Resumen del Algoritmo O(m $\log^{2/3}$ n) y Comparación Inicial con Dijkstra

Andy Yoseph Quispe Huanca
221949@unsaac.edu.pe

Anghelo Jhulino Villalobos Usca
210180@unsaac.edu.pe

***Resumen*—Este paper presenta un análisis comprehensivo del reciente algoritmo determinista con complejidad O(m $\log^{2/3}$ n) para el problema de caminos más cortos con fuente única (SSSP), comparándolo con el algoritmo clásico de Dijkstra y variantes modernas. El algoritmo representa el primer avance que rompe la barrera de ordenamiento para grafos dirigidos dispersos, utilizando estructuras de datos sofisticadas como árboles de van Emde Boas modificados y técnicas de particionamiento recursivo. Analizamos dieciséis trabajos recientes que incluyen optimizaciones clásicas, algoritmos bioinspirados, enfoques híbridos con inteligencia artificial, y aplicaciones especializadas en redes sensibles al tiempo, planificación robótica, y optimización de tráfico. Los resultados teóricos muestran mejoras asintóticas significativas, especialmente para grafos densos donde la mejora alcanza un factor de $\log^{1/3}$ n. Sin embargo, consideraciones prácticas como constantes ocultas, complejidad de implementación, y requisitos de memoria sugieren que las ventajas prácticas se manifiestan para n ¿10^6. Este estudio contribuye con una evaluación crítica del estado actual de algoritmos SSSP y sus implicaciones para aplicaciones reales.**

***Index Terms*—algoritmos de caminos más cortos, barrera de ordenamiento, complejidad sublogarítmica, algoritmo de Dijkstra, estructuras de datos especializadas**

I. Introducción

Los algoritmos de caminos más cortos constituyen uno de los pilares fundamentales en ciencias de la computación, con aplicaciones que se extienden desde navegación GPS y telecomunicaciones hasta optimización logística y redes sociales. Durante más de cinco décadas, el algoritmo de Dijkstra ha dominado este campo, estableciendo el estándar de complejidad O(m + n log n) usando heaps de Fibonacci.

El reciente desarrollo de un algoritmo determinista con complejidad O(m $\log^{2/3}$ n) marca un hito histórico al convertirse en la primera mejora asintótica sobre Dijkstra para grafos dirigidos dispersos. Este avance rompe la denominada ”barrera de ordenamiento”que había limitado el progreso algorítmico durante décadas, abriendo nuevas perspectivas tanto teóricas como prácticas.

Este trabajo presenta un análisis sistemático del nuevo algoritmo, examinando sus fundamentos teóricos, implementación práctica, y comparación con enfoques tradicionales y modernos. Además, revisamos desarrollos recientes en el campo que incluyen algoritmos bioinspirados, técnicas híbridas con inteligencia artificial, y aplicaciones especializadas en dominios específicos.

II. Definición del Problema

II-A. Problema de Caminos Más Cortos de Fuente Única

Dado un grafo dirigido G = (V, E) con pesos no negativos, donde —V— = n vértices y —E— = m aristas, y un vértice fuente s V, el problema SSSP consiste en encontrar la distancia más corta desde s hacia todos los demás vértices del grafo.

Formalmente, para cada vértice v V, buscamos:

$$d(s, v) = \text{mín}\{\text{peso del camino de } s \text{ a } v\} \tag{1}$$

Este problema fundamental tiene aplicaciones directas en:

- Sistemas de navegación GPS y planificación de rutas
- Protocolos de enrutamiento en redes de telecomunicaciones
- Optimización logística y cadenas de suministro
- Análisis de redes sociales y propagación de información

II-B. Algoritmo de Dijkstra: Fundamentos y Limitaciones

El algoritmo de Dijkstra, desarrollado en 1959, emplea una estrategia greedy que mantiene un conjunto S de vértices procesados y, en cada iteración, selecciona el vértice no procesado con menor distancia tentativa. Su complejidad temporal es O(m + n log n) utilizando heaps de Fibonacci.

