A ES
ISSN: 2096-3246
Volume 58, Issue 01, 2026

ADAPTIVE COST-AWARE SHORTEST PATH OPTIMIZATION IN ELECTRONIC AND TRANSPORTATION NETWORKS: A COMPARATIVE STUDY WITH FLOYD–WARSHALL AND FORD–FULKERSON

[1]Saiful Islam, [2]Dr. Anupam Dutta
[1]Department of Mathematics, Research Scholar, Department of Mathematics. University of Science and Technology Meghalaya, Email: maths.saiful@gmail.com
ORCID id: https://orcid.org/0009-0005-7361-7532
[2]Department of Mathematics, University of Science and Technology Meghalaya.
State: Meghalaya, Dist: RiBhoi, India
Email: anupa.dutta@gmail.com | ORCID id: https://orcid.org/ 0009-0003-1803-3652

ABSTRACT
This study addresses a key challenge in modern networks corresponding to finding the optimal path. We first examined classical algorithms like Floyd–Warshall and Ford–Fulkerson but found them limited in scalability, flexibility, and real-time adaptability. To overcome these constraints, we developed an adaptive routing method namely ACASPO (Adaptive Cost-Aware Shortest Path Optimization) that integrates real-time updates and intelligent navigation. Simulations across diverse network structures show our approach delivers greater path efficiency with lower computational cost, particularly in large-scale or dynamic settings. Our work provides a practical, context-aware framework that connects classical theory with contemporary needs for responsive and efficient routing.

KEYWORDS: Floyd–Warshall algorithm, Ford–Fulkerson algorithm, Shortest path problem, Network flow optimization, Traveller optimization, Transportation networks.

INTRODUCTION
Optimization of paths in networks is a fundamental problem that has attracted continuous interest in computer science, operations research, and applied mathematics. From the design of efficient electronic communication systems to the development of intelligent transportation frameworks, shortest path algorithms serve as the foundation for decision-making and optimization. In electronic networks, efficient routing is critical for minimizing latency, energy consumption, and congestion, while in transportation systems, optimized traveler routes reduce overall travel time, minimize costs, and enhance system-wide efficiency. These diverse but related applications necessitate robust algorithms capable of handling dynamic conditions and complex structures.

Classical graph algorithms, particularly the Floyd–Warshall algorithm [1, 2] and the Ford–Fulkerson method [3, 4], have historically served as benchmarks in this domain. The Floyd–Warshall algorithm provides an elegant solution for computing all-pairs shortest paths in weighted graphs, operating with a time complexity of $O(n^3)$), making it highly effective for dense graphs. Similarly, the Ford–Fulkerson method addresses the maximum flow problem in networks, a fundamental approach to capacity-constrained optimization. These algorithms are mathematically rigorous and have stood the test of time, offering reliable solutions to well-defined static problems. However, their application to modern electronic and transportation systems reveals certain

limitations. Specifically, they assume static edge weights and lack adaptability to real-time changes, such as fluctuating bandwidth in communication systems or dynamic congestion in urban transportation.

To overcome these restrictions, research has increasingly turned toward heuristic and metaheuristic methods. Algorithms such as A* search [5**],** Genetic Algorithms (GA**)** [6], and Ant Colony Optimization (ACO), [7] have been broadly studied for their flexibility in handling dynamic and large-scale optimization problems. Heuristic methods often incorporate problem-specific knowledge to accelerate convergence, while metaheuristics offer robustness across diverse optimization landscapes. In spite of their effectiveness, these approaches introduce new challenges. Heuristic algorithms may struggle with scalability in very large graphs, and metaheuristic algorithms often sacrifice guaranteed optimality for approximate solutions [8,9]. Moreover, their computational overhead can be significant, limiting real-time applicability in systems requiring rapid decision-making.

Parallel to these efforts, research into multi-objective optimization has gained traction, particularly in transportation and traveler routing. These methods aim to minimize not only travel time but also reasons such as fuel consumption, cost, and congestion simultaneously [10]. Multi-objective evolutionary algorithms have shown capacity in balancing these trade-offs, yet their application to electronic networks has been limited. This highlights a key gap: while transportation optimization has benefited from adaptive and multi-objective approaches, electronic networks still rely heavily on static or single-objective methods.

