Digital Technologies Research and Applications
http://ojs.ukscip.com/index.php/dtra

Article

Extended Dijkstra Algorithm and Moore-Bellman-Ford Algorithm

Congdian Cheng

School of Intelligence Technology, Geely University of China, Chengdu 641400, China; zhiyang918@163.com

Received: 29 October 2025; **Revised:** 25 November 2025; **Accepted:** 24 December 2025; **Published:** 12 February 2026

Abstract: The shortest path problem which can not be solved by classical Dijkstra algorithm and Moore-Bellman-Ford algorithm appears frequently, for example, the Anti-risk Path Problem proposed by Xiao et al. To address this kind of shortest path problem, the present work proposes and studies a general single-source shortest path problem, which is motivated by current interest in needing to extend the total weight function of paths on a network and the classical shortest path problem. Firstly, define the path functional on a set of certain paths with same source on a graph; introduce a few concepts of the defined path functional; and make some discussions on the properties of the path functional. Secondly, develop a kind of general single-source shortest path problem (GSSSP). Thirdly, following respectively the approaches of the well known Dijkstra's algorithm and Moore-Bellman-Ford algorithm, design an extended Dijkstra's algorithm (EDA) and an extended Moore-Bellman-Ford algorithm (EMBFA) to solve the problem GSSSP under certain given conditions. Fourthly, under the assumption that the value of related path functional for any path can be obtained in $M(n)$ time, prove respectively the algorithm EDA solving the problem GSSSP in $O(n^2)M(n)$ time and the algorithm EMBFA solving the problem GSSSP in $O(mn)M(n)$ time. Finally, some applications of the designed algorithms are shown with a few examples. What we done can not only improve both the researches and the applications of the shortest path theory, but also promote the development of the researches and the applications of other combinatorial optimization problems, promote the development of the algorithm theory and promote the development of the artificial intelligence.

Keywords: Graph; Network; Path Functional; Shortest Path; Algorithm

1. Introduction

Shortest path problems are the best known class of combinatorial optimization problems and have been extensively studied for more than half a century, which have many applications in network, electrical routing, transportation, robot motion planning, critical path computation in scheduling, quick response to urgent relief, etc.; and can also unify framework for many optimization problems such as knapsack, sequence alignment in molecular biology, inscribed polygon construction, and length-limited Huffman-coding, etc. For the basic knowledge of shortest path problems, please refer to chapter 7 of the monograph of Korte and Vygen [1] and the other literatures afterwords.

The classical single-source shortest path problem of network, denoted by CSSSP, is the most famous one of shortest path problems, and a lot of works have been done to study and solve this kind of shortest path problem. Among many algorithms for the problem CSSSP, Dijkstra's Algorithm (DA) and Moore-Bellman-Ford Algorithm (MBFA), called also by Bellman-Ford Algorithm, are two well-known and most fundamental, which have now been the core technique to solve many optimization problems. As we all know, the first one can solve the problem CSSSP with nonnegative edge weights in $O(n^2)$ time and the second one can deal with the problem CSSSP with arbitrary conservative weights in $O(nm)$ time. Here, n and m denote respectively the number of vertices and the number of edges on the underlying graph. See, e.g., chapter 7 of the monograph, and the studies of Bellman [2], Dijkstra [3], Ford Jr. [4], and Moore [5].

The study of shortest paths is a research area with a long history. And there are still substantial research works in the area in this century. On the work over the last two decades, please see the following introduction, for example.

Motivated by the recent interest in pricing networks and other computational problems, Hershberger and Suri [6] studied the problem how to determine the Vickrey payments of all the agents between the given two nodes with a pricing network in less time, which is closely related to the shortest path algorithm and the replacement path problem, see the study of Hershberger et al. [7], and they proposed an algorithm to complete the computation in essentially the same time bound as a single-source shortest path computation. Hershberger et al. [7] also explained and investigated the replacement path problem and the other shortest path problems; and made some results on the time complexity of computing the related problems. Cho et al. [8] developed and studied a hybrid shortest path algorithm of navigation system. Du et al. [9] based on the known heuristic algorithm CST for Euclidean Steiner Tree (EST), proposed the algorithms CST(A) to find EST, and by making the worst-case analysis of algorithms CST and CST(A), presented the restricted submodularity technique to analyze approximation algorithm with nonsubmodular functions; for Connected Dominating Set (CDS), based on the known 1—greedy algorithm, they also proposed $(2k-1)$—greedy algorithm to compute a minimum CDS, and by making the worst-case analysis of $(2k-1)$—greedy algorithm, presented the shifted submodularity technique to analyze approximation algorithm with nonsubmodular functions; in the process they obtain some excellent results.

Noted the fact that the transportation system of a city and the roads of the city can be respectively modeled by a network and its edges, some roads of which may be blocked at certain times; and the traveler only observes that upon reaching an adjacent site of the blocked road. Xiao et al. [10] (2009) introduced the definition of the risk of paths on a network, which is really a function on the set of all the paths with a same source on the network; and introduced also the anti-risk path (ARP) problem of finding a path such that the solution of its has minimum risk, which, on the one hand, is a kind of single-source shortest path problem, and on the other hand, is different from the classical single-source shortest path problem, so cannot be solved by the classical DA and MBFA. They showed also the ARP problem can be solved in $O(mn + n^2 logn)$ time supposed that at most one edge may be blocked. Afterwards, Mahadeokar and Saxena [11] (2014) proposed a faster algorithm to solve the ARP problem.

