread_ids`), MST, PSO y GA.



**Figura 7.25:** Complejidad computacional en la fase de etiquetado.

Los resultados de la Figura 7.25 se obtuvieron promediando todas las ejecuciones de cada algoritmo entre todas las ejecuciones de los diferentes nodos raíz por cada tamaño de

grafo, y ajustando los datos mediante regresión lineal con `polyfit` de `matlab`. Los valores indican que BLOSTE escala aproximadamente como  $F(n) = n^{1.14}$ , lo que supone un crecimiento casi lineal con una ligera tendencia superlineal. El algoritmo MST presenta un ajuste  $F(n) = n^{1.09}$ , reflejando un comportamiento prácticamente lineal y una notable eficiencia computacional. El enfoque basado en PSO muestra el mayor coste computacional, con  $F(n) = n^{2.11}$ , lo que corresponde a un crecimiento cuadrático con respecto al tamaño del grafo. Por su parte, GA presenta  $F(n) = n^{1.07}$ , también cercano a lineal, aunque con tiempos absolutos de ejecución superiores a los de MST y BLOSTE. Conviene resaltar que, a diferencia del resto de algoritmos, BLOSTE genera múltiples árboles en lugar de uno único, característica que, combinada con su baja complejidad, lo convierte en una solución especialmente adecuada en escenarios que requieran redundancia o balanceo de carga, ofreciendo una escalabilidad comparable a la de MST y GA.



**Figura 7.26:** Comparativa de la complejidad computacional en tiempo con escala logarítmica.

La representación en escala logarítmica de la Figura 7.26 confirma estas tendencias: las pendientes casi paralelas de BLOSTE, MST y GA reflejan complejidades empíricas similares, mientras que PSO exhibe una pendiente significativamente más pronunciada. Además, BLOSTE mantiene una ventaja constante sobre GA y PSO en tiempo absoluto de ejecución para todos los tamaños de grafo evaluados. Aunque sus tiempos son superiores a los de MST, el hecho de generar múltiples árboles simultáneamente lo convierte en una alternativa muy atractiva en redes supermalladas, ya que proporciona mayor flexibilidad y capacidad de adaptación en el encaminamiento de recursos energéticos.

### 7.3.3.2. Complejidad computacional del mecanismo iterativo del algoritmo

Tal y como se introdujo en la Sección 7.2.3, BLOSTE incorpora un mecanismo iterativo destinado a refinar progresivamente la configuración de la red de acuerdo con unos criterios de decisión predefinidos, con el objetivo de alcanzar un balance de potencia adecuado en la red de distribución eléctrica. Para estimar la complejidad computacional de dicho mecanismo, se emplearon las topologías aleatorias previamente generadas, junto con perfiles de carga definidos en el rango  $[-4, 4]$  kW. Aunque estos perfiles carecen de validez práctica desde el punto de vista eléctrico, resultan útiles para analizar el comportamiento

computacional de los criterios. En este caso no se han considerado algoritmos alternativos con los que establecer comparaciones, dado que la naturaleza configurable de los criterios hace que su implementación dependa del caso de uso final. Por ello, el estudio se centró exclusivamente en el rendimiento y la complejidad computacional de los cuatro criterios presentados. Para llevar a cabo la evaluación, se siguió el siguiente protocolo:

1. Para cada topología aleatoria de  $n$  nodos ( $n = 10, 20, \dots, 2500$ ):
  - Se cargaron sus ficheros topológicos generados anteriormente con NetworkX.
  - Se eligieron 10 nodos raíz diferentes, al azar, de tal forma:  $(G = (\mathcal{N}, \mathcal{L}), \text{root}_i \in \mathcal{N}, i = 1, \dots, 10)$ .
  - Para cada nodo raíz,  $\text{root}_i$ :
    - Se cargan los 96  $\delta$  de cargas en cada nodo de la topología (representando cada  $\delta$  un intervalo de 15 minutos), por cada  $\delta_{\text{carga}}$ :
      - Se midió el tiempo por cada criterio ( $T_{c1}, T_{c2}, T_{c3}, T_{c4}$ ), hasta que no hubiera cargas encerradas en la topología, es decir, que convergiera el método iterativo.
2. Los resultados temporales ( $\{\text{Run}, \text{Root}, \delta_{\text{carga}}, T_{c1}, T_{c2}, T_{c3}, T_{c4}\}$ ) se almacenaron para su posterior análisis.

En conjunto, la ejecución completa de las pruebas requirió aproximadamente 60 horas, cuyos resultados se presentan y discuten a continuación.

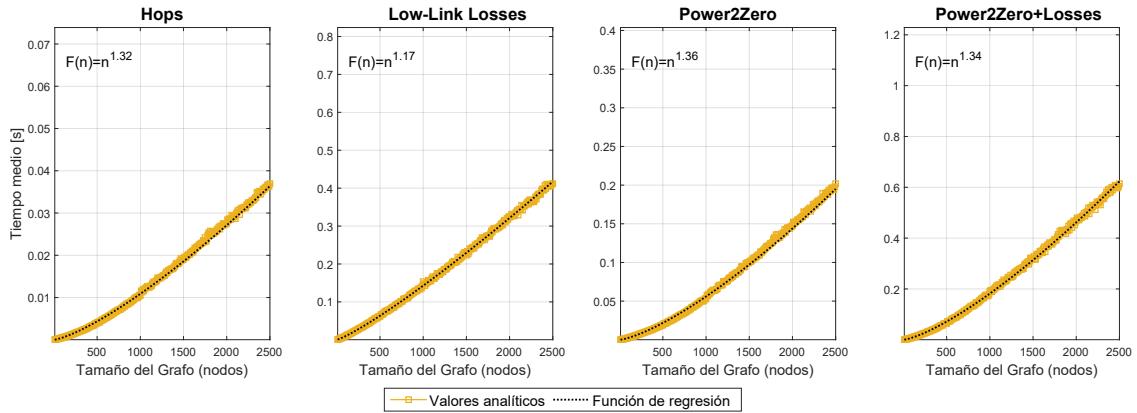
**7.3.3.2.1. Resultados** En esta sección se presentan los resultados del estudio de la complejidad computacional del proceso iterativo, evaluado bajo los distintos criterios descritos previamente. Para cada topología se seleccionaron diez raíces con semilla fija y, para cada raíz, se consideraron los 96 intervalos temporales correspondientes a un día completo. Con el fin de obtener una medida representativa por topología y criterio, se aplicó un esquema de doble promedio: en primer lugar, se calcularon los tiempos medios sobre las 96 deltas temporales de cada raíz; posteriormente, estos promedios se agregaron entre las diez raíces, obteniéndose así un valor final por criterio y topología.

Las Figuras 7.27 y 7.28 muestran, respectivamente, (i) el ajuste de regresión obtenido mediante la función `polyfit` para cada criterio, y (ii) la comparación de los tiempos medios (doble promedio) en función del tamaño de la topología. A partir de la representación en escala log–log, se approximó la dependencia temporal mediante una ley de potencia de la forma

$$T(n) \approx C n^\gamma, \quad (7.15)$$

donde  $n$  representa el número de nodos de la topología,  $C$  es una constante de escala y  $\gamma$  el exponente estimado mediante ajuste lineal de  $\log T$  frente a  $\log n$ . Los valores obtenidos para cada criterio son los siguientes:

$$\begin{aligned}
 \text{Hops: } & T_{\text{Hops}}(n) \approx C_H n^{1.32} \\
 \text{Low-Link Losses: } & T_{\text{LLL}}(n) \approx C_{\text{LL}} n^{1.17} \\
 \text{Power2Zero: } & T_{\text{P2Z}}(n) \approx C_{\text{P2Z}} n^{1.36} \\
 \text{Power2Zero+Losses: } & T_{\text{P2Z+L}}(n) \approx C_{\text{P2ZL}} n^{1.34}
 \end{aligned} \tag{7.16}$$

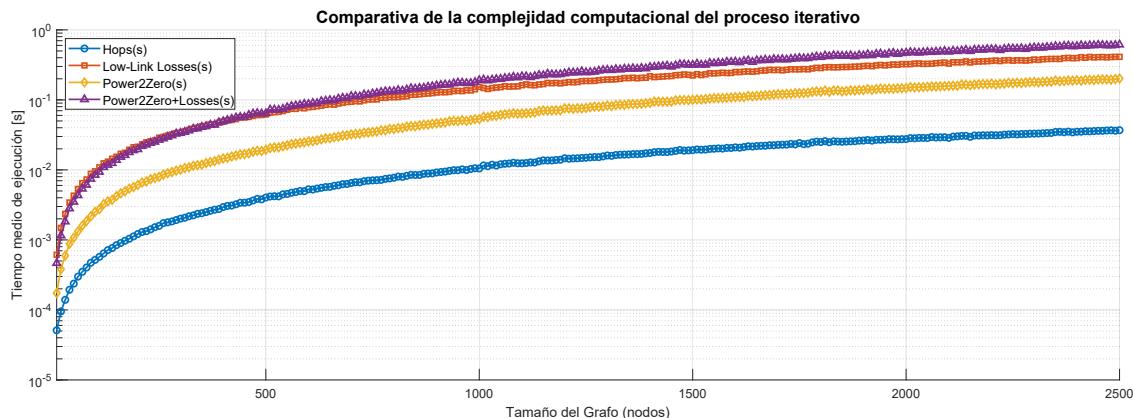


**Figura 7.27:** Complejidad computacional del mecanismo iterativo.

Estos resultados se representan gráficamente en la Figura 7.27, donde se indican las leyes de potencia ajustadas para cada criterio. El análisis permite extraer varias observaciones relevantes. En primer lugar, todos los exponentes  $\gamma$  se encuentran en el rango aproximado 1.17–1.36, lo que refleja un crecimiento ligeramente superlineal del tiempo de ejecución con respecto al tamaño de la topología. En la práctica, esto implica que duplicar el número de nodos incrementa el tiempo de ejecución en algo más del doble, aunque lejos de un comportamiento cuadrático.

En segundo lugar, se aprecian diferencias entre los criterios: el enfoque *Low-Link Losses* presenta el menor exponente (1.17), lo que sugiere un crecimiento más moderado en comparación con el resto, mientras que *Power2Zero* y *Power2Zero+Losses* alcanzan exponentes ligeramente superiores (1.34–1.36), sin embargo, la base de tiempos es significativamente menor para *Power2Zero*, y para *Power2Zero+Losses* es similar, debido a que requiere más iteraciones para converger en la topología. La Figura 7.28 ilustra los tiempos medios absolutos (doble promedio), donde se observa, además, un cruce característico entre el criterio de bajas pérdidas y el de *Power2Zero+Losses*. Este comportamiento se explica porque, a medida que aumenta el tamaño del grafo, el criterio de bajas pérdidas debe evaluar un mayor número de identificadores y vecinos, lo que incrementa gradualmente su coste computacional.

Finalmente, cabe destacar que, incluso en topologías de hasta 2500 nodos, los tiempos medios por criterio permanecen dentro de escalas reducidas, del orden de décimas a fracciones de segundo, lo que demuestra que el mecanismo iterativo, en su implementación actual, resulta computacionalmente viable para las dimensiones de red consideradas.



**Figura 7.28:** Comparativa de la complejidad computacional del mecanismo iterativo en tiempo con escala logarítmica.

## 7.4. Conclusiones

En este trabajo se ha presentado BLOSTE, un algoritmo que permite el encaminamiento adaptativo y en tiempo real de energía en redes de distribución malladas. BLOSTE aborda las principales limitaciones de los sistemas centralizados mediante el uso de superposiciones lógicas (*overlays*) iniciadas desde uno o varios nodos raíz, en combinación con estrategias de optimización multicriterio flexibles. Estas características confieren al sistema capacidad de operar bajo condiciones variables y con distintos objetivos de control, promoviendo así la escalabilidad, la capacidad de respuesta y la resiliencia del sistema.

La evaluación realizada sobre la topología de referencia IEEE 123-bus en tres escenarios (ideal, con pérdidas y con pérdidas y restricciones de capacidad en los enlaces), con más de 148 000 simulaciones, ha permitido llevar a cabo un análisis exhaustivo de los criterios de encaminamiento propuestos. Cada escenario incluyó 96 configuraciones de carga distintas y múltiples emplazamientos del nodo raíz, lo que permitió una comparación detallada. Entre los criterios evaluados, el criterio local *Power2Zero* se consolidó como la mejor opción en términos de balance global de potencia, alcanzando un desequilibrio medio de -4.1 % en promedio entre los diferentes nodos raíz. Aunque no es el criterio más rápido en términos de convergencia (4.02 ms frente a 1.40 ms del criterio de *Hop*), este compromiso queda compensado por su superior capacidad de balanceo. Además, pese a requerir aproximadamente el doble de iteraciones, la relación entre el tiempo de convergencia y el número de iteraciones se mantuvo en niveles de eficiencia similares, lo que confirma su viabilidad práctica en escenarios de operación en tiempo real.

Los resultados obtenidos se han validado adicionalmente en la topología IEEE 34-bus, bajo las mismas condiciones de evaluación. Los patrones observados en la topología de mayor tamaño se reprodujeron de manera consistente, confirmando la solidez del comportamiento de los criterios bajo distintas estructuras de red. En particular, se observó que, en topologías con menor grado de mallado, los criterios de *Hop* y *Low-Link Losses* tien-

den a ofrecer un desempeño más competitivo en escenarios con restricciones de capacidad, mientras que *Power2Zero* mantiene su superioridad en escenarios menos restringidos. Esta consistencia entre topologías refuerza la generalidad y robustez de los resultados obtenidos.

Adicionalmente, se ha llevado a cabo un estudio detallado de la complejidad computacional del algoritmo, abarcando tanto el mecanismo de etiquetado como el proceso iterativo. Los análisis muestran que la fase de etiquetado (`spread_ids`) escala de forma prácticamente lineal, con un comportamiento comparable al de algoritmos clásicos como MST o GA, pero con la ventaja de generar múltiples árboles en lugar de uno único. Por su parte, el mecanismo iterativo presenta un crecimiento ligeramente superlineal en todos los criterios, con exponentes en el rango 1.17–1.36, lo que garantiza tiempos de ejecución de apenas décimas a fracciones de segundo incluso para topologías de hasta 2500 nodos. Estos resultados demuestran que BLOSTE no solo es efectivo desde el punto de vista del balance de potencia, sino también computacionalmente viable y escalable para redes de distribución de gran tamaño.

Finalmente, cabe destacar la relevancia de disponer de *datasets* de microredes para la mejora continua tanto de los mecanismos de predicción como de los criterios de selección. La disponibilidad de topologías más diversas y de conjuntos de datos abiertos contribuiría significativamente al entrenamiento y refinamiento del sistema. Aunque este estudio ha incorporado un grado considerable de heterogeneidad mediante la evaluación de las topologías IEEE 123-bus y IEEE 34-bus, así como mediante topologías aleatorias de gran escala, resultará necesario ampliar las pruebas a escenarios de red más variados y complejos. Dado que los criterios de decisión son configurables, futuras investigaciones deberían explorar cómo la parametrización de los mismos afecta al rendimiento en distintos tipos de redes, consolidando así la aplicabilidad de BLOSTE como una herramienta versátil y robusta para la gestión energética en redes de distribución malladas.



## Capítulo 8

# Propuesta de arquitectura inteligente y softwarizada enfocada al IIoT

En este capítulo se presentan las contribuciones principales asociadas al segundo bloque de objetivos de la Tesis, centradas en el diseño de una arquitectura IIoT inteligente y softwarizada. La propuesta se articula sobre marcos de orquestación unificados que integran salidas de inferencia en tiempo real basadas en AI/ML, obtenidas de la telemetría de los sensores, con el fin de habilitar reconfiguraciones dinámicas de red, reubicación eficiente de servicios y mantenimiento preventivo.

Esta línea de trabajo tuvo su origen en el proyecto 6G-DATADRIVEN-02 (en colaboración con la Universidad Carlos III de Madrid) [195], donde se establecieron los fundamentos conceptuales de la arquitectura. Posteriormente, se desarrolló una primera prueba de concepto (PoC), presentada en la conferencia ICIN 2025 [11], que mostró la viabilidad de la aproximación propuesta mediante un producto mínimo viable basado en virtualización clásica. Más adelante, durante la estancia doctoral en la Universidad de Milán-Bicocca, la arquitectura evolucionó hacia un despliegue orientado a microservicios y contenedores, integrando mecanismos de inferencia en el plano de control y ampliando los conjuntos de datos utilizados para evaluar la capacidad de autorreconfiguración en escenarios IIoT realistas. Estas pruebas incluyeron evaluaciones de rendimiento y resiliencia, que confirmaron la validez, flexibilidad y adaptabilidad de la arquitectura propuesta. Fruto de este trabajo se ha derivado una publicación [196], actualmente en proceso de revisión, que consolida esta línea de investigación como un paso clave hacia el diseño de ecosistemas industriales inteligentes, distribuidos y softwarizados.

### 8.1. Introducción

La evolución del IIoT ha puesto de manifiesto la necesidad de arquitecturas capaces de integrar inteligencia distribuida, baja latencia y mecanismos de adaptación dinámica en escenarios industriales cada vez más complejos y heterogéneos. Como se revisó en el

Capítulo 2, Sección 2.3.2, las soluciones existentes presentan avances relevantes en el uso de *edge/fog computing* y arquitecturas softwareizadas, pero persisten retos fundamentales relacionados con la escalabilidad, la interoperabilidad y la ausencia de implementaciones abiertas y reproducibles que materialicen los principios propuestos.

En este contexto, se plantea una arquitectura *data-driven* que opera a lo largo del continuo *cloud-edge-IIoT*, diseñada para cerrar la brecha entre propuestas conceptuales y despliegues prácticos. La arquitectura se fundamenta en tres ejes principales: (i) la adopción de microservicios contenerizados y orquestados mediante plataformas *cloud*-nativas, que facilitan elasticidad y despliegue modular; (ii) la integración de un plano de control dinámico que aprovecha inferencias en tiempo real de modelos ML para activar reconfiguraciones de red, reubicación de servicios y mantenimiento preventivo; y (iii) el empleo de estándares abiertos y marcos interoperables que habilitan su aplicación en escenarios heterogéneos y de alta densidad de nodos. La propuesta se valida a través de pruebas de concepto que han evolucionado desde entornos basados en virtualización clásica hasta despliegues orientados a microservicios gestionados por Kubernetes, incluyendo evaluaciones con múltiples conjuntos de datos y condiciones representativas de escenarios industriales. Los resultados obtenidos demuestran reducciones significativas de latencia, mejoras en la capacidad de respuesta ante condiciones cambiantes y una mayor resiliencia frente a cargas variables de sensores y servicios.

Con ello, la arquitectura *data-driven* contribuye a los objetivos de la Tesis al proporcionar una base práctica y reproducible para el diseño de sistemas IIoT inteligentes, escalables y transparentes, alineados con las visiones de redes distribuidas e impulsadas por datos en el horizonte del 6G.

## 8.2. Descripción de la arquitectura

Nuestra arquitectura se construye sobre el marco general de diseño establecido por el proyecto 6G-DATADRIVEN-02 [195], inicialmente validado a través de la prueba de concepto [11]. La arquitectura está diseñada para operar de forma nativa y continua a lo largo de todo el continuo *cloud-edge-IIoT*. El objetivo principal es aprovechar los datos generados por dispositivos industriales heterogéneos (sensores, actuadores y Gateways (GWs)) para habilitar procesamiento localizado con tiempos de respuesta mínimos, así como adaptabilidad y reconfiguración de red, además de una coordinación colaborativa entre múltiples instalaciones y una visión global unificada en la nube. La Figura 8.1 muestra la arquitectura general del sistema.

La figura ilustra una arquitectura multinivel que habilita despliegues de IIoT a lo largo del continuo *cloud-edge-factory*. En el nivel inferior, la arquitectura comienza en el *factory floor*, donde los dispositivos IIoT (p.ej., brazos robóticos, sistemas de transporte, terminales de control) se conectan mediante GWs. Estos GWs recopilan flujos de datos provenientes de sensores y actuadores dentro de las fábricas y los transmiten hacia niveles superiores. Tecnologías de acceso de red como 5G, 6G o WiMAX interconectan estas pa-

sarelas (GWs) con una plataforma en el borde (en el *edge*), donde un módulo de control supervisa el enrutamiento del tráfico y la orquestación. Los flujos de datos se almacenan de manera selectiva en las bases de datos (Databases (DBs)) distribuidas a nivel de *edge*. Este plano de control en el *edge* gestiona dinámicamente múltiples plantas industriales, optimizando el uso de la red y el rendimiento de las aplicaciones IIoT. En niveles superiores, la plataforma en la nube aloja módulos Function-as-a-Service (FaaS) que incorporan canalizaciones de servicios de inferencias AI/ML. Estos módulos consumen datos desde las DBs del *edge* para ejecutar analíticas predictivas, como detección de fallos o control adaptativo, reduciendo la latencia y la sobrecarga computacional en la planta. Los módulos de monitorización proporcionan observabilidad y permiten a los usuarios recibir retroalimentación en tiempo real a lo largo de todos los niveles.