METHODOLOGY

This study investigates advanced pathfinding and optimization algorithms across two domains: electronic network topologies and transportation networks. The methodology combines classical algorithms with novel, adaptive approaches to evaluate efficiency, adaptability, and scalability.

1. Problem Modeling

- Represent the system as a weighted graph $G(V, E)$, where vertices V correspond to nodes (network devices or locations) and edges E represent connections (links or routes) with associated costs
- Definition of the optimization objective according to the domain:
  - **Electronic networks:** Minimize adaptive cost including latency, congestion, and energy consumption.
  - **Traveler networks:** Minimize travel time, cost, or other multi-objective criteria.

2. Baseline Algorithms

- **Floyd–Warshall Algorithm:** Computes all-pairs shortest paths efficiently in dense graphs; used as a reference in both electronic and transportation networks.
- **Ford–Fulkerson Algorithm:** Applied in transportation networks to model capacity-constrained flow scenarios.

Adaptive and Modern Optimization Approaches

To bridge this gap, we propose the Adaptive Cost-Aware Shortest Path Optimization (ACASPO) algorithm. Unlike classical methods, ACASPO introduces a dynamic edge cost function that adapts in real time to system conditions. In electronic networks, this accounts for latency, congestion, and energy efficiency, while in transportation networks, it captures factors such as travel time, cost, and route congestion. Inspired by the Floyd–Warshall algorithm, ACASPO employs a modified

relaxation mechanism to iteratively update path costs until convergence is achieved. This allows the algorithm to retain computational efficiency while adapting to dynamic cost variations.
Furthermore, ACASPO is evaluated against both Floyd–Warshall and Ford–Fulkerson to provide a fair comparison with classical baselines. To demonstrate its versatility, the algorithm is applied across different graph models: electronic network topologies (mesh, random, and scale-free networks) and transportation structures (grid-based and route-based graphs). In addition, we include heuristic and metaheuristic approaches such as A*, GA, and ACO to benchmark performance against modern alternatives.

- **ACASPO (Adaptive Cost-Aware Shortest Path Optimization):**
  - Introduces a dynamic edge cost function $C(e)$ that adapts based on network conditions.
  - Iteratively updates shortest paths using a modified relaxation mechanism inspired by Floyd–Warshall.
  - Converges when path costs stabilize within a threshold.
- **Modern Traveler Optimization Algorithms:**
  - Heuristic-based: A* search, Dijkstra with adaptive heuristics.
  - Metaheuristic-based: Genetic Algorithms, Ant Colony Optimization.
  - Multi-objective optimization to balance travel time, cost, and congestion.

4. Comparative Framework

- Implement both classical and adaptive algorithms on representative graph datasets:
  - Electronic networks: Random graphs, mesh networks, scale-free topologies.
  - Traveler networks: Transportation grids, city road networks, route-based graphs.
- Compare performance across metrics:
  - Optimality of path (shortest or minimal cost)
  - Computational complexity (time and space)
  - Scalability with graph size
  - Robustness to dynamic changes (e.g., network congestion, blocked routes)
  - Adaptability of ACASPO to changing edge costs

5. Evaluation and Analysis

- Quantitative analysis: Execution time, memory usage, path quality, and flow efficiency.
- Qualitative analysis: Algorithm adaptability to real-world scenarios, such as variable network conditions or dynamic travel demands.
- Benchmarking against classical algorithms ensures objective assessment of improvements offered by adaptive and modern approaches.

OPTIMIZATION ALGORITHMS

We explore in detail the operational principles of the Floyd–Warshall and Ford–Fulkerson algorithms, followed by an extensive evaluation of the proposed ACASPO cost-aware path optimization method as given below

Floyd–Warshall Algorithm: Computes all-pairs shortest paths in a weighted graph using dynamic programming for complete path optimization.

Ford–Fulkerson Algorithm: Determines the maximum flow in a network by iteratively augmenting flow along available paths.

ACASPO Algorithm (Proposed): Provides a cost-aware shortest-path optimization approach that integrates dynamic cost factors for more efficient and adaptive routing.

