Universidad Nacional de Asunción
Facultad Politécnica

Ingeniería en Informática

Scalable and Decentralized Urban Traffic Optimization with Verifiable Storage for Smart City Deployments

Kevin Mathias Galeano Saldivar
María José Duarte Kowalewski

Proyecto de Trabajo de Grado presentado en conformidad a los requisitos para obtener el grado de Ingeniería en Informática
San Lorenzo - Paraguay
Febrero - 2026

Universidad Nacional de Asunción
Facultad Politécnica

Ingeniería en Informática

Scalable and Decentralized Urban Traffic Optimization with Verifiable Storage for Smart City Deployments

Kevin Mathias Galeano Saldivar
María José Duarte Kowalewski

San Lorenzo - Paraguay
Febrero - 2026

Universidad Nacional de Asunción
Facultad Politécnica

Ingeniería en Informática

Scalable and Decentralized Urban Traffic Optimization with Verifiable Storage for Smart City Deployments

Kevin Mathias Galeano Saldivar
María José Duarte Kowalewski

Asesor:
PhD Marcos Villagra

Proyecto de Trabajo de Grado presentado en conformidad a los
requisitos para obtener el grado de Ingeniería en Informática
San Lorenzo - Paraguay
Febrero - 2026

Dedico a mi blah blah blah,
a mi blah blah blah.

AGRADECIMIENTOS

Titulo1
Titulo2

Autor: Nombre y apellido

Asesor: Nombre y apellido

RESUMEN

Este es el resumen

Palabras Clave:

Title

Author: full name

Advisor: full name

ABSTRACT

This is the abstract

Keywords:

ÍNDICE

Página

LISTA DE FIGURAS

Página

LISTA DE TABLAS

Página

LISTA DE SÍMBOLOS

CAPÍTULO 1

INTRODUCCIÓN

1.1. Introducción

CAPÍTULO 2
MARCO TEÓRICO

2.1. Ingeniería de Tráfico y Control Semafórico

La ingeniería de tráfico es la rama de la ingeniería del transporte que se ocupa del planeamiento, diseño, operación y gestión de instalaciones viales, con el objetivo de proporcionar un movimiento seguro y eficiente de personas y mercancías [1, 2]. Su enfoque combina el análisis de la interacción entre usuarios, vehículos e infraestructura con el estudio de parámetros macroscópicos de flujo, como volumen, velocidad y densidad, para evaluar y mejorar el desempeño de redes viales urbanas y periurbanas.

Dentro de este campo, el control semafórico constituye una de las herramientas centrales para regular el derecho de paso en intersecciones donde confluyen flujos vehiculares y peatonales conflictivos [1, 3]. El diseño de planes de señalización semafórica abarca la definición de fases, la asignación de tiempos de verde, amarillo y rojo, y la coordinación entre intersecciones vecinas, con el fin de equilibrar seguridad, capacidad y niveles de servicio. La evolución desde esquemas de tiempos fijos hacia controles actuados y adaptativos refleja la necesidad de responder a patrones de demanda cada vez más variables, especialmente en entornos urbanos complejos.

2.1.1. Semáforos para el Control del Tránsito de Vehículos

Los semáforos constituyen el principal dispositivo de control del derecho de paso en intersecciones urbanas y periurbanas con conflictos significativos de circulación vehicular y peatonal [4]. En el Manual de Carreteras del Paraguay se establece que los semáforos para el control del tránsito de vehículos deben emplearse para asignar tiempos de paso y de detención a los distintos movimientos que confluyen en la intersección, con el objetivo de reducir conflictos, ordenar las maniobras y aumentar la seguridad vial [4]. Esta función se complementa con el control de flujos peatonales y, cuando corresponde, con fases específicas para giros, carriles exclusivos y

prioridades especiales (por ejemplo, transporte público o vehículos de emergencia) [4].

Desde el punto de vista de la ingeniería de tráfico, un semáforo vehicular puede interpretarse como un sistema de control discreto en el tiempo que alterna estados de indicación verde, amarilla y roja para cada grupo de movimientos, de forma tal que en ningún instante se autoricen simultáneamente movimientos incompatibles [4, 1]. El diseño del plan de fases considera la geometría de la intersección, los volúmenes de tránsito por aproximación, la presencia de giros a la izquierda y a la derecha, los flujos peatonales y, en su caso, la coordinación con semáforos próximos en el corredor [4, 2]. Las indicaciones luminosas verde, amarilla y roja poseen significado normado asociado a las obligaciones legales de detenerse o avanzar, y se complementan con flechas direccionales cuando es necesario discriminar maniobras permitidas dentro de una misma aproximación [4].

En la normativa paraguaya se especifican además criterios de ubicación, número de caras y área de influencia de cada cabezal semafórico, de modo que las indicaciones resulten claramente visibles para todos los usuarios a los que se dirigen [4]. Estos lineamientos aseguran que el dispositivo de control implementado por el sistema desarrollado en el presente trabajo sea compatible con la infraestructura semafórica existente y respetuoso de las disposiciones vigentes en el país.

2.1.2. Semáforos de Tiempos Fijos o Predeterminados

Los semáforos de tiempos fijos o predeterminados operan con un plan de señales cuya secuencia de fases y tiempos se define previamente y se repite cíclicamente, sin interacción directa con el tránsito en tiempo real [4]. El Manual de Carreteras del Paraguay indica que estos equipos asignan a cada fase un tiempo de verde y un tiempo intermedio (amarillo y, si corresponde, tiempo de despeje en rojo) calculados a partir de volúmenes de tránsito de diseño, análisis de capacidad y niveles de servicio objetivo [4]. Los tiempos de fase se mantienen constantes durante el período de operación del plan, aunque se contempla el uso de distintos planes fijos según la franja horaria (pico de la mañana, valle, pico de la tarde, periodo nocturno) [4, 1].

La principal ventaja de los semáforos de tiempos fijos reside en su simplicidad de diseño, facilidad de implementación y previsibilidad del patrón de señales, lo que se traduce en menores requerimientos de equipamiento (no se requieren detectores de vehículos) y en un mantenimiento relativamente sencillo [1, 2]. Sin embargo, este tipo de control presenta limitaciones en entornos con variaciones significativas de la demanda, como redes urbanas con fuertes diferencias entre horas pico y valle o con eventos recurrentes que alteran los patrones de flujo, ya que el plan fijo no se adapta a la ocupación real de las aproximaciones y puede generar tiempos de verde desaprovechados o demoras excesivas en ciertas corrientes [5, 6].

Desde la perspectiva de la optimización, el diseño de planes de tiempos fijos puede formularse como un problema de programación entera mixta (Mixed-Integer Linear Programming, MILP) en el que las variables de decisión —duración del ciclo, tiempos de verde por fase (splits) y, en el

caso de redes coordinadas, los desplazamientos temporales entre intersecciones (offsets)— son esencialmente discretas o deben satisfacer relaciones de discretización [7, 8]. En este marco, la función objetivo típicamente busca minimizar el retraso total de la red o maximizar la banda de progresión de pelotones a lo largo de un corredor, sujeta a restricciones de consistencia de ciclo, tiempos mínimos de verde, capacidad y sincronización [7, 9]. Los métodos analíticos clásicos, como las fórmulas de Webster [10], constituyen soluciones cerradas para versiones simplificadas de este problema: asumen una intersección aislada, demanda constante y condiciones de estado estacionario, permitiendo obtener expresiones analíticas para el ciclo óptimo y los splits que minimizan el retraso medio. Sin embargo, estas soluciones ignoran las interdependencias entre intersecciones, la naturaleza estocástica de los arribos y la presencia de restricciones operativas complejas [8, 5].

Cuando se considera la optimización de redes semafóricas con múltiples intersecciones coordinadas, el problema adquiere dimensiones combinatorias elevadas: el número de configuraciones posibles crece exponencialmente con el número de nodos, fases y rangos de valores admisibles para ciclos y offsets, lo que convierte al problema en NP–difícil [7, 8]. En estos casos, los métodos exactos de programación entera mixta pueden resultar computacionalmente intratables para redes de tamaño medio o grande, particularmente cuando se requiere resolver el problema en tiempo razonable para permitir ajustes frecuentes de los planes en respuesta a cambios de demanda. Por ello, la literatura ha recurrido a técnicas heurísticas y metaheurísticas —como algoritmos genéticos, simulated annealing y búsqueda local— para encontrar soluciones de buena calidad sin garantizar la optimalidad global [11, 12]. En particular, los algoritmos genéticos han mostrado eficacia para optimizar planes de tiempos fijos en condiciones de demanda sobresaturada, donde los modelos analíticos tradicionales pierden validez y se requiere exploración robusta del espacio de soluciones [11].

Diversos estudios en ingeniería de tráfico han mostrado que, en condiciones de demanda altamente variable, los esquemas de control fijo tienden a producir mayores tiempos de retraso medio, un mayor número de paradas y colas más extensas que los sistemas actuados o adaptativos [5, 6]. Este comportamiento motiva la búsqueda de estrategias de control más flexibles, como las basadas en lógica difusa o algoritmos de optimización metaheurística, que constituyen el foco del trabajo.

2.1.3. Semáforos Accionados por el Tránsito

Los semáforos accionados por el tránsito utilizan información en tiempo real captada por detectores (por ejemplo, lazos inductivos, sensores de radar o cámaras de video) ubicados en las aproximaciones para modificar la duración de las fases dentro de límites mínimos y máximos definidos [4]. El Manual de Carreteras del Paraguay distingue entre control parcialmente actuado (al menos una aproximación bajo control por detectores) y totalmente actuado (todas las aproximaciones controladas por demanda), estableciendo criterios de diseño para la ubicación de detectores, la distancia respecto de la línea de detención y los períodos iniciales mínimos y

extensiones de verde en función de la velocidad que comprende el 85 % del tránsito en el acceso [4].

En estos sistemas, el controlador decide en cada ciclo si extiende o termina el verde de una fase atendiendo a la detección de vehículos, lo que permite reducir tiempos de verde desaprovechados en movimientos con baja demanda y asignar capacidad adicional a las aproximaciones más cargadas [2, 5]. La literatura sobre control semafórico actuado y adaptativo destaca que este tipo de sistemas mejora el desempeño frente a planes fijos, especialmente en condiciones de demanda fluctuante, al disminuir el retraso medio por vehículo y el número de paradas, y al reducir la probabilidad de colas críticas en aproximaciones dominantes [5, 6].

El uso de detectores también permite implementar estrategias avanzadas como extensiones de verde para "limpiar" colas residuales, prioridad al transporte público o prioridad a vehículos de emergencia, resultando particularmente relevante en corredores urbanos con alta variabilidad de flujos [5, 13]. Estos conceptos constituyen la base sobre la cual se articula el sistema de control inteligente propuesto en este trabajo, donde un controlador difuso tipo Mamdani y algoritmos de optimización como Particle Swarm Optimization (PSO) se integran para ajustar dinámicamente parámetros de operación semafórica en respuesta a condiciones de tráfico cambiantes.

Entre las técnicas metaheurísticas aplicadas al control semafórico, Particle Swarm Optimization (PSO) ha demostrado eficacia en la exploración del espacio de parámetros de señalización, gracias a su capacidad para escapar de óptimos locales y su relativa simplicidad de implementación [14, 15, 16]. Cuando se combina con sistemas de inferencia difusa tipo Mamdani [17, 18, 19], que mapean variables lingüísticas de tráfico (congestión "baja", "media", "alta") a decisiones de control, se obtiene un enfoque híbrido capaz de incorporar conocimiento experto y adaptarse dinámicamente a condiciones variables, superando las limitaciones de los planes fijos y aproximándose al comportamiento de sistemas totalmente adaptativos. Este paradigma de control híbrido difuso–metaheurístico aprovecha la transparencia e interpretabilidad de los sistemas difusos para modelar relaciones complejas entre variables de tráfico y decisiones de control, mientras que PSO actúa como mecanismo de ajuste fino de los parámetros del sistema (funciones de pertenencia, pesos de reglas, ganancias de control), optimizando el desempeño global sin requerir modelos matemáticos explícitos del comportamiento del tráfico [20, 5].

2.1.4. Técnicas Algorítmicas y Computacionales para Control Adaptativo

La creciente disponibilidad de sensores de tráfico en tiempo real, el incremento de la capacidad de procesamiento computacional y los avances en inteligencia artificial han posibilitado el desarrollo de estrategias de control semafórico que van más allá de los enfoques clásicos basados en planes de tiempo fijo o en lógicas actuadas simples [5, 6]. Estas técnicas algorítmicas y computacionales permiten abordar el problema de optimización de tiempos semafóricos en condiciones de demanda estocástica y no estacionaria, superando las limitaciones de los métodos analíticos tradicionales que asumen flujos uniformes y equilibrio de estado [5, 21].

Entre las técnicas más consolidadas se encuentran los sistemas de control adaptativo en tiempo real, como SCOOT (Split, Cycle and Offset Optimization Technique) y SCATS (Sydney Coordinated Adaptive Traffic System), que ajustan automáticamente los parámetros de señalización a partir de información de ocupación y flujo captada por detectores ubicados aguas arriba de las intersecciones [22, 23]. SCOOT, desarrollado en el Reino Unido, emplea modelos de tráfico microscópicos para predecir la llegada de pelotones y ajusta ciclo, splits y offsets de forma continua, minimizando una función de costo basada en demoras y paradas [22, 5]. SCATS, implementado originalmente en Australia, utiliza un enfoque de biblioteca de planes (plan library) en el que un controlador central selecciona dinámicamente el plan más apropiado de un conjunto predefinido en función de las condiciones de tráfico observadas, y ajusta los tiempos de verde dentro de límites establecidos [23, 13]. Ambos sistemas han demostrado reducciones significativas en demoras y emisiones en redes urbanas de mediana y gran escala, aunque requieren infraestructura de detección extensa y calibración cuidadosa de sus parámetros operativos.