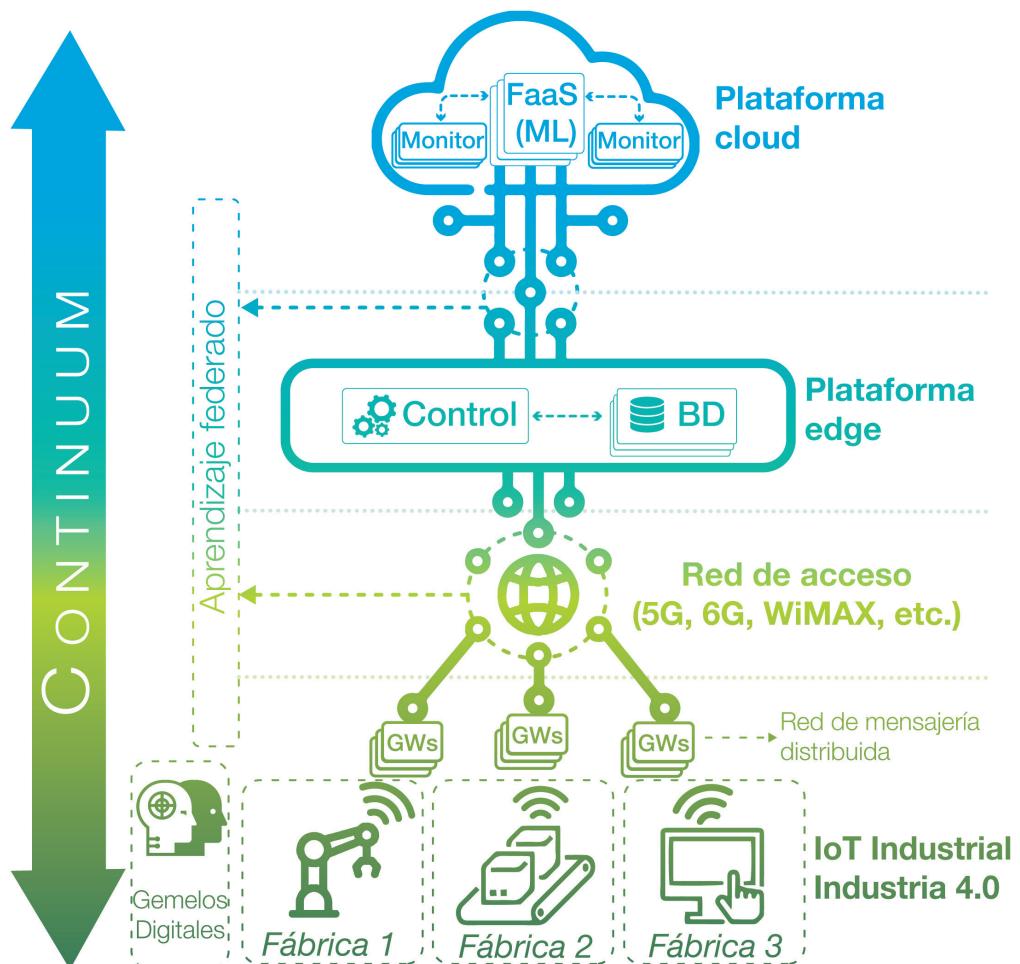


Figura 8.1: Visión general de la arquitectura del sistema.

La arquitectura soporta *federated learning* (FL), donde el conocimiento derivado de una planta puede contribuir a modelos globales, garantizando inteligencia compartida entre instalaciones. La plataforma en la nube consolida los resultados de todas las fábricas,

proporcionando una visión operacional unificada y habilitando que el módulo de control en el *edge* aplique políticas globales de ubicación de servicios, enrutamiento y asignación de recursos. Además, la arquitectura incorpora varios servicios fundamentales para reforzar la resiliencia y la extensibilidad en todos los niveles. Una red distribuida de mensajería desacopla las pasarelas IIoT de los consumidores, permitiendo que nuevos módulos de inferencia o control se suscriban a los flujos de sensores sin interrumpir los flujos de datos existentes. Cada planta industrial mantiene un gemelo digital local que refleja de manera continua sus activos físicos, y participa en la capa de *federated learning*, mejorando así la precisión de las predicciones. Finalmente, uno de los puntos más importantes es la observabilidad extremo a extremo (mediante registros, métricas y trazas) proporciona visibilidad en todo el continuo, soportando monitorización proactiva y planificación estratégica de largo plazo.

En conjunto, esta arquitectura continua, softwareizada y orientada a datos, habilita:

- Reconfiguración automática de redes IIoT basada en el estado operativo de cada planta.
- Procesamiento distribuido y de baja latencia en el borde, garantizando acciones reactivas inmediatas.
- Colaboración federada entre fábricas para aprendizaje conjunto y transferencia de conocimiento.
- Visión operacional unificada, control global y optimización continua desde la nube y el borde.

### 8.3. Implementación y despliegue de la arquitectura

En esta sección se detallan la implementación y el despliegue de la arquitectura introducida en la Sección 8.2. Tal y como se señaló previamente, la arquitectura está concebida de manera fundamentalmente orientada a los datos. En consecuencia, la Sección 8.3.1 comienza con un análisis del conjunto de datos empleado para simular las lecturas de sensores IIoT en el entorno de la fábrica, junto con los modelos de aprendizaje automático utilizados para tareas de clasificación. Esta subsección se introduce en primer lugar, dado que ciertos aspectos clave basados en datos, como la noción de estado de eficiencia de los sensores y su papel en el desencadenamiento de eventos de reconfiguración de la red, resultan esenciales para comprender la lógica y el comportamiento de los componentes descritos posteriormente.

La Sección 8.3.2 describe la implementación de los componentes funcionales básicos de la arquitectura. Finalmente, la Sección 8.3.3 presenta el despliegue basado en Kubernetes, el cual adopta un enfoque orientado a microservicios que garantiza escalabilidad, resiliencia y flexibilidad a lo largo del continuo fábrica–*edge*–cloud.

### 8.3.1. Análisis del conjunto de datos y entrenamiento del modelo de clasificación

El funcionamiento de la arquitectura está impulsado por los datos generados a partir de los sensores IIoT. Ante la falta de acceso a un entorno industrial físico, se recurre a un conjunto de datos IIoT de acceso público que contiene lecturas temporales de sensores y actuadores. El conjunto de datos seleccionado, disponible en Kaggle [197], es sometido a un análisis estadístico con el fin de extraer las características relevantes, las cuales se utilizan posteriormente para simular el comportamiento de los sensores en el despliegue. A partir de dichas características, se entrena un modelo de clasificación capaz de detectar estados anómalos en los nodos, activando los mecanismos de reconfiguración de red cuando resulte necesario.

#### 8.3.1.1. Análisis de conjuntos de datos IIoT

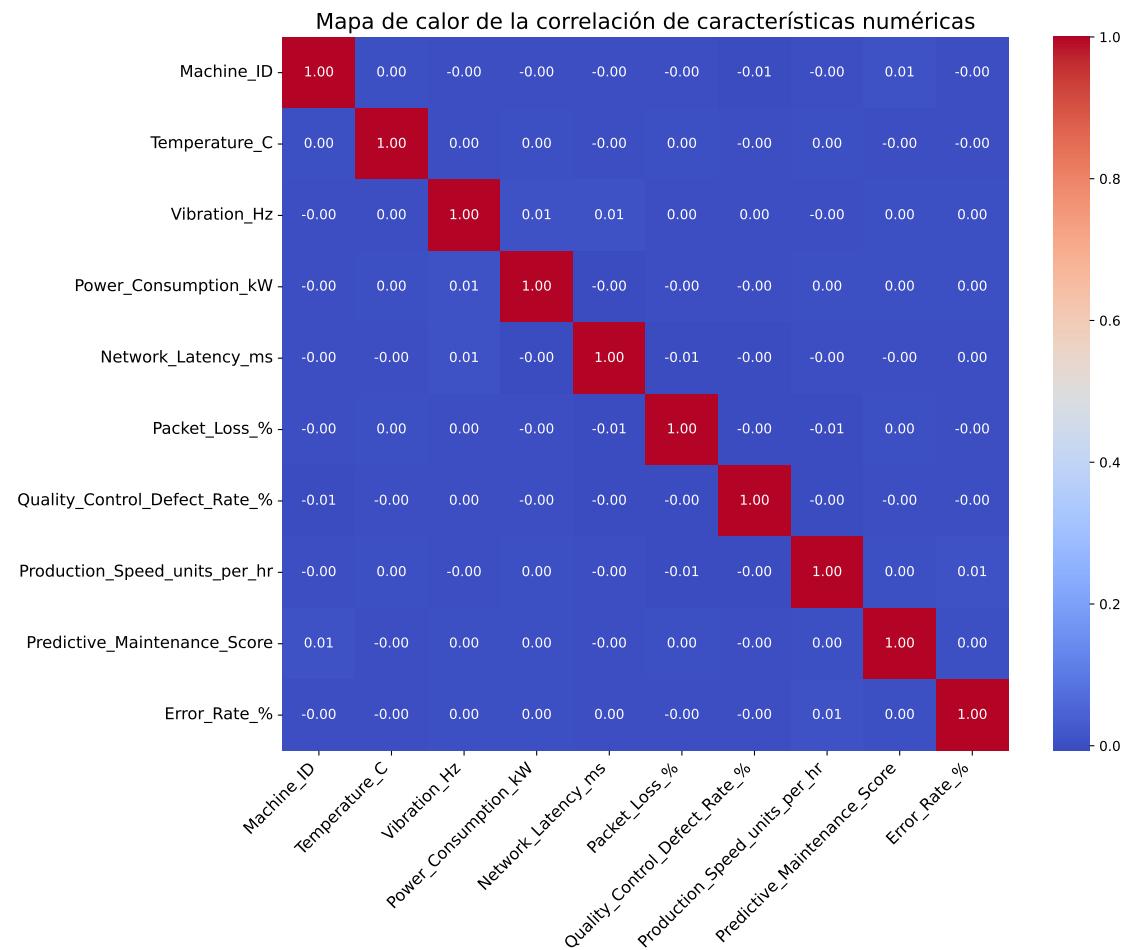
El conjunto de datos seleccionado incluye mediciones completas en forma de series temporales de procesos industriales. El dataset recopila datos físicos de sensores (temperatura, vibración, consumo energético), indicadores de rendimiento de red (latencia, pérdida de paquetes, eficiencia en la comunicación), así como métricas de producción (tasas de defectos, errores operativos). Dicho conjunto, nos permite evaluar la eficiencia de las máquinas considerando simultáneamente las lecturas de sensores, las condiciones de red y los indicadores de desempeño productivo. La Tabla 8.1 resume las características principales del conjunto de datos.

Característica	Tipo de dato	Descripción
Timestamp	Datetime	Marca temporal de cada lectura.
Machine_ID	Integer	Identificador único de cada máquina IIoT.
Operation_Mode	String	Modo de operación de cada máquina IIoT.
Temperature_C	Float	Temperatura (°C).
Vibration_Hz	Float	Vibración (Hz).
Power_Consumption_kW	Float	Consumo energético (kW).
Network_Latency_ms	Float	Latencia de red (ms).
Packet_Loss_%	Float	Pérdida de paquetes (%).
Quality_Ctrl_Defect_Rate	Float	Tasa de defectos en control de calidad (%).
Prod_Speed_units_per_hr	Float	Velocidad de producción (unidades/hora).
Predict_Maintenance_Score	Float	Predicción de mantenimiento (%).
Error_Rate_%	Float	Tasa de errores (%).
Efficiency_Status	String	Clasificación de la eficiencia de fabricación basada en métricas de rendimiento.

**Tabla 8.1:** Características principales del conjunto de datos.

El conjunto de datos se ha sometido a un análisis estadístico para extraer las características relevantes, las cuales se emplean posteriormente para emular el comportamiento de los sensores en nuestro despliegue. Un análisis estructural del conjunto de datos, basado en la matriz de correlación (véase la Figura 8.2), revela que las características numéricas listadas en la Tabla 8.1 presentan una correlación lineal muy baja entre sí. Esta ausencia

de redundancia entre características resulta ventajosa, ya que sugiere que cada característica aporta información distinta y no solapada entre diferentes características del conjunto de datos.

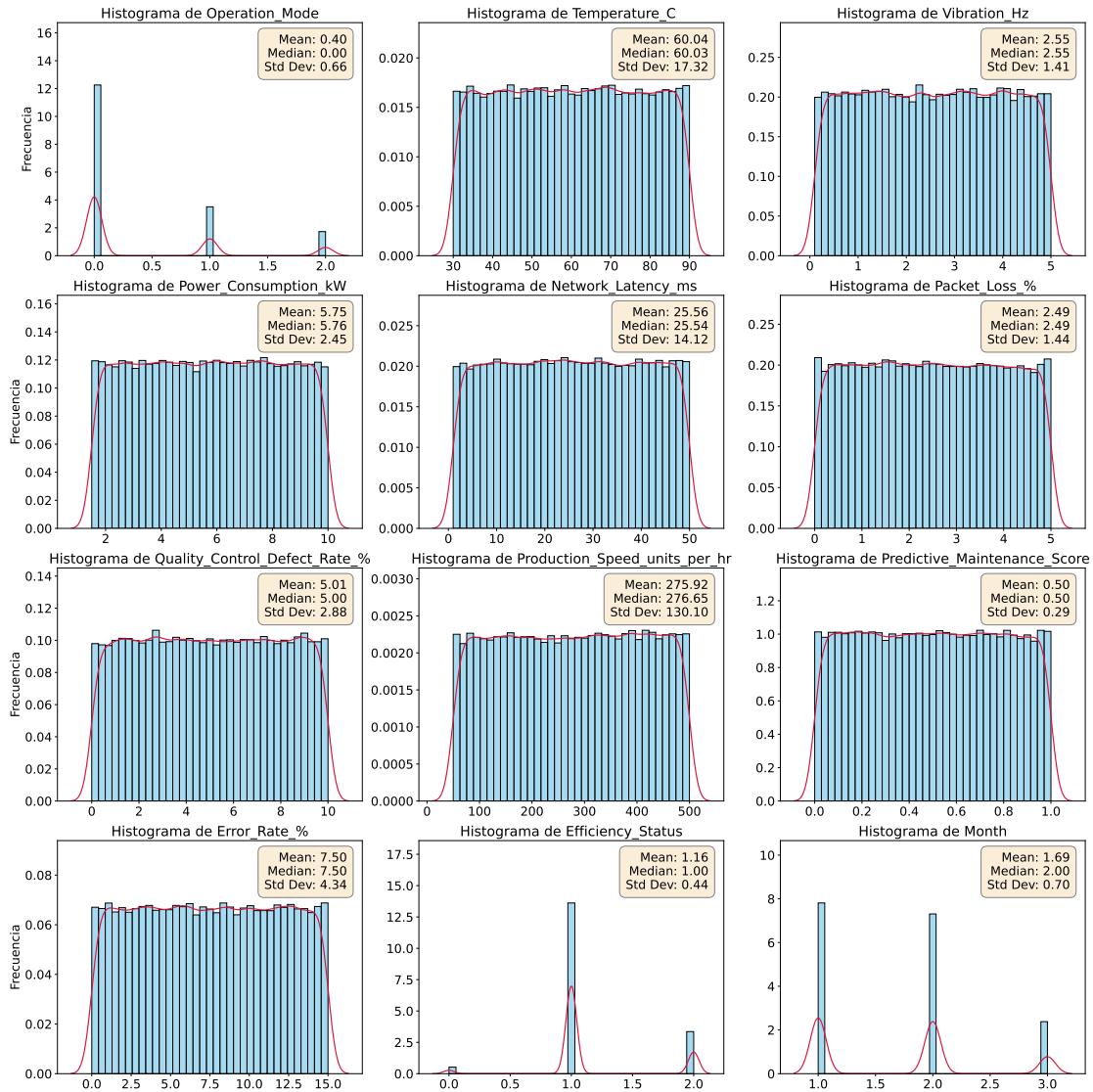


**Figura 8.2:** Mapa de calor de la correlación de las características numéricas del dataset.

Un análisis estadístico adicional de las distribuciones de características muestra que todas las variables numéricas siguen una distribución aproximadamente uniforme (véase la Figura 8.3). Este aspecto resulta especialmente relevante para nuestra arquitectura, ya que respalda la extracción de estadísticas descriptivas con el fin de simular datos de sensores realistas en ausencia de hardware físico.

En contraste, las variables categóricas, `Operation_Mode` y `Efficiency_Status`, presentan distribuciones tipo delta, en las que ciertas categorías están notablemente sobrerepresentadas (como se ilustra en la Figura 8.3). Además, el conjunto de datos exhibe irregularidades temporales, puesto que el atributo `Month` muestra una fuerte concentración de registros entre los meses de enero, febrero y marzo.

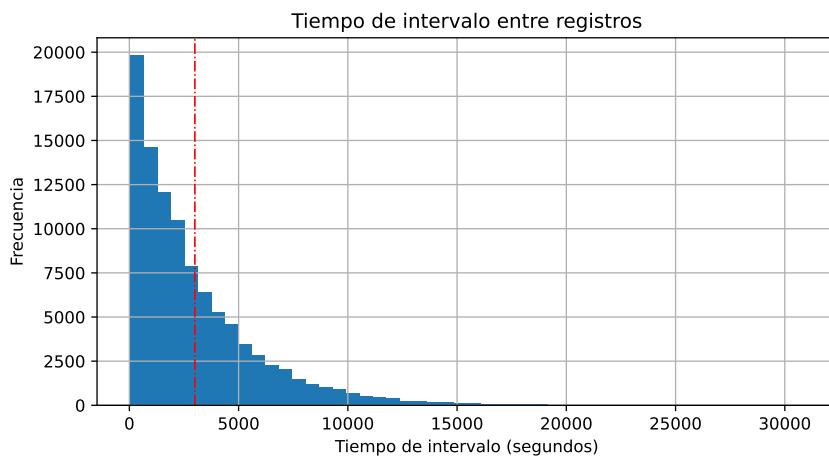
### 8.3 IMPLEMENTACIÓN Y DESPLIEGUE DE LA ARQUITECTURA



**Figura 8.3:** Histogramas de las distribuciones y estadísticas principales para cada característica del conjunto de datos.

Para lograr una simulación realista de los datos, se analizaron los intervalos de tiempo entre lecturas consecutivas de los sensores. El histograma de estos intervalos, mostrado en la Figura 8.4, revela una distribución sesgada a la derecha, con la mayoría de los intervalos concentrados en valores bajos. Sin embargo, el conjunto de datos también contiene brechas esporádicas de hasta 8,5 horas (aproximadamente 514 minutos o 30.840 segundos). El intervalo de tiempo medio es de aproximadamente 2.998,40 segundos (alrededor de 50 minutos), mientras que la mediana se sitúa en 2.100,00 segundos. La diferencia entre la media y la mediana confirma la prevalencia de intervalos más cortos debido al sesgo de la distribución. Asimismo, la elevada desviación estándar indica una variabilidad considera-

ble en los tiempos de reporte de los sensores. Para fines de implementación, este intervalo medio ha sido normalizado a un intervalo configurable de simulación de 1 segundo, preservando los patrones temporales relativos y permitiendo pruebas en tiempo real de la arquitectura.