Srivastava and Tyagi [12] by modifying the Prim's Algorithm, proposed an algorithm to find the shortest path in a specific network that consists of host systems on land and satellites in air. Murota and Shioura [13] from the viewpoint of discrete convex analysis and linear programming formulation, showed that the shortest path problem can be seen as a special case of L-concave function maximization; and solving the LP dual of the shortest path problem with the steepest ascent algorithm for L-concave function maximization is exactly coincident to solving the shortest path problem with Dijkstra's algorithm. Feng [14] presented a new exact algorithm to solve k shortest simple paths (KSP) in a network; and demonstrated that the algorithm performs significantly better than the existing exact polynomial time algorithm that have polynomial worst-case complexity. Basing on the concept of the replacement path and the real time detour path, Zhang et al. [15] proposed the definition of shortest path set and the definition of optimal shortest path set, on an undirected graph with source node s and destination node t; they also, under some conditions, investigated and presented the polynomial time algorithms to compute the optimal shortest path set. By combining the flow shop scheduling problem and the shortest path problem, Nip et al. [16] first developed a synthetical optimization problem; then they discussed the complexity to compute the solutions and separately proposed two approximation algorithms, for the case that the number of machines is an input and for the case that the number of machines is fixed. Meng et al. [17] reported the results of their experiment and research on the multiple shortest path algorithms. By combining Bellman-Ford algorithm and Dijkstra algorithm, Dinitz and Itzhak [18] presented a new hybrid algorithm for the single-source shortest path problem with general edge costs, which can improve the running time bound of Bellman-Ford algorithm for graphs with a sparse distribution of negative cost edges; and made some related researches; in addition, they also suggested a new straightforward proof that the Bellman-Ford algorithm produces a shortest paths tree. Noting the wide and important applications of Dijkstra algorithm and Bellman-Ford algorithm in the field of computer and software sciences, and many researches interested in the problem to solve the shortest path problem in these applications, AbuSalim et al. [19] made a more profound comparison between the two popular algorithms on the complexity and the performance in terms of shortest path optimization. Xia et al. [20], based Ant Colony Optimization ACO algorithm, proposed BiA*-ACO algorithm to recommend the fastest route for taxicabs in a complex urban road network; and showed it is at least 49.81% more efficient than the algorithms ACO, DA and MBFA. Gajjar et al. [21] developed and studied a problem of reconfiguring shortest paths.

In addition, the types and applications of functional are expanding, e.g., see the studies of Wu [22] and Xu et al. [23], and the types and applications of networks are expanding, see the studies of Bäusing et al. [24], Deen et al. [25], Henke and Wulf [26], Hooshmand and Huchaiah [27], and Kovács [28] for example. Moreover, in view of the critical role of Manhattan Steiner Problem in VLSI-design, we can clearly know that algorithms and networks are crucial to the intelligent technology. At present, much work has been done to develop the intelligent technology through the research of algorithms and networks, for example, see the studies of Acuña et al. [29], Mao et al. [30], and Jahanmanesh et al. [31].

Motivated by the stated background of researches above, in particular the studies of Xiao et al. [10,11], to meet practical needs, to deepen algorithm theory and to improve intelligence level, the present paper will carry out the following work.

1. Through generalizing the total weight path function of networks as the path functional of graphs, develop a general single-source shortest path problem (GSSSP), which include the classical problem CSSSP and the ARP problem as its special cases.
2. Try to design an Extended Dijkstra's Algorithm (EDA) and an Extended Moore-Bellman-Ford Algorithm (EMBFA) to solve the problem GSSSP under certain conditions, which respectively reduce to Dijkstra's Algorithm and Moore-Bellman-Ford Algorithm when the problem GSSSP is the classical problem CSSSP.
3. Moreover, make some related studies to analyse and believe the two designed algorithms.

2. Preliminaries

This section provides some preliminaries for our sequel research.

Artificial intelligence (AI) is a key modern technology. Digitalization is the foundation of AI. To effectively support AI development, we will strive to enhance the level of digitalization of content, here.

2.1. Conceptual Framework

(1) Suppose V is a set of $n(>1)$ points.

Let $u, v \in V$ and $u \neq v$. We use $[u, v]$ to denote an edge connecting two points u and v. And use (u, v) ((v, u)) to denote a road from u to v (v to u) on the edge $[u, v]$. (Note: $[u, v] = [v, u]$, while $(u, v) \neq (v, u)$.)

When there are more than one edge between u and v, namely there are the parallel edges between u and v, $[u, i, v]$ may be used to denote the ith edge; and (u, i, v) may be used to denote the ith road from u to v. However, to simplify in notation afterwards, $[u, i, v]$ is denoted as $[u, v]$ and (u, i, v) is denoted as (u, v) when i needn't be indicated.

A edge $[u, v]$ is called undirected (/directed) if there are two roads (u, v) and (v, u) (/there is only one road (u, v) or (v, u)) on it.

For set A, we use $|A|$ denotes the number of all elements in the set A.

Let E be all the edges and R be all the roads. The triple (V, E, R) is called as a graph, which is also denoted by the tuple (V, E) $(/(V, R))$ when all the roads R (/edges E) needn't be indicated for clearness and briefness. Graphs without parallel edges are called simple. For graph (V, E, R), an element of V is called a vertex of the graph. In this work we always assume that $|E|$ is finite.

A graph is called as an undirected graph (/directed graph) if it has only undirected edges (/has only directed edges).

(2) Suppose $G = (V, E, R)$ is a graph.

When $[v_{(i-1)}, v_i] \in E, i = 1, 2, \cdots, k$, the orderly combination of edges,

$$\{[v_0, v_1], [v_1, v_2], \cdots, [v_{k-1}, v_k]\}([v_{(i-1)}, v_i] \neq [v_{(j-1)}, v_j], i \neq j)$$

is called as a chain connecting v_0 and v_k, denoted by $C[v_0, v_1, v_2, \cdots, v_{k-1}, v_k]$. (Note: $C[v_0, v_1, v_2, \cdots, v_{k-1}, v_k] = C[v_k, v_{k-1}, \cdots, v_2, v_1, v_0]$.) If $v_0 = v_k$, the chain is called as a cycle (chain).