Paralelamente, el campo de la optimización metaheurística ha aportado algoritmos capaces de explorar eficientemente el espacio de soluciones de problemas combinatorios complejos, evitando quedar atrapados en óptimos locales. Los algoritmos genéticos (Genetic Algorithms, GA) emulan procesos de evolución biológica —selección, cruce y mutación— para hacer evolucionar poblaciones de soluciones candidatas (configuraciones de ciclo, splits y offsets) hacia regiones de alta calidad del espacio de búsqueda, y han sido aplicados con éxito a la optimización de redes semafóricas coordinadas y a la optimización multiobjetivo que balancea demoras, emisiones y equidad [12, 5]. Particle Swarm Optimization (PSO), inspirado en el comportamiento social de bandadas de aves o cardúmenes de peces, representa cada solución candidata como una "partícula" que se desplaza en el espacio de parámetros guiada por su propia experiencia (mejor posición histórica individual) y por la experiencia del conjunto (mejor posición global del enjambre), logrando convergencia rápida y robusta en problemas de optimización de tiempos semafóricos [14, 15, 16]. Otros algoritmos metaheurísticos aplicados al dominio incluyen *simulated annealing*, búsqueda tabú y optimización por colonias de hormigas, cada uno con fortalezas específicas en términos de diversidad de exploración, velocidad de convergencia y manejo de múltiples objetivos [5].

En el ámbito de la inteligencia artificial, los sistemas basados en lógica difusa (fuzzy logic) permiten incorporar conocimiento experto y razonamiento aproximado en el control de semáforos, modelando conceptos lingüísticos como "congestión alta", "demanda moderada" o "cola larga" mediante funciones de pertenencia y reglas de inferencia del tipo "si–entonces" [17, 18]. Los controladores difusos tipo Mamdani mapean variables de entrada continuas (por ejemplo, volumen, velocidad, densidad) a decisiones de control (extensión de verde, cambio de fase) de forma intuitiva y transparente, facilitando la interpretación y el ajuste de los parámetros por parte de operadores humanos [24, 25]. Las redes neuronales artificiales (Artificial Neural Networks, ANN) han sido exploradas como aproximadores universales de funciones complejas en el contexto de predicción de tráfico y control adaptativo, aunque su naturaleza de "caja negra"

dificulta la interpretabilidad de las decisiones [5]. Más recientemente, las técnicas de aprendizaje por refuerzo (Reinforcement Learning, RL) —en particular Q-learning y Deep Q-Networks (DQN)— han ganado prominencia al permitir que un agente de control aprenda políticas óptimas de asignación de verde a partir de la interacción directa con el entorno de tráfico, sin requerir un modelo explícito del sistema y adaptándose automáticamente a patrones de demanda cambiantes [26, 21]. Estos enfoques de aprendizaje profundo han mostrado resultados prometedores en simulaciones, superando en algunos casos a SCOOT y SCATS en escenarios de alta variabilidad, aunque aún enfrentan desafíos de estabilidad, seguridad y aceptación regulatoria para su despliegue en sistemas reales [21].

Finalmente, enfoques más radicales como los sistemas autoorganizados proponen que cada intersección opere de forma autónoma, coordinándose con sus vecinas mediante intercambio de información local y reglas de decisión descentralizadas, sin depender de un controlador central [27]. Esta perspectiva se inspira en fenómenos naturales de autoorganización y busca mayor robustez frente a fallos y escalabilidad en redes extensas, aunque plantea interrogantes sobre garantías de desempeño y estabilidad global del sistema.

En conjunto, estas técnicas algorítmicas y computacionales representan una evolución desde los paradigmas tradicionales de control fijo y actuado hacia estrategias verdaderamente adaptativas, capaces de responder de forma dinámica y autónoma a la variabilidad inherente del tráfico urbano moderno. El presente trabajo se inscribe en esta línea al combinar un sistema de inferencia difusa tipo Mamdani —que captura conocimiento experto sobre congestión— con el algoritmo metaheurístico PSO —que optimiza los parámetros de operación semafórica—, conformando un enfoque híbrido que aprovecha las fortalezas de ambas técnicas y que será descrito en detalle en secciones posteriores.

2.1.5. Parámetros Clásicos de Diseño de Tiempos Semafóricos

En el diseño clásico de control semafórico, la configuración de una intersección se describe mediante un conjunto reducido de parámetros numéricos que determinan su desempeño operativo: la longitud de ciclo, la repartición de verdes por fase (splits), los tiempos de amarillo y todo–rojo, y los tiempos perdidos en cada etapa del ciclo [1, 2]. La longitud de ciclo C representa la duración total de una repetición completa de las fases del semáforo, mientras que los splits determinan qué fracción de ese ciclo se asigna al verde efectivo de cada movimiento o grupo de movimientos. Los tiempos de amarillo y todo–rojo se definen para advertir el fin del derecho de paso y garantizar el despeje seguro de la intersección antes de otorgar el verde a flujos conflictivos, y los tiempos perdidos agrupan intervalos en los que, por razones de seguridad o arranque, la capacidad efectiva de la intersección es menor a la teórica [1].

Estos parámetros influyen directamente en la capacidad y el retraso en cada aproximación. Para una misma demanda, ciclos más largos tienden a aumentar la capacidad por fase pero también pueden incrementar el tiempo medio de espera si los usuarios deben esperar largos intervalos de rojo, mientras que ciclos muy cortos reducen el retraso pero pueden volverse

ineficientes si el tiempo perdido ocupa una proporción significativa del ciclo [2, 5]. La capacidad se relaciona con la proporción de verde asignada a cada movimiento y con la saturación de los carriles, en tanto que el retraso medio por vehículo, el número de paradas y la longitud de colas son métricas fundamentales para evaluar la calidad de servicio de un plan de tiempos [1, 3].

Métodos de diseño tradicional, como la formulación de Webster, proporcionan expresiones para estimar un ciclo "óptimo" fijo a partir de los volúmenes de diseño y de los tiempos perdidos, buscando un compromiso entre capacidad y retraso medio [10, 3]. En la práctica, estos valores se calculan para condiciones de flujo representativas y se implementan como planes de tiempo fijo, eventualmente diferenciados por franja horaria. En contraste, el enfoque desarrollado en este trabajo utiliza un sistema de inferencia difusa tipo Mamdani y un algoritmo de optimización PSO para ajustar dinámicamente la longitud de ciclo efectiva, los splits y otros parámetros de operación en función de mediciones de tráfico en tiempo real, con el objetivo de reducir retrasos y mejorar el desempeño frente a condiciones de demanda variables [5, 6].

2.1.6. Modelado del Control Semafórico como Problema de Optimización

El diseño y operación de sistemas de control semafórico puede formularse como un problema de optimización combinatoria en el que se busca determinar los valores de un conjunto de variables de decisión que minimicen (o maximicen) una función objetivo, respetando un conjunto de restricciones operativas y de seguridad [8, 9]. Esta perspectiva permite trascender los enfoques tradicionales basados en fórmulas empíricas y abordar de manera sistemática la complejidad inherente a redes urbanas con demanda variable y múltiples intersecciones interdependientes.

En su formulación más general, el problema de optimización de control semafórico define como variables de decisión los parámetros temporales que gobiernan la operación del sistema: la duración del ciclo C, los tiempos de verde efectivo asignados a cada fase o movimiento (splits g_i), los desplazamientos temporales entre intersecciones consecutivas (offsets ϕ_j), y en algunos casos la secuencia misma de fases [8, 1]. Estas variables deben elegirse dentro de rangos factibles definidos por la geometría, la capacidad de las aproximaciones y los requerimientos normativos de tiempos mínimos y máximos.

La función objetivo del problema representa la métrica que se desea optimizar y puede adoptar diversas formas según el criterio de desempeño que se privilegie. Entre las formulaciones más comunes se encuentran la minimización del retraso total experimentado por todos los vehículos en la red, la minimización del número total de paradas, la maximización del throughput (número de vehículos que atraviesan el sistema por unidad de tiempo), o la minimización de emisiones contaminantes y consumo de combustible derivados de aceleraciones y frenados [8, 6, 5]. En aplicaciones recientes se han propuesto funciones objetivo multicriterio que balancean simultáneamente eficiencia operativa, equidad entre corrientes de tráfico, seguridad y sostenibilidad ambiental, empleando técnicas de optimización multiobjetivo como frentes de Pareto o funciones de utilidad ponderadas [12, 6].

El problema se completa con un conjunto de restricciones que garantizan la factibilidad y seguridad de las soluciones. Estas restricciones incluyen tiempos mínimos de verde para permitir que los peatones crucen la intersección de forma segura, tiempos máximos de verde para evitar demoras excesivas en movimientos conflictivos, tiempos de amarillo y todo–rojo calculados en función de la velocidad de aproximación y las distancias de despeje, y restricciones de capacidad que limitan los flujos admisibles en cada carril o aproximación [1, 3]. En el caso de redes coordinadas, se añaden restricciones de consistencia de ciclo (todas las intersecciones deben operar con el mismo ciclo común o con múltiplos enteros del ciclo base) y restricciones de sincronización que determinan los valores admisibles de los offsets para lograr progresión de pelotones [28, 13].

Los métodos analíticos tradicionales, como las fórmulas de Webster [10] o el método del Highway Capacity Manual (HCM) [2], constituyen soluciones cerradas para versiones simplificadas del problema de optimización, en las que se asumen condiciones de estado estacionario, demanda uniforme y geometrías regulares. Estas formulaciones proporcionan estimaciones rápidas del ciclo óptimo y de los splits correspondientes, y resultan útiles para el diseño preliminar de planes de tiempo fijo. Sin embargo, en la práctica, las condiciones reales de tráfico urbano se caracterizan por una alta variabilidad temporal (diferencias entre horas pico y valle, fluctuaciones aleatorias de la demanda), heterogeneidad espacial (diferencias de volumen entre aproximaciones), y la presencia de eventos no recurrentes (accidentes, obras, eventos especiales) que invalidan las hipótesis de estacionariedad de los modelos analíticos [5, 6].

Esta variabilidad convierte al problema de optimización semafórica en un desafío computacional complejo, especialmente cuando se considera la operación coordinada de múltiples intersecciones a lo largo de corredores o redes completas. En estos casos, el espacio de búsqueda de soluciones crece exponencialmente con el número de intersecciones y de fases, y el problema adquiere la estructura de un problema de optimización combinatoria NP–difícil, para el cual no se conocen algoritmos exactos con tiempo de ejecución polinomial [8, 12]. Ante esta complejidad, las técnicas algorítmicas avanzadas —como algoritmos genéticos, Particle Swarm Optimization (PSO), simulated annealing y métodos de búsqueda local— se han consolidado como herramientas eficaces para explorar el espacio de soluciones, escapar de óptimos locales y encontrar configuraciones de alta calidad en tiempos de cómputo razonables [12, 5].

En síntesis, la formulación del control semafórico como problema de optimización combinatoria proporciona un marco conceptual riguroso que permite integrar distintos criterios de desempeño, considerar múltiples restricciones operativas y de seguridad, y justificar el empleo de técnicas computacionales avanzadas —como los sistemas de inferencia difusa y los algoritmos metaheurísticos— que serán presentados en secciones posteriores y que constituyen la base del sistema de control inteligente desarrollado en este trabajo.

2.1.7. Coordinación de Semáforos y Progresión de Pelotones

En redes urbanas, los semáforos rara vez operan de manera completamente aislada; por el contrario, se busca coordinar varios equipos a lo largo de un corredor para facilitar la progresión de pelotones de vehículos y reducir el número de paradas sucesivas [1, 2]. La coordinación se logra ajustando parámetros como la longitud de ciclo común entre intersecciones, los offsets (desplazamientos temporales entre inicios de verde) y, en algunos casos, la secuencia de fases, de modo que un grupo de vehículos que parte con verde en una intersección encuentre verdes sucesivos en las siguientes, configurando la llamada "onda verde" [3, 28]. Estos esquemas buscan compatibilizar la operación de múltiples nodos de control, sacrificando a veces el óptimo local de una intersección en beneficio del desempeño global del corredor.

La coordinación introduce dependencias fuertes entre los parámetros de distintas intersecciones: cambios en la longitud de ciclo o en los splits de una intersección pueden degradar la progresión en tramos adyacentes, generando nuevas demoras o incrementando el número de paradas en el corredor [28, 2]. Por ello, el diseño de planes coordinados suele apoyarse en herramientas de simulación y optimización que consideran simultáneamente varias intersecciones, y en prácticas de monitoreo operativo que permiten ajustar offsets y ciclos en respuesta a cambios de demanda. En este contexto, la idea de un sistema de control y monitoreo distribuido, apoyado en tecnologías como BlockDAG e IPFS para registrar configuraciones y métricas de desempeño, resulta especialmente pertinente: aunque el presente trabajo se centre en la optimización a nivel de una intersección, la misma arquitectura puede escalar a corredores coordinados donde las decisiones de tiempo de verde, ciclo y offset deben gestionarse de forma conjunta y trazable.