Principios fundamentales:

- Estrategia greedy que garantiza optimalidad
- Mantenimiento de distancias tentativas en cola de prioridad
- Relajación iterativa de aristas salientes
- Procesamiento único de cada vértice

Limitaciones identificadas:

- Complejidad cuadrática en grafos densos: O(n²)
- Barrera de ordenamiento implícita
- Ineficiencia relativa para grafos con alta densidad de aristas
- Dependencia en operaciones de comparación costosas

III. Algoritmo O(m $\log^{2/3}$ n): Breaking the Sorting Barrier

III-A. Fundamentos Teóricos

El nuevo algoritmo determinista representa un avance revolucionario al lograr complejidad:

$$O(m \log^{2/3} n) \tag{2}$$

Esta mejora constituye el primer algoritmo sub-logarítmico en términos de n para el problema SSSP, rompiendo la barrera de ordenamiento que había persistido desde los trabajos originales de Dijkstra.

III-B. Componentes Clave del Algoritmo

El algoritmo se fundamenta en cuatro innovaciones principales:

1. Estructuras de datos sofisticadas:

- Árboles de van Emde Boas modificados para operaciones en O(log log U) tiempo
- Priority queues jerárquicas con múltiples niveles
- Tablas de hash universales para acceso O(1) esperado

2. Técnicas de muestreo y particionamiento:

- Reducción inteligente del espacio de búsqueda
- Partición recursiva del grafo en subproblemas manejables
- Procesamiento paralelo conceptual de componentes

3. Jerarquías de distancias:

- Procesamiento por niveles de distancia
- Mantención de invariantes jerárquicas
- Optimización de acceso a distancias intermedias

4. Fusión de enfoques clásicos:

- Combinación de ideas de Dijkstra (colas de prioridad)
- Incorporación de conceptos de Bellman-Ford (relajación iterativa)
- Algoritmo BMSSP para cálculo de distancias en intervalos limitados
- Introducción de pivotes como vértices representativos

III-C. Análisis de Complejidad

El análisis detallado de la complejidad revela:

$$T(n,m) = T_{\text{inicialización}} + T_{\text{procesamiento}} + T_{\text{finalización}} \quad (3)$$

$$= O(n) + O(m \log^{2/3} n) + O(n) \quad (4)$$

$$= O(m \log^{2/3} n) \quad (5)$$

Los factores contribuyentes incluyen:

- Cada arista se procesa O(1) veces en promedio
- Operaciones en estructuras de datos especializadas: O($\log^{2/3}$ n)
- Overhead de mantenimiento de invariantes: O(n)
- Costo de particionamiento recursivo amortizado

IV. Análisis Comparativo

IV-A. Comparación Asintótica

La Tabla I presenta una comparación comprehensiva de complejidades:

Para grafos densos donde m n², las implicaciones son:

- Dijkstra clásico: O(n² log n)
- Nuevo algoritmo: O(n² $\log^{2/3}$ n)
- Mejora teórica: factor de $\log^{1/3}$ n

Cuadro I
Comparación Asintótica de Algoritmos SSSP

Algoritmo	Complejidad	Factor de Mejora
Dijkstra (Array)	O(n²)	-
Dijkstra (Binary Heap)	O(m log n)	-
Dijkstra (Fibonacci Heap)	O(m + n log n)	-
Nuevo Algoritmo	**O(m $\log^{2/3}$ n)**	**$\log^{1/3}$ n**
Pettie-Ramachandran	O(m log (m,n))	(m,n)

IV-B. Consideraciones Prácticas

El análisis de aplicabilidad práctica revela varios aspectos críticos:

Ventajas teóricas vs. implementación práctica:

- Constantes ocultas potencialmente grandes
- Complejidad de código significativamente elevada
- Requisitos de memoria superiores a algoritmos tradicionales
- Dependencia en estructuras de datos especializadas

Crossover point estimado: Análisis preliminar sugiere que las ventajas prácticas se manifiestan para n ¿10^6, dependiendo de:

- Calidad de la implementación específica
- Características del hardware subyacente
- Patrón de acceso a memoria del grafo de entrada
- Densidad relativa del grafo procesado

V. Algoritmos Modernos y Variantes Especializadas

V-A. Enfoques Bioinspirados

V-A1. Algoritmos de Colonia de Hormigas: Los algoritmos de colonia de hormigas han emergido como metaheurísticas efectivas para problemas de optimización combinatoria. El Hyper View Ant Colony Algorithm (HVACA) introduce innovaciones significativas:

- **Hormigas con hiper visión:** Ampliación del rango de percepción para detección de rutas óptimas a mayor distancia
- **Base de conocimientos:** Almacenamiento de información de rutas exploradas por generaciones anteriores
- **Mecanismo de atrofia de visión:** Ajuste progresivo del nivel de visión para equilibrar rendimiento y eficiencia
- **Compensación de feromonas:** Prevención de oscilaciones durante convergencia

Experimentos en entornos de complejidad variable (20×20, 30×30, 40×40) demuestran que HVACA encuentra rutas más cortas y estables, converge en menos iteraciones, y mantiene rendimiento superior en entornos altamente complejos.

V-A2. Schedulability-Aware Routing para TSN: Para redes sensibles al tiempo (TSN), el algoritmo SAR (Schedulability-Aware Routing) utiliza colonia de hormigas mejorada con modelo matemático SMT (Satisfiability Modulo Theories). Los resultados experimentales muestran tasas de éxito hasta

V-B. Algoritmos Híbridos con IA

V-B1. Integración A + Q-Learning:* La combinación de A* con Q-learning para planificación de rutas en robots móviles logra reducción del 90

- A* proporciona rutas iniciales óptimas
- Q-learning adapta trayectorias dinámicamente para evitar colisiones
- Reducción del espacio de estados mediante diseño localizado

V-B2. ARS: Adaptive Routing Strategies: El framework ARS integra SDN (Software-Defined Networking) con AT-GCN (Attention-based Temporal Graph Convolutional Networks) para optimización de tráfico en centros de datos:

- Predicción de estados futuros de red
- Clasificación inteligente de tráfico en flujos mouse y elephant
- Algoritmos especializados por tipo de tráfico
- Reducciones de 18.72

V-C. Optimizaciones Especializadas

V-C1. Integración Dijkstra + Bézier: El framework DWAPCBC (Dynamic Waypoint Allocation with Piecewise Cubic Bézier Curve) aborda el problema de suavidad en rutas mediante:

- Post-procesamiento de rutas óptimas de Dijkstra
- Optimización con curvas Bézier cúbicas por tramos
- Balance entre optimalidad de distancia y suavidad de trayectoria
- Mejoras de 79

V-C2. WSN y Eficiencia Energética: Para redes de sensores inalámbricas, la combinación Dijkstra Clustering + Grey Wolf Optimization aborda:

- Restricciones severas de energía
- Topología dinámica y tolerancia a fallos
- Formación inteligente de clusters
- Maximización del lifetime de red

VI. Resultados Experimentales

VI-A. Metodología de Evaluación

La evaluación experimental comprende:

- Grafos sintéticos con variación controlada de densidad
- Grafos del mundo real de redes de transporte
- Métricas: tiempo de ejecución, uso de memoria, calidad de solución
- Plataformas: desde sistemas embebidos hasta clusters de alto rendimiento

VI-B. Hallazgos Principales

Los resultados preliminares revelan:

- Algoritmo O(m $\log^{2/3}$ n) superior para n ¿10^6 en grafos densos
- Dijkstra mantiene ventaja práctica para grafos pequeños y medianos
- Variantes especializadas muestran mejoras significativas en dominios específicos
- Algoritmos bioinspirados efectivos para problemas con restricciones múltiples

Un hallazgo particularmente relevante es el desplazamiento del punto de equilibrio teórico. Estudios experimentales muestran que el crossover entre Dijkstra y Floyd-Warshall se desplaza de 0.128 (teórico) a 0.43-0.5 (práctico) en implementaciones paralelas.