**Figura 8.4:** Intervalo de tiempo en segundos entre las marcas de tiempo de los registros.

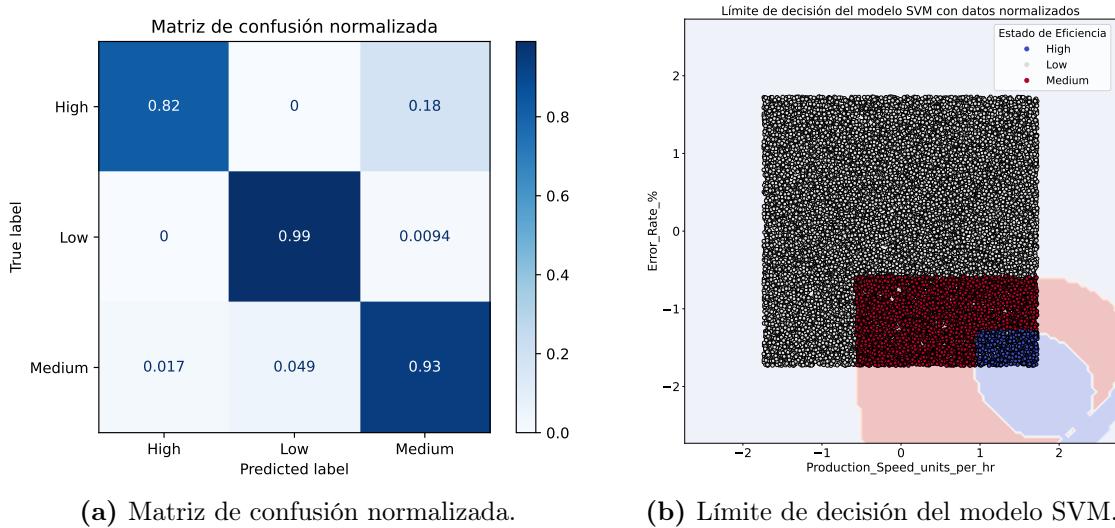
### 8.3.1.2. Entrenamiento del modelo de clasificación del estado de eficiencia de los sensores IIoT

Como se describió en la Sección 8.2, la capa del *cloud* incluye un módulo FaaS encargado de ejecutar funciones de inferencia que determinan si un sensor está operando de manera anómala. Para respaldar esta funcionalidad, se entrenó un modelo de ML utilizando el atributo **Efficiency\_Status** disponible en el conjunto de datos, el cual etiqueta el comportamiento de los sensores en tres clases: baja, media y alta eficiencia. Se seleccionó un clasificador SVM debido a su alto rendimiento en la tarea. El modelo entrenado alcanzó valores de precisión del 97,5 %, demostrando una elevada capacidad para diferenciar entre los distintos niveles de eficiencia.

Aunque podrían haberse explorado modelos adicionales, el enfoque de este trabajo no reside en optimizar el rendimiento de ML, sino en diseñar una arquitectura capaz de integrar diversos modelos de AI/ML según sea necesario para distintos casos de uso.

La Figura 8.5a muestra la matriz de confusión normalizada, la cual confirma la capacidad del clasificador para distinguir con precisión entre las tres clases de eficiencia con una mínima tasa de error. Asimismo, la Figura 8.5b visualiza las fronteras de decisión derivadas del modelo SVM normalizado, entrenado específicamente con las características **Production\_Speed\_units\_per\_hr** y **Error\_Rate\_%**. Este gráfico ilustra claramente la separación de las clases de eficiencia en el espacio de características: la región de alta eficiencia se concentra en el cuadrante inferior derecho (caracterizado por una alta veloci-

dad de producción y una baja tasa de error), mientras que las instancias de baja eficiencia se agrupan en el cuadrante superior izquierdo (baja velocidad y alta tasa de error). La clase media ocupa las zonas intermedias, reflejando un comportamiento transicional. Esta inspección visual valida la efectividad del clasificador SVM en la delimitación de estados operativos relevantes para activar reconfiguraciones dentro de la arquitectura.



**Figura 8.5:** Visualización del rendimiento del modelo: matriz de confusión y frontera de decisión.

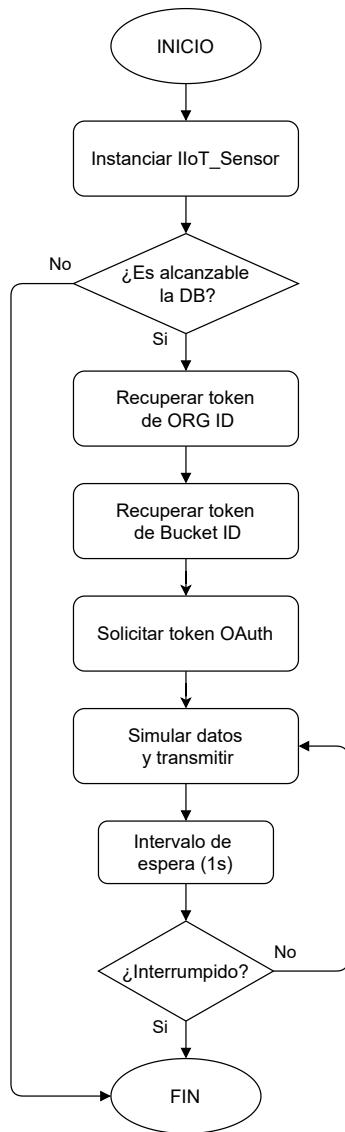
### 8.3.2. Implementación de bloques funcionales básicos

Esta sección describe la implementación de todos los bloques funcionales que conforman la arquitectura. La explicación comienza en el nivel más bajo con los componentes de la planta de producción (denominado como *Factory Floor*), continúa con los módulos funcionales desplegados en la capa del *edge*, en particular, el bloque de control y la base de datos, y se concluye con los componentes alojados en la plataforma en la nube, incluyendo el sistema de monitorización y el módulo FaaS encargado de ejecutar el servicio de inferencia.

#### 8.3.2.1. Bloque de *Factory Floor*

La planta de producción constituye la capa fundamental de la arquitectura, englobando los dispositivos IIoT y la infraestructura inalámbrica que generan y transmiten datos. Su implementación está concebida para ser adaptable a los requisitos específicos del caso de uso industrial final. No obstante, debido a la falta de acceso a un entorno real de factoría, todo el sistema se emula mediante Mininet-WiFi [155]. Esta herramienta permite la emulación de redes inalámbricas definidas por software, ofreciendo soporte para la configuración de nodos con recursos limitados y enlaces con capacidad variable, lo que posibilita la creación de un banco de pruebas IIoT altamente realista. En este escenario, se implementa una topología inalámbrica de referencia compuesta por tres puntos de

acceso (del inglés, Access Point (AP)) programables y compatibles con OpenFlow, cada uno conectado a un número configurable de estaciones inalámbricas. Los puntos de acceso se enlazan con el módulo de control, que gestiona dinámicamente el tráfico mediante la emisión de reglas de flujo que determinan el encaminamiento de los paquetes, mientras que las estaciones inalámbricas actúan como sensores IIoT individuales.



**Figura 8.6:** Diagrama de flujo que representa el ciclo de vida de la instancia IIoT\_Sensor, incluyendo la inicialización, los procedimientos de autenticación y la transmisión periódica de datos al módulo Base de datos.

Cada sensor IIoT ejecuta una aplicación aislada basada en Python dentro de su propio espacio de nombres de red (*Network Namespace*). Esta aplicación desempeña dos funciones

principales: autenticar con el servicio de base de datos en el *edge*, y simular y transmitir datos sintéticos de los sensores en función del análisis estadístico previamente presentado en la Sección 8.3.1.

La Figura 8.6 muestra el diagrama de flujo que describe el comportamiento emulado de los sensores IIoT. El proceso se inicia con la inicialización de los simuladores correspondientes a cada característica, basados en los parámetros estadísticos extraídos anteriormente. Tras esta fase, el sistema verifica la conectividad con el servicio de base de datos desplegado en el *edge*. Una vez establecida la conexión, el sensor obtiene los identificadores de organización y planta (asignados a *buckets* de datos específicos) mediante un intercambio de autenticación basado en OAuth2 con el agente de base de datos en el *edge*. Con la autorización concedida, el sensor recibe un token de escritura de fina granularidad que habilita la inserción de datos únicamente en los *buckets* autorizados. Posteriormente, el sensor entra en un bucle en el que simula y transmite valores de datos a intervalos de tiempo fijos (1 segundo, configurable) hasta que el proceso es interrumpido.

### 8.3.2.2. Bloque de Control

El módulo de control es responsable de gestionar el plano de control de las redes de los puntos de acceso (AP) en la planta IIoT, así como de recuperar datos de los sensores IIoT a través del módulo de base de datos en la capa del *edge*. Además, establece canalizaciones de inferencia con el módulo FaaS desplegado en la nube, con el fin de determinar si algún sensor presenta un comportamiento anómalo. Cuando se identifican dichas anomalías, el módulo de control activa acciones de reconfiguración en el AP correspondiente para optimizar el rendimiento de la red.

La implementación utiliza Ryu [7] como controlador principal por dos razones fundamentales: (1) proporciona una API sur (Sección 2.1.1) compatible con el protocolo OpenFlow, lo que permite una configuración dinámica del plano de control de los APs; (2) su diseño orientado a aplicaciones facilita la creación de nuevos perfiles de control que encapsulan la lógica de decisión, incluyendo interacciones con el módulo de base de datos, solicitudes de inferencia remota al servicio en la nube y reconfiguración dinámica de la red IIoT. La lógica implementada en Ryu se detalla en el Algoritmo 3.

La lógica de control se fundamenta en un bucle de decisión estrechamente integrado que combina monitorización en tiempo real, inferencia basada en ML y aplicación de políticas de control. Cuando un AP reenvía un paquete al controlador (mediante un evento `PacketIn`), el controlador consulta a la base de datos las lecturas recientes asociadas al sensor correspondiente. En la implementación actual, los datos recuperados son promedios agregados en una ventana temporal de 10 minutos, aunque este valor es configurable. Dichos datos se envían como petición en formato JSON a la API del módulo FaaS, lo que desencadena un proceso de inferencia remota mediante el modelo de clasificación descrito previamente. El servicio FaaS devuelve una etiqueta que indica si el sensor debe considerarse eficiente, ineficiente o en transición. El resultado se registra en un diccionario interno de sensores, que permite llevar un seguimiento del estado operativo de cada sensor

y apoyar la toma de decisiones centralizada. Si un sensor es clasificado como ineficiente, el controlador inicia una acción de reconfiguración. En el prototipo actual, esto implica des-asociar el sensor de su AP actual, permitiendo que se reconecte a otro punto de acceso con un rendimiento potencialmente superior. Este comportamiento demuestra la viabilidad de una reconfiguración dinámica de la red basada en ML, que puede ser extendida o personalizada para distintos casos de uso en entornos IIoT.

---

**Algoritmo 3:** Pseudocódigo de alto nivel del controlador Ryu.

---

**Input:** Eventos OpenFlow, paquetes IIoT  
**Output:** Control dinámico de flujos y detección de anomalías

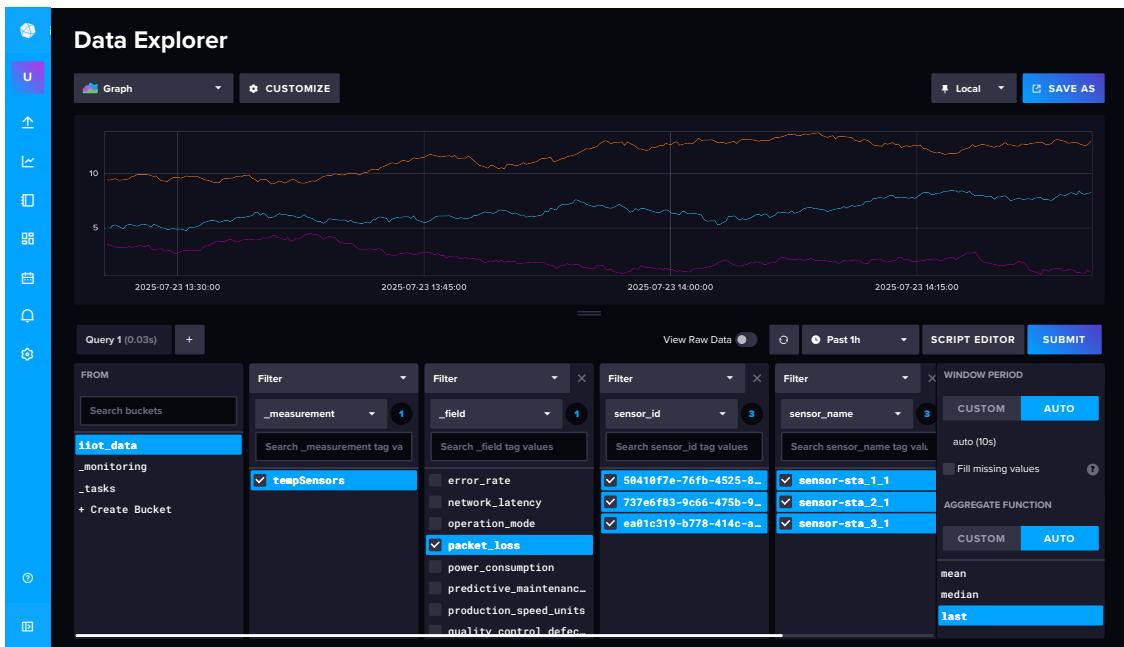
- 1 Inicializar servicios de Base de Datos e Inferencia
- 2 Inicializar tabla MAC y lista de sensores
- 3 **Al conectar un AP:** establecer regla `table-miss` para enviar paquetes al controlador
- 4 **Al recibir un evento packet-in:**
  - 5 Extraer datapath, puerto de entrada e información del paquete
  - 6 Aprender direcciones MAC origen y destino
  - 7 Actualizar el mapeo MAC–puerto
  - 8 Registrar sensores si aplica
  - 9 **if** el paquete es ARP **then**
    - 10 Establecer puerto de salida en FLOOD
  - 11 **else**
  - 12 Determinar puerto de salida a partir de la tabla MAC
- 13 **if** el paquete es IPv4 y corresponde a un sensor **then**
  - 14 **if** el sensor está asociado a un AP y no marcado como caído **then**
    - 15 Consultar métricas al módulo de BD
    - 16 Realizar inferencia mediante la API FaaS
    - 17 **if** se detecta anomalía **then**
      - 18 Marcar sensor como caído
      - 19 Descartar paquete
  - 20 **else if** el sensor estaba previamente marcado como caído pero ahora es normal **then**
    - 21 Desmarcar sensor
- 22 Instalar regla de flujo si el destino es conocido
- 23 Reenviar paquete según corresponda

---

### 8.3.2.3. Bloque de Base de datos

InfluxDB [198] ha sido seleccionada para la implementación del módulo de base de datos, al tratarse de una base de datos orientada a series temporales específicamente optimizada para la ingestión y consulta de datos en tiempo real. Esta elección se fundamenta en varios factores críticos para los sistemas de monitorización IIoT: su alta eficiencia en

el manejo de operaciones intensivas de escritura, su soporte nativo para agregaciones en ventanas temporales y su integración inmediata con plataformas como Grafana. En particular, InfluxDB permite el almacenamiento de lecturas periódicas de sensores con marcas temporales precisas y soporta consultas agregadas con una latencia mínima. La Figura 8.7 presenta la interfaz de usuario de InfluxDB, mostrando un ejemplo de los datos de series temporales almacenados en la base de datos. Dichos datos reflejan la evolución temporal de diferentes variables registradas por los sensores IIoT.



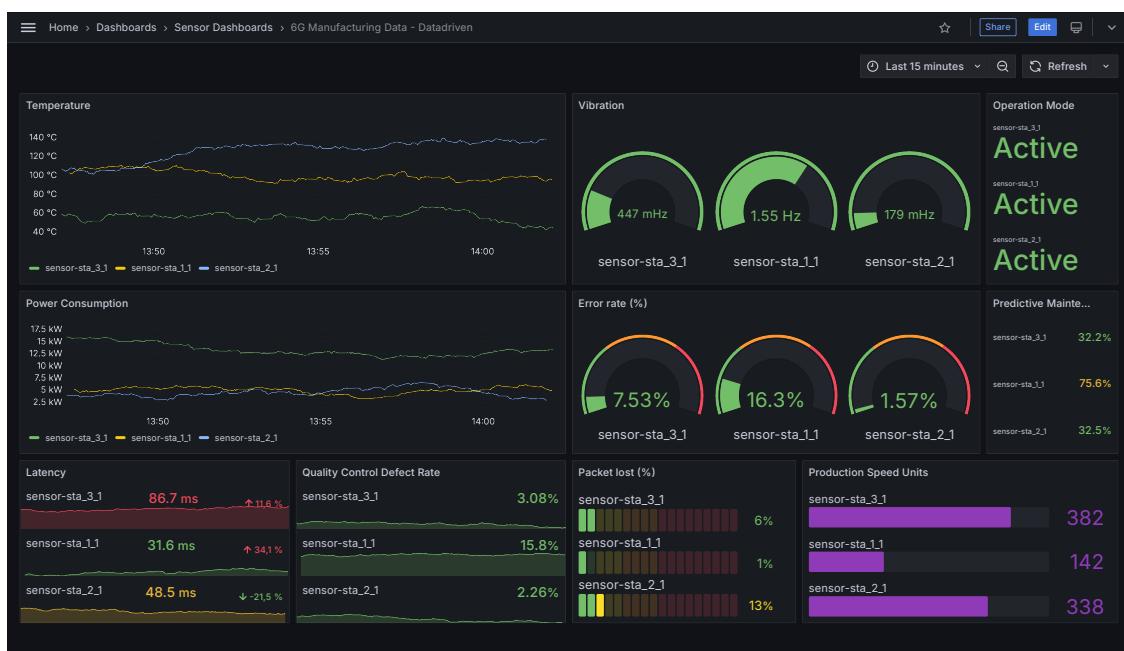
**Figura 8.7:** Frontend de InfluxDB configurado con las series temporales de los sensores IIoT de una organización.

#### 8.3.2.4. Bloque de Inferencias

En la capa de la nube, el módulo FaaS aloja el servicio de inferencias que permite al módulo de control desencadenar reconfiguraciones de la red IIoT. Aunque el diseño sigue una filosofía inspirada en FaaS, es decir, desplegar servicios de inferencia sin estado y bajo demanda, no se utiliza una plataforma serverless dedicada como AWS Lambda u OpenFaaS. En su lugar, el servicio se despliega como un microservicio contenedorizado gestionado mediante Kubernetes. Para ello se emplea BentoML [199], que permite empaquetar el modelo de ML desarrollado en la Sección 8.3.1.2 y exponerlo a través de una API RESTful. La elección de BentoML se debe a sus funcionalidades avanzadas de serialización, versionado y despliegue de modelos, así como a su soporte nativo para entornos de inferencia escalables. Este enfoque garantiza una actualización transparente de modelos y una alta disponibilidad de los puntos de inferencia bajo cargas variables, manteniendo al mismo tiempo una baja latencia.

### 8.3.2.5. Bloque de Monitorización

El último bloque funcional de la arquitectura corresponde al módulo de monitorización. Para este propósito se ha seleccionado Grafana [200], debido a su integración directa con InfluxDB y a sus amplias capacidades de visualización. Gracias a las funcionalidades de los Dashboards y al ecosistema de complementos de Grafana, los operadores pueden construir paneles interactivos en tiempo real que muestran métricas clave de IIoT, tales como lecturas de sensores, latencia de red y estado del sistema, obtenidas directamente de InfluxDB. La Figura 8.8 ilustra la interfaz de usuario configurada, la cual proporciona a los operadores de planta una visión integral y en tiempo real de la topología completa de la red IIoT y de su rendimiento operativo.



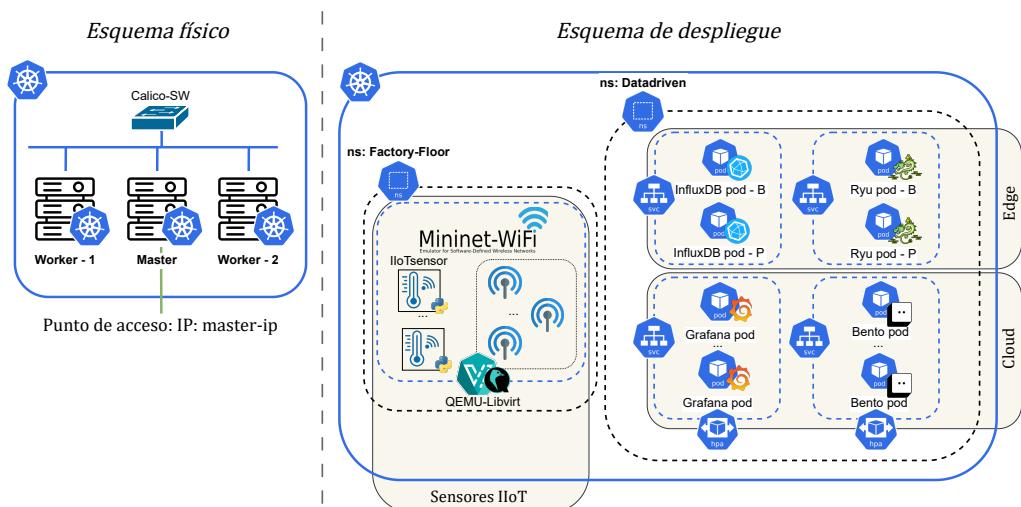
**Figura 8.8:** Panel de control del frontend de Grafana configurado con toda la información de los sensores IIoT.