Floyd–Warshall Algorithm

Purpose:

Calculating the shortest paths between every pairs of vertices in a weighted directed graph (can handle negative weights, but no negative cycles).

Algorithm

Let the graph be $G = (V, E)$ with edge weights $w(u, v)$.
Let $D^{(k)}[i][j]$ = shortest distance from i to j using only vertices $\{1,2,\ldots,k\}$ as intermediates.

Steps:

1. **Initialization:**

$$D^{(k)}[i][j] = \begin{Bmatrix} \mathrm{D}^{(0)}[\mathrm{i}][\mathrm{j}] = 0, & \text{if } \mathrm{i} = \mathrm{j} \\ \mathrm{w}(\mathrm{i},\mathrm{j}), & \text{if } (\mathrm{i},\mathrm{j}) \in \mathrm{E} \\ \infty, & \text{otherwise} \end{Bmatrix}$$

2. **Main Iteration:**
   For $k = 1\ to\ n$:
   For all pairs (i, j)):

$$D^{(k)}[i][j] = min \quad (D(k-1)[i][j],\ D(k-1)[i][k] + D(k-1)[k][j])$$

   Result:

$$D^{(n)}[i][j]$$

   gives the shortest path distance between all pair (i, j).
3. **Complexity:** $O(V^3)$

Ford–Fulkerson Algorithm

Purpose:

Computes the maximum flow in a flow network from a source (s) to a sink (t)**.**

Algorithm

Let $G = (V, E)$ with capacity $c(u, v) \geq 0$.

Steps:

a. Initialize flow $f(u, v) = 0$ for all $(u, v) \in E$.
b. While there exists an augmenting path P from s to t in the residual graph G_f: Find the residual capacity along the path:
   -
   - Augment flow along P:

$$f(u, v) = f(u, v) + c_f(P) f(u, v)$$

   When no augmenting path remains, f is a maximum flow.
c. **Complexity:** $O(E \cdot |\ f^*\ |)$ (depends on path-finding strategy)
   With Edmonds–Karp: $O(VE^2)$

PROPOSED METHOD: A Hybrid Graph Optimization Algorithm

COST-AWARE SHORTEST PATH OPTIMIZATION (ACASPO) ALGORITHM

Objective

To determine an optimal path (or solution) in a weighted network that minimizes a composite objective function incorporating multiple criteria such as distance, cost, time, energy, congestion, risk, or reliability.

Purpose:

Finds a shortest path that minimizes total cost while considering dynamic edge costs such as congestion, energy, or reliability.
(Used in network optimization and transport systems where $\neq$ $distance$.)

Problem Formulation

Let, $G = (V, E)$ be a directed or undirected graph, where:
For each edge $(u, v) \in E$, define a set of attributes:

- $a_1(u, v)$: distance or length
- $a_2(u, v)$: monetary or energy cost
- $a_3(u, v)$: congestion, delay, or risk
- ……
- $a_k(u, v)$: any user-defined metric

Here we define a generalized cost function:

$$F(u, v) = \sum_{i=1}^{k} w_i \, a_i(u, v) \, , w_i \geq 0$$

where w_i represents the importance (weight) of the $i - th$ criterion.

Generalised Algorithm

Step 1: Initialization

For all vertices $v \in V$:
$Cost[v] = \infty$, $Parent[v] = NIL$ Set $Cost[s] = 0$ where s is the source node.

Step 2: Iterative Relaxation

Repeat for $i = 1 \; to \; | V | - 1$:
For each edge $(u, v) \in E$:
If $Cost[v] > Cost[u] + F(u, v)$ then update:

$$Cost[v] = Cost[u] + F(u, v)$$
$$Parent[v] = u$$

This step ensures gradual improvement of multi-criteria costs across the network.

Step 3: Path Reconstruction

Starting from destination node t, reconstruct the solution path by iteratively tracing:

$$v \leftarrow Parent[v]$$

until the source node s is reached.

Step 4: Dynamic Adaptation (Optional)

Update edge attributes $a_i(u, v)$ dynamically to reflect:

- Traffic load variations
- Energy availability
- Reliability or failure probability

- User preferences

Recomputed paths periodically or when significant changes occur.