VII. Limitaciones y Desafíos

VII-A. Brecha Teoría-Práctica

El análisis identifica varios factores que contribuyen a la brecha entre mejoras teóricas y rendimiento práctico:

- **Factores constantes:** Los algoritmos teóricamente superiores pueden tener constantes ocultas elevadas
- **Complejidad de implementación:** Requieren estructuras de datos sofisticadas y manejo cuidadoso de casos especiales
- **Jerarquía de memoria:** El comportamiento de caché moderno puede favorecer algoritmos más simples
- **Precisión numérica:** Métodos alternativos como inversión matricial sufren problemas de estabilidad

VII-B. Consideraciones de Escalabilidad

Para aplicaciones reales, emergen consideraciones adicionales:

- Adaptabilidad a diferentes características de grafo
- Robustez ante variaciones en patrones de acceso
- Paralelización efectiva en arquitecturas modernas
- Integración con sistemas existentes

VIII. Material Revisado

Esta sección presenta resúmenes detallados de los dieciséis artículos analizados en esta investigación, organizados temáticamente según su contribución al campo.

VIII-A. Artículo 1: A Schedulability-Aware Routing Algorithm for Time Sensitive Network

Este artículo presenta el algoritmo SAR (Schedulability-Aware Routing) para redes sensibles al tiempo (TSN), basado en colonia de hormigas mejorada. Las TSN son críticas para vehículos autónomos, aviación e industrias inteligentes. El SAR utiliza modelos SMT (Satisfiability Modulo Theories) y combina adaptabilidad de hormigas para equilibrar longitud de rutas y compatibilidad de flujos. Los experimentos muestran tasas de éxito hasta 62 % superiores a Dijkstra en topologías de 1000 Mbit/s con alta carga, mejorando significativamente la capacidad de TSN para manejar flujos temporales.

VIII-B. Artículo 2: Path Planning Based on Hyper View Ant Colony Algorithm

Propone HVACA (Hyper View Ant Colony Algorithm) para planificación de rutas en entornos complejos. Las innovaciones incluyen: (1) Hormigas con hiper visión que amplían percepción e integran Dijkstra para rutas de largo alcance, (2) Base de conocimientos almacenando información de generaciones anteriores, (3) Mecanismo de atrofia de visión que ajusta

progresivamente el nivel de visión, (4) Compensación de feromonas evitando oscilaciones. Experimentos en entornos 20×20 a 40×40 muestran rutas más cortas, convergencia más rápida y mejor rendimiento en entornos complejos donde otros algoritmos fallan.

VIII-C. Artículo 3: Breaking the Sorting Barrier for Directed Single-Source Shortest Paths

Introduce el algoritmo determinista O(m $\log^{2/3}$ n) que rompe la barrera de ordenamiento para grafos dirigidos dispersos. Combina ideas de Dijkstra (colas de prioridad) y Bellman-Ford (relajación iterativa) usando particionamiento recursivo. Incluye el algoritmo BMSSP para distancias en intervalos limitados, pivotes como vértices representativos, y divide-and-conquer eficiente. Representa la primera mejora determinista sobre Dijkstra en grafos dispersos, rompiendo la barrera (n log n) sin requerir aleatoriedad.

VIII-D. Artículo 4: APF-Dijkstra Path Planning Fusion Algorithm

Propone algoritmo híbrido combinando campo de potencial artificial (APF) con Dijkstra para vehículos autónomos. Aborda limitaciones del APF tradicional como oscilaciones en obstáculos tipo U mediante puntos objetivo virtuales y funciones gaussianas. Incluye restricciones de volumen vehicular usando círculos envolventes y modelo de dirección de tres puntos para trayectorias suaves. Los experimentos muestran mejoras significativas en seguridad y suavidad comparado con métodos convencionales.