When $(v_{(i-1)}, v_i) \in R, i = 1, 2, \cdots, k$, the orderly combination of roads,

$$\{(v_0, v_1), (v_1, v_2), \cdots, (v_{k-1}, v_k)\}$$

is called as a path from v_0 to v_k, denoted by $P(v_0, v_1, v_2, \cdots, v_{k-1}, v_k)$, in briefness denoted by $P(v_0, v_k)$, if there will be no confusion; v_k is said to be reachable from v_0. (Note: $P(v_0, v_1, v_2, \cdots, v_{k-1}, v_k) \neq P(v_k, v_{k-1}, \cdots, v_2, v_1, v_0)$.) We also use $(v_0, v_1, \cdots, v_{k-1}, v_k)$, or $(v_0, v_1) + (v_1, v_2) + \cdots + (v_{k-1}, v_k)$, or $(v_0, v_1, \cdots, v_{k-1}) + (v_{k-1}, v_k)$ to denote $P(v_0, v_k)$. That is,

$$\begin{aligned} & P(v_0, v_k) = \{(v_0, v_1), (v_1, v_2), \cdots, (v_{k-1}, v_k)\} = (v_0, v_1, \cdots, v_{k-1}, v_k) \\ = \quad & (v_0, v_1) + (v_1, v_2) + \cdots + (v_{k-1}, v_k) = (v_0, v_1, \cdots, v_{k-1}) + (v_{k-1}, v_k). \end{aligned}$$

v_0 and v_k are respectively called the source and the terminal of path P, denoted by $s(P)$ and $t(P)$; $(v_{(i-1)}, v_i)$, $0 < i \leq k$, is called a road of P, denoted by $(v_{(i-1)}, v_i) \in P$. For path $P = (v_0, v_1, \cdots, v_{k-1})$ and path $P' = (v_0, v_1, \cdots, v_{k-1}, v_k) = P + (v_{k-1}, v_k)$, we call P as the father of P', denoted by $F(P')$; and P' as a son of P, denoted by $S(P)$. A path $P = (v_0, v_1, \cdots, v_{k-1}, v_k)$ is called no cycle if $v_i \neq v_j$ while $i \neq j$.

A mapping $w : E(/R) \to (-\infty, \infty)$ is called as a weight of edges (/roads). And the triple (V, w, E) $(/(V, w, R))$ is called as a network with edge (/road) weight. The tuple (G, w) is used to represent both the networks (V, w, E) and (V, w, R) when $w([u, v]) = w((u, v)), \forall u, v \in V$.

Let $s \in V$ (called source point). A path of graph G with the source point s is called a path of $[G, s]$. Some paths of $[G, s]$ is called a path system on $[G, s]$. All the paths of $[G, s]$ is called the complete path system on $[G, s]$. All the no cycle paths of $[G, s]$ is called the no cycle path system on $[G, s]$. To be convenient and clear, we stipulate (s, s) is a special road and a special path with the source point s, and $(s, s) + (s, v) = (s, v)$ for each road (s, v).

(3) Suppose $\mathcal{P}$ is a path system on $[G, s]$.

Put $\mathcal{P}(u) = \{P \in \mathcal{P} | t(P) = u\}$ (particulary, $\mathcal{P}(s) = \{(s, s)\}$), $V(\mathcal{P}) = \{u | \mathcal{P}(u) \neq \emptyset\}$, $R(\mathcal{P}) = \{(u, v) \in P | P \in \mathcal{P}\}$ and $E(\mathcal{P}) = \{[u, v] | (u, v) \in R(\mathcal{P})\}$. Moreover, $\mathcal{P}_{nc}$ denotes all the no cycle paths in $\mathcal{P}$, and $\mathcal{P}_{nc}(u)$ denotes all the no cycle paths in $\mathcal{P}(u)$.

$\forall P \in [\mathcal{P} \setminus \{(s, s)\}]$, define $(s, s) \prec P$; $\forall P, P' \in [\mathcal{P} \setminus \{(s, s)\}]$, define $P \prec P'$ if and only if $P = (v_0, v_1, \cdots, v_k)$, $P' = (v_0, v_1, \cdots, v_k, v_{k+1}, \cdots, v_{k'})$, $0 \leq k < k'$; and $\forall P, P' \in \mathcal{P}$, define $P \preceq P'$ if and only if $P \prec P'$ or $P = P'$.

A mapping $f : \mathcal{P} \to (-\infty, \infty)$ is called as a path functional on $\mathcal{P}$.

Finally, $\forall v \in V(\mathcal{P})$, define $m_f(v) = \inf\{f(P) | P \in \mathcal{P}(v)\}$. And $P \in \mathcal{P}$ is called a shortest (minimum) path on f if $f(P) = m_f(t(P))$.

Below, we always assume that G is a graph, $s \in V(G)$ is a source, $\mathcal{P}$ is a path system on $[G, s]$, f is a path functional on the system $\mathcal{P}$ and $f((s, s)) = 0$. In addition, please note: $P(\in \mathcal{P})$ denotes a path of system $\mathcal{P}$; $s(P)$ denotes the source of P and $t(P)$ denotes the terminal of P; $S(P)$ denotes the son of P and $F(P)$ denotes the father of P; $f(P)$ represents the value of functional f at path P. In addition, we should note: 1. P denotes a path with source s; 2. $s(P)$ denotes the source of P and $t(P)$ denotes the terminal of P; 3. $S(P)$ denotes the son of P and $F(P)$ denotes the father of P; 4. f denotes a path functional of $\mathcal{P}$; 5. $f(P)$ indicates the value of path functional f in path P.