Generalisation Features

1. **Multi-Objective Capability**
   Handles multiple competing objectives within a single unified framework.
2. **Problem Agnostic**
   Can be adapted to:

- Transportation networks ($time + distance + tolls$)
- Communication networks ($latency + packet\ loss + energy$)
- Supply chains ($cost + risk + delivery\ time$)
- Power grids ($loss + reliability + congestion$)

3. **Extensible Cost Function**
   New constraints or criteria can be added without modifying the core algorithm.
4. **Compatibility with Classical Algorithms**

- Reduces to Dijkstra when all weights are non-negative and k=1
- Reduces to Bellman–Ford when dynamic or negative components exist

Complexity Analysis

- Time Complexity:

$$O(|\ V\ ||\ E\ |)$$

- Space Complexity:

$$O(|\ V\ |)$$

Comparable to Bellman–Ford, but with **greater expressive power**.

EXPERIMENT AND RESULTS ANALYSIS

The experimental problem considered in this study is a directed network optimization problem involving distance, monetary cost, and capacity constraints. The network structure used in the experiment is a representative graph model designed to evaluate the behavior of classical algorithms such as Floyd–Warshall and Ford–Fulkerson, along with the proposed Adaptive Cost-Aware Shortest Path Optimization (ACASPO) method.

The problem instance is inspired by classical network flow and shortest path problems studied in graph theory and combinatorial optimization [21]

The problem instance consists of four vertices representing a simplified transportation or communication network, where the objective is to determine optimal routing and flow decisions under multiple constraints. Such graph-based optimization problems frequently arise in transportation systems, logistics planning, communication networks, and supply chain management.

Consider the vertices: A (source / s), B, C, D (sink / t).

For each directed edge are shown in the below given list (distance, cost, and capacity)**:**

- $A \rightarrow B$: $(3,\ 2,\ 3)$
- $A \rightarrow C$: $(1,\ 5,\ 2)$
- $B \rightarrow C$: $(1,\ 1,\ 1)$
- $B \rightarrow D$: $(6,\ 2,\ 2)$
- $C \rightarrow D$: $(4,\ 3,\ 3)$

(If an entry is missing the edge does not exist. No negative weights here.)

1) Floyd–Warshall (all-pairs shortest distances using distance metric)

Order of vertices: A, B, C, D. Initialize distance matrix $D^{(0)}$

	A	B	C	D
A	0	3	1	∞
B	∞	0	1	6
C	∞	∞	0	4
D	∞	∞	∞	0

Run Floyd–Warshall stepwise ($k = A, B, C, D$):

- $k = A$: no improvements (no useful paths that go through A change anything).
- $k = B$: check pairs using B as intermediate:
- $A \to D$ via A□−□B□−□$D = 3 + 6 = 9$.
- $k = C$: $check$ via C:
  - $A \to D$: via A□−□C□−□$D = 1 + 4 = 5$. $5 < 9 \ \to$ update A → D = 5
- $k = D$: no outgoing edges from D, nothing changes.
- Final all-pairs shortest distance matrix:

	A	B	C	D
A	0	3	1	5
B	∞	0	1	5
C	∞	∞	0	4
D	∞	∞	∞	0

So, for example:

- $Shortest\ distance\ A \to D = 5$(via A□−□C□−□D)
- $Shortest\ distance\ B \to D = 5$ (via B□−□C□−□D

2) Ford–Fulkerson (maximum flow from s=A to t=D using capacities)

Capacities:

$$A \to B: 3, \mathrm{A} \to \mathrm{C}: 2, \mathrm{B} \to \mathrm{C}: 1\mathrm{B}, \mathrm{B} \to \mathrm{D}: 2, \mathrm{C} \to \mathrm{D}: 3.$$

We find augmenting paths and send flow until no path remains (classic Ford–Fulkerson with simple path choices).

1. Choose path $A \to B \to D$.
   - Residual capacity = $min(3, 2) = 2$.
   - Augment 2 units.
   - Flows now: $f(A$□−□$B) = 2,\ f(B$□−□$D) = 2$
   - Remaining capacities: A□−□$B: 1,\ B$□−□$D: 0$
2. Choose path $A \to C \to D$.
   - Residual cap = $min(2, 3) = 2$.
   - Augment 2 units.