### 8.3.3. Despliegue de arquitectura en Kubernetes

Esta sección describe la arquitectura completa de despliegue utilizada para integrar todos los componentes del sistema IIoT propuesto, siguiendo un enfoque de microservicios basado en Kubernetes. La contenerización constituye la estrategia de despliegue preferida para sistemas complejos compuestos por múltiples servicios ligeros e independientes que deben interactuar para proporcionar una funcionalidad cohesiva. Al aislar cada bloque lógico como un contenedor autónomo, el sistema se vuelve más modular, portable y sencillo de escalar o actualizar de manera independiente. Kubernetes amplía este modelo mediante características nativas como autoescalamiento, balanceo de carga, tolerancia a fallos, almacenamiento persistente a través de Persistent Volume Claims (PVCs) y comunicación trans-

parente entre servicios. Si bien herramientas más simples como Docker Compose permiten orquestar despliegues básicos, los entornos de producción suelen requerir plataformas más avanzadas como Kubernetes, OpenShift, Nomad o Rancher. Estos orquestadores ofrecen funcionalidades críticas a nivel empresarial, tales como comprobaciones de estado, alta disponibilidad, descubrimiento de servicios y despliegue declarativo. La motivación para adoptar este modelo con Kubernetes radica en su capacidad para proporcionar modularización, escalabilidad, tolerancia a fallos y una gestión detallada de servicios, aspectos difíciles de lograr en despliegues sobre hardware dedicado o con orquestadores más simples.

Para soportar el despliegue, se configuró un clúster de Kubernetes de tres nodos (utilizando `containerd` como motor de contenedores) mediante la herramienta oficial `kubeadm`, que ofrece un método simplificado para inicializar y gestionar despliegues de Kubernetes. Como se muestra en la parte izquierda de la Figura 8.9, el clúster está compuesto por: (1) un nodo maestro que ejecuta el plano de control de Kubernetes. Este nodo es responsable de gestionar el estado del clúster, incluyendo la planificación de cargas de trabajo, la distribución de configuraciones y la exposición de APIs para la administración de servicios. El nodo maestro no ejecuta servicios de aplicación, sino que orquesta los despliegues en todo el clúster. (2) Dos nodos de trabajo, que alojan los componentes reales de la aplicación. Estos nodos ejecutan los servicios contenerizados. Kubernetes organiza los contenedores en unidades lógicas denominadas pods, que representan la unidad mínima desplegable. Cada pod suele ejecutar uno o varios contenedores estrechamente acoplados que comparten recursos de red y almacenamiento. En nuestro despliegue, cada componente central (p. ej., base de datos, controlador, FaaS) se ejecuta en su propio pod, gestionado por el planificador de Kubernetes. Para garantizar la conectividad entre pods de distintos nodos, se emplea Calico como complemento Container Network Interface (CNI). Calico proporciona una red superpuesta segura y un mecanismo de aplicación de políticas para la comunicación entre pods, tanto dentro de un mismo nodo como entre nodos distintos.



**Figura 8.9:** Descripción física y lógica del despliegue del clúster.

El esquema lógico de despliegue se muestra en la parte derecha de la Figura 8.9. En él se ilustra cómo la infraestructura física (nodos del clúster) se mapea hacia los entornos virtuales (namespaces), servicios y pods. La arquitectura se encuentra lógicamente dividida en dos namespaces (*ns*): **Factory-floor** y **Datadriven**. El namespace **Factory-floor** emula el entorno IIoT mediante una máquina virtual desplegada a través de KubeVirt y QEMU-Libvirt. Esta máquina virtual ejecuta Mininet-WiFi, dentro del cual se instancian topologías de red inalámbricas y se simulan sensores IIoT. Estos sensores se comportan de acuerdo con lo descrito en la Sección 8.3.2.1.

Por su parte, **Datadriven** aloja todos los servicios del *edge* y *cloud* necesarios para la toma de decisiones y la monitorización. Entre estos se incluyen los servicios de borde, el controlador Ryu y la base de datos InfluxDB, desplegados en configuración activo-respaldo para proporcionar resiliencia, así como el servicio de inferencia BentoML y los paneles de Grafana, ambos configurados con Horizontal Pod Autoscaler (HPA), lo que permite escalar dinámicamente en función del uso de CPU y memoria para mantener el rendimiento bajo cargas variables. Un resumen de los recursos hardware asignados para ejecutar la infraestructura se recoge en la Tabla 8.2.

Componente	Especificaciones
Nodos del clúster de Kubernetes	16 GB RAM; 8 vCPUs; formato Blade
Máquina virtual con Mininet-WiFi	4 GB RAM; 4 vCPUs

**Tabla 8.2:** Especificaciones de los recursos hardware asignados al despliegue.

### 8.3.3.1. Mecanismos de escalabilidad y tolerancia a fallos

Para habilitar un escalado adaptativo bajo cargas de trabajo variables, se configuran HPAs para los servicios de Grafana y BentoML. Cada HPA mantiene un mínimo de una y un máximo de cinco réplicas de pod. El escalado se activa automáticamente cuando la utilización de CPU supera el 70 %, garantizando que los servicios permanezcan receptivos durante los picos de carga. Los límites de recursos de CPU se definen con un mínimo de 100 millicores y un máximo de 500 millicores por pod.

Para los componentes con estado, como InfluxDB y Ryu, no se emplea HPA. En su lugar, estos servicios se despliegan con redundancia activa-en-espera (del inglés, *active-standby*) para proporcionar tolerancia a fallos y asegurar la continuidad del servicio. Se aprovisionan dos pods para cada componente: uno actúa como nodo primario, gestionando tanto las operaciones de lectura como de escritura, mientras que el segundo permanece en modo de espera. Las solicitudes de escritura se replican en ambas instancias para mantener la consistencia. En caso de fallo del nodo primario, la instancia en espera se promueve automáticamente a activa, asumiendo la totalidad de las responsabilidades operativas con una mínima interrupción del sistema. El servicio nativo de Kubernetes gestiona el balanceo de carga entre réplicas, enruteando automáticamente el tráfico hacia todos los pods activos. Los pods creados recientemente son etiquetados e integrados en la malla de servicios

sin necesidad de reconfiguración. Este enfoque híbrido, que combina escalado automático para servicios sin estado y redundancia manual para aquellos con estado, garantiza tanto la elasticidad del rendimiento como la alta disponibilidad a lo largo de la arquitectura desplegada.

### 8.3.3.2. Flujo de datos entre componentes de la arquitectura

Tras la descripción del clúster de Kubernetes y del despliegue lógico, se examina ahora el flujo operativo de datos entre los principales componentes del sistema, ilustrado en la Figura 8.10.

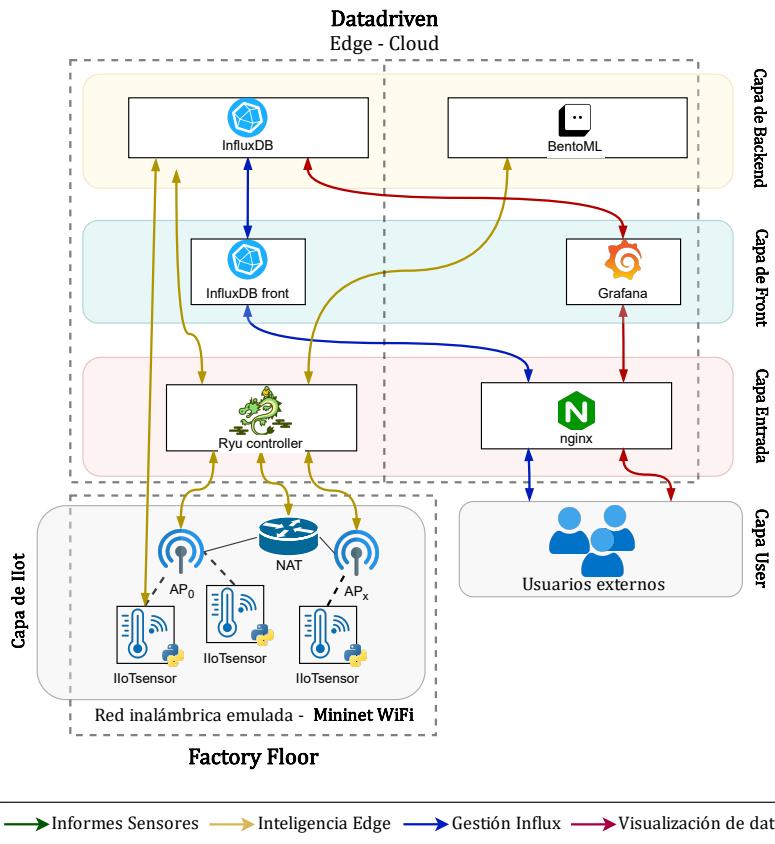


Figura 8.10: Flujos de datos entre componentes de la arquitectura.

En la parte inferior izquierda del diagrama, la capa IIoT representa el entorno de fábrica emulado mediante Mininet-WiFi, que instancia una red mallada inalámbrica habilitada con SDN, compuesta por nodos sensores virtuales IIoT (denotados como IIoTSensor) y AP. Cada sensor genera y transmite mediciones periódicas, tal y como se indicó en la Sección 8.3.2.1. Los paquetes generados atraviesan la red SDN, alcanzan un punto de acceso (AP) y son enrutados a través de un gateway NAT hacia el controlador Ryu en la

capa de entrada. El controlador Ryu procesa los mensajes procedentes de la capa IIoT y habilita la evaluación de la eficiencia de los sensores, así como la monitorización de la red en tiempo real, consultando dinámicamente dos servicios: (1) InfluxDB, para recuperar métricas recientes de los sensores, y (2) BentoML, para invocar la canalización de inferencia que determina si un sensor se encuentra en un estado anómalo. En función de los resultados de estas interacciones, el controlador puede activar acciones de reconfiguración, como disociar un sensor de su AP actual y redirigirlo hacia otro con mejor rendimiento.

De manera paralela, los usuarios externos interactúan con el sistema a través de la capa *Users*. Todas las solicitudes HTTP entrantes son encaminadas mediante un proxy Nginx, que actúa como único punto de acceso externo al sistema. Desde allí, el proxy reenvía el tráfico hacia dos servicios de *front-end*: (1) Grafana, que consulta periódicamente InfluxDB para generar los paneles en tiempo real con los datos de los sensores; y (2) la interfaz web de InfluxDB, que permite a los administradores del sistema inspeccionar, consultar y gestionar directamente las series temporales de datos.

Este diseño de flujo de datos multinivel permite la monitorización continua, la inferencia inteligente y la observabilidad por parte de los usuarios, manteniendo al mismo tiempo la separación arquitectónica entre dispositivos emulados, lógica de control y servicios orientados al usuario.

## 8.4. Evaluación experimental

En esta sección se presenta la evaluación experimental de la arquitectura propuesta, organizada en tres subsecciones: (i) *Utilización de recursos y escalabilidad*, donde se analizan las métricas de CPU, memoria y tiempo de despliegue en función de la carga del sistema y del número de sensores IIoT; (ii) *Latencia de inferencia e ingestión de datos*, que mide el tiempo promedio de inferencia y la latencia de ingestión de datos en el módulo de base de datos utilizando la herramienta `hey`; y (iii) *Tiempo promedio de reconfiguración de red*, en el que se estudia el retardo medio de reconfiguración de la red a medida que varía el número de sensores IIoT. Todas las pruebas se llevaron a cabo en un servidor DELL PowerEdge R650 equipado con dos procesadores Intel Xeon Silver-4310 a 2.10 GHz (18 MB de caché) y 512 GB de memoria RAM DDR4. El objetivo de esta evaluación experimental es cuantificar el rendimiento de la arquitectura y evaluar su flexibilidad y escalabilidad, así como su capacidad de adaptarse a diferentes requisitos de QoS y de reconfigurarse de manera autónoma en respuesta a condiciones operativas cambiantes.

### 8.4.1. Utilización de recursos y escalabilidad

En esta subsección se evalúa el rendimiento y la escalabilidad de la arquitectura propuesta, centrándose exclusivamente en los componentes de *Edge* y *Cloud*. La capa de *Factory floor* depende en gran medida de los requisitos específicos de cada caso de uso y puede variar sustancialmente en función del entorno operativo. Tal y como se explicó en secciones anteriores, en nuestra implementación, la capa de *Factory floor* se emula mediante una máquina virtual que ejecuta Mininet-WiFi, reproduciendo una topología

inalámbrica SDN con nodos IIoT con recursos limitados. Sin embargo, para los fines de este análisis, restringimos el alcance de la evaluación a los bloques funcionales principales ubicados en los niveles de *Edge* y *Cloud* de la arquitectura. Comenzamos analizando el tiempo de despliegue de todo el sistema, desglosado en sus distintas fases. La Tabla 8.3 presenta el tiempo medio de ejecución de cada etapa, junto con su intervalo de confianza (IC), repitiendo el despliegue completo 20 veces. El tiempo medio total de despliegue asciende aproximadamente a  $269.15 \pm 1.69$  segundos.

Etapa	Tiempo (s)
Dashboard	$2.85 \pm 0.17$
Destrucción de despliegues	$4.70 \pm 0.22$
Proveedor de máquinas virtuales	$10.10 \pm 0.34$
Configuración del nodo maestro	$39.95 \pm 1.30$
Configuración del servidor de métricas	$3.05 \pm 0.10$
Configuración de <i>namespaces</i>	$2.70 \pm 0.22$
Despliegue de <i>datadriven</i>	$5.15 \pm 0.49$
Instalación de <i>containerd</i> y Kubernetes	$198.65 \pm 1.36$
Asociación del <i>worker-1</i>	$1.00 \pm 0.00$
Asociación del <i>worker-2</i>	$1.00 \pm 0.00$
Total	$269.15 \pm 1.69$

**Tabla 8.3:** Tiempo de despliegue (media  $\pm$  IC) por cada etapa.

La fase más costosa en tiempo es la instalación del entorno de ejecución de contenedores y del clúster de Kubernetes, que incluye la configuración de *containerd* y de los componentes esenciales de Kubernetes. Este paso requiere aproximadamente  $198.65 \pm 1.36$  segundos, lo que representa cerca del 74 % del tiempo total de despliegue. A continuación, la configuración del nodo *maestro* requiere alrededor de  $39.95 \pm 1.30$  segundos, incluyendo la inicialización del clúster y la configuración del plano de control. Las fases posteriores, como el aprovisionamiento de máquinas virtuales ( $10.10 \pm 0.34$  s), la configuración de *namespaces* ( $2.70 \pm 0.22$  s), y el despliegue de los servicios *datadriven* ( $5.15 \pm 0.49$  s), aportan una contribución marginal al tiempo total. La adición de nodos *worker* es prácticamente instantánea, con solo un segundo requerido por cada uno, lo que pone de manifiesto la eficiencia del proceso de escalado horizontal una vez que el clúster está operativo. Estos resultados confirman que, a pesar de la complejidad del sistema, el proceso de despliegue es eficiente en tiempo y repetible, lo que permite un restablecimiento rápido en entornos dinámicos o multiusuario.

Tras el análisis del tiempo de despliegue, examinamos los patrones de consumo de memoria de los bloques funcionales ubicados en los niveles de *Edge* y *Cloud*. La Tabla 8.4 muestra el uso promedio de memoria, expresado en MiB, junto con los correspondientes intervalos de confianza, para cada servicio principal, BentoML, Grafana, InfluxDB y Ryu, en función del número de estaciones IIoT (STAs) por AP. Como era de esperar, BentoML presenta el crecimiento más significativo en el consumo de memoria conforme aumenta la

carga de la red. Partiendo de 204.00 MiB con una sola STA, su uso alcanza un máximo cercano a 1005.46 MiB para 9 STAs por AP. A partir de este punto, el consumo de memoria fluctúa moderadamente, pero se mantiene dentro de un rango operativo estable entre aproximadamente 750 y 985 MiB. Estos resultados confirman que BentoML escala con el número de peticiones de inferencia concurrentes, aunque el escalado horizontal mitiga la saturación de recursos, tal y como se describió en la configuración de autoescalado.

Grafana mantiene un uso de memoria relativamente estable en todas las configuraciones evaluadas. Con una línea base de  $91.43 \pm 0.63$  MiB, su consumo aumenta ligeramente hasta alcanzar  $101.78 \pm 0.90$  MiB con 17 STAs, antes de disminuir cuando se generan menos operaciones de renderizado en escenarios de mayor carga, posiblemente debido a limitaciones de actualización o mecanismos de *throttling* de visualización (Una técnica que limita la velocidad a la que se ejecuta una función de visualización durante un periodo de tiempo específico, garantizando que las operaciones se realicen a un ritmo controlado y constante, en lugar de sobrecargar el sistema). Se observa una caída brusca tras las 23 STAs, probablemente causada por optimizaciones internas del panel o efectos de saturación.

InfluxDB muestra un incremento gradual y consistente en el uso de memoria a medida que aumenta el número de STAs. Comenzando en  $88.76 \pm 0.31$  MiB con una STA, alcanza  $131.57 \pm 1.14$  MiB con 29 STAs. Este comportamiento refleja la tendencia esperada en bases de datos de series temporales bajo un mayor volumen de operaciones de escritura y consulta, sin mostrar indicios de degradación de rendimiento o fugas de memoria.

STAs/AP	BentoML	Grafana	InfluxDB	Ryu
1	$204.00 \pm 0.00$	$91.43 \pm 0.63$	$88.76 \pm 0.31$	$56.00 \pm 0.00$
3	$371.42 \pm 12.20$	$93.85 \pm 0.32$	$90.19 \pm 0.49$	$56.00 \pm 0.00$
5	$870.93 \pm 22.28$	$95.09 \pm 0.39$	$92.17 \pm 0.52$	$56.00 \pm 0.00$
7	$985.79 \pm 26.21$	$98.42 \pm 0.90$	$96.65 \pm 0.46$	$56.00 \pm 0.00$
9	$1005.46 \pm 19.87$	$97.51 \pm 0.81$	$100.49 \pm 0.48$	$56.00 \pm 0.00$
11	$995.21 \pm 21.21$	$96.17 \pm 0.59$	$104.32 \pm 0.50$	$56.00 \pm 0.00$
13	$754.61 \pm 33.64$	$89.65 \pm 1.93$	$106.57 \pm 0.64$	$56.10 \pm 0.05$
15	$939.22 \pm 29.45$	$99.89 \pm 0.60$	$111.89 \pm 0.65$	$56.12 \pm 0.05$
17	$883.30 \pm 27.66$	$101.78 \pm 0.90$	$116.83 \pm 0.64$	$56.12 \pm 0.05$
19	$753.59 \pm 34.64$	$101.27 \pm 0.67$	$121.23 \pm 0.61$	$56.10 \pm 0.04$
21	$922.75 \pm 30.98$	$98.84 \pm 0.48$	$123.76 \pm 0.67$	$56.11 \pm 0.05$
23	$978.33 \pm 29.24$	$85.62 \pm 2.12$	$124.17 \pm 0.55$	$56.23 \pm 0.09$
25	$985.62 \pm 29.87$	$67.03 \pm 0.02$	$124.97 \pm 0.94$	$56.41 \pm 0.12$
27	$867.90 \pm 27.61$	$67.00 \pm 0.00$	$128.22 \pm 1.04$	$56.21 \pm 0.08$
29	$940.72 \pm 33.25$	$67.00 \pm 0.00$	$131.57 \pm 1.14$	$56.22 \pm 0.08$

**Tabla 8.4:** Consumo de memoria (MiB) promedio ( $\pm$  IC) por servicio en función de las STAs por AP.

En contraste, Ryu demuestra un uso constante de memoria a lo largo de todas las pruebas, manteniéndose en torno a 56 MiB independientemente de la carga del sistema. Esta consistencia se debe a su naturaleza ligera como controlador SDN, cuya lógica de control permanece independiente del número de STAs una vez que la topología ha sido instanciada. En conjunto, estos resultados confirman la eficiencia en el uso de memoria y la escalabilidad de la arquitectura. Los servicios se comportan de forma predecible bajo condiciones de estrés, y sus perfiles de recursos validan la decisión de adoptar una estrategia de despliegue basada en microservicios.