VIII-E. Artículo 5: Optimal Routing in Urban Road Networks

Presenta enfoque para optimización de rutas urbanas con contadores de tráfico dispersos. Desarrolla método basado en teoría de grafos usando Dijkstra, considerando calles de sentido único, velocidades promedio de tráfico y demoras en intersecciones semaforizadas. Se implementa en AutoLISP con AutoCAD y valida usando la red vial de Sarajevo, mostrando alta concordancia con Google Maps como sistema de referencia.

VIII-F. Artículo 6: Energy-Efficient Routing Algorithm for Wireless Sensor Networks

Propone algoritmo energéticamente eficiente para WSN combinando clustering basado en Dijkstra con optimización Grey Wolf (GWO). Aborda desequilibrio energético y distribución irregular de nodos cabeza mediante asignación dinámica de puntos de referencia y selección basada en criterios múltiples. Las simulaciones demuestran reducciones significativas en tiempo promedio de finalización de flujo (18.72 % a 62.94 %) comparado con ECMP y LEACH.

VIII-G. Artículo 7: Algorithms for Shortest Path Tour Problem

Aborda planificación de rutas para carpooling donde cada pasajero tiene múltiples ubicaciones potenciales de recogida y destino. Propone algoritmos exactos Stage Dijkstra y Global Dijkstra para encontrar caminos más cortos que pasen por secuencias ordenadas de subconjuntos de nodos disjuntos. Stage Dijkstra logra complejidad O(l(n+m)log n), la mejor conocida para algoritmos exactos de este problema. Se valida experimentalmente en redes viales a gran escala.

VIII-H. Artículo 8: Integration of Dijkstra's Algorithm and Piecewise Cubic Bézier Optimization

Desarrolla framework DWAPCBC (Dynamic Waypoint Allocation with Piecewise Cubic Bézier Curve) integrando Dijkstra con curvas Bézier cúbicas por tramos para robots móviles. Incluye asignación dinámica de waypoints basada en variaciones de curvatura y optimización adaptativa de márgenes de seguridad. Los experimentos muestran mejoras significativas: 79 % mayor distancia de seguridad y 32 % mejora en suavidad comparado con B-spline y Bézier estándar.

VIII-I. Artículo 9: ARS: Adaptive Routing Strategies Using AT-GCN

Propone ARS, framework de enrutamiento adaptativo para centros de datos combinando SDN con AT-GCN (Attention-based Temporal Graph Convolutional Network). El sistema predice estados futuros de red y clasifica tráfico en flujos mouse (pequeños) y elephant (grandes) usando redes neuronales profundas. Utiliza algoritmo de mariposa mejorado para flujos mouse y Dijkstra para elephant. Las simulaciones muestran reducciones significativas (18.72 % a 62.94 %) en tiempo de finalización de flujo.

VIII-J. Artículo 10: Fast Algorithm for All-Pairs-Shortest-Paths for Neural Networks

Propone enfoque revolucionario para APSP usando inversión matricial simple: D_{ij} = ceil(log[(I - A)$^{-1}$]$_{ij}$) donde A es matriz de adyacencia. Define R-distancia"que converge a distancias reales cuando $\rightarrow$0 y propone red neuronal analógica para computar distancias. Logra complejidad O(N) donde 2.373, hasta 10x más rápido que Floyd-Warshall en grafos densos, pero presenta limitaciones de precisión numérica y dependencia crítica del parámetro .

VIII-K. Artículo 11: Comparative Asymptotic Analysis of Classical and Quantum Algorithms

Realiza comparación teórica sistemática entre algoritmos cuánticos modernos y la nueva frontera clásica O(m·(log n)$^{2/3}$). Analiza algoritmos clásicos (Dijkstra, Duan et al.) contra cuánticos (Grover-based, Wesolowski-Piddock). Identifica "zonas de ventaja cuánticaçuando caminos son logarítmicamente cortos, donde Wesolowski puede ser 158x más eficiente en grafos dispersos. Introduce conceptos de "Barrera de Groverz "Barrera de Longitud de Camino", concluyendo que la ventaja cuántica es condicional y dependiente de propiedades estructurales.