- Flows now: $f(A \to C) = 2,\ f(C \to D) = 2.$
- Remaining caps: $A \to C: 0,\ C \to D: 1$

3. There's still residual capacity $A \to B = 1, B \to C = 1, C \to D = 1.$ *So choose path*

$$A \to B \to C \to D$$

- Residual cap $= min(1,1,1) = 1.$
- Augment 1 unit.
-
- Remaining caps: $A \to B: 0,\ B \to C: 0,\ C \to D: 0$
- Now source A has no outgoing residual capacity ($both\ A \to B\ and\ A \to C\ are\ saturated$). So no further augmenting path exists.

Maximum flow = sum of augmentations = 2 + 2 + 1 = 5.
Final flows on edges:

- $f(A \to B) = 3\ (full), f(A \to C) = 2\ (full)$
- $f(B \to C) = 1$
- $f(B \to D) = 2$
- $f(C \to D) = 3$

Check: capacity out of source $= 3 + 2 = 5, matches\ max\ flow\ 5.$

3) ACASPO — Cost-aware shortest path (minimize combined function F)

We combine distance and cost into one scalar: choose

$$F(u, v) = \alpha \cdot distance(u, v) + \beta \cdot cost(u, v),$$

take $\alpha = 1,\ \beta = 1$ (equal weighting). Compute F per edge:

- $A \to B$: $F = 3 + 2 = 5$
- $A \to C$: $F = 1 + 5 = 6$
- $B \to C$: $F = 1 + 1 = 2$
- $B \to D$: $F = 6 + 2 = 8$
- $C \to D$: $F = 4 + 3 = 7$

Now find the shortest path from A to D under these combined edge weights. Evaluate $A \to B \to D$: $A \to B \to D$: $cost = 5 + 8 = 13$

  - $(Total\ distance = 3 + 6 = 9, total\ cost = 2 + 2 = 4.)$
- $A \to C \to D$: $A \to C \to D$: $cost = 6 + 7 = 13.$
  - $(Total\ distance = 1 + 4 = 5, total\ cost = 5 + 3 = 8)$
- $A \to B \to C \to D$: $A \to B \to C \to D$: $cost = 5 + 2 + 7 = 14$
- So under $F = distance + cost$, there is a tie: two paths have the same total combined cost 13:
- $A \to B \to D$ $(combined\ 13;\ distance\ 9, cost\ 4)$
- $A \to C \to D$ $(combined\ 13;\ distance\ 5, cost\ 8)$

If we want to break ties by preferring a smaller distance, we choose the route $A \to C \to D$ (distance $5 < 9$). If we prefer a smaller monetary cost, we select the route $A \to B \to D$ (monetary cost $4 < 8$).

ACASPO remains flexible; after computing the combined scores, we can adopt a tie-breaking policy. By using the distance tie-breaker, ACASPO selects $A \rightarrow C \rightarrow D$ with:

- Combined $F = 13$
- Distance = 5 (which matches the Floyd-Warshall shortest distance from ($A \rightarrow D$)
- Monetary $cost = 8$

We can also adjust the weights α, β ; this choice can influence the outcome. For instance, if we weigh the cost more heavily than the distance, we might prefer the route $A \rightarrow B \rightarrow D$.

COMPARISON ANALYSIS

The performance of the three algorithms Floyd–Warshall**,** Ford–Fulkerson, and the proposed ACASPO is compared on the same network topology as follows:

- Floyd–Warshall Algorithm (Distance-Based): The shortest path from $A \rightarrow D$ is obtained through $A - C - D$, with a minimum total distance of 5**.**
- Ford–Fulkerson Algorithm (Capacity / Flow-Based): The maximum flow from A → D is 5 units, with the following individual edge flows: $A - B = 3, A - C = 2, B - D = 2, B - C = 1, C - D = 3.$
- ACASPO Algorithm ($Hybrid\ Distance + Cost, with\ \alpha = \beta = 1$): The algorithm identifies two equivalent optimal paths, $A - B - D$ and $A - C - D$**,** each with a combined cost of 13. When prioritizing shorter distance, the tie is resolved in favor of $A - C - D$.