2.2. Basic Definitions

Definition 1. *(i.)* *f is said to be non-decreasing if and only if $\forall P, P' \in \mathcal{P}$, provided $P \preceq P'$, we have $f(P) \leq f(P')$.*

(ii.) *f is said to be increasing if and only if $\forall P, P' \in \mathcal{P}$, provided $P \prec P'$, we have $f(P) < f(P')$.*

(iii.) *f is said to be weak order-preserving (WOP) if and only if $\forall u, v \in V$, provided $\forall P, P' \in \mathcal{P}(u)$, $P + (u, v), P' + (u, v) \in \mathcal{P}(v)$ and $f(P) < f(P')$, we have $f(P + (u, v)) < f(P' + (u, v))$.*

(iv.) *f is said to be semi -order-preserving (SOP) if and only if $\forall u, v \in V$, provided $\forall P, P' \in \mathcal{P}(u)$, $P + (u, v), P' + (u, v) \in \mathcal{P}(v)$ and $f(P) \leq f(P')$, we have $f(P + (u, v)) \leq f(P' + (u, v))$.*

(v.) *f is said to be order-preserving (OP) if and only if f is WOP, and $\forall u, v \in V$, provided $\forall P, P' \in \mathcal{P}(u)$, $P + (u, v), P' + (u, v) \in \mathcal{P}(v)$ and $f(P) = f(P')$, we have $f(P + (u, v)) = f(P' + (u, v))$.*

Definition 2. *f is said to have no negative (/non-positive) cycle if and only if $\forall v \in V$, provided $\forall P \in \mathcal{P}(v)$ and $\forall P' = P + (v, v_1, \cdots, v_k, v) \in \mathcal{P}(v)$, we have $f(P') - f(P) \geq 0$ $(/f(P') - f(P) > 0)$. f is called conservative if it has no negative cycle.*

Definition 3. *f is said to be weak inherited on shortest path (WISP) if $\forall v \in [V(\mathcal{P}) \setminus \{s\}]$, provided that the shortest path from s to v exists, then there must be a path $P = (v_0, v_1, \cdots, v_k) \in \mathcal{P}(v), k \geq 1$, such that $P_i = (v_0, v_1, \cdots, v_i)$ is the shortest path, $i = 1, 2, \cdots, k$. (Note: $v_k = v$). f is said to be inherited on shortest path (ISP) if $\forall P \in [\mathcal{P} \setminus \{(s, s)\}]$, provided that P is the shortest path, then $F(P)$ must be the shortest path.*

2.3. Basic Propositions

For the above definitions, we have the following statements.

Proposition 1. $\forall v \in V$, *$|\mathcal{P}_{nc}(v)|$ is finite.*

Proposition 2. *If f is increasing, then it must have no non-positive cycle. If f has no non-positive cycle or is non-decreasing, then it must have no negative cycle; that is, f is conservative.*

Proposition 3. *If f is OP, then it must be WOP and SOP.*

For the proofs of the three propositions above is trivial, here we omit them.

Proposition 4. *Let $\mathcal{P}$ be the complete path system. (i) If f has no non-positive cycle and is WOP, then, $\forall v \in [V(\mathcal{P})\setminus\{s\}]$, the shortest path from s to v has no cycle, and $m_f(v) = \min\{f(P)|P \in \mathcal{P}_{nc}(v)\}$. (ii) If f is conservative and SOP, then, $\forall v \in [V(\mathcal{P}) \setminus \{s\}]$, $m_f(v) = \min\{f(P)|P \in \mathcal{P}_{nc}(v)\}$. (iii) If f is conservative, SOP and WISP, then, $\forall v \in [V(\mathcal{P}) \setminus \{s\}]$, there is a path $P = (v_0, v_1, \cdots, v_{k-1}, v_k) \in \mathcal{P}_{nc}(v)(k \geq 1)$ such that $P_i = (v_0, v_1, \cdots, v_i)$ is the shortest path for any $i = 0, 1, 2, \cdots, k$.*

Proof. $\forall v \in [V(\mathcal{P}) \setminus \{s\}]$, let P be a path from s to v and have cycles. We can assume

$$P = (v_0, v_1, \cdots, v_l, v'_1, v'_2, \cdots, v'_{l'}, v_l, v_{l+1}, v_{l+2}, \cdots, v_k)$$

$(v_k = v)$, $l' \geq 1$. Put $P_1 = (v_0, v_1, \cdots, v_l)$, $P_2 = (v_0, v_1, \cdots, v_l, v'_1, v'_2, \cdots, v'_{l'}, v_l)$ and $P_3 = P_1 + (v_l, v_{l+1}) + \cdots + (v_{k-1}, v_k) = (v_0, v_1, \cdots, v_l, v_{l+1}, v_{l+2}, \cdots, v_{k-1}, v_k)$. For $\mathcal{P}$ is the complete path system, $P_1, P_2, P_3 \in \mathcal{P}$.

(1) For f has no non-positive cycle, we have $f(P_2) > f(P_1)$. Since also f is WOP, we further have $f(P_2+(v_l, v_{l+1})) > f(P_1 + (v_l, v_{l+1})), \cdots, f(P_2 + (v_l, v_{l+1}) + \cdots + (v_{k-1}, v_k)) > f(P_1 + (v_l, v_{l+1}) + \cdots + (v_{k-1}, v_k))$. Thus, $f(P) > f(P_3)$. This implies that P can not be a shortest path from s to v and leads to $m_f(v) = \inf\{f(P)|P \in \mathcal{P}(v)\} \leq \min\{f(P)|P \in \mathcal{P}_{nc}(v)\} \leq \inf\{f(P)|P \in \mathcal{P}(v)\} = m_f(v) \Rightarrow m_f(v) = \min\{f(P)|P \in \mathcal{P}_{nc}(v)\}$. Hence (i) holds.
(2) For f is conservative, we have $f(P_2) \geq f(P_1)$. Since also f is SOP, we further have $f(P_2 + (v_l, v_{l+1})) \geq f(P_1+(v_l, v_{l+1})), \cdots, f(P_2+(v_l, v_{l+1})+\cdots+(v_{k-1}, v_k)) \geq f(P_1+(v_l, v_{l+1})+\cdots+(v_{k-1}, v_k))$. Thus $f(P) \geq f(P_3)$. This implies that there must be a shortest path from s to v such that it has no cycle. Hence (ii) holds.
(3) Let $v \in [V(\mathcal{P}) \setminus \{s\}]$. In terms of the conclusion (ii), there must be a shortest path from s to v. Since also f is WISP, we can further know there is a path $(v_0, v_1, v_2, \cdots, v_k) \in \mathcal{P}(v)$ such that $(v_0, v_1, v_2, \cdots, v_i), i = 0, 1, 2, \cdots, k$, are all the shortest pathes. Finally, following the approach to prove conclusion (ii), we can easily prove that there is a path $P = (v_0, v'_1, v'_2, \cdots, v'_{k'}) \in \mathcal{P}_{nc}(v)$ such that $(v_0, v'_1, v'_2, \cdots, v'_i), i = 0, 1, 2, \cdots, k'$, are all the shortest paths. Hence (iii) holds. □