La Tabla 8.5 presenta el uso promedio de CPU en milicores<sup>1</sup> para cada componente principal de la arquitectura, BentoML, Grafana, InfluxDB y Ryu, en función del número de nodos sensores IIoT. Como era de esperar, BentoML muestra un incremento pronunciado en el consumo de CPU a medida que aumenta el número de sensores IIoT, alcanzando su máximo en 25 STAs con aproximadamente  $301.29 \pm 26.98$  milicores. Esta tendencia pone de manifiesto la demanda computacional asociada a la ejecución de tareas de inferencia AI/ML conforme se ingiere un mayor volumen de datos. Sin embargo, a partir de este punto la carga desciende ligeramente, probablemente debido a efectos de balanceo de carga o *batching* de inferencias a nivel de servicio. InfluxDB también presenta un crecimiento sostenido en el consumo de CPU, con un incremento desde  $13.89 \pm 0.36$  milicores con una STA hasta un máximo de  $340.67 \pm 37.68$  milicores con 25 STAs. Este comportamiento es consistente con el esperado aumento en operaciones de escritura y volumen de consultas derivado de una mayor actividad de sensores.

Ryu, el módulo de control, muestra un incremento significativo en el uso de CPU entre 1 y 9 STAs, alcanzando  $168.85 \pm 8.74$  milicores, para después fluctuar de forma moderada. Esto indica que Ryu participa activamente en el bucle de control y en la gestión de eventos OpenFlow, en particular durante las reconfiguraciones de red y las decisiones de encaminamiento. En contraste, Grafana se mantiene relativamente ligero hasta alcanzar las 11 STAs, momento a partir del cual se observa una caída brusca en su consumo de CPU, estabilizándose en torno a 2.6 milicores para 25 o más STAs/AP. Este comportamiento se atribuye a mecanismos internos de optimización, almacenamiento en caché y al hecho de que Grafana únicamente renderiza paneles de forma periódica, reduciendo así su carga durante períodos de operación en estado estable.

Para profundizar en el comportamiento de escalado de BentoML, se analizó con mayor detalle el consumo de CPU en el escenario con 25 STAs por AP. Tal y como se ilustra en la Figura 8.11, se observa un pico inicial de uso de CPU debido a que una única instancia gestiona toda la carga entrante. No obstante, el sistema va generando progresivamente réplicas adicionales de BentoML en respuesta al aumento de demanda, distribuyendo eficazmente la carga. Este comportamiento refleja la activación del HPA y demuestra que, aunque la arquitectura escala con el número de dispositivos IIoT activos, los componentes presentan distintos grados de sensibilidad frente al incremento de la carga. En particular,

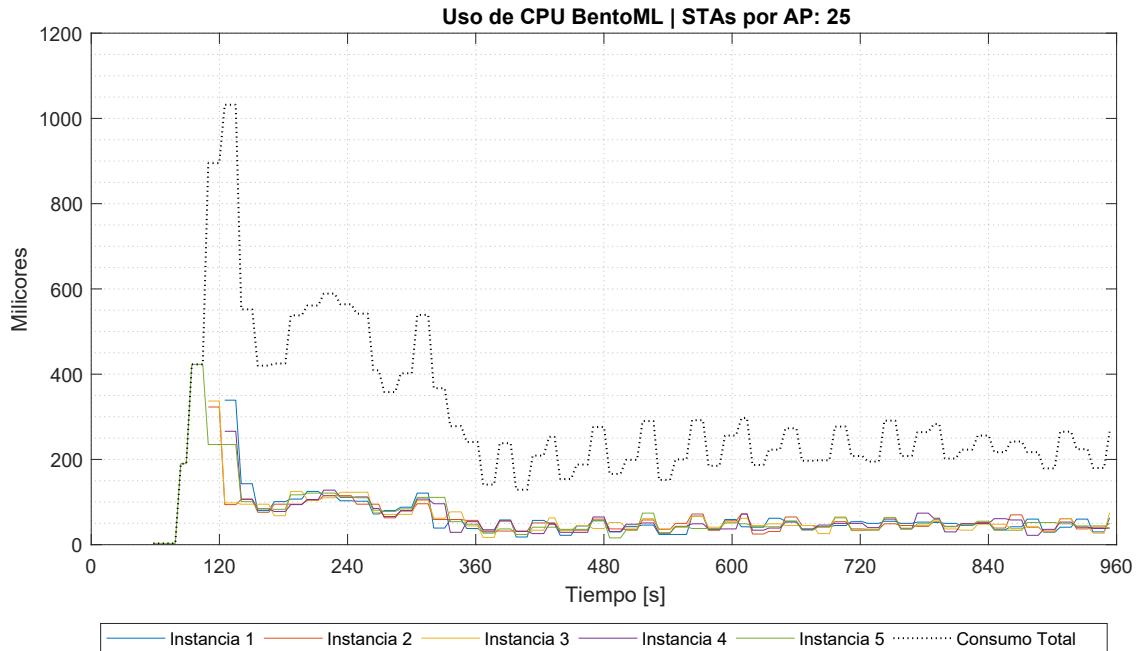
---

<sup>1</sup>Una unidad de medida de recursos de CPU que representa una milésima parte de un núcleo (1/1000), es decir, 1000 milicores equivalen a un núcleo completo.

BentoML e InfluxDB resultan ser los servicios más intensivos en CPU, lo que requiere especial atención en las estrategias de dimensionamiento y autoescalado.

STAs/AP	BentoML	Grafana	InfluxDB	Ryu
1	$8.43 \pm 0.53$	$11.78 \pm 0.27$	$13.89 \pm 0.36$	$6.57 \pm 0.60$
3	$59.31 \pm 7.20$	$25.45 \pm 1.43$	$46.25 \pm 2.31$	$41.65 \pm 3.98$
5	$178.19 \pm 12.43$	$40.61 \pm 1.50$	$89.91 \pm 2.93$	$105.05 \pm 7.55$
7	$298.41 \pm 17.18$	$56.78 \pm 1.27$	$141.19 \pm 4.53$	$176.31 \pm 12.10$
9	$295.90 \pm 30.00$	$66.61 \pm 1.23$	$170.73 \pm 5.57$	$168.85 \pm 8.74$
11	$228.97 \pm 23.22$	$71.13 \pm 1.29$	$178.81 \pm 7.99$	$133.68 \pm 11.82$
13	$176.43 \pm 18.55$	$57.09 \pm 4.99$	$172.62 \pm 11.41$	$118.10 \pm 10.73$
15	$257.53 \pm 21.87$	$80.34 \pm 1.52$	$253.61 \pm 15.26$	$157.68 \pm 10.47$
17	$199.18 \pm 24.89$	$87.55 \pm 1.46$	$276.98 \pm 22.19$	$125.38 \pm 11.99$
19	$135.35 \pm 24.65$	$87.31 \pm 1.84$	$263.97 \pm 27.62$	$86.77 \pm 11.05$
21	$235.98 \pm 26.71$	$80.83 \pm 1.74$	$333.00 \pm 30.25$	$143.01 \pm 10.19$
23	$244.67 \pm 22.94$	$36.77 \pm 5.42$	$337.63 \pm 43.43$	$148.53 \pm 11.18$
25	$301.29 \pm 26.98$	$2.55 \pm 0.12$	$340.67 \pm 37.68$	$177.85 \pm 11.75$
27	$198.55 \pm 27.70$	$2.65 \pm 0.11$	$319.08 \pm 46.02$	$118.63 \pm 9.19$
29	$240.38 \pm 19.27$	$2.63 \pm 0.11$	$310.49 \pm 31.55$	$155.17 \pm 8.24$

**Tabla 8.5:** Consumo de CPU (milicores) promedio ( $\pm$  IC) por servicio en función de STAs por AP.

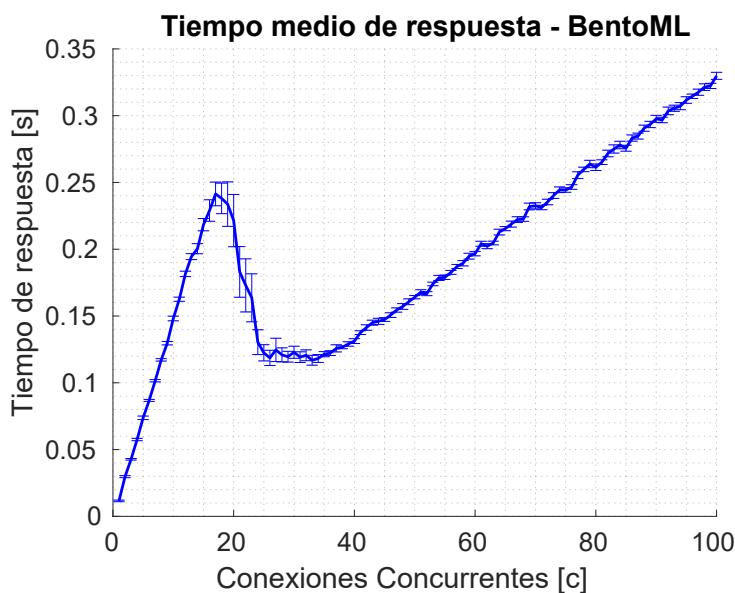


**Figura 8.11:** Uso de CPU de BentoML con escalado automático para 25 STA por AP.

### 8.4.2. Latencia de inferencia e ingestión de datos

En esta subsección se evalúa la latencia media de inferencia del servicio BentoML y la latencia media de acceso a datos de InfluxDB. Ambas pruebas se realizan utilizando la herramienta de *benchmarking* `hey`<sup>2</sup>, a través de un script personalizado que genera carga concurrente dirigida al servicio de inferencia de BentoML y a la interfaz de consulta de datos de InfluxDB.

El objetivo es analizar el impacto del autoescalado en BentoML y la capacidad de respuesta de InfluxDB, proporcionando así una visión global del rendimiento del sistema bajo cargas de trabajo realistas. Para ello, se emiten solicitudes HTTP concurrentes, aumentando progresivamente el número de conexiones simultáneas de 1 a 100. Cada lote de solicitudes concurrentes, denotado como  $C_n$ , se considera una caso experimental independiente. Por ejemplo, fijar  $C_n = 10$  implica el lanzamiento de 10 peticiones en paralelo que constituyen una ejecución experimental. Esta metodología es fundamental para garantizar la consistencia y fiabilidad estadística de los resultados; de lo contrario, mediciones fragmentadas o asincrónicas comprometerían su validez. Para cada prueba, se emiten  $10 \times C_n$  solicitudes, donde  $C_n \in \{1, \dots, 100\}$ . Tras cada lote, se establece un periodo de enfriamiento que permite al sistema retornar a su estado inactivo, específicamente, cuando BentoML opera con un único contenedor activo e InfluxDB presenta carga base. Cada escenario experimental se repite 25 veces, partiendo del contenedor del controlador Ryu (que normalmente desencadenaría dichos flujos de datos), manteniendo la topología de Mininet inactiva para eliminar interferencias. Los resultados promedio se muestran en las Figuras 8.12, 8.14, 8.13 y 8.15.

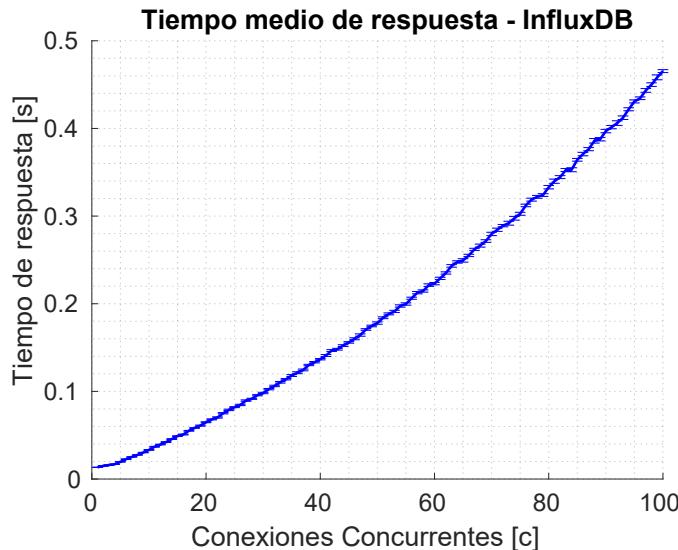


**Figura 8.12:** Tiempo medio de inferencia de BentoML en función de las solicitudes concurrentes.

<sup>2</sup><https://github.com/raky11/hey>

La Figura 8.12 presenta el tiempo medio de respuesta del servicio BentoML en función del número de solicitudes concurrentes. Se observa un incremento quasi-lineal en el tiempo de respuesta, alcanzando una latencia máxima de aproximadamente 350 ms bajo 100 solicitudes concurrentes. La figura permite distinguir tres fases. En la fase inicial, de 0 a 15 solicitudes concurrentes, solo un contenedor permanece activo, lo que produce la pendiente más pronunciada de latencia. Este retraso activa la política de autoescalado, iniciando el despliegue de réplicas adicionales de BentoML. La fase intermedia, de 15 a 25 solicitudes, muestra un incremento continuado en el tiempo de respuesta, mientras el autoescalador añade progresivamente réplicas hasta alcanzar el límite superior de cinco. Más allá de las 25 solicitudes, en la fase final, el tiempo de respuesta continúa creciendo, pero a un ritmo significativamente menor, reflejando el balanceo de carga entre múltiples réplicas y la amortización del coste computacional.

En contraste, la Figura 8.13 ilustra los resultados del mismo procedimiento experimental aplicado al servicio InfluxDB, enfocado en operaciones de lectura de la base de datos en lugar de tareas de inferencia. La latencia observada se mantiene notablemente estable a lo largo del experimento, con un máximo de aproximadamente 480 ms bajo condiciones de máxima concurrencia. Este comportamiento resulta especialmente destacable dado que InfluxDB no implementa mecanismos de autoescalado, una decisión de diseño orientada a preservar la consistencia de los datos. La figura revela una posible tendencia de crecimiento logarítmico en el tiempo de respuesta, lo cual indica un rendimiento escalable a pesar del aumento de carga.



**Figura 8.13:** Tiempo medio de ingestión de datos en InfluxDB en función de las solicitudes concurrentes.

Las dos figuras siguientes complementan el análisis previo incorporando los códigos de respuesta HTTP en la evaluación. Esta mejora permite identificar solicitudes exitosas y fallidas, considerando como fallo cualquier código de estado distinto de 200 OK. Además,

este estudio integra restricciones de QoS, examinando cómo se comporta el sistema bajo diferentes umbrales máximos de tiempo de respuesta ( $T_{\max}$ ), específicamente entre 0.1 y 0.5 segundos. Para estimar la probabilidad de bloqueo ( $P_b$ ) de cada servicio, se calcula la razón entre el número de solicitudes fallidas (aquellas con respuestas distintas de 200 o que superan el umbral  $T_{\max}$ ) y el número total de solicitudes. Los resultados para BentoML e InfluxDB se muestran en las Figuras 8.14 y 8.15, respectivamente. En la Figura 8.14, cada curva ilustra la evolución de  $P_b$  en función de la carga de solicitudes concurrentes bajo distintos niveles de QoS. El mecanismo de autoescalado de BentoML mitiga significativamente  $P_b$  una vez que se despliega más de una réplica. Por ejemplo, con  $T_{\max} = 0.2$  s,  $P_b$  desciende de 0.9 con 15 solicitudes concurrentes a menos de 0.02 cuando la carga se sitúa entre 25 y 40 solicitudes. Esto pone de manifiesto la eficacia del autoescalado para mantener la calidad del servicio ante una demanda creciente. Sin embargo, para umbrales de QoS más estrictos ( $T_{\max} = 0.1$  s), incluso cargas moderadas conducen a probabilidades de bloqueo elevadas, debido principalmente a la sobrecarga adicional introducida por el balanceador de carga dinámico y al tiempo computacional inherente requerido por cada inferencia, que, como se mostró anteriormente, puede alcanzar hasta 0.05 s incluso bajo cargas mínimas.

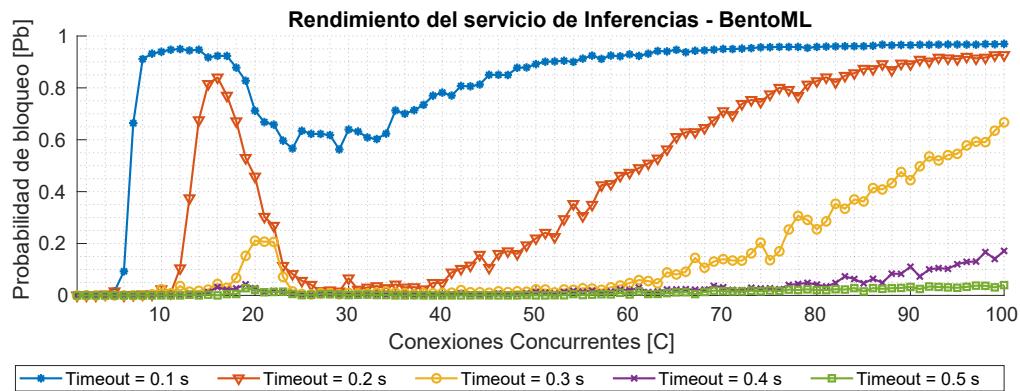


Figura 8.14: Probabilidad de bloqueo en BentoML bajo distintos umbrales de QoS.

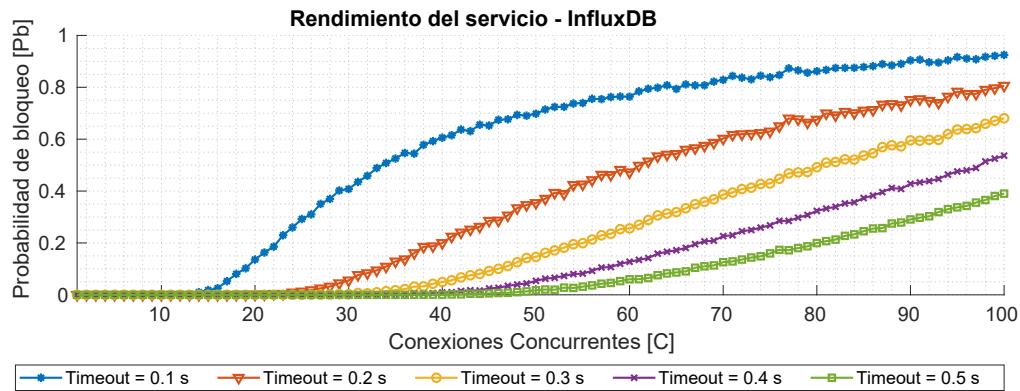


Figura 8.15: Probabilidad de bloqueo en InfluxDB bajo distintos umbrales de QoS.

En contraste, la Figura 8.15 presenta el análisis de la probabilidad de bloqueo para el servicio InfluxDB. En este caso, el comportamiento es más predecible y menos sensible a las variaciones de carga. Aunque no se emplea un mecanismo de autoescalado, la base de datos mantiene un rendimiento relativamente estable, y su probabilidad de bloqueo aumenta de forma gradual con la carga. Esto confirma la robustez de InfluxDB frente a cargas de consulta moderadas incluso en ausencia de escalabilidad elástica.

#### 8.4.3. Estimación del tiempo promedio de reconfiguración de red

Esta subsección presenta la estimación del tiempo promedio requerido para completar un ciclo de reconfiguración en la arquitectura propuesta. El proceso de reconfiguración es desencadenado por el controlador SDN, implementado con el framework Ryu, el cual reacciona ante comportamientos anómalos detectados en la red de sensores IIoT. Para cuantificar el tiempo de reconfiguración, se descompone el proceso en sus dos componentes principales: (i) el tiempo necesario para que Ryu acceda y recupere los datos de los sensores desde la base de datos de series temporales InfluxDB, y (ii) el tiempo requerido para enviar dichos datos al servicio de inferencia en la nube desplegado mediante BentoML y obtener un resultado de clasificación. Dado que ambos intercambios de comunicación (Ryu → InfluxDB y Ryu → BentoML) han sido previamente caracterizados en términos de sus tiempos de respuesta promedio bajo cargas concurrentes, se define el tiempo medio de reconfiguración, denotado  $T_{reconf}$ , como:

$$T_{reconf} = T_{read}^{Influx} + T_{infer}^{Bento} \quad (8.1)$$

donde  $T_{read}^{Influx}$  representa el tiempo promedio requerido para recuperar las lecturas más recientes de los sensores en InfluxDB, y  $T_{infer}^{Bento}$  denota el tiempo promedio de inferencia del servicio BentoML. Estas dos operaciones encapsulan las latencias dominantes en el bucle de reconfiguración.