The ACASPO algorithm is particularly advantageous for travelers because it considers both travel distance and monetary cost simultaneously, rather than optimizing only a single parameter. Traditional algorithms such as Floyd–Warshall focus solely on minimizing distance, which may lead to routes that are shorter but more expensive. Similarly, approaches that focus only on cost may result in routes that are cheaper but significantly longer.

By combining these two factors into a unified evaluation function, ACASPO provides a balanced decision-making framework that better reflects real-world travel preferences. In practical travel scenarios, individuals often need to choose routes that offer a reasonable compromise between travel time and travel expenses.

In the present proposed algorithmic experiment, ACASPO identifies two feasible routes from A to D, namely $A - B - D$ and $A - C - D$, both having the same combined score. This demonstrates the flexibility of the algorithm in recognizing multiple viable travel options. When the decision criterion prioritizes shorter travel distance, the algorithm selects $A - C - D$, which aligns with the shortest path identified by Floyd–Warshall. Conversely, if minimizing cost is preferred, the route $A - B - D$ becomes more attractive due to its lower monetary expense.

Therefore, ACASPO is favourable for travellers because it:

- Balances travel distance and cost
- Provides multiple optimal route options
- Allows travelers to choose routes based on personal preferences
- Adapts easily by adjusting weighting parameters

This flexibility makes ACASPO a more practical and traveler-oriented routing strategy compared to traditional single-objective algorithms.

CONCLUSION

This research has presented a comprehensive comparative study of classical pathfinding algorithms, namely, the Floyd–Warshall algorithm for all-pairs shortest path computation and the Ford–Fulkerson algorithm for network flow optimization alongside the proposed Adaptive Cost-Aware Shortest Path Optimization (ACASPO) algorithm. Through theoretical analysis and simulation-based evaluation, the study demonstrated that while traditional methods remain foundational for static and well-defined network structures, they exhibit limitations when applied to dynamic or large-scale environments where real-time adaptability is crucial.

The proposed experiment through ACASPO algorithm effectively addresses these challenges by incorporating adaptive edge-weight adjustment, heuristic guidance, and real-time network feedback mechanisms. The results indicate significant improvements in computational efficiency, path optimality, and responsiveness under fluctuating network conditions such as congestion, latency variation, and cost dynamics. Consequently, ACASPO brings together the mathematical strength of old algorithms (Floyd–Warshall and Ford–Fulkerson) helps to give the flexibility of modern adaptive systems to create a smarter and more practical pathfinding approach.

FUTURE WORKS AND RESEARCH EXTENSIONS

Building on the promising outcomes of this study, several avenues for future research emerge:

1. **Integration with AI and Machine Learning Models:**
   Incorporating predictive analytics or reinforcement learning into ACASPO could enhance its ability to forecast network conditions and proactively adjust routing strategies.
2. **Scalability in Real-World Networks:**
   Further experimentation on large-scale datasets, such as smart city transportation systems or cloud-based communication infrastructures, can validate the algorithm's scalability and real-time deployment feasibility.
3. **Multi-Objective Optimization:**
   Extending ACASPO to consider multiple simultaneous objectives such as minimizing energy consumption, environmental impact, or user discomfort would make it applicable to sustainable and intelligent mobility systems.
4. **Hybridization with Meta-heuristics:**
   Combining ACASPO with evolutionary or swarm-based optimization techniques could yield hybrid models capable of handling uncertain and nonlinear optimization landscapes.
5. **Implementation in IoT and Edge Environments:**
   Future work could explore ACASPO's adaptability in Internet of Things (IoT) and edge-computing contexts, where low latency and localized decision-making are essential.

REFERENCES

1. Alsubaie, N., & Diabat, A. (August 2020). "Hybrid metaheuristics for dynamic shortest path problems" . *Applied Soft Computing*.Bulletin of Electrical Engineering and Informatics Vol. 10 Issue 4: pages 2152-2162. DOI:10.11591/eei.v10i4.2836.
2. Bast, Hannah, et al. (April 2015) "Route planning in transportation networks." *Algorithm engineering: Selected results and surveys*. Cham: Springer International Publishing. P.P. 19-80. DOI:10.48550/arXiv.1504.05140.