Corollary 1. *Let $\mathcal{P}$ be the complete path system. If f is non-decreasing and SOP, then $\forall v \in V(\mathcal{P}), m_f(v) = \min\{f(P)|P \in \mathcal{P}_{nc}(v)\}$.*

Lemma 1. *Let $\mathcal{P}$ be the complete path system. Suppose also f has no non-positive cycle. If the path*

$$P = (v_0, v_1, \cdots, v_k, v_{(k+1)}, v_{(k+2)}, \cdots, v_{(k+l)}) \quad (k \geq 0, l \geq 1)$$

such that

$$P_{(k+i)} = (v_0, v_1, \cdots, v_k, v_{(k+1)}, v_{(k+2)}, \cdots, v_{(k+i)}), i = 1, 2, \cdots, l,$$

are all the shortest paths, then $v_{(k+i)} \neq v_{(k+j)}$ while $1 \leq i < j \leq l$.

Proof. It is obvious that $v_{(k+i)} \neq v_{(k+j)}$ when $j = i+1$. Assume that $1 \leq i < j \leq l, j-i \geq 2$, and $v_{(k+i)} = v_{(k+j)} = v$. Then, $f(P_{(k+i)}) = f(P_{(k+j)}) = m_f(v)$. For f has no non-positive cycle, we have $f(P_{(k+j)}) > f(P_{(k+i)})$. This is contradictory to $f(P_{(k+i)}) = f(P_{(k+j)})$. Hence the lemma holds. □

Proposition 5. *Let $\mathcal{P}$ be the complete path system. If f has no non-positive cycle and is SOP, then f is WISP.*

Proof. $\forall v \in [V(\mathcal{P}) \setminus \{s\}]$, let $P = (v_0, v_1, \cdots, v_k, v), k \geq 0$, be a shortest path. If $(v_0, v_1, \cdots, v_k)$ is not the shortest path, then, from the term (ii) of Proposition 4, there must be a shortest path $P' = (v_0, v'_1, \cdots, v'_{k'}, v_k)$. For $\mathcal{P}$ is complete, $[P' + (v_k, v)] \in \mathcal{P}$. Since f is SOP, we have $f(P' + (v_k, v)) \leq f(P)$. This implies that $[P' + (v_k, v)]$ is also the shortest path. If $(v_0, v'_1, \cdots, v'_{k'})$ is not the shortest path, then, in the same way, there must be a shortest path $P'' = (v_0, v''_1, \cdots, v''_{k''}, v'_{k'})$ such that $[P'' + (v'_{k'}, v_k)]$, $[P'' + (v'_{k'}, v_k) + (v_k, v)]$ are all the shortest path. $\cdots$ Assume that the step has been performed l times. Then we can obtain a path $P^* = (v_0, v^*_1, \cdots, v^*_{k^*}, v^*_{(k^*+1)}, v^*_{(k^*+2)}, \cdots, v^*_{(k^*+l)}) \in \mathcal{P}(v)$ $(k^* \geq 0, l \geq 1)$ such that,

$$P_{(k^*+i)} = (v_0, v^*_1, \cdots, v^*_{k^*}, v^*_{(k^*+1)}, v^*_{(k^*+2)}, \cdots, v^*_{(k^*+i)}), i = 1, 2, \cdots, l,$$

are all the shortest paths. By Lemma 1, we have $v^*_{(k^*+i)} \neq v^*_{(k^*+j)}$ while $1 \leq i < j \leq l$. This implies that $l \leq n$. Therefore, we can eventually get a shortest path $\bar{P} = (v_0, \bar{v}_1, \cdots, \bar{v}_{\bar{k}}, v)$ such that,

$$\bar{P}_i = (v_0, \bar{v}_1, \cdots, \bar{v}_{\bar{k}}), i = 1, 2, \cdots, \bar{k},$$

are all the shortest paths. Hence the proposition holds. □

Proposition 6. *Let $\mathcal{P}$ be the complete path system. If f is conservative, SOP and WOP, then f is ISP.*

Proof. Let $P = (v_0, v_1, \cdots, v_k) \in \mathcal{P}, k \geq 1$, be a shortest path. If $F(P)$ is not the shortest path, then, from the term (ii) of Proposition 4, there must be another path $P' \in \mathcal{P}_{nc}(v_{k-1})$ such that $f(P') = m_f(v_{k-1}) < f(F(P))$. For $\mathcal{P}$ is complete path system, we have $[P' + (v_{k-1}, v_k)] \in \mathcal{P}(v_k)$. Since also f is WOP and $[P' + (v_{k-1}, v_k)], [F(P) + (v_{k-1}, v_k)] \in \mathcal{P}(v_k)$, we further have $f(P' + (v_{k-1}, v_k)) < f(F(P) + (v_{k-1}, v_k)) = f(P)$, which is contradictory with that P is the shortest path. Hence the proposition holds. □

Corollary 2. *Let $\mathcal{P}$ be the complete path system. If f is conservative and OP, then $\forall v \in V(\mathcal{P}), m_f(v) = \min\{f(P) | P \in \mathcal{P}_{nc}(v)\}$ and f is ISP.*

Proof. From Propositions 4 and 6, we can easily know the corollary holds. □

Remark 1. *The propositions above are not only the basis for our to design and study the next algorithms EDA and EMBFA, but also fully shows that the contents of the path functional, especially its order relations, are very profound, extensive and interesting, which means that for the path functional, there are still many problems needing to research.*

3. Problem

Definition 4. *The problem to find a path $P \in \mathcal{P}(v)$ such that $f(P) = m_f(v)$ for all $v \in V(\mathcal{P})$ is called as general single-source shortest path problem (GSSSP) on $[G, s, \mathcal{P}, f]$.*

It is clear that the problem GSSSP is just the problem CSSSP when G is the graph with weight w and f is the path functional d in the Example 1 of Section 6. It is also clear that the ARP problem, see Xiao et al. [10] (2009) or Example 3, is an instance of the problem GSSSP. The two facts show that the problem GSSSP is really generalization of the problem CSSSP.