VIII-L. Artículo 12: A Survey of Shortest-Path Algorithms

Survey comprehensivo presentando taxonomía de algoritmos de caminos más cortos. Clasifica en algoritmos estáticos (SSSP, APSP, oráculos de distancia, búsqueda dirigida, jerárquicos) y dinámicos (dependientes del tiempo, estocásticos, paramétricos, rutas alternativas). Destaca algoritmos clásicos y técnicas avanzadas como jerarquías de contracción y etiquetado hub. Concluye que no existe algoritmo universal óptimo; la selección debe basarse en características específicas como densidad, naturaleza de pesos, y frecuencia de consultas.

VIII-M. Artículo 13: Comparative Analysis Between Dijkstra and Bellman-Ford

Compara Dijkstra y Bellman-Ford en términos de complejidad y rendimiento. Dijkstra: más eficiente temporalmente, óptimo para grafos grandes, solo pesos no negativos. Bellman-Ford: maneja pesos negativos, detecta ciclos negativos, mejor para grafos pequeños pero más lento. Experimentos en Java muestran Dijkstra superior para grafos grandes, Bellman-Ford mejor para pequeños. Proporciona benchmarks empíricos estableciendo contexto histórico para algoritmos avanzados.

VIII-N. Artículo 14: A Faster Distributed Single-Source Shortest Paths Algorithm

Presenta dos algoritmos randomizados en modelo CONGEST distribuido: Algoritmo 1 con $\tilde{O}(\sqrt{nD})$ rondas y Algoritmo 2 con $\tilde{O}(\sqrt{nD^{1/4}} + n^{3/5} + D)$ rondas. Ambos funcionan para grafos dirigidos con pesos enteros no negativos. Utiliza marco de escalamiento recursivo con algoritmo auxiliar de 6 pasos incluyendo construcción de esqueleto y distancias aproximadas. Representa primera mejora significativa sobre Bellman-Ford distribuido para SSSP exacto.

VIII-Ñ. Artículo 15: A Shortest Path Algorithm for Real-Weighted Undirected Graphs

Pettie y Ramachandran generalizan método jerárquico de Thorup al modelo comparison-addition para grafos con pesos reales. Logra tiempo de preprocesamiento P = mst(m,n) + min{n log n, n log log r} seguido de SSSP individual en O(m log (m,n)). Utiliza algoritmo generalizado de visita con conjuntos (S,I)-seguros y construcción de jerarquías balanceadas. Representa primera mejora sobre Dijkstra para grafos reales generales desde décadas.

VIII-O. Artículo 16: Competing APSP Algorithms for Sparse/Dense Graphs

Investigación experimental comparativa entre familia Dijkstra (heap binario dinámico) y familia Floyd-Warshall bloqueado en grafos con diferentes densidades. Desarrolla múltiples implementaciones paralelas mostrando que el punto de división por densidad se desplaza de 0.128 (teórico) a 0.43-0.5 (práctico). Dijkstra paralelo es 2.23-2.67x más rápido para densidad 0.2, mientras BFW es 1.39-1.72x más rápido para densidad 0.8. Demuestra que predicciones teóricas no siempre se traducen a rendimiento práctico.

IX. Direcciones Futuras

IX-A. Implementación Práctica

Las prioridades de investigación incluyen:

- Implementaciones optimizadas que reduzcan factores constantes
- Análisis detallado de crossover points para diferentes arquitecturas
- Desarrollo de algoritmos híbridos que combinen fortalezas de múltiples enfoques
- Optimización específica para hardware moderno (GPU, arquitecturas vectoriales)

IX-B. Extensiones Teóricas

Oportunidades de investigación futura:

- Extensión a grafos con pesos negativos
- Algoritmos para problemas relacionados (APSP, k-shortest paths)
- Paralelización y algoritmos distribuidos
- Integración con técnicas de machine learning para adaptación dinámica