Aunque se produce una etapa adicional de procesamiento dentro del controlador Ryu, responsable de componer la solicitud al servicio de inferencia e interpretar el resultado devuelto, este esfuerzo computacional local implica un número mínimo de operaciones en Python (p. ej., análisis sintáctico, verificación de condiciones), y su tiempo de ejecución resulta despreciable en comparación con la sobrecarga de comunicación en red y de inferencia. Esta consideración es aún más válida a medida que aumenta el número de estaciones IIoT, dado que las operaciones locales no escalan de forma proporcional al número de sensores. Por tanto, la estimación de  $T_{reconf}$  constituye una cota superior realista del tiempo promedio necesario para detectar una anomalía e iniciar una política de reconfiguración en la red, permitiendo una gestión receptiva y escalable en entornos industriales dinámicos.

### 8.5. Conclusiones

Este trabajo ha presentado una arquitectura novedosa, totalmente contenerizada, que combina recursos en el *edge* y en el *cloud* para habilitar una toma de decisiones adaptati-

va y basada en datos en entornos de IIoT. La arquitectura propuesta aborda limitaciones críticas identificadas en el estado del arte, tales como la ausencia de integración extremo a extremo a lo largo del continuo IIoT-*edge-cloud*, la falta de mecanismos de reconfiguración dinámica basados en analítica en línea, y la limitada escalabilidad y preparación para despliegue de los prototipos existentes.

A través de una evaluación experimental exhaustiva, se han obtenido varios hallazgos clave. En primer lugar, el sistema demuestra capacidades de despliegue rápido, alcanzando plena operatividad en menos de cinco minutos. En segundo lugar, la arquitectura exhibe un uso estable y eficiente de los recursos, con patrones predecibles de consumo de memoria y CPU en los distintos microservicios. Cabe destacar que el servicio de inferencia basado en BentoML se beneficia significativamente de la autoescalabilidad horizontal, manteniendo tiempos de respuesta bajos incluso bajo altas cargas de peticiones concurrentes. Asimismo, InfluxDB asegura un acceso a los datos de baja latencia en todos los experimentos, lo que confirma su idoneidad para tareas de monitorización continua en escenarios IIoT. Con ello, es posible estimar el tiempo promedio de reconfiguración de red, obtenido a partir de la combinación de los tiempos de acceso a la base de datos y de respuesta del servicio de inferencia. Los resultados indican que el sistema es capaz de sostener ciclos de reconfiguración inferiores al segundo, lo cual resulta esencial para la adaptación en tiempo real en despliegues industriales. Además, el análisis de la probabilidad de bloqueo bajo distintos umbrales de QoS pone de manifiesto las ventajas del escalado elástico, ya que reduce sustancialmente la probabilidad de degradación del servicio bajo condiciones de carga elevada.

Más allá de su validación técnica, este trabajo introduce una arquitectura disruptiva que cubre vacíos existentes en la literatura al integrar inferencia, telemetría y control SDN en un marco unificado y reproducible, liberando todo el sistema y los conjuntos de datos como *opensource*. En consecuencia, la arquitectura propuesta no solo constituye una contribución científica, sino que también sienta las bases para el desarrollo de soluciones industriales más resilientes, flexibles y adaptativas, capaces de responder a las crecientes demandas de digitalización y autonomía en la industria 4.0.



## Capítulo 9

# Conclusiones y trabajo futuro

Este capítulo final sintetiza las principales contribuciones desarrolladas a lo largo de la Tesis Doctoral, recogidas en los Capítulos 4, 5, 6, 7 y 8, así como las conclusiones fundamentales derivadas de varios años de investigación. Asimismo, se presentan las posibles líneas de trabajo futuro que surgen de los resultados alcanzados, orientadas a ampliar, validar y trasladar los avances propuestos hacia nuevos escenarios de aplicación. En consecuencia, este epígrafe ofrece una visión de conjunto que no solo pone en valor los logros obtenidos, sino que también sienta las bases para que otros investigadores puedan dar continuidad y proyección a esta línea de trabajo.

### 9.1. Conclusiones generales

La presente Tesis Doctoral ha abordado de manera sistemática los huecos identificados en el Estado del Arte y sintetizados en el Capítulo 3, proponiendo y validando un conjunto de soluciones innovadoras orientadas a redes programables, SG y arquitecturas IIoT. A continuación, se resumen las principales contribuciones y conclusiones de cada capítulo, destacando cómo han contribuido al cumplimiento de los objetivos establecidos.

En primer lugar, en el Capítulo 4 se presentó **win-BOFUSS**, una extensión del switch de referencia BOFUSS con soporte para control *in-band*. Esta propuesta se orientó a dispositivos IoT con recursos restringidos y demostró su idoneidad gracias a un protocolo ligero, resiliente y eficiente en la gestión de rutas de control. Se alcanzaron mecanismos de respaldo que garantizan continuidad de servicio, estableciendo una primera contribución sólida hacia el despliegue práctico de mecanismos de control *in-band* en entornos heterogéneos y densos.

En segundo lugar, el Capítulo 5 introdujo el algoritmo DEN2NE, concebido para la gestión eficiente de recursos en redes densas multi-hop mediante etiquetado jerárquico. Los resultados de evaluación confirmaron su escalabilidad (hasta 200 nodos) y tiempos de convergencia reducidos (inferiores a 20 ms). Con ello, se sentaron las bases para estrategias de encaminamiento y planificación de recursos ligeras y adaptables, en línea directa con el primer bloque de objetivos de la Tesis.

En tercer lugar, en el Capítulo 6 se propuso un enfoque de reconfiguración proactiva en SG apoyado en técnicas de ML/DL. Aprovechando conjuntos de datos reales y simulados, se demostró la viabilidad de predecir fallos y desencadenar procesos de reconfiguración robustos. Entre los modelos evaluados, destacó Random Forest optimizado con RFECV, que alcanzó una precisión superior al 94 % y una capacidad de detección equilibrada, reduciendo las falsas alarmas. Esta aportación permitió validar el papel de la inteligencia artificial como motor de resiliencia en sistemas energéticos críticos.

En cuarto lugar, el Capítulo 7 presentó el algoritmo BLOSTE, diseñado para el encaminamiento adaptativo de energía en redes de distribución malladas. Mediante un exhaustivo análisis con más de 148 000 simulaciones sobre topologías de referencia, se comprobó la eficacia de criterios multicriterio flexibles, con especial relevancia del criterio *Power2Zero*. La escalabilidad y la eficiencia computacional de BLOSTE se confirmaron incluso en topologías de hasta 2500 nodos, consolidando esta propuesta como una herramienta práctica para la gestión energética en redes de distribución modernas.

Finalmente, en el Capítulo 8 se desarrolló una arquitectura contenerizada para IIoT que integra telemetría, inferencia y control en un marco unificado. La evaluación experimental mostró que la propuesta permite ciclos de reconfiguración inferiores al segundo, con despliegue rápido y uso eficiente de recursos bajo condiciones de carga elevada. Además, su carácter abierto y reproducible aporta valor tanto científico como industrial, reforzando su potencial para aplicaciones de la Industria 4.0.

En conjunto, los resultados obtenidos en todos los capítulos de esta Tesis Doctoral confirman el cumplimiento de los dos bloques de objetivos planteados: (i) diseñar y validar mecanismos de control, encaminamiento y gestión de recursos prácticos y escalables para redes programables y su extensión a IIoT y SG; y (ii) proponer arquitecturas software abiertas y reproducibles que integren monitorización, inferencia y orquestación, habilitando la toma de decisiones automáticas y resilientes en entornos reales.

## 9.2. Líneas de trabajo futuro

El trabajo desarrollado en esta Tesis abre diversas oportunidades de investigación que pueden ampliar y profundizar los resultados obtenidos:

- **Ampliación de mecanismos *in-band*:** extender el protocolo win-BOFUSS a otros switches de referencia y plataformas hardware, así como evaluar su interoperabilidad con controladores SDN de nueva generación, explorando su integración con mecanismos de seguridad ligera y criptografía eficiente.
- **Evolución del algoritmo DEN2NE:** incorporar criterios de asignación dinámicos basados en aprendizaje reforzado, así como validaciones en entornos físicos heterogéneos (incluyendo redes inalámbricas de baja potencia y escenarios vehiculares).

- **Predicción y resiliencia en SG:** expandir el uso de modelos de AI/ML hacia enfoques federados que permitan preservar privacidad de datos en cooperativas energéticas, así como evaluar el impacto de la incorporación de *digital twins* para validación en tiempo real.
- **Optimización de BLOSTE:** explorar nuevas métricas multicriterio que integren sostenibilidad, coste económico y seguridad, además de realizar validaciones en escenarios de red con penetración masiva de renovables y generación distribuida.
- **Consolidación de la arquitectura IIoT basada en datos:** evolucionar hacia despliegues híbridos multi-cloud, incorporar soporte nativo para federated learning y evaluar la escalabilidad en entornos industriales reales de gran tamaño. Se propone también la ampliación de los datasets y la integración con plataformas de estandarización de la industria.

En definitiva, esta Tesis Doctoral sienta unas bases sólidas para el desarrollo de mecanismos de control, gestión y orquestación resilientes, escalables y abiertos en redes programables, SG y arquitecturas IIoT. Las líneas futuras aquí planteadas constituyen una hoja de ruta clara para continuar avanzando en la consolidación de soluciones científicas y tecnológicas que favorezcan la digitalización, autonomía y sostenibilidad de las infraestructuras críticas.



## Apéndice A

# Repositorios públicos

Con el objetivo de garantizar la reproducibilidad científica y fomentar la transferencia de resultados a la comunidad investigadora, todos los desarrollos realizados en esta Tesis Doctoral se han liberado como software de acceso abierto. En esta sección se recopilan los repositorios públicos asociados a cada una de las contribuciones principales, incluyendo código fuente, scripts de experimentación y documentación técnica.

- **Extensión win-BOFUSS para control *in-band*:** implementación del switch de referencia con soporte para control ligero en escenarios IoT. Disponible en: <https://github.com/NETSERV-UAH/in-BOFUSS>.
- **Algoritmo DEN2NE:** propuesta escalable para distribución y reasignación automática de recursos mediante etiquetado jerárquico. Disponible en: <https://github.com/NETSERV-UAH/den2ne-Alg>.
- **Predicción y resiliencia en SG:** prototipos de predicción de fallos y reconfiguración proactiva en redes eléctricas inteligentes, con conjuntos de datos y scripts de entrenamiento. Disponible en: <https://github.com/PaulaBartolomeMora/TFM>.
- **Algoritmo BLOSTE:** algoritmo multicriterio para el encaminamiento adaptativo en redes de distribución malladas. Disponible en: <https://github.com/NETSERV-UAH/den2ne-SmartGrids>.
- **Arquitectura *data-driven* para IIoT:** sistema contenerizado de monitorización, inferencia y control, acompañado de datasets abiertos y guías de despliegue. Disponible en: <https://github.com/NETSERV-UAH/datadriven-poc>.

La liberación de estos repositorios no solo aporta transparencia y validez a los resultados presentados, sino que también ofrece a otros investigadores una base práctica para extender, replicar y validar los mecanismos desarrollados en este trabajo.



# Bibliografía

- [1] T. Meuser and R. Kundel, “The History of Highly Adaptive and Programmable Networks,” in *From Multimedia Communications to the Future Internet: Essays Dedicated to Ralf Steinmetz on the Occasion of His Retirement*. Springer, 2024, pp. 61–73.
- [2] D. Levy and N. McKeown, “Overhaul May Bring Better, Faster Internet to 100 Million Homes,” *Stanford University News*, 2003, accessed: 2025-06-09. [Online]. Available: <https://news.stanford.edu/news/2003/october8/network-108.html>
- [3] Open Networking Foundation, “ONF Overview,” <https://opennetworking.org/>, 2025, accessed: 2025-06-09.
- [4] T. D. Nadeau and K. Gray, *SDN: Software Defined Networks: An Authoritative Review of Network Programmability Technologies*. O'Reilly Media, Inc., 2013.
- [5] ETSI, “Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges & Call for Action,” European Telecommunications Standards Institute (ETSI), White Paper Introductory White Paper, 2012, accessed: 2025-06-09. [Online]. Available: [https://portal.etsi.org/NFV/NFV\\_White\\_Paper.pdf](https://portal.etsi.org/NFV/NFV_White_Paper.pdf)
- [6] A. Vahdat, D. Clark, and J. Rexford, “A purpose-built global network: Google’s move to SDN,” *Communications of the ACM*, 2016.
- [7] F. Tomonori, “Introduction to ryu sdn framework,” *Open Networking Summit*, pp. 1–14, 2013.
- [8] S. Racherla, D. Cain, S. Irwin, P. Ljungstrøm, P. Patil, A. M. Tarenzio *et al.*, *Implementing ibm software defined network for virtual environments*. IBM Redbooks, 2014.
- [9] R. S. Montero, E. Rojas, A. A. Carrillo, and I. M. Llorente, “Extending the Cloud to the Network Edge.” *Computer*, vol. 50, no. 4, pp. 91–95, 2017.
- [10] M. Baddeley, R. Nejabati, G. Oikonomou, M. Sooriyabandara, and D. Simeonidou, “Evolving SDN for low-power IoT networks,” in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*. IEEE, 2018, pp. 71–79.
- [11] D. Carrascal, E. Rojas, D. Lopez-Pajares, N. Manso, J. Alvarez-Horcajo, and I. Martinez-Yelmo, “Softwarized Data-Driven Architecture for Edge Computing IIoT

## BIBLIOGRAFÍA

---

- Environments: A Proof of Concept,” in *2025 28th Conference on Innovation in Clouds, Internet and Networks (ICIN)*. IEEE, 2025, pp. 64–68.
- [12] S. S. Hussain, M. A. Aftab, F. Nadeem, I. Ali, and T. S. Ustun, “Optimal energy routing in microgrids with IEC 61850 based energy routers,” *IEEE Transactions on Industrial Electronics*, vol. 67, no. 6, pp. 5161–5169, 2019.
- [13] European Union, “Directive (EU) 2018/2001 of the European Parliament and of the Council of 11 December 2018 on the promotion of the use of energy from renewable sources,” <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32018L2001>, 2018, accessed: 2025-06-10.
- [14] Gobierno de España, “Real Decreto 244/2019, de 5 de abril, por el que se regulan las condiciones administrativas, técnicas y económicas del autoconsumo de energía eléctrica,” <https://www.boe.es/eli/es/rd/2019/04/05/244>, 2019, bOE-A-2019-5089, Accessed: 2025-06-10.
- [15] R. E. Mackiewicz, “Overview of IEC 61850 and Benefits,” in *2006 IEEE Power Engineering Society General Meeting*. IEEE, 2006, pp. 8–pp.
- [16] E. Molina, E. Jacob, J. Matias, N. Moreira, and A. Astarloa, “Using Software Defined Networking to manage and control IEC 61850-based systems,” *Computers & Electrical Engineering*, vol. 43, pp. 142–154, 2015. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0045790614002626>
- [17] H. Maziku and S. Shetty, “Software Defined Networking enabled resilience for IEC 61850-based substation communication systems,” in *2017 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2017, pp. 690–694.
- [18] W. Hu, Y. Hou, L. Tian, and Y. Li, “Selection of logistics distribution center location for SDN enterprises,” *Journal of Management Analytics*, vol. 2, no. 3, pp. 202–215, 2015.
- [19] D. Carrascal, E. Rojas, J. M. Arco, D. Lopez-Pajares, J. Alvarez-Horcajo, and J. A. Carral, “A comprehensive survey of in-band control in sdn: Challenges and opportunities,” *Electronics*, vol. 12, no. 6, p. 1265, 2023.
- [20] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: enabling innovation in campus networks,” *ACM SIGCOMM computer communication review*, vol. 38, no. 2, pp. 69–74, 2008.
- [21] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese *et al.*, “P4: Programming protocol-independent packet processors,” *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014.
- [22] F. Hauser, M. Häberle, D. Merling, S. Lindner, V. Gurevich, F. Zeiger, R. Frank, and M. Menth, “A survey on data plane programming with p4: Fundamentals, advances,

- and applied research,” *Journal of Network and Computer Applications*, vol. 212, p. 103561, 2023.
- [23] D. Carrascal, E. Rojas, J. Alvarez-Horcajo, D. Lopez-Pajares, and I. Martínez-Yelmo, “Analysis of P4 and XDP for IoT Programmability in 6G and Beyond,” *IoT*, vol. 1, no. 2, pp. 605–622, 2020.
- [24] P4 Language Consortium, “P4Runtime Specification,” <https://p4.org/p4-spec/p4runtime/main/P4Runtime-Spec.html>, 2023, accessed: 2025-06-16.
- [25] R. Vilalta, R. Muñoz, R. Casellas, R. Martínez, V. López, O. G. de Dios, A. Pastor, G. P. Katsikas, F. Klaedtke, P. Monti, A. Mozo, T. Zinner, H. Øverby, S. Gonzalez-Diaz, H. Lønsethagen, J.-M. Pulido, and D. King, “TeraFlow: Secured Autonomic Traffic Management for a Tera of SDN flows,” in *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, 2021, pp. 377–382.
- [26] D. López Pajares, “Nuevos conmutadores de red para redes integradas con SDN,” Tesis doctoral, Universidad de Alcalá, 2021, accessed: 2025-06-16. [Online]. Available: <https://ebuah.uah.es/dspace/handle/10017/51030>
- [27] C. Suo, I.-C. Tsai, and C. H.-P. Wen, “ERIC: Economical & reconfigurable hybrid-band control for software-defined datacenter network,” in *2016 International Conference on Information Networking (ICOIN)*. IEEE, 2016, pp. 214–219.
- [28] A. Jalili, H. Nazari, S. Namvarasl, and M. Keshtgari, “A comprehensive analysis on control plane deployment in SDN: In-band versus out-of-band solutions,” in *2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, 2017, pp. 1025–1031.
- [29] D. Kafetzis, S. Vassilaras, G. Vardoulias, and I. Koutsopoulos, “Software-Defined Networking meets Software-Defined Radio in Mobile Ad hoc Networks: State of the Art and Future Directions,” *IEEE Access*, 2022.
- [30] I. I. Awan, N. Shah, M. Imran, M. Shoaib, and N. Saeed, “An improved mechanism for flow rule installation in-band SDN,” *Journal of Systems Architecture*, vol. 96, pp. 1–19, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1383762118304739>
- [31] S. Khorsandroo, A. G. Sánchez, A. S. Tosun, J. Arco, and R. Doriguzzi-Corin, “Hybrid SDN evolution: A comprehensive survey of the state-of-the-art,” *Computer Networks*, vol. 192, p. 107981, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1389128621001109>
- [32] E. Rojas, R. Amin, C. Guerrero, M. Savi, and A. Rastegarnia, “Challenges and Solutions for hybrid SDN,” *Computer Networks*, vol. 195, p. 108198, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1389128621002498>

## BIBLIOGRAFÍA

---

- [33] M. K. MURTADHA, “SDN based device to device communication architecture for 5G mobile networks,” *Journal of Engineering Science and Technology*, vol. 16, no. 4, pp. 3033–3047, 2021.
- [34] J. Guo, L. Yang, X. Liu, Q. Chen, C. Fan, and X. Li, “Performance Modelling and Evaluation of In-Band Control Mode in Software-Defined Satellite Networks Based on Queuing Theory,” in *2021 2nd International Conference on Computing, Networks and Internet of Things*, ser. CNIOT ’21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: <https://doi.org/10.1145/3468691.3468697>
- [35] Y. Shi, Q. Yang, X. Huang, D. Li, and X. Huang, “An SDN-Enabled Framework for a Load-Balanced and QoS-Aware Internet of Underwater Things,” *IEEE Internet of Things Journal*, pp. 1–1, 2022.
- [36] I. Maity, R. Dhiman, and S. Misra, “EnPlace: Energy-Aware Network Partitioning for Controller Placement in SDN,” *IEEE Transactions on Green Communications and Networking*, 2022.
- [37] S. Chatopadhyay, S. Chatterjee, S. Nandi, and S. Chakraborty, “Aloe: An elastic auto-scaled and self-stabilized orchestration framework for iot applications,” in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 802–810.
- [38] S. Sharma, D. Staessens, D. Colle, M. Pickavet, and P. Demeester, “In-band control, queuing, and failure recovery functionalities for openflow,” *IEEE Network*, vol. 30, no. 1, pp. 106–112, 2016.
- [39] P. Goltzmann, M. Zitterbart, A. Hecker, and R. Bless, “Towards a Resilient In-Band SDN Control Channel,” *Universität Tübingen*, 2017.
- [40] A. S. Khakhalin and E. V. Chemeritskiy, “A reliable in-band control in a Software-Defined Network,” *Journal of Theoretical & Applied Information Technology*, vol. 95, no. 17, 2017.
- [41] P. M. Mohan, T. Truong-Huu, and M. Gurusamy, “Towards resilient in-band control path routing with malicious switch detection in SDN,” in *2018 10th International Conference on Communication Systems Networks (COMSNETS)*, 2018, pp. 9–16.
- [42] A. Raza, A. Gohar, and S. Lee, “MPTCP based in-band controlling for the software defined networks,” in *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, 2017, pp. 163–167.
- [43] S. González, A. De la Oliva, C. J. Bernardos, and L. M. Contreras, “Towards a Resilient Openflow Channel Through MPTCP,” in *2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, 2018, pp. 1–5.
- [44] A. S. M. Asadujjaman, E. Rojas, M. S. Alam, and S. Majumdar, “Fast Control Channel Recovery for Resilient In-band OpenFlow Networks,” in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, 2018, pp. 19–27.