Theorem 1. *Let $\mathcal{P}$ be the complete path system. If f is conservative and SOP, then the problem GSSSP can be solved. That is, $\forall v \in V(\mathcal{P})$, there is a path $P \in \mathcal{P}(v)$ such that $f(P) = m_f(v)$, namely P is a shortest path from s to v.*

Proof. From the term (ii) of Proposition 4, the Theorem 1 holds. □

4. Algorithms

For problem GSSSP, following the approaches of the algorithms DA and MBFA, an extended Dijkstra's algorithm (EDA) and an extended Moore-Bellman-Ford algorithm (EMBFA) can be respectively designed to solve it under certain conditions. We accomplish the tasks in this section (**Algorithms 1**–**3**).

Algorithm 1 Extended Dijkstra's Algorithm (EDA)

Input: graph $G = (V, E, R)$ and vertex $s \in V$, with a path functional f on the complete path system $\mathcal{P}$ of $[G, s]$, which is non-decreasing and SOP.
Output: a path system $\mathcal{T}$ on $[G, s]$ and the graph $T = (V(\mathcal{T}), E(\mathcal{T}), R(\mathcal{T}))$.
Process:
1. Put $P_T[s] = (s, s), f(P_T[s]) = f((s, s)) = 0$;
   $\forall v \in [V \setminus \{s\}]$, set $P_T[v] \leftarrow (s, \infty, v), f(P_T[v]) \leftarrow +\infty$.
   Set $C \leftarrow \emptyset, \mathcal{T} \leftarrow \emptyset$. (Set $k \leftarrow (-1)$.)
2.
   (1) Find a $u \in [V \setminus C]$ such that $f(P_T[u]) = \min\{f(P_T[v]) | v \in [V \setminus C]\}$.
   (2) Set $C \leftarrow [C \cup \{u\}], \mathcal{T} \leftarrow [\mathcal{T} \cup \{P_T[u]\}]$.
       $\forall v \in [V \setminus C]$, if $(u, v) \in \mathcal{P}$ and $f(P_T[v]) > f(P_T[u] + (u, v))$, set
       $P_T[v] \leftarrow (P_T[u] + (u, v))$.
       (Set $k \leftarrow (k + 1)$, then put $v(k) = u$.)
   (3) If $[V \setminus C] = \emptyset$ or $\min\{f(P_T[v]) | v \in [V \setminus C]\} = +\infty$, go to the step 3.
       Otherwise, return to step 2.
3. Output the path system $\mathcal{T}$ and the graph $T = (V(\mathcal{T}), E(\mathcal{T}), R(\mathcal{T}))$.
   (Output the vertices: $v(i), i = 0, 1, 2, \cdots, |V(\mathcal{T})| - 1$.) Then stop.

In order to simplify the analytical process of algorithm EDA, we also propose the next algorithm STA.

Algorithm 2 Spanning Tree Algorithm (STA)

Input: graph $G = (V, E, R)$, point $s \in V$.
Output: a graph T.
Process:
1. Set $C \leftarrow \{s\}, R(T) \leftarrow \{(s, s)\}$.
   (Set $k \leftarrow 0$, then put $v(k) = s, T_k = (C, R(T))$.)
   If $\{(u, v) | u \in C, v \in [V \setminus C], (u, v) \in R\} \neq \emptyset$, implement the next step. Otherwise, go to step 3.
2. Find a $u \in C$ and a $v \in [V \setminus C]$ such that $(u, v) \in R$. Then set

$$C \leftarrow [C \cup \{v\}], R(T) \leftarrow [R(T) \cup \{(u, v)\}].$$

   (Set $k \leftarrow (k + 1), v(k) = v, T_k = (C, R(T))$.)
   If $\{(u, v) | u \in C, v \in [V \setminus C], (u, v) \in R(\mathcal{P})\} \neq \emptyset$, return to step 2. Otherwise, implement the next step.
3. Put $V(T) = C$ and $T = (V(T), R(T))$. Output the graph T. Then stop.

Algorithm 3 Extended Moore-Bellman-Ford Algorithm (EMBFA)

Input: graph $G = (V, E, R)$ and vertex $s \in V$, with a path functional f on the complete path system $\mathcal{P}$ of $[G, s]$, which is conservative and OP.
Output: a path system $\mathcal{T}$ on $[G, s]$ and the graph $T = (V(\mathcal{T}), E(\mathcal{T}), R(\mathcal{T}))$.
Process:
1. Put $P_T[s] = (s, s), f(P_T[s]) = 0$;
   set $P_T[v] \leftarrow (s, \infty, v), f(P_T[v]) \leftarrow +\infty, \forall v \in [V \setminus \{s\}]$.
   Set $\mathcal{T} \leftarrow \{P_T[s]\}$.
2. For $i = 1, 2, \cdots, n$, do:
   for each road $(u, v) \in [R(\mathcal{P}) \setminus \{(s, s)\}]$, if $P_T[u] \in \mathcal{T}$ and

$$f(P_T[v]) > f(P_T[u] + (u, v)),$$

   then in turn set: $\mathcal{T}' \leftarrow [\mathcal{T} \setminus \{P_T[v]\}]$; $P_T[v] \leftarrow P_T[u] + (u, v)$;
   and $\mathcal{T} \leftarrow [\mathcal{T}' \cup \{P_T[v]\}]$.
   (Set $T = (V(\mathcal{T}), E(\mathcal{T}), R(\mathcal{T}))$.)
3. Output the path system $\mathcal{T}$ and the graph $T = (V(\mathcal{T}), E(\mathcal{T}), R(\mathcal{T}))$.
   Then stop.