2.1.8. Consideraciones Operativas y Métricas de Desempeño

La evaluación de un plan de tiempos semafóricos se basa en un conjunto de métricas operativas que permiten cuantificar su impacto sobre los usuarios de la vía. Entre las más utilizadas se encuentran el retraso medio por vehículo, el número de paradas, la longitud de colas y el nivel de servicio (Level of Service, LOS); en estudios recientes se añaden además indicadores de consumo de combustible y emisiones contaminantes [1, 2, 5]. El retraso medio mide la diferencia entre el tiempo de viaje real y el tiempo teórico sin interferencias, mientras que el número de paradas y la longitud de colas caracterizan la discontinuidad del flujo y la probabilidad de bloqueo de accesos o de intersecciones adyacentes [1, 3].

El nivel de servicio resume de forma cualitativa la calidad de operación en categorías que van desde "A" (pocas demoras, operación confortable) hasta "F" (congestión severa), definidas a partir de rangos de retraso medio por vehículo y otras variables [2, 3]. Estas métricas permiten comparar diferentes esquemas de control (planes fijos, control actuado, control adaptativo) bajo condiciones de demanda similares y constituyen la base para justificar cambios en la programación de semáforos desde una perspectiva de ingeniería.

En el contexto de este trabajo, el desempeño de los planes semafóricos se evalúa a partir de indicadores derivados de la salida del sistema difuso de congestión (valor numérico de congestión y su categoría lingüística), junto con variables macroscópicas como volumen por minuto, velocidad media y densidad vehicular. El controlador difuso Mamdani proporciona una medida agregada de congestión a partir de estas variables, y el algoritmo PSO ajusta los tiempos de verde para minimizar directamente dicho índice de congestión y mejorar las condiciones de flujo respecto a una asignación base de tiempos calculada mediante fórmulas clásicas de ingeniería de tráfico. La automatización de la recolección de estas métricas y su análisis en tiempo real plantea desafíos de almacenamiento, trazabilidad e integridad de datos. En sistemas de control inteligente distribuido, donde múltiples nodos (intersecciones) operan de forma coordinada, resulta fundamental contar con mecanismos que aseguren la auditabilidad de las decisiones de control y la inmutabilidad de los registros históricos de configuración y desempeño [29, 30]. Este requisito motiva el uso de arquitecturas descentralizadas como BlockDAG e IPFS, que serán analizadas en secciones posteriores.

2.1.9. Integración de Técnicas de Optimización y Control Difuso

La convergencia entre técnicas de optimización metaheurística y sistemas de inferencia difusa ha dado lugar a enfoques híbridos que combinan la capacidad de razonamiento aproximado y manejo de incertidumbre de la lógica difusa con la potencia de búsqueda y ajuste paramétrico de algoritmos como Particle Swarm Optimization (PSO), algoritmos genéticos y otras metaheurísticas [5, 20]. En el contexto del control semafórico, estos sistemas híbridos fuzzy–metaheurísticos permiten abordar de manera integrada dos desafíos complementarios: por un lado, la captura y formalización del conocimiento experto sobre las relaciones entre variables de tráfico y decisiones de control mediante reglas lingüísticas y funciones de pertenencia difusas; por otro, la optimización automática de los parámetros del sistema difuso (formas y umbrales de las funciones de pertenencia, pesos de las reglas, ganancias de salida) de forma que se maximice una función objetivo operativa, como la minimización de retrasos o la maximización del throughput de la red [31, 32].

En un sistema de control difuso tipo Mamdani aplicado a semáforos, las funciones de pertenencia de las variables de entrada (por ejemplo, volumen vehicular, velocidad media, densidad, longitud de cola) y de salida (extensión de verde, ajuste de ciclo, cambio de fase) se definen inicialmente con base en conocimiento experto o en datos históricos, pero su sintonización fina puede requerir ajustes iterativos costosos y poco sistemáticos. PSO ofrece una alternativa eficiente y automática: cada partícula del enjambre representa una configuración específica de parámetros del sistema difuso, y la posición de la partícula se actualiza iterativamente en función de su desempeño histórico y del desempeño del mejor individuo global, explorando el espacio de configuraciones hasta encontrar un conjunto de parámetros que optimice la métrica de desempeño deseada [14, 15, 16]. Este enfoque de optimización de parámetros difusos mediante PSO ha demostrado reducir significativamente el tiempo de diseño del controlador, mejorar

la robustez frente a variaciones de demanda y permitir la adaptación online de parámetros en respuesta a condiciones cambiantes [20, 31].

Extensiones más complejas incluyen sistemas neuro–fuzzy, en los que se combinan redes neuronales artificiales con lógica difusa para permitir aprendizaje automático de las reglas de inferencia y funciones de pertenencia a partir de datos, y que pueden ser entrenados o ajustados mediante metaheurísticas como PSO, algoritmos genéticos o búsqueda cuco (Cuckoo Search) [33]. Estos enfoques híbridos neuro–fuzzy–metaheurísticos aprovechan la capacidad de generalización y aprendizaje de las redes neuronales, la interpretabilidad de los sistemas difusos y la eficiencia de búsqueda de las metaheurísticas, constituyendo una vía prometedora para el diseño de controladores adaptativos en dominios complejos como el control de tráfico urbano [5, 32].

El sistema propuesto en este trabajo se inscribe en esta línea de sistemas híbridos fuzzy–PSO: emplea un controlador difuso tipo Mamdani para evaluar el nivel de congestión a partir de variables macroscópicas de tráfico (volumen, velocidad, densidad) y utiliza PSO para optimizar los tiempos de verde de las fases semafóricas de forma que se minimice el índice de congestión estimado por el sistema difuso. Este enfoque permite incorporar conocimiento experto sobre la relación entre variables de tráfico y estados de congestión en las reglas difusas, mientras que PSO asegura que la asignación de tiempos de verde sea óptima respecto de la configuración de demanda observada. La arquitectura resultante es adaptativa, interpretable y computacionalmente eficiente, características esenciales para su despliegue en entornos operativos reales.

Más allá de las ventajas operativas del enfoque híbrido, la implementación de sistemas de control inteligente en infraestructuras críticas como redes semafóricas plantea requisitos adicionales de trazabilidad, auditabilidad e integridad de datos. En un sistema distribuido donde múltiples intersecciones operan de forma coordinada o autónoma, resulta fundamental registrar de manera inmutable y verificable las decisiones de control tomadas (tiempos de verde asignados, parámetros del controlador difuso, resultados de la optimización PSO), las condiciones de tráfico observadas (volúmenes, velocidades, densidades capturadas por sensores) y las métricas de desempeño resultantes (retrasos, emisiones, niveles de servicio). Estos registros permiten auditar el comportamiento del sistema, identificar configuraciones problemáticas, validar la conformidad con normativas y políticas de operación, y facilitar la depuración y mejora continua del controlador. Las arquitecturas descentralizadas basadas en BlockDAG (Directed Acyclic Graph de bloques) e IPFS (InterPlanetary File System) ofrecen soluciones escalables y confiables para el registro y almacenamiento distribuido de estos datos, temas que serán analizados en profundidad en secciones posteriores del presente trabajo. En síntesis, la integración de técnicas de optimización metaheurística y control difuso no solo mejora el desempeño operativo del sistema semafórico, sino que abre la puerta a arquitecturas distribuidas y auditables que responden a las demandas de transparencia y confiabilidad de las ciudades inteligentes modernas.

CAPÍTULO 3

VISIÓN GENERAL DEL SISTEMA

El sistema propuesto se concibe como un ecosistema modular de microservicios para el monitoreo, análisis y optimización del tráfico urbano en tiempo (casi) real, diseñado para operar en entornos de ciudad inteligente. La arquitectura se compone de cuatro módulos principales, cada uno con responsabilidades bien definidas y comunicándose mediante interfaces estandarizadas (APIs REST y mensajes JSON). Esta organización modular facilita la escalabilidad, el mantenimiento independiente de componentes y la integración con fuentes de datos heterogéneas.

3.0.1. Módulos Principales

El sistema se estructura en torno a cuatro módulos fundamentales:

- **traffic-sim**: Módulo de simulación de tráfico urbano, que genera escenarios de prueba controlados y reproducibles para evaluar el comportamiento del sistema bajo diferentes condiciones de demanda vehicular. Permite emular redes viales con semáforos y flujos vehiculares variables.
- **traffic-control**: Módulo de orquestación que actúa como punto de entrada unificado al sistema. Recibe observaciones de tráfico, valida los datos, coordina las invocaciones a los servicios especializados de optimización y almacenamiento, y mantiene metadatos operativos en una base de datos relacional.
- **traffic-sync**: Módulo de análisis y optimización que procesa las métricas de tráfico recibidas, evalúa el nivel de congestión mediante técnicas de inteligencia computacional, y genera configuraciones optimizadas de tiempos semafóricos que buscan reducir demoras y mejorar el flujo vehicular.

- **traffic-storage**: Módulo de persistencia verificable que almacena datos del sistema en infraestructuras descentralizadas, garantizando trazabilidad, integridad e inmutabilidad de los registros históricos mediante el uso de tecnologías de almacenamiento distribuido y registro de auditoría.

3.0.2. Flujo General de Información

El flujo operativo del sistema sigue un ciclo cerrado de captura, análisis, optimización y registro:

1. **Captura de datos**: El módulo de simulación (o sensores reales en un despliegue futuro) genera observaciones de tráfico que incluyen métricas agregadas por intersección semafórizada.

2. **Validación y orquestación**: El módulo de control recibe las observaciones, valida su estructura y coherencia, y coordina el flujo de información hacia los servicios especializados.

3. **Análisis y optimización**: El módulo de sincronización procesa las métricas, clasifica el estado de congestión y determina configuraciones de tiempos semafóricos que mejoren el desempeño operativo.

4. **Persistencia verificable**: Los datos originales y los resultados de optimización se almacenan de forma distribuida, registrando identificadores de contenido en estructuras inmutables que permiten auditar y verificar el comportamiento histórico del sistema.

3.0.3. Ventajas de la Arquitectura Modular

La separación de responsabilidades en módulos independientes proporciona varios beneficios arquitectónicos y operativos:

- **Bajo acoplamiento**: Los módulos se comunican exclusivamente mediante interfaces REST bien definidas, lo que permite evolucionar o reemplazar componentes individuales sin afectar al resto del sistema.

- **Escalabilidad**: Cada módulo puede desplegarse de forma independiente en diferentes servidores o contenedores, permitiendo distribuir la carga computacional según las necesidades operativas.

- **Trazabilidad**: La combinación de almacenamiento distribuido con registros inmutables permite auditar el comportamiento del sistema, verificar la integridad de los datos y validar la conformidad con políticas operativas.

- **Reproducibilidad**: El uso de simulación permite evaluar el sistema bajo condiciones controladas, generando evidencias objetivas sobre el impacto de las estrategias de control propuestas.
- **Interoperabilidad**: La arquitectura basada en APIs REST facilita la integración con dispositivos IoT, plataformas de gestión urbana y otros sistemas externos que requieran consumir o producir datos de tráfico.

La Figura 3.1 ilustra de forma esquemática la organización modular del sistema, mostrando los cuatro módulos principales y sus relaciones.

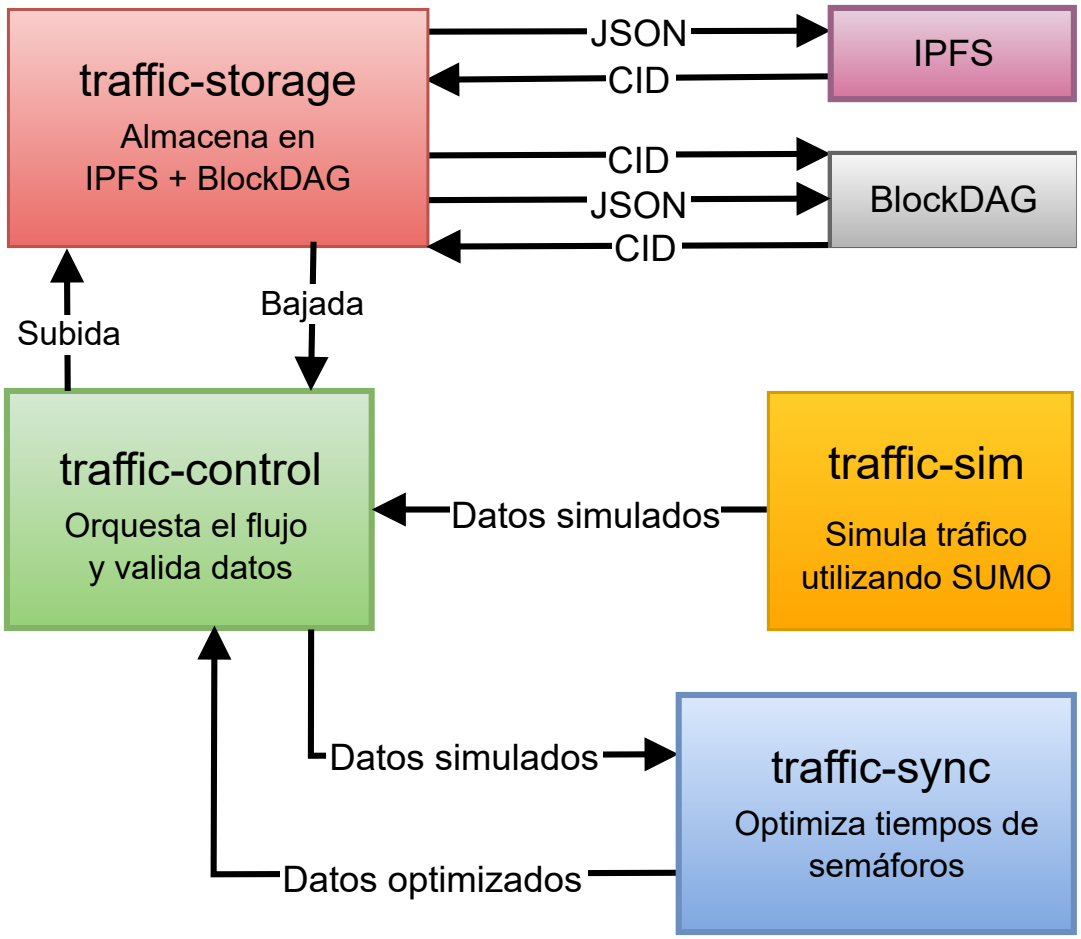

Figura 3.1: Arquitectura general del sistema mostrando los cuatro módulos principales y el flujo de información.