3. Chen, Bi Yu, et al. (2013) "Finding reliable shortest paths in road networks under uncertainty." *Networks and spatial economics.* Volume 13, Issue 2, pages 123-148.
4. Cormen, Thomas H., et al. (April- 2022), *Introduction to algorithms*. MIT press.ISBN: 9780262046305.
5. Deb, Kalyanmoy. (2015) "Multi-objective evolutionary algorithms." *Springer handbook of computational intelligence*. Berlin, Heidelberg: Springer Berlin Heidelberg. : P.P. 995-1015.
6. Deb, Kalyanmoy, et al. (2002), "A fast and elitist multiobjective genetic algorithm: NSGA-II." *IEEE transactions on evolutionary computation* 6.2 :P.P. 182-197
7. Delling, Daniel, et al. (2009) "Engineering route planning algorithms." *Algorithmics of large and complex networks: design, analysis, and simulation*. Berlin, Heidelberg: Springer Berlin Heidelberg, P.P. 117-139.
8. Dorigo, Marco, and Luca Maria Gambardella.(1997) "Ant colonies for the travelling salesman problem." *biosystems* Volume 43, Issue.2 :P.P. 73-81.
9. Dijkstra, Edsger Wybe. (1996). "Edsger W. Dijkstra." *Great Papers in Computer Science* 378.
10. Edmonds, Jack, and Richard M. Karp. (1972). "Theoretical improvements in algorithmic efficiency for network flow problems." *Journal of the ACM (JACM)* Volume 19, Issue.2 : P.P. 248-264.
11. Floyd, Robert W. (1962) "Algorithm 97: shortest path." *Communications of the ACM* . Volume 5, Issue 6 : P.P. 345-345.
12. Ford Jr, Lester R., and Delbert R. Fulkerson. (1956) "Maximal flow through a network." *Canadian journal of Mathematics* Volume 8 : Pages 399-404.
13. Goldberg, David E. (1989) "Genetic algorithm in search, optimization and machine learning, addison." *W esley Publishing Company, Reading, MA*. P.P. 9.
14. Gutjahr, Walter J. (2011) "Ant Colony Optimization: Recent Developments in Theoretical Analysis." Theory of Randomized Search Heuristics : P.P. 225-254. https://doi.org/10.1142/9789814282673_0008
15. Hart, Peter E., Nils J. Nilsson, and Bertram Raphael.(1968) "A formal basis for the heuristic determination of minimum cost paths." *IEEE transactions on Systems Science and Cybernetics* Volume 4.issue 2, P.P. 100-107.
16. Kirkpatrick, Scott, C. Daniel Gelatt Jr, and Mario P. Vecchi.(1983) "Optimization by simulated annealing." *science* 220.4598. Vol 220, Issue 4598. PP. 671-680 :.DOI: 10.1126/science.220.4598.671
17. Lancia, Giuseppe, and Marcello Dalpasso. (2025) "Speeding Up Floyd–Warshall's Algorithm to Compute All-Pairs Shortest Paths and the Transitive Closure of a Graph." *Algorithms* Volume 18 issue 9 : pages 560.
18. Lin, Zhiyi, et al. (2025) "Heuristic-Based Computing-Aware Routing for Dynamic Networks." *Electronics* Volume 14. Issue 18 : P.P. 3724.
19. Liu, Gangli. (Dec-2024) "Solving the all pairs shortest path problem after minor update of a large dense graph." *arXiv preprint arXiv :* P.P. 2412.15122. DOI:10.48550/arXiv.2412.15122

20. Ritzinger, Ulrike, and Jakob Puchinger. (2013) "Hybrid metaheuristics for dynamic and stochastic vehicle routing." *Hybrid metaheuristics*. Berlin, Heidelberg: Springer Berlin Heidelberg, P.P. 77-95.
21. Thomas H. Cormen, Charles E.”Introduction to Algorithms” MIT Press. ISBN: 9780262046305