Remark 2. *To be concise, please first understand the EDA and the related proof (see next section) under the condition that G is a simple graph. When G is not a simple graph, we should consider the parallel edges. For instance, when there exist two parallel roads $(u, 1, v), (u, 2, v) \in R(\mathcal{P})$ and u is $v(k)$, on the update of $P_T[v]$ in the $(k + 1)$ time iteration, $P_T[u] + (u, 1, v)$ and $P_T[u] + (u, 2, v)$ should be simultaneously involved in the comparison and the replacement. More specifically, we should understand and execute the term (2) of step 2 as follows.*

Set $C \leftarrow [C \cup \{u\}], \mathcal{T} \leftarrow [\mathcal{T} \cup \{P_T[u]\}]$.

$\forall v \in [V \setminus C]$, if $(u, i, v) \in \mathcal{P}, i = 1, 2, \cdots, |I(u, v)|, I(u, v) = \{$all the roads $(u, i, v)\}$, and $f(P_T[v]) > \min\{f(P_T[u] + (u, i, v)) | i = 1, 2, \cdots, |I(u, v)|\}$, find i' such that $f(P_T[u] + (u, i', v)) = \min\{f(P_T[u] + (u, i, v)) | i = 1, 2, \cdots, |I(u, v)|\}$, set

$$P_T[v] \leftarrow (P_T[u] + (u, i', v)).$$
$$(Set k \leftarrow (k+1), then put v(k) = u.)$$

Remark 3. *For EDA, $\forall v \in V(\mathcal{P})$, $P_T[v]$ will not change after it become a member of $\mathcal{T}$. However, for EMBFA, some $P_T[v]$ may change after it become a member of $\mathcal{T}$. The two facts are useful for us to understand the algorithms EDA and EMBFA.*

5. Analysis of Algorithms

Lemma 2. *For algorithm STA, we have the following conclusions.*

(i) The algorithm works well. (That is, it can effectively input, output and stop.) (ii) The output T of STA is a spanning tree of $\bar{G}$. (iii) T is an arborescence rooted at s. (iv) The running time of STA is no more than $O(mn)$. Here $\bar{G} = (V(\mathcal{P}), E(\mathcal{P}), R(\mathcal{P}))$, $\mathcal{P}$ is the complete path system on $[G, s]$ and $m = |E(V)|$.

Proof. Obviously, we always have that $1 \leq |C| \leq n = |V|$ and $|C|$ is increasing. This implies that the step 2 is performed no more than n times. Hence (i) holds.

In terms of the process of Algorithm STA and (i), it is obvious that T is connected subgraph of $\bar{G}$. Note that $\{(u,v)|u \in C, v \in [V \setminus C], (u,v) \in R\} \neq \emptyset$ so long as $|C| < |V(\bar{G})|$. We can also know that $|V(T)| = |V(\bar{G})|$. Therefore, in order to show (ii) is true, we need only to prove that T has no cycle. Clearly, T_0 has no cycle. Suppose that T_k has no cycle. Then, for $v(k+1) \neq v(0), v(1), \cdots, v(k)$, $T_{(k+1)}$ also has no cycle. This implies that T has no cycle. Hence (ii) holds.

By the process of STA, we can easily know that for any edge $[u, v]$ of T, there exists only one road ((u, v) or (v, u)), that is, T is a directed graph. Moreover, we can easily know also that $|\delta^-(s)| = 0$ and $|\delta^-(v)| = 1, \forall v \in [V(T) \setminus \{s\}]$. Here $|\delta^-(v)|$ is the in-degree of v in the directed graph T. Therefore T is an arborescence rooted at s. Hence (iii) holds.

Finally, it is obvious that the running time of STA depends on the complexity of the step 2. However, the number of total iterations of the step 2 do not exceed n and the complexity of each run of the step 2 does not exceed $O(m)$. Hence (iv) holds. □

Remark 4. *For the necessary basic concepts of graphs, such as spanning tree, arborescence, induced subgraph, the degree of a vertex, the in-degree of a vertex, so on, please see 2.1 and 2.2 of the monograph [1] or other related books.*

Remark 5. *For algorithm STA, by properly designing the method to find the road (v, u) in step 2, the running time can be greatly simplified. Please see the Graph Scanning Algorithm in 2.3 of the monograph [1]. However, the current STA can more effectively implement its functionality in the present work, which helps to prove the following Theorem 2, so we propose it in the current pattern.*

Theorem 2. *For algorithm EDA, we have the following conclusions.*

(i) The algorithm works well. (ii) The output T is a spanning tree of graph $\bar{G}$. (iii) T is an arborescence rooted at s. (iv) $\forall v \in V(\mathcal{P})$, the path from s to v on the tree T is $P_T[v]$. (v) $\forall v \in V(\mathcal{P}), f(P_T[v]) = m_f(v)$. (vi) The running time is $M(n)O(n^2)$, provided $\Delta(G) = O(n)$ and $f(P)$ can be obtained in $M(n)$ time of calculation for any $P \in \mathcal{P}$.

Here, $\bar{G} = (V(\mathcal{P}), E(\mathcal{P}), R(\mathcal{P}))$, which is the subgraph of G induced by $V(\mathcal{P})$; $n = |V|$; $\Delta(G) = \max\{|\delta(v)||v \in V\}$, $|\delta(v)|$ is the degree of vertex v, which is called as the maximum degree of graph G; $M(n)$ is a constant related to n.