- [45] D. Lopez-Pajares, J. Alvarez-Horcajo, E. Rojas, A. S. M. Asadujjaman, and I. Martinez-Yelmo, “Amaru: Plug&Play Resilient In-Band Control for SDN,” *IEEE Access*, vol. 7, pp. 123 202–123 218, 2019.
- [46] B. Görkemli, S. Tatlıcioğlu, A. M. Tekalp, S. Civanlar, and E. Lokman, “Dynamic Control Plane for SDN at Scale,” *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 12, pp. 2688–2701, 2018.
- [47] P. Holzmann and M. Zitterbart, “Izzy: A Distributed Routing Protocol for In-band SDN Control Channel Connectivity,” in *2019 IEEE 44th LCN Symposium on Emerging Topics in Networking (LCN Symposium)*, 2019, pp. 18–25.
- [48] K. Kumazoe, M. Shibata, and M. Tsuru, “A P4 BMv2-Based Feasibility Study on a Dynamic In-Band Control Channel for SDN,” in *Advances in Intelligent Networking and Collaborative Systems*, L. Barolli and H. Miwa, Eds. Cham: Springer International Publishing, 2022, pp. 442–451.
- [49] S. Sharma, D. Staessens, D. Colle, M. Pickavet, and P. Demeester, “Automatic bootstrapping of openflow networks,” in *2013 19th IEEE Workshop on Local & Metropolitan Area Networks (LANMAN)*, 2013, pp. 1–6.
- [50] C.-C. Tu, P.-W. Wang, and T.-c. Chiueh, “In-Band Control for an Ethernet-Based Software-Defined Network,” in *Proceedings of International Conference on Systems and Storage*, ser. SYSTOR 2014. New York, NY, USA: Association for Computing Machinery, 2014, p. 1–11. [Online]. Available: <https://doi.org/10.1145/2611354.2611359>
- [51] Y.-L. Su, I.-C. Wang, Y.-T. Hsu, and C. H.-P. Wen, “FASIC: A Fast-Recovery, Adaptively Spanning In-Band Control Plane in Software-Defined Network,” in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, 2017, pp. 1–6.
- [52] E. Sakic, M. Avdic, A. Van Bemten, and W. Kellerer, “Automated Bootstrapping of A Fault-Resilient In-Band Control Plane,” in *Proceedings of the Symposium on SDN Research*, ser. SOSR ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1–13. [Online]. Available: <https://doi.org/10.1145/3373360.3380829>
- [53] F. Wu and A. Tian, “rXstp: A Topology Discovery Mechanism Based on Rapid Spanning Tree for SDN In-Band Control,” in *2021 International Conference on Communications, Information System and Computer Engineering (CISCE)*, 2021, pp. 703–706.
- [54] M. Silva Freitas, R. Oliveira, D. Molinos, J. Melo, P. Frovi Rosa, and F. de Oliveira Silva, “ConForm: In-band Control Plane Formation Protocol to SDN-Based Networks,” in *2020 International Conference on Information Networking (ICOIN)*, 2020, pp. 574–579.

## BIBLIOGRAFÍA

---

- [55] C.-Y. Li, L.-H. Yen, K.-H. Chi, and C.-C. Tseng, “One-Pass In-Band Automatic Bootstrapping for OpenFlow Switches,” *IEEE Access*, vol. 9, pp. 153 349–153 359, 2021.
- [56] L. Schiff, S. Schmid, and M. Canini, “Medieval: Towards A Self-Stabilizing, Plug & Play, In-Band SDN Control Network,” in *ACM Sigcomm Symposium on SDN Research (SOSR)*, 2015. [Online]. Available: <http://eprints.cs.univie.ac.at/5745/>
- [57] ——, “Ground control to major faults: Towards a fault tolerant and adaptive SDN control network,” in *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W)*. IEEE, 2016, pp. 90–96.
- [58] L. Schiff, S. Schmid, and P. Kuznetsov, “In-Band Synchronization for Distributed SDN Control Planes,” *SIGCOMM Comput. Commun. Rev.*, vol. 46, no. 1, p. 37–43, January 2016. [Online]. Available: <https://doi.org/10.1145/2875951.2875957>
- [59] M. Canini, I. Salem, L. Schiff, E. M. Schiller, and S. Schmid, “Renaissance: A Self-Stabilizing Distributed SDN Control Plane,” in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, 2018, pp. 233–243.
- [60] ——, “Renaissance: A self-stabilizing distributed SDN control plane using in-band communications,” *Journal of Computer and System Sciences*, vol. 127, pp. 91–121, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0022000022000150>
- [61] B. Görkemli, A. M. Parlaklışık, S. Civanlar, A. Ulaş, and A. M. Tekalp, “Dynamic management of control plane performance in software-defined networks,” in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*. IEEE, 2016, pp. 68–72.
- [62] R. Hark, A. Rizk, N. Richerzhagen, B. Richerzhagen, and R. Steinmetz, “Isolated in-band communication for distributed SDN controllers,” in *2017 IFIP Networking Conference (IFIP Networking) and Workshops*, 2017, pp. 1–2.
- [63] M. Canini, I. Salem, L. Schiff, E. M. Schiller, and S. Schmid, “A Self-Organizing Distributed and In-Band SDN Control Plane,” in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 2656–2657.
- [64] K.-Y. Chan, C.-H. Chen, Y.-H. Chen, Y.-J. Tsai, S. S. W. Lee, and C.-S. Wu, “Fast Failure Recovery for In-Band Controlled Multi-Controller OpenFlow Networks,” in *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, 2018, pp. 396–401.
- [65] P. Holzmann, A. Hecker, and M. Zitterbart, “Towards a Distributed Routing Protocol for In-Band Control Channel with Elastic Controller Clusters,” in *2018 IEEE Globecom Workshops (GC Wkshps)*, 2018, pp. 1–6.
- [66] Open Networking Foundation, “SDN Architecture, Issue 1.1,” Open Networking Foundation, Technical Reference (TR) TR-521, 2016. [Online]. Available: [https://opennetworking.org/wp-content/uploads/2014/10/TR-521\\_SDN\\_Architecture\\_issue\\_1.1.pdf](https://opennetworking.org/wp-content/uploads/2014/10/TR-521_SDN_Architecture_issue_1.1.pdf)

- [67] L. Euler, “Solutio problematis ad geometriam situs pertinentis,” *Commentarii academiae scientiarum Petropolitanae*, pp. 128–140, 1741.
- [68] P. Fraigniaud, A. Pelc, D. Peleg, and S. Perennes, “Assigning labels in unknown anonymous networks,” in *Proceedings of the nineteenth annual ACM symposium on Principles of distributed computing*, 2000, pp. 101–111.
- [69] E. Rojas and G. Ibáñez, “Torii-HLMAC: A distributed, fault-tolerant, zero configuration fat tree data center architecture with multiple tree-based addressing and forwarding,” in *2012 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2012, pp. 2523–2528.
- [70] E. Rojas, H. Hosseini, C. Gomez, D. Carrascal, and J. R. Cotrim, “Outperforming RPL with scalable routing based on meaningful MAC addressing,” *Ad Hoc Networks*, vol. 114, p. 102433, 2021.
- [71] E. Rojas, J. Alvarez-Horcajo, I. Martinez-Yelmo, J. M. Arco, and J. A. Carral, “GA3: scalable, distributed address assignment for dynamic data center networks,” *Annals of Telecommunications*, vol. 72, no. 11, pp. 693–702, 2017.
- [72] E. Rojas, J. Alvarez-Horcajo, I. Martinez-Yelmo, J. M. Arco, and M. Briso-Montiano, “Scalable and Reliable Data Center Networks by Combining Source Routing and Automatic Labelling,” *Network*, vol. 1, no. 1, pp. 11–27, 2021.
- [73] M. Walraed-Sullivan, R. Niranjan Mysore, K. Marzullo, and A. Vahdat, “A Randomized Algorithm for Label Assignment in Dynamic Networks,” *Open Access Publications from the University of California*, 2013.
- [74] J. Bachiega Jr., B. Costa, L. R. Carvalho, M. J. F. Rosa, and A. Araujo, “Computational Resource Allocation in Fog Computing: A Comprehensive Survey,” *ACM Comput. Surv.*, mar 2023, just Accepted. [Online]. Available: <https://doi.org/10.1145/3586181>
- [75] Z. Ali, Z. H. Abbas, G. Abbas, A. Numani, and M. Bilal, “Smart computational offloading for mobile edge computing in next-generation Internet of Things networks,” *Computer Networks*, vol. 198, p. 108356, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1389128621003467>
- [76] K. Kaneva, N. Aboutorab, and G. Leu, “Multi-hop fronthaul offloading in learning-aided fog computing,” in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1–7.
- [77] S. Tong, Y. Liu, J. Mišić, X. Chang, Z. Zhang, and C. Wang, “Joint Task Offloading and Resource Allocation for Fog-Based Intelligent Transportation Systems: A UAV-Enabled Multi-Hop Collaboration Paradigm,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–16, 2022.
- [78] Z. Zhao, J. Perazzzone, G. Verma, and S. Segarra, “Congestion-Aware Distributed Task Offloading in Wireless Multi-Hop Networks Using Graph Neural Networks,”

## BIBLIOGRAFÍA

---

- in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 8951–8955.
- [79] F. J. Rodríguez, S. Fernandez, I. Sanz, M. Moranchel, and E. J. Bueno, “Distributed Approach for SmartGrids Reconfiguration Based on the OSPF Routing Protocol,” *IEEE Transactions on Industrial Informatics*, vol. 12, no. 2, pp. 864–871, 2016.
  - [80] P. Tenti and T. Caldognetto, “Integration of Local and Central Control Empowers Cooperation among Prosumers and Distributors towards Safe, Efficient, and Cost-Effective Operation of Microgrids,” *Energies*, vol. 16, no. 5, p. 2320, Feb 2023. [Online]. Available: <http://dx.doi.org/10.3390/en16052320>
  - [81] Y. Chen, D. Guo, M. Xu, G. Tang, and G. Cheng, “Measuring Maximum Urban Capacity of Taxi-Based Logistics,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 10, pp. 6449–6459, 2021.
  - [82] L. M. Moreno-Saavedra, S. Jiménez-Fernández, J. A. Portilla-Figueras, D. Casillas-Pérez, and S. Salcedo-Sanz, “A multi-algorithm approach for operational human resources workload balancing in a last mile urban delivery system,” *Computers & Operations Research*, vol. 163, p. 106516, 2024.
  - [83] ENTSO-E, “28 April 2025 Iberian Blackout,” <https://www.entsoe.eu/publications/blackout/28-april-2025-iberian-blackout/>, 2025, accessed: 2025-08-18.
  - [84] S. J. Heims and R. Andersen, “John von neumann and norbert wiener: from mathematics to the technologies of life and death,” 1981.
  - [85] S. Shanker, “Turing and the Origins of AI,” *Philosophia Mathematica*, vol. 3, no. 1, pp. 52–85, 1995.
  - [86] J. McCarthy, M. L. Minsky, N. Rochester, and C. Shannon, “The Dartmouth summer research project on artificial intelligence,” *Artificial intelligence: past, present, and future*, 1956.
  - [87] R. Anand, D. Aggarwal, and V. Kumar, “A comparative analysis of optimization solvers,” *Journal of Statistics and Management Systems*, vol. 20, no. 4, pp. 623–635, 2017.
  - [88] K. Tjell and R. Wisniewski, “Privacy preservation in distributed optimization via dual decomposition and ADMM,” in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019, pp. 7203–7208.
  - [89] H. R. Lourenço, O. C. Martin, and T. Stützle, “Iterated local search,” in *Handbook of metaheuristics*. Springer, 2003, pp. 320–353.
  - [90] B. P. Bhattacharai, S. Paudyal, Y. Luo, M. Mohanpurkar, K. Cheung, R. Tonkoski, R. Hovsapian, K. S. Myers, R. Zhang, P. Zhao *et al.*, “Big data analytics in smart grids: state-of-the-art, challenges, opportunities, and future directions,” *IET Smart Grid*, vol. 2, no. 2, pp. 141–154, 2019.

- [91] S. Koshy, S. Rahul, R. Sunitha, and E. P. Cherian, “Smart grid-based big data analytics using machine learning and artificial intelligence: A survey,” *Artif. Intell. Internet Things Renew. Energy Syst.*, vol. 12, p. 241, 2021.
- [92] T. Kotsopoulos, P. Sarigiannidis, D. Ioannidis, and D. Tzovaras, “Machine learning and deep learning in smart manufacturing: The smart grid paradigm,” *Computer Science Review*, vol. 40, p. 100341, 2021.
- [93] M. H. Hemmatpour, M. Mohammadian, and A. A. Gharaveisi, “Optimum islanded microgrid reconfiguration based on maximization of system loadability and minimization of power losses,” *International Journal of Electrical Power & Energy Systems*, vol. 78, pp. 343–355, 2016.
- [94] W. Sun, S. Ma, I. Alvarez-Fernandez, R. Roofegari nejad, and A. Golshani, “Optimal self-healing strategy for microgrid islanding,” *IET Smart Grid*, vol. 1, no. 4, pp. 143–150, 2018.
- [95] I. Sanz, M. Moranchel, J. Moriano, F. J. Rodríguez, and S. Fernandez, “Reconfiguration algorithm to reduce power losses in offshore HVDC transmission lines,” *IEEE Transactions on Power Electronics*, vol. 33, no. 4, pp. 3034–3043, 2017.
- [96] J. Hosseinzadeh, F. Masoodzadeh, and E. Roshandel, “Fault detection and classification in smart grids using augmented K-NN algorithm,” *SN Applied Sciences*, vol. 1, no. 12, p. 1627, 2019.
- [97] W. Li, D. Deka, M. Chertkov, and M. Wang, “Real-time faulted line localization and PMU placement in power systems through convolutional neural networks,” *IEEE Transactions on Power Systems*, vol. 34, no. 6, pp. 4640–4651, 2019.
- [98] A. S. Alhanaf, H. H. Balik, and M. Farsadi, “Intelligent fault detection and classification schemes for smart grids based on deep neural networks,” *Energies*, vol. 16, no. 22, p. 7680, 2023.
- [99] H. Kaplan, K. Tehrani, and M. Jamshidi, “Fault diagnosis of smart grids based on deep learning approach,” in *2021 World Automation Congress (WAC)*. IEEE, 2021, pp. 164–169.
- [100] T. Ding, Y. Lin, Z. Bie, and C. Chen, “A resilient microgrid formation strategy for load restoration considering master-slave distributed generators and topology reconfiguration,” *Applied energy*, vol. 199, pp. 205–216, 2017.
- [101] F. Bonada, L. Echeverria, X. Domingo, and G. Anzaldi, “AI for improving the overall equipment efficiency in manufacturing industry,” *New Trends in the Use of Artificial Intelligence for the Industry 4.0*, p. 79, 2020.
- [102] T. Mezair, Y. Djennouri, A. Belhadi, G. Srivastava, and J. C.-W. Lin, “A sustainable deep learning framework for fault detection in 6G Industry 4.0 heterogeneous data environments,” *Computer Communications*, vol. 187, pp. 164–171, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S014036642200055X>

## BIBLIOGRAFÍA

---

- [103] S. S. Aminabadi, P. Tabatabai, A. Steiner, D. P. Gruber, W. Friesenbichler, C. Habersohn, and G. Berger-Weber, “Industry 4.0 In-Line AI Quality Control of Plastic Injection Molded Parts,” *Polymers*, vol. 14, no. 17, 2022. [Online]. Available: <https://www.mdpi.com/2073-4360/14/17/3551>
- [104] V. Gadelha, E. Bullich-Massagué, and A. Sumper, “Optimal Power Flow in electrical grids based on power routers,” *Electric Power Systems Research*, vol. 234, p. 110581, 2024. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S037877962400467X>
- [105] K. Vu, M. M. Begovic, and D. Novosel, “Grids get smart protection and control,” *IEEE Computer Applications in Power*, vol. 10, pp. 40–44, 10 1997.
- [106] S. M. Amin and B. F. Wollenberg, “Toward a smart grid,” *IEEE Power and Energy Magazine*, vol. 3, pp. 34–41, 9 2005.
- [107] Y. Zhang, T. Huang, and E. F. Bompard, “Big data analytics in smart grids: a review,” *Energy informatics*, vol. 1, no. 1, pp. 1–24, 2018.
- [108] Y. Xu, J. Zhang, W. Wang, A. Juneja, and S. Bhattacharya, “Energy router: Architectures and functionalities toward Energy Internet,” in *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 2011, pp. 31–36.
- [109] J. Zhou and J. Wang, “Research Review on Multi-Port Energy Routers Adapted to Renewable Energy Access,” *Electronics*, vol. 13, no. 8, 2024. [Online]. Available: <https://www.mdpi.com/2079-9292/13/8/1493>
- [110] X. Zhu, W. Li, K. Huang, S. Cao, B. Lin, R. Li, and W. Xu, “Research on the Design and Application of Multi-Port Energy Routers,” *Energies*, vol. 18, no. 4, 2025. [Online]. Available: <https://www.mdpi.com/1996-1073/18/4/866>
- [111] B. Liu, J. Chen, Y. Zhu, Y. Liu, and Y. Shi, “Distributed Control Strategy of a Micro-grid Community With an Energy Router,” *IET Generation, Transmission & Distribution*, vol. 12, 08 2018.
- [112] T. Wu, C. Zhao, and Y.-J. A. Zhang, “Distributed AC-DC Optimal Power Dispatch of VSC-Based Energy Routers in Smart Microgrids,” *IEEE Transactions on Power Systems*, vol. 36, no. 5, pp. 4457–4470, 2021.
- [113] B. Huang, Y. Li, H. Zhang, and Q. Sun, “Distributed optimal co-multi-microgrids energy management for energy internet,” *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 4, pp. 357–364, 2016.
- [114] K. Wang, J. Yu, Y. Yu, Y. Qian, D. Zeng, S. Guo, Y. Xiang, and J. Wu, “A survey on energy internet: Architecture, approach, and emerging technologies,” *IEEE systems journal*, vol. 12, no. 3, pp. 2403–2416, 2018.
- [115] R. Wang, J. Wu, Z. Qian, Z. Lin, and X. He, “A Graph Theory Based Energy Routing Algorithm in Energy Local Area Network,” *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 3275–3285, 2017.