En síntesis, esta arquitectura modular proporciona una plataforma flexible y escalable para la gestión inteligente de tráfico urbano, combinando capacidades de simulación, análisis adaptativo y persistencia verificable en un sistema distribuido que puede integrarse con infraestructuras reales de monitoreo vehicular. Los capítulos subsiguientes describen en detalle la implementación, las tecnologías subyacentes y el funcionamiento de cada módulo.

CAPÍTULO 4
SIMULACIÓN DE TRÁFICO

4.1. Módulo de Simulación de Tráfico

El módulo `traffic-sim` implementa un entorno de simulación de tráfico urbano basado en SUMO (Simulation of Urban MObility), utilizado para evaluar el comportamiento del sistema propuesto bajo escenarios controlados y reproducibles. SUMO es un simulador microscópico de tráfico ampliamente adoptado en la comunidad científica, que permite modelar redes viales, rutas de vehículos y programas semafóricos, y ejecutar simulaciones paso a paso mediante una interfaz de control remoto (`traci`) [34]. En este trabajo, `traffic-sim` se integra con el servicio de optimización `traffic-sync` para cerrar el ciclo "simulación–detección–optimización–aplicación" sobre redes generadas a partir de datos reales o escenarios de prueba.

4.1.1. Arquitectura del Módulo

La estructura de `traffic-sim` se organiza en torno a un orquestador de simulación y un conjunto de componentes especializados para detección, control y comunicación externa. El orquestador de simulación encapsula la lógica principal: carga la configuración de SUMO (archivos de configuración de red, aristas, nodos, rutas y semáforos), inicializa la simulación mediante la interfaz de control remoto y gestiona el avance de la simulación en pasos discretos [34]. El punto de entrada recibe como argumento un archivo comprimido de escenario (generado, por ejemplo, mediante la herramienta auxiliar de construcción de mapas) y ejecuta la simulación en modo sin interfaz gráfica o con interfaz gráfica.

Los detectores de cuellos de botella monitorean métricas como densidad de vehículos, velocidad promedio y longitud de cola por tramo o aproximación. Los umbrales y parámetros de detección (por ejemplo, densidad mínima, velocidad máxima para considerar congestión, intervalo entre detecciones, duración mínima del evento) se centralizan en la configuración del

sistema, lo que permite ajustar la sensibilidad del sistema sin modificar la lógica de detección. El controlador de semáforos expone métodos para actualizar dinámicamente los tiempos de verde y de ciclo de los semáforos modelados en SUMO, aplicando las optimizaciones recibidas desde el servicio de optimización.

El cliente HTTP se comunica con la API del módulo de optimización: cuando se detecta un cuello de botella, este cliente construye un payload con métricas agregadas de tráfico para la intersección crítica y lo envía al endpoint correspondiente (por ejemplo, `/evaluate`). Las utilidades de cálculo y validación de métricas calculan y validan las métricas macroscópicas a partir de los datos de SUMO (por ejemplo, viajes completados, tiempos de viaje, velocidades y colas), garantizando que la información enviada al módulo de optimización siga el esquema esperado por el sistema difuso y el optimizador PSO.

Las herramientas de visualización proporcionan análisis posterior de resultados: parsers de archivos de salida de SUMO (información de viajes, resúmenes, datos de vehículos), funciones para generar gráficos de distribución de tiempos de viaje, evolución temporal de congestión y comparaciones entre diferentes configuraciones de semáforos, así como envoltorios sobre herramientas nativas de SUMO. Estas utilidades permiten cuantificar el impacto de las estrategias de control sobre métricas clave como tiempo de viaje, tiempos de espera y niveles de congestión.

La Figura 4.1 ilustra la arquitectura del módulo `traffic-sim`, mostrando los componentes principales y sus interacciones con SUMO y el servicio `traffic-sync`.

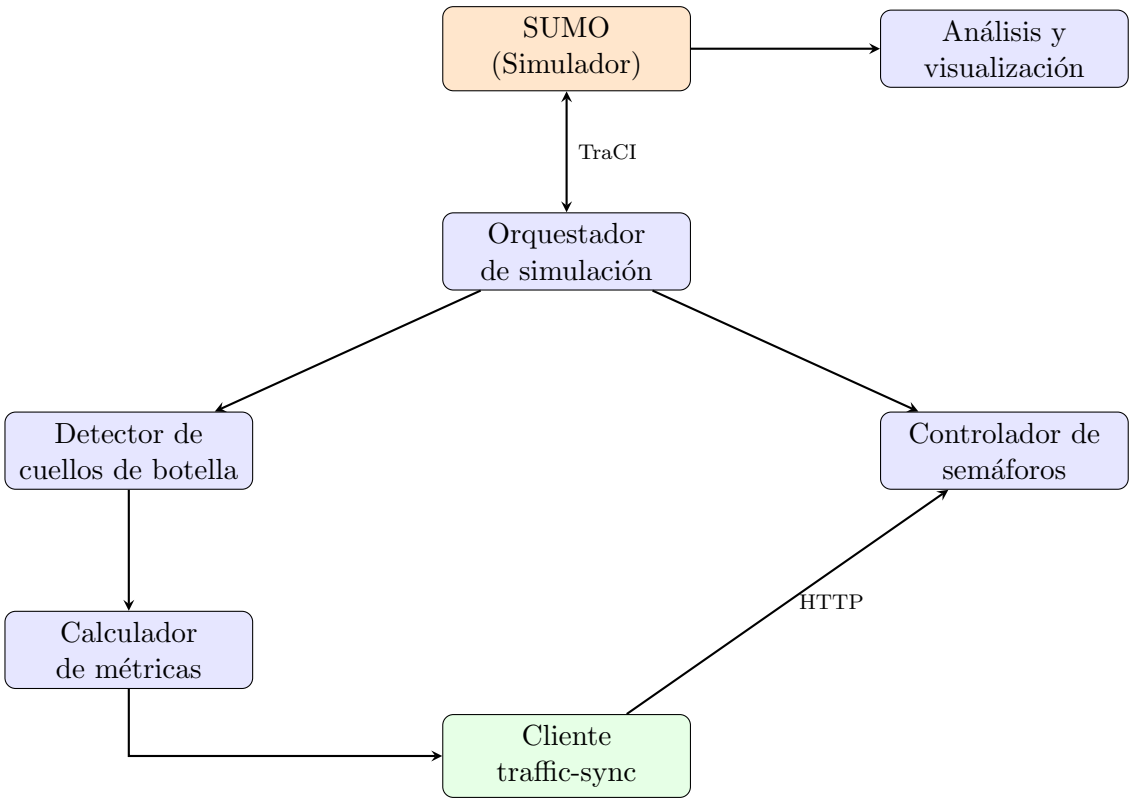

Figura 4.1: Arquitectura del módulo `traffic-sim` y sus componentes principales [34].

4.1.2. Flujo de Integración

El flujo de integración entre `traffic-sim` y `traffic-sync` sigue una secuencia cíclica que replica el comportamiento de un sistema de control de tráfico en línea:

1. **Ejecución de la simulación**: el orquestador inicia SUMO con el escenario suministrado y avanza la simulación paso a paso, según las rutas y patrones de demanda definidos en los archivos de configuración.

2. **Detección de cuellos de botella**: en intervalos configurables, el detector analiza para cada aproximación variables como densidad, velocidad y longitud de cola, y decide si existe un cuello de botella según los umbrales establecidos en la configuración del sistema.

3. **Construcción de métricas de tráfico**: cuando se confirma un cuello de botella en una intersección con semáforo, se calcula un conjunto de métricas agregadas para esa localización (vehículos por minuto, velocidad media en km/h, tiempo medio de circulación, densidad y estadísticos por tipo de vehículo), que se empaquetan en un payload JSON siguiendo el formato consumido por el módulo de optimización.

4. **Comunicación con el módulo de optimización**: el cliente HTTP envía el payload al servicio de optimización mediante una petición HTTP (por ejemplo, al endpoint `/evaluate`), y espera la respuesta con los tiempos de verde optimizados y el impacto estimado sobre la congestión.

5. **Aplicación de optimizaciones**: el controlador de semáforos actualiza, a través de la interfaz de control remoto, los tiempos de verde y rojo de los programas semafóricos correspondientes en la simulación SUMO, utilizando los valores recibidos desde el módulo de optimización.

6. **Continuación de la simulación**: la simulación se reanuda con los nuevos parámetros de control y se repite el proceso de detección y optimización mientras existan vehículos en red o hasta alcanzar el tiempo límite definido.

De esta manera, `traffic-sim` se comporta como un "laboratorio virtual" donde se puede evaluar en bucle el desempeño del controlador difuso Mamdani y del optimizador PSO integrados en el módulo de optimización, utilizando escenarios sintéticos o construidos a partir de mapas reales. La salida de herramientas auxiliares de construcción de escenarios (por ejemplo, generadores de redes y rutas compatibles con SUMO) sirve como entrada directa para el módulo de simulación, lo que facilita la creación sistemática de casos de prueba. Los resultados de las simulaciones, incluyendo métricas temporales y distribuciones de tiempos de viaje, se analizan posteriormente mediante las utilidades de visualización para cuantificar de forma objetiva el impacto de las políticas de control propuestas.

CAPÍTULO 5

ALMACENAMIENTO DISTRIBUIDO Y TECNOLOGÍAS DESCENTRALIZADAS

5.1. Módulo de Almacenamiento de Datos

El módulo `traffic-storage` actúa como capa de persistencia verificable del sistema, recibiendo métricas de tráfico y resultados de optimización, almacenando los datos en un repositorio distribuido y registrando huellas inmutables en infraestructuras descentralizadas. En este contexto, blockchain, BlockDAG e IPFS se utilizan de forma complementaria: blockchain y BlockDAG como ledgers inmutables para trazabilidad, e IPFS como almacén de archivos orientado a contenido [35, 36, 37].

5.1.1. Blockchain

Blockchain puede describirse como un libro mayor distribuido donde las transacciones se agrupan en bloques enlazados criptográficamente en una cadena lineal, de modo que cada bloque referencia al anterior mediante un hash, proporcionando inmutabilidad y resistencia a manipulaciones retrospectivas [35, 38]. Los nodos de la red ejecutan un mecanismo de consenso (como Proof-of-Work o Proof-of-Stake) para acordar qué bloques se consideran válidos, garantizando que todos mantengan una vista consistente del historial de transacciones [39, 40].

La Figura 5.1 ilustra la estructura básica de una cadena de bloques, mostrando cómo cada bloque referencia criptográficamente al bloque anterior mediante su hash, creando una cadena inmutable donde cualquier modificación en un bloque anterior invalida todos los bloques subsiguientes.

En este trabajo, blockchain se utiliza para registrar metadatos de almacenamiento de tráfico de forma inmutable, actuando como capa de "prueba de existencia" para datos que residen

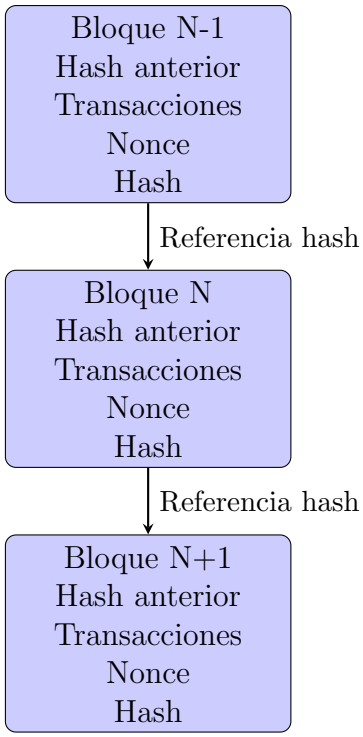

Figura 5.1: Estructura básica de una cadena de bloques, mostrando la dependencia criptográfica entre bloques consecutivos [35].

físicamente fuera de la cadena. El contrato inteligente de almacenamiento expone operaciones de alto nivel para registrar eventos de almacenamiento: cada llamada almacena, al menos, un identificador de semáforo, una marca de tiempo y un identificador de contenido (CID de IPFS), junto con el tipo de dato al que hace referencia. Este contrato se despliega y prueba utilizando un entorno de desarrollo estándar, con scripts de despliegue y pruebas automatizadas que verifican su funcionamiento básico.