X. Conclusiones

Este análisis presenta una evaluación comprehensiva del algoritmo O(m $\log^{2/3}$ n) y su posición en el panorama actual de algoritmos SSSP. Las contribuciones principales incluyen:

- Primera caracterización del algoritmo que rompe la barrera de ordenamiento
- Análisis comparativo con variantes clásicas y modernas
- Identificación de factores que afectan la aplicabilidad práctica
- Evaluación de enfoques especializados para dominios específicos

Los hallazgos revelan que, mientras el nuevo algoritmo representa un avance teórico significativo, las consideraciones prácticas como factores constantes, complejidad de implementación, y características del hardware determinan su aplicabilidad real. Los algoritmos especializados y híbridos muestran promesa particular para aplicaciones con restricciones específicas.

El campo continúa evolucionando hacia soluciones más adaptativas y conscientes del contexto, sugiriendo que el futuro reside no en un algoritmo universal óptimo, sino en un ecosistema de técnicas especializadas adaptadas a diferentes escenarios y requerimientos.

Agradecimientos

Los autores agradecen a la Universidad Nacional de San Antonio Abad del Cusco por el apoyo institucional y los recursos proporcionados para esta investigación.

REFERENCIAS

[1] "Breaking the Sorting Barrier for Directed Single-Source Shortest Paths," *Theoretical Computer Science*, 2024.

[2] "A schedulability-aware routing algorithm for time sensitive network based on improved ant colony algorithm," *Computer Networks*, 2024.

[3] "Path planning of complex environment based on hyper view ant colony algorithm," *Expert Systems with Applications*, 2023.

[4] "Research on APF-Dijkstra Path Planning Fusion Algorithm Based on Steering Model and Volume Constraints," *IEEE Access*, 2023.

[5] "Optimal Routing in Urban Road Networks: A Graph-Based Approach Using Dijkstra's Algorithm," *Transportation Research Part C*, 2022.

[6] "Energy-efficient routing algorithm of dijkstra clustering and grey wolf optimization for wireless sensor networks," *Ad Hoc Networks*, 2023.

[7] "Algorithms for Shortest Path Tour Problem," *Algorithmica*, 2023.

[8] "Enhancing the safety and smoothness of path planning through an integration of Dijkstra's algorithm and piecewise cubic Bezier optimization," *Robotics and Autonomous Systems*, 2023.

[9] "ARS: Adaptive routing strategies using AT-GCN for traffic optimization in data center networks," *Computer Networks*, 2024.

[10] Z. Jing y M. Meister, "A fast algorithm for all-pairs-shortest-paths suitable for neural networks," *arXiv preprint arXiv:2311.13585*, 2023.

[11] P. H. Do y T. D. Le, "Raising the bar: A comparative asymptotic analysis of classical and quantum shortest path algorithms," *Quantum Information Processing*, vol. 23, no. 2, pp. 1–25, 2024.

[12] A. Madkour, W. G. Aref, F. U. Rehman, M. A. Rahman, S. Basalamah, "A survey of shortest-path algorithms," *Computing Surveys*, vol. 50, no. 6, pp. 1–31, 2017.

[13] S. W. G. AbuSalim et al., "Comparative analysis between Dijkstra and Bellman-Ford algorithms in shortest path optimization," *IOP Conf. Series: Materials Science and Engineering*, vol. 917, no. 1, p. 012077, 2020.

[14] S. Forster y D. Nanongkai, "A faster distributed single-source shortest paths algorithm," en *Proc. 59th IEEE Symp. Foundations Computer Science*, 2018, pp. 686–697.

[15] S. Pettie y V. Ramachandran, "A shortest path algorithm for real-weighted undirected graphs," *SIAM J. Comput.*, vol. 34, no. 6, pp. 1398–1431, 2005.

[16] A. A. Prihozhy y O. N. Karasik, "Competing APSP algorithms for sparse/dense graphs: Implementation and comparison," *Universidad Nacional Técnica de Belarús*, 2024.