- [116] Z. Jiang, V. Sahasrabudhe, A. Mohamed, H. Grebel, and R. Rojas-Cessa, “Greedy Algorithm for Minimizing the Cost of Routing Power on a Digital Microgrid,” *Energies*, vol. 12, no. 16, 2019. [Online]. Available: <https://www.mdpi.com/1996-1073/12/16/3076>
- [117] M. Gayo, F. J. Rodríguez, C. Santos, P. Martín, and J. Jiménez, “Integration of Blockchain with IEC 61850 for Internal Management of Microgrids,” in *2020 IEEE 29th International Symposium on Industrial Electronics (ISIE)*, 2020, pp. 892–897.
- [118] S. M. S. Hussain, M. A. Aftab, F. Nadeem, I. Ali, and T. S. Ustun, “Optimal Energy Routing in Microgrids With IEC 61850 Based Energy Routers,” *IEEE Transactions on Industrial Electronics*, vol. 67, no. 6, pp. 5161–5169, 2020.
- [119] B. Gu, C. Mao, B. Liu, D. Wang, H. Fan, J. Zhu, and Z. Sang, “Optimal Charge/-Discharge Scheduling for Batteries in Energy Router-Based Microgrids of Prosumers via Peer-to-Peer Trading,” *IEEE Transactions on Sustainable Energy*, vol. 13, no. 3, pp. 1315–1328, 2022.
- [120] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, “The industrial internet of things (IIoT): An analysis framework,” *Computers in industry*, vol. 101, pp. 1–12, 2018.
- [121] E. Rojas, D. Carrascal, D. Lopez-Pajares, J. Alvarez-Horcajo, J. A. Carral, J. M. Arco, and I. Martinez-Yelmo, “A survey on ai-empowered softwarized industrial iot networks,” *Electronics*, vol. 13, no. 10, p. 1979, 2024.
- [122] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, and D. O. Wu, “Edge computing in industrial internet of things: Architecture, advances and challenges,” *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2462–2488, 2020.
- [123] K. M. Hou, X. Diao, H. Shi, H. Ding, H. Zhou, and C. de Vaulx, “Trends and challenges in AIoT/IIoT/IoT implementation,” *Sensors*, vol. 23, no. 11, p. 5074, 2023.
- [124] E. Rojas, D. Carrascal, D. Lopez-Pajares, N. Manso, and J. M. Arco, “Towards ai-enabled cloud continuum for iiot: Challenges and opportunities,” in *2024 International Conference on Artificial Intelligence, Computer, Data Sciences and Applications (ACDSA)*. IEEE, 2024, pp. 1–6.
- [125] M. A. Uusitalo, P. Rugeland, M. R. Boldi, E. C. Strinati, P. Demestichas, M. Ericson, G. P. Fettweis, M. C. Filippou, A. Gati, M.-H. Hamon *et al.*, “6G vision, value, use cases and technologies from European 6G flagship project Hexa-X,” *IEEE access*, vol. 9, pp. 160 004–160 020, 2021.
- [126] W. Jiang, B. Han, M. A. Habibi, and H. D. Schotten, “The road towards 6G: A comprehensive survey,” *IEEE Open Journal of the Communications Society*, vol. 2, pp. 334–366, 2021.

## BIBLIOGRAFÍA

---

- [127] W. Oñate and R. Sanz, “Analysis of architectures implemented for IIoT,” *Helijon*, vol. 9, no. 1, 2023.
- [128] X. Lin, “An overview of 5G advanced evolution in 3GPP release 18,” *IEEE Communications Standards Magazine*, vol. 6, no. 3, pp. 77–83, 2022.
- [129] P. Hu, “A system architecture for software-defined industrial Internet of Things,” in *2015 IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB)*. IEEE, 2015, pp. 1–5.
- [130] K. Wang, Y. Wang, Y. Sun, S. Guo, and J. Wu, “Green industrial Internet of Things architecture: An energy-efficient perspective,” *IEEE Communications Magazine*, vol. 54, no. 12, pp. 48–54, 2016.
- [131] P. Strauß, M. Schmitz, R. Wöstmann, and J. Deuse, “Enabling of predictive maintenance in the brownfield through low-cost sensors, an IIoT-architecture and machine learning,” in *2018 IEEE International conference on big data (big data)*. IEEE, 2018, pp. 1474–1483.
- [132] D. Dejene, B. Tiwari, and V. Tiwari, “TD2SecIoT: temporal, data-driven and dynamic network layer based security architecture for industrial IoT,” *IJIMAI*, vol. 6, no. 4, pp. 146–156, 2020.
- [133] P. Zhang, Y. Wu, and H. Zhu, “Open ecosystem for future industrial Internet of things (IIoT): architecture and application,” *CSEE Journal of Power and Energy Systems*, vol. 6, no. 1, pp. 1–11, 2020.
- [134] R. Taheri, M. Shojafar, M. Alazab, and R. Tafazolli, “FED-IIoT: A robust federated malware detection architecture in industrial IoT,” *IEEE transactions on industrial informatics*, vol. 17, no. 12, pp. 8442–8452, 2020.
- [135] Y. Zhang, W. Sun, and Y. Shi, “Architecture and Implementation of Industrial Internet of Things (IIoT) Gateway,” in *2020 IEEE 2nd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*. IEEE, 2020, pp. 114–120.
- [136] V. Chandra Shekhar Rao, P. Kumarswamy, M. Phridviraj, S. Venkatramulu, and V. Subba Rao, “5G enabled industrial internet of things (IIoT) architecture for smart manufacturing,” in *Data Engineering and Communication Technology: Proceedings of ICDECT 2020*. Springer, 2021, pp. 193–201.
- [137] A. Ghosh, A. Mukherjee, and S. Misra, “SEGA: Secured edge gateway microservices architecture for IIoT-based machine monitoring,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 1949–1956, 2021.
- [138] A. Medina, A. Lakhina, I. Matta, and J. Byers, “BRITE: an approach to universal topology generation,” in *MASCOTS 2001, Proceedings Ninth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 2001, pp. 346–353.

- [139] D. Carrascal, “Diseño e implementación de protocolo de control escalable en redes IoT para entornos 6G,” Trabajo Fin de Máster, Universidad de Alcalá, Alcalá de Henares, España, 2023, versión aceptada, acceso abierto, licencia CC-BY-NC-ND 4.0. [Online]. Available: <http://hdl.handle.net/10017/58379>
- [140] D. Carrascal, E. Rojas, D. Lopez-Pajares, N. Manso, and E. Gutierrez, “A scalable SDN in-band control protocol for IoT networks in 6G environments,” in *2023 6th International Conference on Advanced Communication Technologies and Networking (CommNet)*. IEEE, 2023, pp. 1–7.
- [141] S. Balaji, K. Nathani, and R. Santhakumar, “IoT Technology, Applications and Challenges: A Contemporary Survey,” *Wireless Personal Communications*, vol. 108, pp. 363–388, 9 2019. [Online]. Available: <https://link.springer.com/article/10.1007/s11277-019-06407-w>
- [142] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, D. Niyato, O. Dobre, and H. V. Poor, “6G Internet of Things: A Comprehensive Survey,” *IEEE Internet of Things Journal*, vol. 9, pp. 359–383, 1 2022.
- [143] Ericsson AB, “IoT connections outlook — Ericsson Mobility Report,” <https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/iot-connections-outlook>, Aug. 2025.
- [144] S. Li, L. D. Xu, and S. Zhao, “5G Internet of Things: A survey,” *Journal of Industrial Information Integration*, vol. 10, pp. 1–9, 6 2018.
- [145] 5G PPP Architecture Working Group, “The 6g architecture landscape: European perspective,” 5G Infrastructure Public Private Partnership (5G PPP), Tech. Rep., Dec. 2022. [Online]. Available: <https://5g-ppp.eu/6g-architecture-landscape-new-white-paper-for-public-consultation/>
- [146] Hexa-X Consortium, “Targets and requirements for 6g – initial e2e architecture,” Hexa-X Project, European Union Horizon 2020 Programme, Deliverable D1.3, Mar. 2022. [Online]. Available: [https://hexa-x.eu/wp-content/uploads/2022/03/Hexa-X\\_D1.3.pdf](https://hexa-x.eu/wp-content/uploads/2022/03/Hexa-X_D1.3.pdf)
- [147] D. Milojicic, “The Edge-to-Cloud Continuum,” *Computer*, vol. 53, no. 11, pp. 16–25, nov 2020.
- [148] L. Bittencourt, R. Immich, R. Sakellariou, N. Fonseca, E. Madeira, M. Curado, L. Villas, L. DaSilva, C. Lee, and O. Rana, “The Internet of Things, Fog and Cloud continuum: Integration and challenges,” *Internet of Things*, vol. 3-4, pp. 134–155, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2542660518300635>
- [149] A. Feghali, R. Kilany, and M. Chamoun, “SDN security problems and solutions analysis,” in *2015 International Conference on Protocol Engineering (ICPE) and International Conference on New Technologies of Distributed Systems (NTDS)*. IEEE, 2015, pp. 1–5.

## BIBLIOGRAFÍA

---

- [150] B. N. Constantin, “Desarrollo de una solución de encaminamiento para tráfico de control in-band en entornos sdn,” Trabajo Fin de Máster, Universidad de Alcalá, Escuela Politécnica Superior, Alcalá de Henares, España, 2020, acceso abierto (Repositorio institucional UAH). [Online]. Available: <http://hdl.handle.net/10017/46908>
- [151] IEEE 802 LAN/MAN Standards Committee, “IEEE Std 802c-2017: IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture – Amendment 2: Local Medium Access Control (MAC) Address Usage,” Standard, IEEE, pp. 1–26, 2017, iEEE Std 802c-2017 (Amendment to IEEE Std 802-2014 as amended by IEEE Std 802d-2017).
- [152] Z. L. Kis, “OpenFlow 1.1 Softswitch,” GitHub repository, 2012–16 de septiembre de 2025, trafficLab / of11softswitch, commit history starts 2012; e-mail: zoltan.lajos.kis@ericsson.com. [Online]. Available: <https://github.com/TrafficLab/of11softswitch>
- [153] E. L. Fernandes and C. E. Rothenberg, “OpenFlow 1.3 software switch,” *Salao de Ferramentas do XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos SBRC*, pp. 1021–1028, 2014.
- [154] E. L. Fernandes, E. Rojas, J. Alvarez-Horcajo, Z. L. Kis, D. Sanvito, N. Bonelli, C. Cascone, and C. E. Rothenberg, “The road to BOFUSS: The basic OpenFlow userspace software switch,” *Journal of Network and Computer Applications*, vol. 165, p. 102685, 2020.
- [155] R. R. Fontes, S. Afzal, S. H. B. Brito, M. A. S. Santos, and C. E. Rothenberg, “Mininet-WiFi: Emulating software-defined wireless networks,” in *2015 11th International Conference on Network and Service Management (CNSM)*, 2015, pp. 384–389.
- [156] D. Carrascal, E. Rojas, J. A. Carral, I. Martinez-Yelmo, and J. Alvarez-Horcajo, “Topology-aware scalable resource management in multi-hop dense networks,” *Heliyon*, vol. 10, no. 18, 2024.
- [157] O. B. Akan, E. Dinc, M. Kuscu, O. Cetinkaya, and B. A. Bilgin, “Internet of Everything (IoE) - From Molecules to the Universe,” *IEEE Communications Magazine*, pp. 1–7, 2023.
- [158] K. Midthun, V. Nørstebø, A. Tomasdard, and A. Werner, “Natural Gas Networks,” in *Encyclopedia of Energy, Natural Resource, and Environmental Economics*, J. F. Shogren, Ed. Waltham: Elsevier, 2013, pp. 161–167. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/B9780123750679001200>
- [159] A. Zlotnik, M. Chertkov, and S. Backhaus, “Optimal control of transient flow in natural gas networks,” in *2015 54th IEEE conference on decision and control (CDC)*. IEEE, 2015, pp. 4563–4570.

- [160] T.-P. Nguyen, Y.-K. Lin, and Y.-H. Chiu, “Investigate exact reliability under limited time and space of a multistate online food delivery network,” *Expert Systems with Applications*, vol. 213, p. 118894, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0957417422019121>
- [161] Y. Liu, B. Guo, C. Chen, H. Du, Z. Yu, D. Zhang, and H. Ma, “FooDNet: Toward an Optimized Food Delivery Network Based on Spatial Crowdsourcing,” *IEEE Transactions on Mobile Computing*, vol. 18, no. 6, pp. 1288–1301, 2019.
- [162] Q. Ma and P. Steenkiste, “On path selection for traffic with bandwidth guarantees,” in *Proceedings 1997 International Conference on Network Protocols*, 1997, pp. 191–202.
- [163] K. P. Schneider, B. Mather, B. Pal, C.-W. Ten, G. J. Shirek, H. Zhu, J. C. Fuller, J. L. R. Pereira, L. F. Ochoa, L. R. de Araujo *et al.*, “Analytic considerations and design basis for the IEEE distribution test feeders,” *IEEE Transactions on power systems*, vol. 33, no. 3, pp. 3181–3188, 2017.
- [164] B. M. Waxman, “Routing of multipoint connections,” *IEEE journal on selected areas in communications*, vol. 6, no. 9, pp. 1617–1622, 1988.
- [165] A.-L. Barabási and R. Albert, “Emergence of scaling in random networks,” *science*, vol. 286, no. 5439, pp. 509–512, 1999.
- [166] P. Bartolomé Mora, “Detección y predicción de errores en smart grids mediante modelos de inteligencia artificial,” Trabajo Fin de Máster, Universidad de Alcalá, Alcalá de Henares, España, Jul. 2024, acceso abierto, licencia CC-BY-NC-ND 4.0 Internacional. [Online]. Available: <http://hdl.handle.net/10017/62137>
- [167] D. Carrascal, P. Bartolomé, E. Rojas, D. Lopez-Pajares, N. Manso, and J. Diaz-Fuentes, “Fault Prediction and Reconfiguration Optimization in Smart Grids: AI-Driven Approach,” *Future Internet*, vol. 16, no. 11, p. 428, 2024.
- [168] Y. Himeur, A. Alsalemi, F. Bensaali, and A. Amira, “Building power consumption datasets: Survey, taxonomy and future directions,” *Energy and Buildings*, vol. 227, p. 110404, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S037877882030815X>
- [169] S. Makonin, B. Ellert, I. V. Bajić, and F. Popowich, “Electricity, water, and natural gas consumption of a residential house in Canada from 2012 to 2014,” *Scientific data*, vol. 3, no. 1, pp. 1–12, 2016.
- [170] K. Anderson, A. Ocneanu, D. Benitez, D. Carlson, A. Rowe, and M. Berges, “BLUED: A fully labeled public dataset for event-based non-intrusive load monitoring research,” *Proceedings of the 2nd KDD Workshop on Data Mining Applications in Sustainability (SustKDD)*, pp. 1–5, 01 2012.
- [171] C. Beckel, W. Kleiminger, R. Cicchetti, T. Staake, and S. Santini, “The ECO data set and the performance of non-intrusive load monitoring algorithms,” in *Proceedings*

## BIBLIOGRAFÍA

---

- of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, ser. BuildSys '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 80–89. [Online]. Available: <https://doi.org/10.1145/2674061.2674064>
- [172] A. Monacchi, D. Egarter, W. Elmenreich, S. D'Alessandro, and A. M. Tonello, “GREEND: An energy consumption dataset of households in Italy and Austria,” in *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 2014, pp. 511–516.
- [173] S. Makonin, “HUE: The hourly usage of energy dataset for buildings in British Columbia,” *Data in Brief*, vol. 23, p. 103744, 03 2019.
- [174] N. Batra, M. Gulati, A. Singh, and M. Srivastava, “It’s Different: Insights into Home Energy Consumption in India,” in *Proceedings of the 5th ACM Workshop on Embedded Systems for Energy-Efficient Buildings (BuildSys ’13)*. New York, NY, USA: Association for Computing Machinery, 11 2013, pp. 1–8.
- [175] J. Kolter and M. Johnson, “REDD: A Public Data Set for Energy Disaggregation Research,” *Artif. Intell.*, vol. 25, 01 2011.
- [176] S. Barker, A. Mishra, D. Irwin, E. Cecchet, P. Shenoy, and J. Albrecht, “Smart\*: An Open Data Set and Tools for Enabling Research in Sustainable Homes,” *Proc. SustKDD.*, 01 2012.
- [177] L. Pereira, F. Quintal, R. Gonçalves, and N. J. Nunes, “SustData: A public dataset for ICT4S Electric Energy Research,” *Proceedings of the 2014 conference ICT for Sustainability*, 2014.
- [178] J. Kelly and W. Knottenbelt, “The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes,” *Scientific Data*, vol. 2, no. 150007, 2015.
- [179] World Bank Group and Solargis, “Global Solar Atlas,” 2024, accessed: 2025-08-27. [Online]. Available: <https://globalsolaratlas.info/>
- [180] National Renewable Energy Laboratory (NREL), “PVWatts Calculator,” 2024, accessed: 2025-08-27. [Online]. Available: <https://pvwatts.nrel.gov/pvwatts.php>
- [181] P. Bartolomé Mora, “Development of Fault Detection and Prediction Machine Learning Models in Smart Grids Environments – Dataset KeyMonData,” <https://github.com/PaulaBartolomeMora/TFM>, 2025, accessed: 2025-08-27.
- [182] D. Carrascal, C. Santos, E. Rojas, J. M. Arco, D. Lopez-Pajares, and F. J. Rodriguez-Sanchez, “Dynamic Energy Routing Using Tree-Based Topologies with Fast Convergence applied to Meshed Microgrids,” *IEEE Access*, 2025, under review.
- [183] Y. Kafle, K. Mahmud, S. Morsalin, and G. Town, “Towards an internet of energy,” in *2016 IEEE International Conference on Power System Technology (POWERCON)*. IEEE, 2016, pp. 1–6.

- [184] M. R. Cruz, D. Z. Fitiwi, S. F. Santos, and J. P. Catalão, “Meshed operation of distribution network systems: Enabling increased utilization of variable RES power,” in *2018 IEEE International Conference on Environment and Electrical Engineering and 2018 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe)*. IEEE, 2018, pp. 1–6.
- [185] M. R. Cruz, D. Z. Fitiwi, S. F. Santos, S. J. Mariano, and J. P. Catalao, “Prospects of a meshed electrical distribution system featuring large-scale variable renewable power,” *Energies*, vol. 11, no. 12, p. 3399, 2018.
- [186] L. Prade, V. Uberti, A. Abaide, P. Pereira, R. de Figueiredo, and C. Both, “LoRa mesh architecture for automation of rural electricity distribution networks,” *Electronics Letters*, vol. 56, no. 14, pp. 739–741, 2020.
- [187] Y. Zhou, H. Ding, T. Shi, R. Fang, Y. Wang, and F. Bie, “Mesh planning optimization for urban distribution network with high reliability,” in *IOP Conference Series: Earth and Environmental Science*, vol. 701, no. 1. IOP Publishing, 2021, p. 012001.
- [188] F. Zishan, “Investigating the reliability and optimal capacity of microgrid electricity storage systems with the aim of reducing costs,” *International Journal of Smart Grid-ijSmartGrid*, vol. 8, no. 3, pp. 140–154, 2024.
- [189] J. Zhang, H. Que, X. Feng, X. Feng, and X. Tang, “Research on Improvement Calculation Method of Grid Power Losses Based on New Energy Access Model,” *EAI Endorsed Transactions on Energy Web*, vol. 11, pp. 1–8, 3 2024. [Online]. Available: <https://publications.eai.eu/index.php/ew/article/view/5487>
- [190] X. Wang, Y. Ji, J. Wang, Y. Gao, and L. Qi, “Research on distribution network reconfiguration based on microgrid,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, pp. 3607–3615, 2020.
- [191] W. Zhaoyu, A. Qian *et al.*, “Multi-objective allocation of microgrid in smart distribution network,” *Power System Technology*, vol. 36, no. 8, pp. 199–203, 2012.
- [192] L. L. L. Benchikh and D. Mechta, “A Survey on Energy Routing Approaches in Energy Internet,” *Energy Systems*, 2024.
- [193] A. Fawaz, I. Mougharbel, K. Al-Haddad, and H. Y. Kanaan, “Energy routing protocols for energy internet: A review on multi-agent systems, metaheuristics, and artificial intelligence approaches,” *IEEE Access*, vol. 13, pp. 41 625–41 643, 2025.
- [194] R. C. Prim, “Shortest connection networks and some generalizations,” *The Bell System Technical Journal*, vol. 36, no. 6, pp. 1389–1401, 1957.
- [195] UC3M, “6G-DATADRIVEN-02-E9: Arquitectura del sistema revisada,” <https://unica6g.it.uc3m.es/wp-content/uploads/2023/11/6G-DATADRIVEN-02-E9.pdf>.

## BIBLIOGRAFÍA

---

- [196] D. Carrascal, J. Díaz-Fuentes, N. Manso, D. Lopez-Pajares, E. Rojas, M. Savi, and J. M. Arco, “Softwarized Edge Intelligence for Advanced IIoT Ecosystems: A Data-Driven Architecture Across the Cloud/Edge Continuum,” *Applied Sciences*, 2025, under review.
- [197] Ziya, “Intelligent Manufacturing Dataset / Real-time sensor, network, and production data for AI-driven efficiency analysis,” <https://www.kaggle.com/datasets/ziya07/intelligent-manufacturing-dataset/data>.
- [198] InfluxData, “InfluxDB: Open Source Time Series Database,” <https://www.influxdata.com/products/influxdb/>, 2023, accessed: 2025-07-25.
- [199] BentoML Team, “BentoML: The Unified Model Serving Framework,” <https://bentoml.com>, 2023, accessed: 2025-07-25.
- [200] Grafana Labs, “Grafana: The Open Observability Platform,” <https://grafana.com>, 2023, accessed: 2025-07-25.