El módulo `traffic-storage` ofrece una API HTTP que recibe cargas de datos en un formato unificado (versión 2.0), generadas por otros componentes del sistema (como los módulos de optimización y lógica difusa). Cuando llega una nueva medición o reporte, la API valida el payload, delega la persistencia de contenido a IPFS y, una vez obtenido el CID, invoca al contrato inteligente de almacenamiento para registrar ese CID junto con los metadatos relevantes en la blockchain. De este modo, los algoritmos de optimización y los módulos de monitoreo solo interactúan con una interfaz REST, mientras que los detalles de interacción on-chain quedan encapsulados en el módulo de almacenamiento.

5.1.2. BlockDAG

BlockDAG (Block Directed Acyclic Graph) generaliza la estructura lineal de blockchain permitiendo que cada bloque referencie a múltiples bloques padres anteriores, formando un grafo dirigido acíclico en lugar de una cadena única [41, 37]. Esta arquitectura permite que bloques creados en paralelo se integren en el consenso sin ser descartados como huérfanos, mejorando el aprovechamiento del ancho de banda de la red y posibilitando mayores tasas de

transacciones y menores latencias de confirmación que las típicamente alcanzables en cadenas lineales [41].

La Tabla 5.1 sintetiza las principales diferencias entre arquitecturas blockchain tradicionales y BlockDAG basadas en protocolos como GHOSTDAG. La comparación considera protocolos representativos de cada paradigma: Nakamoto consensus implementado en Bitcoin y Ethereum para blockchains lineales [35, 39], y PHANTOM/GHOSTDAG como generalización del consenso distribuido en estructuras DAG [37].

Cuadro 5.1: Comparación entre Blockchain Tradicional y BlockDAG

Característica	**Blockchain Tradicional**	**BlockDAG (GHOSTDAG)**
Topología	Lineal (1 padre)[37]	Grafo acíclico (múltiples padres)[37, 42]
Bloques huérfanos	Descartados[37]	Integrados como padres[37]
Ordenamiento	Total (intrínseco)	Parcial → Total (vía algoritmo)[37]
Escalabilidad	Limitada[37]	Alta (ancho de banda físico)[37, 42]
Confirmación	Probabilística lenta	Probabilística rápida (segundos)[42]
Minería	Competitiva[37]	Cooperativa[37]
Throughput típico	7-30 TPS[35, 39]	100+ TPS[42]
Latencia de bloque	10 min (Bitcoin), 12 s (Ethereum)[43]	Sub-segundo a segundos[42]

En el contexto de este sistema, BlockDAG se considera como infraestructura de registro distribuido complementaria a la blockchain tradicional, pensada para escenarios de mayor escala donde múltiples intersecciones reportan datos con alta frecuencia. El módulo `traffic-storage` incorpora una capa de abstracción que permite, según la configuración, registrar eventos de almacenamiento tanto en un contrato EVM como en una red BlockDAG compatible, utilizando un cliente dedicado que construye y envía mensajes de registro a un nodo remoto. Estos mensajes contienen, de forma análoga a la variante blockchain, el identificador de semáforo, la marca de tiempo y el hash de contenido (CID de IPFS u otro identificador criptográfico).

La lógica de alto nivel en el módulo de almacenamiento se mantiene independiente de la tecnología subyacente: la API recibe métricas y resultados de optimización, delega la persistencia de archivos a IPFS y, posteriormente, invoca una interfaz de registro de evidencia"que puede mapearse a blockchain, BlockDAG o a un esquema dual. Esta separación permite que, a medida que se exploren o adopten redes DAG de mayor throughput para despliegues reales en entornos de ciudad inteligente, el resto del sistema (clasificación de congestión, PSO, paneles de monitoreo) continúe interactuando con un mismo punto de entrada consistente.

5.1.3. IPFS

El InterPlanetary File System (IPFS) es un sistema de archivos distribuido peer-to-peer que utiliza direccionamiento por contenido: cada objeto se identifica por un Content Identifier (CID) que codifica un hash del contenido más metadatos mínimos sobre cómo interpretarlo [36]. El mismo archivo siempre produce el mismo CID, independientemente del nodo que lo procese,

lo que permite verificar integridad de manera determinista y realizar deduplicación automática a escala de red [44].

La Figura 5.2 ilustra el flujo básico de almacenamiento y recuperación en IPFS, mostrando cómo un archivo se fragmenta, se le asigna un CID basado en su contenido, y puede ser recuperado desde cualquier nodo que posea los bloques correspondientes.

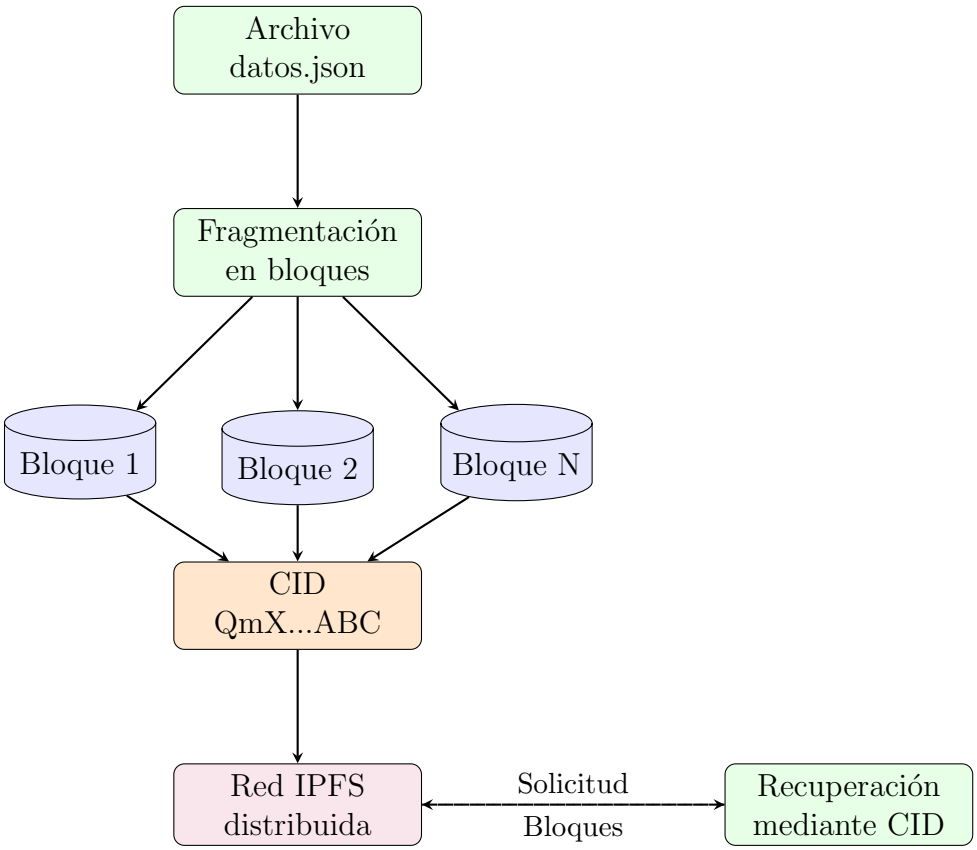

Figura 5.2: Arquitectura básica de IPFS: fragmentación de archivos, generación de CID y recuperación distribuida [36].

En este trabajo, IPFS se emplea como almacén de datos off-chain para los artefactos generados y consumidos por el sistema: mediciones agregadas de tráfico por clúster, indicadores de congestión difusa, configuraciones optimizadas de tiempos semafóricos y reportes experimentales. El módulo `traffic-storage` incluye un cliente específico para interactuar con un nodo IPFS local o con un servicio compatible, ofreciendo operaciones sencillas para subir y recuperar objetos. El flujo básico consiste en que la API recibe un payload de datos, lo serializa (por ejemplo, como JSON), lo envía al nodo IPFS y obtiene de vuelta un CID.

Ese CID se utiliza a continuación como enlace entre IPFS y las capas de consenso: se registra en blockchain o en BlockDAG mediante las interfaces descritas anteriormente, de modo que cualquier módulo que necesite auditar o reanalizar un conjunto de datos solo requiera conocer el CID y el identificador de semáforo asociado. Desde la perspectiva del resto del sistema, IPFS y las tecnologías de registro inmutable quedan encapsuladas detrás del módulo de almacenamiento: los módulos de simulación, clasificación difusa y optimización interactúan únicamente con la API REST, mientras que las garantías de verificabilidad, integridad y persistencia se logran

combinando direccionamiento por contenido en IPFS con registros inmutables en blockchain y/o BlockDAG.

La Figura 5.3 presenta un diagrama de secuencia que ilustra el flujo completo de almacenamiento distribuido: desde la recepción de datos por la API hasta el registro del CID en la capa de consenso, mostrando la interacción entre los componentes del módulo de almacenamiento.

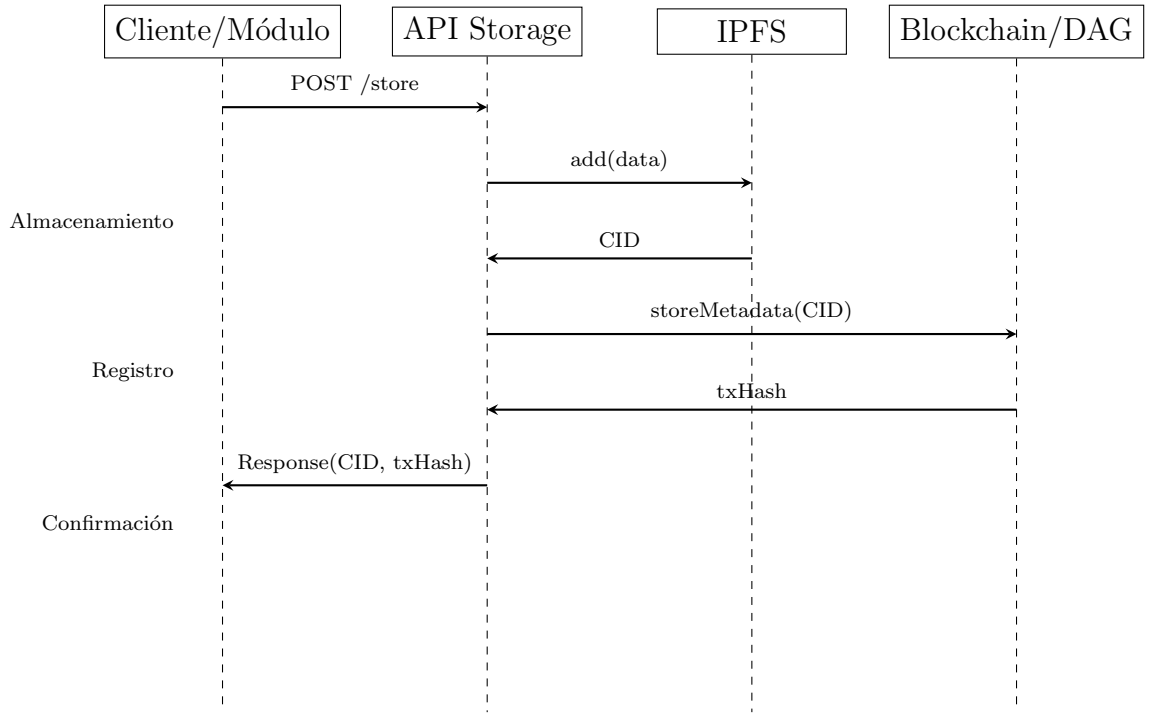

Figura 5.3: Diagrama de secuencia del flujo de almacenamiento distribuido en `traffic-storage`.

CAPÍTULO 6

OPTIMIZACIÓN Y SINCRONIZACIÓN DE TRÁFICO

6.1. Módulo de Optimización de Tiempos Semafóricos

El módulo `traffic-sync` implementa el servicio de optimización de tráfico del sistema, exponiendo una API que recibe mediciones macroscópicas de tráfico por semáforo y devuelve configuraciones de tiempos optimizadas. Para ello combina dos componentes principales: un sistema de inferencia difusa tipo Mamdani, que evalúa el nivel de congestión a partir de variables de flujo, velocidad y densidad; y un algoritmo de optimización Particle Swarm Optimization (PSO), que ajusta dinámicamente los tiempos de verde con el objetivo de reducir la congestión estimada [17, 45, 14].

6.1.1. Lógica Difusa Mamdani

La lógica difusa proporciona un marco formal para representar razonamiento humano impreciso mediante variables lingüísticas y conjuntos difusos con grados de pertenencia entre 0 y 1 [45, 46]. En un sistema de inferencia Mamdani, las reglas tienen la forma "SI condición ENTONCES acción", donde tanto antecedentes como consecuentes se modelan como conjuntos difusos, y la salida final se obtiene tras un proceso de agregación y defuzzificación [17, 24]. Este enfoque resulta especialmente adecuado para describir estados de tráfico como "flujo libre", "denso" o "congestionado" a partir de variables macroscópicas continuas [25, 18].

La Figura 6.1 ilustra un ejemplo de variable lingüística para velocidad promedio con tres términos (baja, media, alta) definidos mediante funciones de membresía, mostrando cómo un mismo valor numérico puede tener diferentes grados de pertenencia a múltiples categorías lingüísticas.

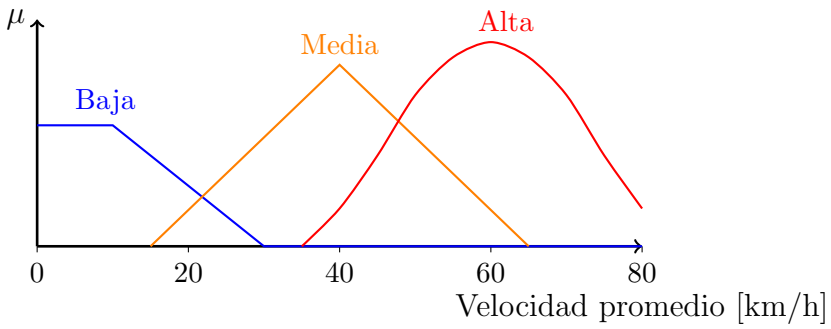

Figura 6.1: Variable lingüística *velocidad promedio* con tres conjuntos difusos modelados mediante funciones de membresía trapezoidal, triangular y aproximadamente gaussiana [24].

En `traffic-sync`, el sistema de inferencia difusa se implementa en el módulo de lógica difusa, que está compuesto principalmente por el módulo de definición del sistema y el módulo de evaluación. El módulo de definición del sistema define las variables lingüísticas de entrada (por ejemplo, vehículos por minuto, velocidad media, tiempo medio de circulación, densidad) y la variable de salida asociada al nivel de congestión, junto con sus funciones de membresía triangulares y trapezoidales calibradas en base a datos de tráfico y criterios de ingeniería [24, 25]. La base de reglas Mamdani captura relaciones cualitativas del tipo "SI flujo es alto Y velocidad es baja ENTONCES congestión es severa", que reflejan tanto conocimiento experto como patrones observados en los datos [18].

La Figura 6.2 resume el flujo de procesamiento en un sistema de inferencia Mamdani aplicado al problema de congestión vehicular, mostrando las etapas desde la fuzzificación de entradas hasta la defuzzificación que produce el nivel de congestión final.

El módulo de evaluación expone funciones de alto nivel que reciben vectores de métricas de tráfico en formato estructurado y devuelven, para cada sensor o semáforo, un valor numérico de congestión y su categoría lingüística asociada. Estas funciones se utilizan en el pipeline principal del servicio web: el endpoint `/evaluate` recibe un lote de sensores, construye las entradas para el sistema difuso, ejecuta la evaluación Mamdani sobre cada uno y genera un conjunto de salidas que se utilizarán como insumo para la etapa de optimización PSO. De esta forma, la lógica difusa actúa como módulo de evaluación de desempeño que transforma mediciones de tráfico en un índice de congestión interpretable, que el optimizador buscará reducir.

6.1.2. Optimización de Enjambre de Partículas (PSO)

Optimización de Enjambre de Partículas (PSO) es un algoritmo de optimización metaheurística inspirado en el comportamiento colectivo de bandadas de aves y bancos de peces, en el que un conjunto de partículas explora el espacio de búsqueda guiado por su mejor experiencia individual y la mejor experiencia global del grupo [14, 48]. Cada partícula representa una solución candidata (en este caso, una posible asignación de tiempo de verde para un grupo de semáforos) y actualiza su posición combinando tres componentes: inercia, atracción hacia su mejor posición personal y atracción hacia la mejor posición global conocida, ponderadas por

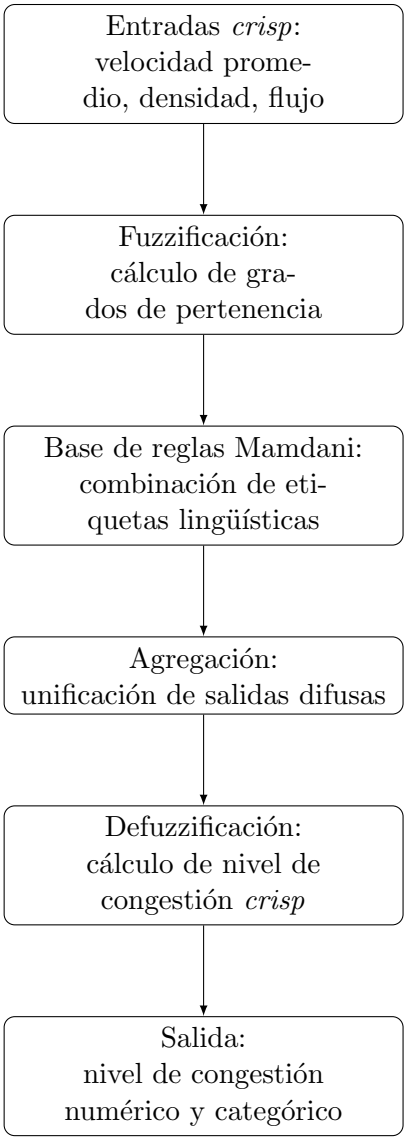

Figura 6.2: Etapas principales de un sistema de inferencia difusa tipo Mamdani aplicado a congestión vehicular [17, 47].

parámetros que controlan el equilibrio entre exploración y explotación [49, 50].

La Figura 6.3 ilustra esquemáticamente el concepto del enjambre PSO en un espacio de búsqueda bidimensional, mostrando cómo las partículas se mueven influenciadas por su mejor experiencia personal (p_i) y la mejor experiencia global del grupo (g).

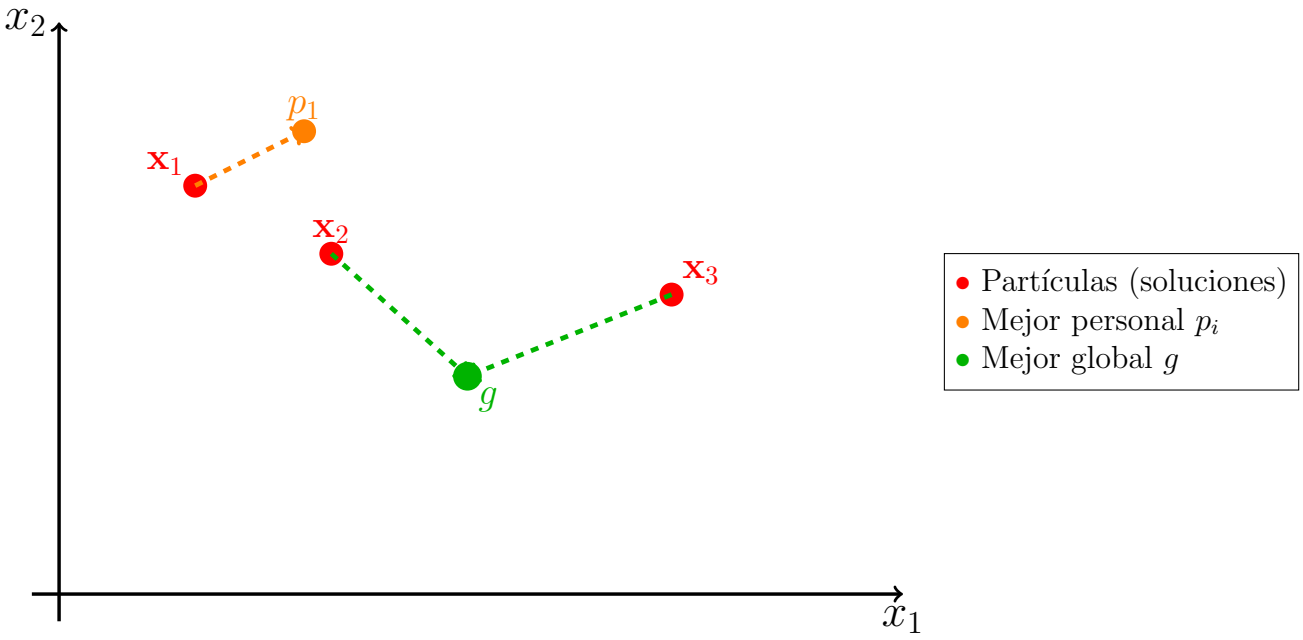

Figura 6.3: Esquema conceptual del enjambre PSO en un espacio de búsqueda bidimensional [14].

En el módulo `traffic-sync`, la lógica de PSO se implementa en el módulo de optimización por enjambre de partículas, que incluye el módulo de función de aptitud y el módulo de optimización. El módulo de optimización define la función principal de PSO, que recibe información agregada de tráfico por grupo (cluster) y aplica PSO para determinar el tiempo de verde óptimo por grupo. Cada partícula codifica un tiempo de verde propuesto para el ciclo semafórico, inicializado alrededor de un valor base calculado mediante fórmulas clásicas de ingeniería de tráfico (función de cálculo de tiempo de verde en el módulo de aptitud), y restringido por límites mínimos y máximos coherentes con la normativa [10, 1]. El algoritmo utiliza configuraciones típicas de PSO (número de partículas, peso de inercia, componentes cognitiva y social, límites de velocidad) y añade mecanismos prácticos como parada temprana y reinicios parciales para evitar estancamiento en óptimos locales pobres.

La función de aptitud evalúa cada propuesta de tiempo de verde combinando la lógica difusa y las métricas de tráfico: dado un tiempo de verde candidato, recalcula métricas derivadas (como volumen efectivo, velocidad y densidad ajustadas) y vuelve a evaluar el nivel de congestión mediante el sistema Mamdani, obteniendo un valor numérico que se utiliza como medida a minimizar [5, 6]. De este modo, PSO busca explícitamente reducir el índice de congestión proporcionado por el módulo difuso, en lugar de optimizar una función analítica cerrada.

En el pipeline de `traffic-sync`, orquestado por la función principal del servicio web, el

flujo es el siguiente: (i) el servicio recibe un lote de sensores en la API `/evaluate`; (ii) las mediciones se pasan al módulo difuso para obtener un nivel de congestión por sensor; (iii) se agrupan sensores con características similares (agrupamiento jerárquico), produciendo un conjunto de grupos de tráfico; (iv) se construye un conjunto de datos por grupo con métricas agregadas (por ejemplo, vehículos por minuto, velocidad media, densidad media, categoría de congestión); y (v) se aplica la función de optimización PSO sobre estos grupos para obtener tiempos de verde optimizados por grupo. La respuesta de la API se construye como un lote de optimizaciones, donde cada entrada incluye el tiempo de verde recomendado, el tiempo de rojo derivado, el nivel de congestión original y optimizado, y las categorías lingüísticas correspondientes, preparados para su consumo posterior por el módulo de control de semáforos y por el módulo de almacenamiento.

En conjunto, la integración entre lógica difusa Mamdani y PSO en `traffic-sync` permite que la optimización de tiempos semafóricos se base en una medida de congestión que captura adecuadamente la percepción y el comportamiento del tráfico urbano, mientras que el metaheurístico proporciona un mecanismo flexible para ajustar los parámetros de control en escenarios donde la función objetivo no es derivable y debe evaluarse a través de simulación o reglas de inferencia [24, 51].

La Figura 6.4 ilustra la arquitectura general del módulo `traffic-sync`, mostrando la interacción entre los componentes de lógica difusa y PSO, así como el flujo de datos desde la recepción de métricas de tráfico hasta la generación de configuraciones optimizadas de tiempos semafóricos.

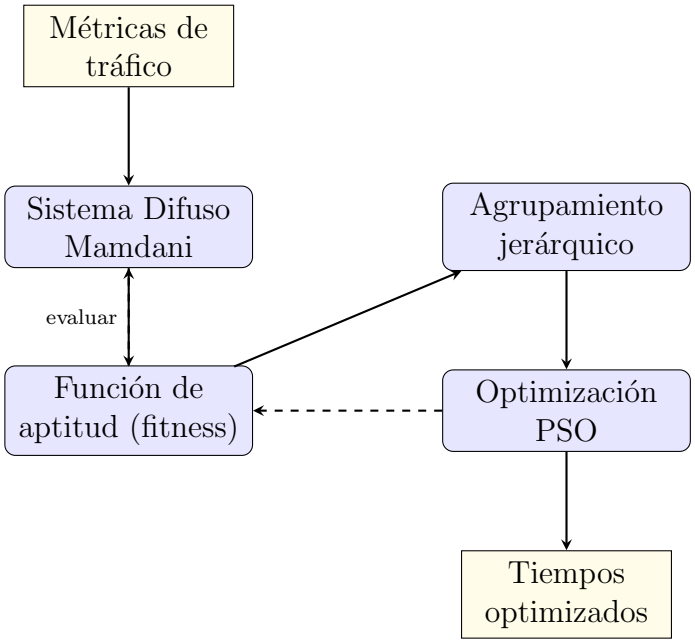

Figura 6.4: Arquitectura del módulo `traffic-sync` integrando lógica difusa Mamdani y optimización PSO.

CAPÍTULO 7

CONTROL Y ORQUESTACIÓN DEL SISTEMA

7.1. Módulo de Control y Orquestación

El módulo `traffic-control` actúa como orquestador central del sistema distribuido de gestión de tráfico: recibe observaciones vehiculares desde simulaciones o despliegues reales, valida y registra los datos, coordina el envío de información hacia los servicios de optimización (`traffic-sync`) y almacenamiento distribuido (`traffic-storage`), y expone una API REST unificada para clientes externos. Su objetivo es desacoplar los productores de datos (por ejemplo, `traffic-sim`) de los módulos especializados de optimización y persistencia, garantizando un flujo coherente y trazable de información a través del sistema.

7.1.1. Arquitectura del Módulo

La estructura de `traffic-control` se organiza en capas bien definidas: configuración, API, modelos de datos, servicios de dominio y utilidades de soporte. El servicio web principal (basado en FastAPI) implementa el endpoint `/process`, que procesa observaciones vehiculares completas, y endpoints auxiliares como `/healthcheck` y operaciones sobre metadatos. El módulo de configuración centraliza parámetros del sistema, como las URLs de los servicios de almacenamiento y optimización, la cadena de conexión a la base de datos y los niveles de logging. El módulo de configuración de logs define la estructura de registro para toda la aplicación.

La capa de modelos incluye esquemas de validación para estructurar peticiones y respuestas (esquemas de datos, optimización y descarga) y modelos de respuesta estandarizados. El módulo de validación encapsula la validación semántica de los datos: verifica versión del payload, formato de marcas de tiempo, tipo de mensaje (datos u optimización), límites de tamaño

de lote (1–10 sensores) e integridad de campos como identificador de semáforo, métricas y estadísticas vehiculares. Cualquier inconsistencia o violación de estos contratos se transforma en una respuesta de error coherente mediante el manejador centralizado de errores.

La persistencia principal del sistema se realiza mediante IPFS y BlockDAG a través del módulo de almacenamiento, que garantiza almacenamiento distribuido, inmutabilidad y trazabilidad verificable. Complementariamente, el módulo de base de datos gestiona un índice auxiliar de metadatos en una base de datos SQL local, con el punto de entrada a la conexión de base de datos y el modelo de metadatos definiendo el esquema de almacenamiento: cada registro guarda, como mínimo, el identificador del semáforo, la marca de tiempo, el tipo de dato (por ejemplo, observación de tráfico u optimización) y referencias a los identificadores de contenido (CIDs y hashes de transacción) almacenados en el módulo de almacenamiento. El servicio de base de datos proporciona operaciones de alto nivel para crear y consultar este índice local, que se exponen a través de endpoints de metadatos (por ejemplo, `/metadata/traffic-light/{id}`, `/metadata/recent`, `/metadata/stats`) para facilitar consultas rápidas, mientras que los datos completos permanecen en el almacenamiento distribuido.

La Figura 7.1 ilustra la arquitectura del módulo `traffic-control`, mostrando las capas de API, validación, servicios de dominio y persistencia, así como la integración con los módulos especializados del sistema.

7.1.2. Flujo de Procesamiento

El endpoint `POST /process` es el punto de entrada principal para observaciones de tráfico, tanto provenientes de `traffic-sim` como de otros clientes. El flujo de procesamiento sigue una secuencia de pasos bien definida:

1. **Recepción y validación**: la API recibe un payload en formato unificado (versión 2.0), que puede contener uno o varios sensores asociados a un mismo identificador de semáforo. El módulo de validación verifica que el mensaje cumpla con la estructura esperada (campos obligatorios en métricas y estadísticas vehiculares, rango de tamaño de lote, formatos de fecha, tipo de mensaje), generando errores explícitos en caso de inconsistencias.

2. **Preprocesamiento de datos**: el servicio de procesamiento de datos puede aplicar transformaciones ligeras sobre los datos (normalización de campos, ajuste de formatos, enriquecimiento con información temporal) antes de enviarlos a los módulos externos. Esta capa asegura que, aunque distintos productores tengan pequeñas variaciones, el sistema trabaje con un formato interno coherente.

3. **Almacenamiento distribuido principal**: el proxy de almacenamiento construye una petición hacia la API del módulo de almacenamiento, que implementa el almacenamiento principal del sistema mediante IPFS y BlockDAG. Los datos completos se almacenan de forma distribuida e inmutable, recibiendo de vuelta identificadores de contenido (CIDs de IPFS) y hashes de transacción en BlockDAG. Este es el mecanismo de persistencia

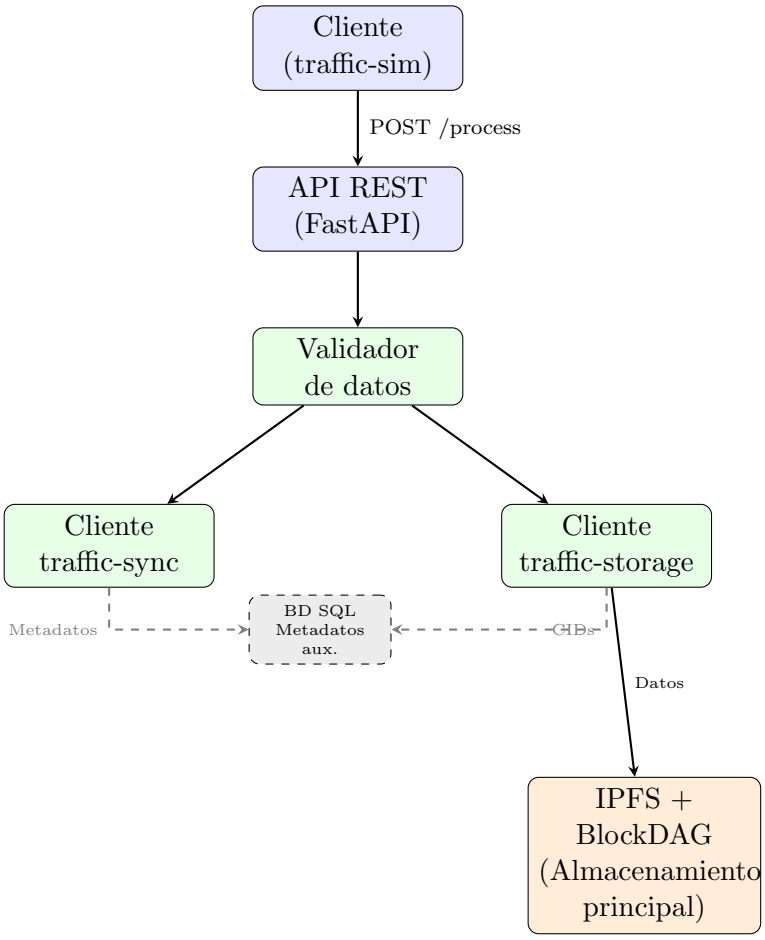

Figura 7.1: Arquitectura del módulo `traffic-control` mostrando el flujo de orquestación. El almacenamiento principal se realiza mediante IPFS y BlockDAG a través de `traffic-storage`, mientras que la base de datos SQL se utiliza únicamente como índice auxiliar de metadatos.

central del sistema, garantizando trazabilidad, integridad e inmutabilidad de los registros históricos.

4. **Índice auxiliar de metadatos**: mediante el servicio de base de datos, se registra en una base de datos SQL local un resumen del evento (identificador de semáforo, marca de tiempo, tipo, estado del procesamiento, CIDs y hashes de transacción), lo que permite consultas rápidas sobre qué datos fueron recibidos, cuándo y con qué resultado. Esta base de datos actúa únicamente como índice auxiliar para facilitar consultas locales, complementando los registros inmutables almacenados en IPFS y BlockDAG gestionados por el módulo de almacenamiento.

5. **Interacción con el módulo de optimización**: el proxy de sincronización envía los datos de tráfico al módulo de optimización mediante el endpoint `/evaluate`, en formato compatible con el sistema difuso y el optimizador PSO. Para lotes de sensores, la función de envío por lotes reempaqueta el lote en el formato esperado por el módulo de optimización (incluyendo el campo de identificador de semáforo de referencia) y recibe un conjunto de optimizaciones, una por grupo detectado.

6. **Respuesta al cliente**: finalmente, el servicio de procesamiento compone una respuesta estandarizada utilizando los modelos de respuesta definidos, indicando el estado del procesamiento (por ejemplo, éxito o error), un mensaje descriptivo y, cuando corresponda, datos derivados como tiempos de verde optimizados e impacto estimado sobre la congestión. Esta respuesta se devuelve al cliente original (por ejemplo, el módulo de simulación), que puede aplicar las decisiones de control correspondientes.

Durante todo el flujo, el manejador centralizado de errores proporciona mecanismos para capturar y clasificar errores en distintas categorías (validación, almacenamiento, sincronización, base de datos, errores genéricos), generando respuestas HTTP adecuadas y registros de log con contexto detallado para facilitar la depuración.

7.1.3. Interacción con los Módulos

Desde el punto de vista del sistema global, `traffic-control` se posiciona como el módulo de orquestación que articula la interacción entre simulación, optimización y almacenamiento distribuido:

- **Relación con `traffic-sim`**: el módulo de simulación basado en SUMO detecta cuellos de botella y, cuando identifica una situación crítica en una intersección, construye un payload con métricas agregadas de tráfico y lo envía a `traffic-control` mediante el endpoint `/process`. De este modo, `traffic-sim` no necesita conocer detalles sobre `traffic-sync` ni `traffic-storage`; sólo interactúa con un servicio central que gestiona validación, almacenamiento y consulta de optimizaciones.

La Figura 7.2 muestra un diagrama de secuencia detallado del flujo completo de procesamiento en `traffic-control`, ilustrando las interacciones entre todos los componentes del sistema desde la recepción de datos hasta la persistencia distribuida.

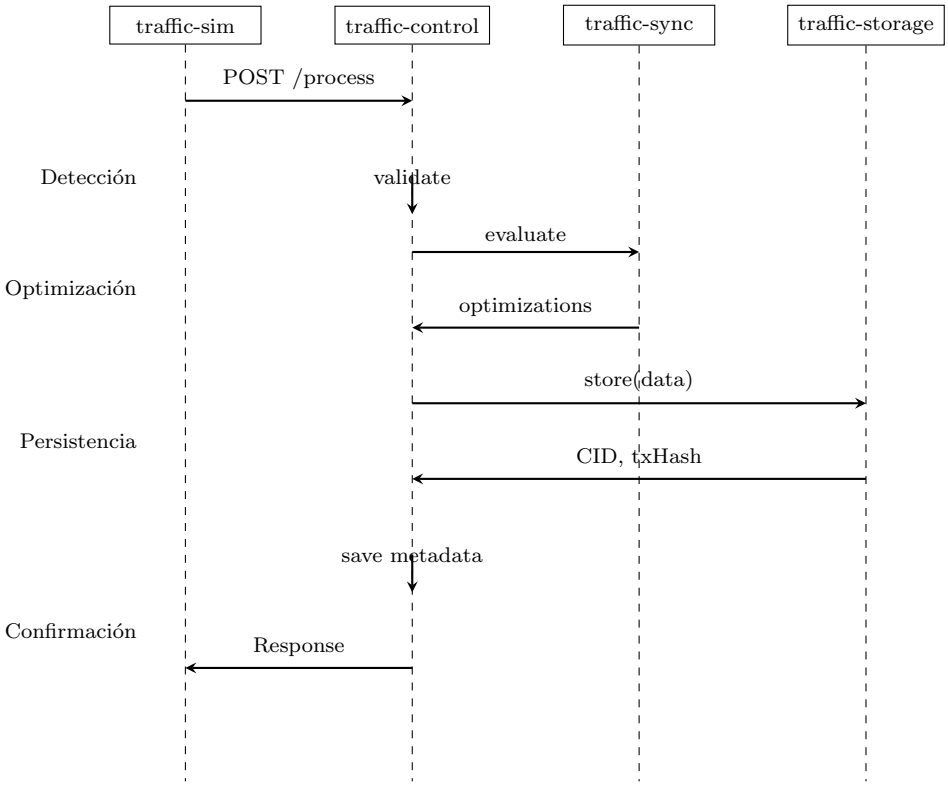

Figura 7.2: Diagrama de secuencia del flujo completo de procesamiento en el sistema distribuido.

- **Relación con el módulo de optimización**: `traffic-control` actúa como cliente del módulo de optimización a través del proxy de sincronización. Para cada observación (o lote) recibida, reenvía los datos relevantes al módulo de optimización, que ejecuta el pipeline de lógica difusa y PSO para producir tiempos de verde optimizados y categorías de congestión. La respuesta del módulo de optimización se integra en la respuesta de `traffic-control`, permitiendo que los consumidores (por ejemplo, el módulo de simulación) apliquen las configuraciones recomendadas.
- **Relación con el módulo de almacenamiento**: mediante el proxy de almacenamiento, `traffic-control` sube y recupera datos hacia/desde el módulo de almacenamiento, que a su vez utiliza IPFS, blockchain y BlockDAG como backend de almacenamiento y registro inmutable. `traffic-control` registra en su base de datos los identificadores de contenido (por ejemplo, CIDs) devueltos por el módulo de almacenamiento, de modo que las consultas de metadatos pueden resolverse localmente y, cuando es necesario, seguir la cadena de referencias hasta los datos almacenados en el sistema distribuido.

Esta arquitectura modular permite que cada componente se especialice en su responsabilidad principal: `traffic-sim` en la generación de escenarios y detección de cuellos de botella; `traffic-sync` en la evaluación de congestión y optimización de tiempos semafóricos

mediante lógica difusa y PSO; `traffic-storage` en la persistencia verificable de datos; y `traffic-control` en la validación, coordinación y exposición de un punto de entrada unificado al sistema. En conjunto, estos módulos forman una plataforma escalable y descentralizada para la gestión inteligente de tráfico urbano.

CAPÍTULO 8

capitulo

CAPÍTULO 9

CONCLUSIÓN

REFERENCIAS

[1] Nicholas J. Garber y Lester A. Hoel. *Traffic and Highway Engineering*. 6th. Boston: Cengage Learning, 2019.

[2] Roger P. Roess, Elena S. Prassas y William R. McShane. *Traffic Engineering*. 5th. Upper Saddle River: Pearson, 2019.

[3] Federal Highway Administration. *Traffic Signal Timing Manual*. `https://ops.fhwa.dot.gov/publications/fhwahop08024/`. FHWA-HOP-08-024, consultado: 2026-01-XX. 2008.

[4] Ministerio de Obras Públicas y Comunicaciones. *Manual de Carreteras del Paraguay. Unidad 3: Diseño de Carreteras. Volumen 3.3: Señalización y Obras Complementarias*. Sección 3.3.2.7: Semáforos. Asunción, Paraguay: MOPC, 2019.

[5] Markos Papageorgiou, Christina Diakaki, Vaya Dinopoulou, Apostolos Kotsialos y Yibing Wang. «Review of Road Traffic Control Strategies». En: *Proceedings of the IEEE* 91.12 (2003), págs. 2043-2067. DOI: `10.1109/JPROC.2003.819610`.

[6] Aleksandar Stevanovic, Jelena Stevanovic, Kai Zhang y Stuart Batterman. «Optimizing Traffic Control to Reduce Fuel Consumption and Vehicular Emissions: Integrated Approach with VISSIM, CMEM, and VISGAOST». En: *Transportation Research Record* 2128 (2010). Incluye comparación de desempeño entre planes fijos y controladores adaptativos, págs. 105-113. DOI: `10.3141/2128-12`.

[7] Nathan H. Gartner, John D. C. Little y Henry Gabbay. «Optimization of Traffic Signal Settings by Mixed-Integer Linear Programming: Part I: The Network Coordination Problem». En: *Transportation Science* 9.4 (1975). Formulación MILP para optimización de redes semafóricas, págs. 321-343. DOI: `10.1287/trsc.9.4.321`.

[8] S. C. Wong, H. Yang y H. K. Lo. «Traffic Signal Optimisation: Modeling and Computation». En: *Transportation Research Part B: Methodological* 32.3 (1998). Formulación matemática del problema de optimización de señales, págs. 157-172. DOI: `10.1016/S0191-2615(97)00028-X`.

[9] J. A. Hillier y R. Rothery. «The Synchronization of Traffic Signals for Minimum Delay». En: *Transportation Science* 1.2 (1967). Clásico sobre formulación de optimización de señales, págs. 81-94. DOI: `10.1287/trsc.1.2.81`.

[10] F. V. Webster. «Traffic Signal Settings». En: *Road Research Technical Paper* 39 (1958). Clásico método de diseño de tiempos semafóricos.

[11] Byungkyu Park, Carroll J. Messer y Thomas Urbanik. «Traffic Signal Optimization Program for Oversaturated Conditions: Genetic Algorithm Approach». En: *Transportation Research Record* 1683 (1999). Uso de algoritmos genéticos para condiciones sobresaturadas, págs. 133-142. DOI: 10.3141/1683-17.

[12] Halim Ceylan y Michael G. H. Bell. «Traffic Signal Timing Optimization Based on Genetic Algorithm Approach, Including Drivers' Routing». En: *Transportation Research Part B: Methodological* 38.4 (2004). Algoritmos genéticos para optimización semafórica, págs. 329-342. DOI: 10.1016/S0191-2615(03)00015-8.

[13] Nathan H. Gartner, Farhad J. Pooran y Carleton M. Andrews. *Optimized Policies for Adaptive Control: A Case Study for Arterial Traffic*. Capítulos sobre control actuado y coordinado. Boston: Springer, 2001.

[14] James Kennedy y Russell Eberhart. «Particle Swarm Optimization». En: *Proceedings of the IEEE International Conference on Neural Networks*. Vol. 4. IEEE, 1995, págs. 1942-1948. DOI: 10.1109/ICNN.1995.488968.

[15] X. Li et al. «A PSO Based Signal Timing Optimization Approach of Phase Combination». En: *Proceedings of the International Conference on Computer Technology and Engineering*. Aplicación de PSO a optimización de tiempos semafóricos. Atlantis Press, 2016, págs. 116-120.

[16] R. Gonçalo et al. «On Tuning the Particle Swarm Optimization for Solving the Traffic Light Time Problem». En: *ICCSA 2022 – Lecture Notes in Computer Science*. PSO integrado con SUMO para optimización de semáforos. Springer, 2022.

[17] Ebrahim H. Mamdani y Sedrak Assilian. «An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller». En: *International Journal of Man-Machine Studies* 7.1 (1975), págs. 1-13. DOI: 10.1016/S0020-7373(75)80002-2.

[18] Rizka Zuliani Musriroh. «Application of Mamdani Method on Fuzzy Logic to Decision of Traffic Light System». En: *Proceedings of International Conference on Informatics and Computational Sciences*. SciTePress, 2020, pág. 85200. DOI: 10.5220/0008520000000000.

[19] Mohamed B. Trabia, Mohamed S. Kaseko y Manish Ande. «A Two-Stage Fuzzy Logic Controller for Traffic Signals». En: *Transportation Research Part C: Emerging Technologies* 7.6 (1999). Control difuso aplicado a semáforos, págs. 353-367. DOI: 10.1016/S0968-090X(00)00004-4.

[20] Mauro Dell'Orco, Ozgur Baskan y Mario Marinelli. «A Harmony Search Algorithm Approach for Optimizing Traffic Signal Timings». En: *PROMET - Traffic & Transportation* 25.4 (2013). Metaheurísticas aplicadas a optimización semafórica, págs. 349-358. DOI: 10.7307/ptt.v25i4.1245.

[21] Li Li, Yisheng Lv y Fei-Yue Wang. «Traffic Signal Timing via Deep Reinforcement Learning». En: *IEEE/CAA Journal of Automatica Sinica* 3.3 (2016). Deep Q-learning para control semafórico adaptativo, págs. 247-254. DOI: 10.1109/JAS.2016.7508798.

[22] P. B. Hunt, D. I. Robertson, R. D. Bretherton y R. I. Winton. «SCOOT—A Traffic Responsive Method of Coordinating Signals». En: *Transport and Road Research Laboratory Report LR 1014* (1982). Sistema de control adaptativo SCOOT.

[23] A. G. Sims y K. W. Dobinson. «The Sydney Coordinated Adaptive Traffic (SCATS) System: Philosophy and Benefits». En: *IEEE Transactions on Vehicular Technology* 29.2 (1980). Sistema SCATS de control adaptativo, págs. 130-137. DOI: 10.1109/T-VT.1980.23833.

[24] Timothy J. Ross. *Fuzzy Logic with Engineering Applications*. 3rd. Chichester, UK: John Wiley & Sons, 2010.

[25] Gökhan Erdinç. «Application of a Mamdani-Based Fuzzy Traffic State Identifier». Tesis sobre identificación de estado de tráfico usando lógica difusa Mamdani. Thesis. Università "La Sapienza"di Roma, 2023. URL: https://iris.uniroma1.it/retrieve/d51daa72-738b-40a8-b0f8-2f25473ebf1b/Erdin%C3%A7_Application-Mamdani-based_2023.pdf.

[26] Baher Abdulhai, Rob Pringle y Grigoris J. Karakoulas. «Reinforcement Learning for True Adaptive Traffic Signal Control». En: *Journal of Transportation Engineering* 129.3 (2003). Q-learning aplicado a control semafórico, págs. 278-285. DOI: 10.1061/(ASCE)0733-947X(2003)129:3(278).

[27] Seung-Bae Cools, Carlos Gershenson y Bart D'Hooghe. «Self-Organizing Traffic Lights: A Realistic Simulation». En: *Advances in Applied Self-Organizing Systems*. Sistemas autoorganizados para control semafórico. Springer, 2013, págs. 45-55. DOI: 10.1007/978-1-4471-5113-5_3.

[28] S. Erdogan, Bashar Al-Omari y Hesham Rakha. «Signal Timing Optimization for Coordinated Arterials: A Review of Progression-Based Methods». En: *Journal of Transportation Engineering* 139.4 (2013), págs. 358-370. DOI: 10.1061/(ASCE)TE.1943-5436.0000506.

[29] Pradeep Kumar Singh, Roshan Singh, Sukumar Nandi y Sukumar Nandi. «Managing Smart Traffic Lights Using Fog Computing and Blockchain». En: *IEEE Internet of Things Journal* 7.7 (2020). Blockchain aplicado a gestión de semáforos inteligentes, págs. 6243-6252. DOI: 10.1109/JIOT.2020.2968410.

[30] Ye Yuan y Fei-Yue Wang. «Blockchain and Cryptocurrencies: Model, Techniques, and Applications». En: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 48.9 (2018). Aplicaciones de blockchain en sistemas ciber-físicos, págs. 1421-1428. DOI: 10.1109/TSMC.2018.2854904.

[31] Vahid Azimirad, Naser Pariz y Saeed Balochian. «A Novel Fuzzy Model and Control of Single Intersection at Urban Traffic Network». En: *IEEE Systems Journal* 4.1 (2010). Modelo difuso para intersecciones urbanas, págs. 107-111. DOI: 10.1109/JSYST.2010.2043159.

[32] Sunil Kumar Yadav y Richa Sharma. «Fuzzy Logic Based Traffic Light Controller for Heterogeneous Traffic». En: *International Journal of Advanced Research in Computer Science* 12.3 (2021). Control difuso para tráfico heterogéneo, págs. 1-5.

[33] Nazri Mohd Nawi, Azlan Khan y M. Z. Rehman. «A New Levenberg-Marquardt based Back Propagation Algorithm Trained with Cuckoo Search». En: *Procedia Technology* 11 (2013). Optimización de sistemas neuro-fuzzy, págs. 18-23.

[34] DLR Institute of Transportation Systems. *SUMO User Documentation*. Consultado: 2025-12-XX. Eclipse SUMO. 2025. URL: https://sumo.dlr.de/docs/.

[35] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. Consultado: 2025-01-XX. 2008. URL: https://bitcoin.org/bitcoin.pdf.

[36] Juan Benet. *IPFS - Content Addressed, Versioned, P2P File System*. Whitepaper fundacional de IPFS. 2014. arXiv: 1407.3561 [cs.DC]. URL: https://arxiv.org/abs/1407.3561.

[37] Yonatan Sompolinsky y Aviv Zohar. «PHANTOM and GHOSTDAG: A Scalable Generalization of Nakamoto Consensus». En: *Proceedings of the 3rd ACM Conference on Advances in Financial Technologies (AFT '21)*. New York, NY, USA: ACM, 2021, págs. 57-70. DOI: 10.1145/3479722.3480990.

[38] Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller y Steven Goldfeder. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press, 2016. ISBN: 978-0-691-17169-2.

[39] Vitalik Buterin. «Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform». En: (2014). Whitepaper, Consultado: 2025-01-XX. URL: https://ethereum.org/en/whitepaper/.

[40] Juan Garay, Aggelos Kiayias y Nikos Leonardos. «The Bitcoin Backbone Protocol: Analysis and Applications». En: *Proceedings of the 19th International Conference on Principles of Distributed Systems (OPODIS 2015)*. Consultado: 2025-01-XX. 2015, págs. 1-15. URL: https://opodis2015.irisa.fr/files/2016/01/Garay_OPODIS_2015.pdf.

[41] Qiang Wang, Jiajing Yu, Shuo Chen y Yong Xiang. «SoK: DAG-based Blockchain Systems». En: *ACM Computing Surveys* 55.12 (2023), Article 256. DOI: 10.1145/3576899.

[42] X. Zhang et al. «SoK: DAG-based Consensus Protocols». En: *arXiv preprint* (2025). Systematization of Knowledge sobre protocolos de consenso basados en DAG. arXiv: 2411.10026 [cs.CR]. URL: https://arxiv.org/abs/2411.10026.

[43] Ethereum Foundation. *Blocks*. Documentación oficial de Ethereum sobre block time de 12 segundos en PoS. 2024. URL: https://ethereum.org/developers/docs/blocks/.

[44] IPFS Project. *IPFS Architecture*. Especificación oficial de arquitectura IPFS, define formalmente nodeID := multihash(publicKey) y esquema de identidad basado en PKI. 2024. URL: https://github.com/ipfs/specs/blob/main/ARCHITECTURE.md.

[45] Lotfi A. Zadeh. «Fuzzy Sets». En: *Information and Control* 8.3 (1965), págs. 338-353. DOI: 10.1016/S0019-9958(65)90241-X.

[46] Lotfi A. Zadeh. «Similarity Relations and Fuzzy Orderings». En: *Information Sciences* 3.2 (1971), págs. 177-200. DOI: 10.1016/S0020-0255(71)80005-1.

[47] Segismundo S. Izquierdo y Luis R. Izquierdo. «Mamdani Fuzzy Systems for Modelling and Simulation». En: *Journal of Artificial Societies and Social Simulation* 21.3 (2018), pág. 2. URL: https://www.jasss.org/21/3/2.html.

[48] Russell C. Eberhart y James Kennedy. «A New Optimizer Using Particle Swarm Theory». En: *Proceedings of the Sixth International Symposium on Micro Machine and Human Science (MHS '95)*. IEEE, 1995, págs. 39-43. DOI: 10.1109/MHS.1995.494215.

[49] Maurice Clerc y James Kennedy. «The Particle Swarm – Explosion, Stability, and Convergence in a Multidimensional Complex Space». En: *IEEE Transactions on Evolutionary Computation* 6.1 (2002), págs. 58-73. DOI: 10.1109/4235.985692.

[50] Yuhui Shi y Russell Eberhart. «Parameter Selection in Particle Swarm Optimization». En: (1998). Proposed inertia weight adaptation, págs. 591-600.

[51] Riccardo Poli, James Kennedy y Tim Blackwell. «Particle Swarm Optimization: An Overview». En: *Swarm Intelligence* 1.1 (2007), págs. 33-57. DOI: 10.1007/s11721-007-0002-0.

APÉNDICE

A.1. A