Proof. Note that V and R are all the finite sets. By observing the process of algorithm EDA, we can easily know that (i) holds. Note that $[V \setminus C] = \emptyset$ or $\min\{f(P_T[v])|v \in [V \setminus C]\} = +\infty$ is equivalent to $\{(u,v)|u \in C, v \in [V \setminus C], (u,v) \in R(\mathcal{P})\} = \emptyset$. By carefully examining the second step of EDA and the second step of STA, it can be easily known that the second step of EDA is a specific implementation of the second step of STA. Hence (ii) and (iii) can be immediately obtained from Lemma 2. Note $P_T[v] \in \mathcal{T}$. According to the relation between T and $\mathcal{T}$, $\mathcal{T}$ is actually the complete path system on $[T, s]$. Hence (iv) holds. As $\Delta(G) = O(n)$ and $f(P)$ can be obtained in $M(n)$ times of calculation for any $P \in \mathcal{P}$, the running time of the step 2 of EDA is $M(n)O(n)$. Note that the total iterations of step 2 is no more than n time. We can further know the running time of EDA is $M(n)O(n^2)$. That is, (vi) holds. Next we focus to prove (v).

Note that the outputted vertices: $v(l), l = 0, 1, \cdots, |V(\mathcal{P})|-1$, actually are all the elements of $V(\mathcal{P})$. The term (v) can be interpreted as: $f(P_T[v(l)]) = m_f(v(l)), l = 0, 1, \cdots, |V(\mathcal{P})| - 1$. We use mathematical induction to complete the proof.

In the first place, it is obvious that (v) holds when $l = 0$, namely $f(P_T[v(0)]) = m_f(v(0))$. In fact, $v(0) = s$. Hence $P_T[v(0)] = (s, s)$ and $f(P_T[v(0)]) = f((s, s)) = 0 = m_f(v(0))$. Assume $|\mathcal{P}| \geq 2$. Then, it is also obvious that (v) is true when $l = 1$, namely $f(P_T[v(1)]) = m_f(v(1))$.

Suppose (v) is true for $0 \leq l \leq k < |V(\mathcal{P})| - 1$ with $k \geq 1$. Then we can prove that it is also true for $0 \leq l \leq (k + 1)$.

In fact, we only need to prove that (v) holds for $(k + 1)$.

Assume that (v) does not holds for $(k + 1)$, namely, $m_f(v(k + 1)) < f(P_T[v(k + 1)])$. We can show that the assumption is not true as follows.

Let $v(k + 1) = v^*$ and $P_T[v(k + 1)] = (s, \cdots, \tilde{v}, v^*)$. Then $m_f(v^*) < f(P_T[v^*])$, and from term (iv), we have $P_T[\tilde{v}] = (s, \cdots, \tilde{v})$.

In terms of the assumption and Corollary 1, there is a path

$$P = (v_0, v_1, \cdots, v_i, v^*) \in \mathcal{P}_{nc}(v^*) \tag{1}$$

such that $f(P) = m_f(v^*) < f(P_T[v^*])$ and $(v_0, v_1, \cdots, v_i) \neq P_T[\tilde{v}]$. For the vertex v_i, there are only three situations: (1) $v_i \in \{v(l)|l = 0, 1, \cdots, k\}$, and $v_i = \tilde{v}$; (2) $v_i \in \{v(l)|l = 0, 1, \cdots, k\}$, but $v_i \neq \tilde{v}$; (3) $v_i \notin \{v(l)|l = 0, 1, \cdots, k\}$.

In the situation (1), $f(P_T[v_i]) = m_f(v_i)$ for $v_i \in \{v(l)|l = 0, 1, \cdots, k\}$, and then $f((v_0, v_1, \cdots, v_i)) \geq m_f(v_i) = f(P_T[v_i]) = f(P_T[\tilde{v}])$. Because f is SOP, we have $m_f(v^*) = f((v_0, v_1, \cdots, v_i, v^*)) = f((v_0, v_1, \cdots, v_i) + (v_i, v^*)) \geq f(P_T[\tilde{v}] + (v_i, v^*)) = f(P_T[v^*])$. This is in contradiction with $m_f(v^*) < f(P_T[v^*])$.

In the situation (2), $f(P_T[v_i]) = m_f(v_i) \leq f((v_0, v_1, \cdots, v_i))$. Because f is SOP,

$$\begin{aligned} & f(P_T[v_i] + (v_i, v^*)) \\ \leq \quad & f((v_0, v_1, \cdots, v_i) + (v_i, v^*)) = f((v_0, v_1, \cdots, v_i, v^*)) < f(P_T[v^*]). \end{aligned} \tag{2}$$

For $v_i \neq \tilde{v}$, we have

$$P_T[v_i] + (v_i, v^*) \neq P_T[\tilde{v}] + (v_i, v^*) = (s, \cdots, \tilde{v}, v^*). \tag{3}$$

Since $v_i \in \{v(l)|l = 0, 1, \cdots, k\}$ and $v^* \notin \{v(l)|l = 0, 1, \cdots, k\}$, on the basis of the process of EDA, from Equations (2) and (3), we have $P_T[v(k + 1)] \neq (s, \cdots, \tilde{v}, v^*)$. This is in contradiction with $P_T[v(k + 1)] = (s, \cdots, \tilde{v}, v^*)$.

In the situation (3), for $v_0 = s$, we can find a vertex

$$v_{i'} \in [\{v_0, v_1, \cdots, v_{i-1}\} \cap \{v(l)|l = 0, 1, \cdots, k\}]$$

such that $v_{i'+1}, v_{i'+2}, \cdots, v_i \notin \{v(l)|l = 0, 1, \cdots, k\}$. Following the approach of situation (2), we can obtain

$$\begin{aligned} & f(P_T[v_{i'}] + (v_{i'}, v_{i'+1})) \\ \leq \quad & f((v_0, v_1, \cdots, v_{i'}) + (v_{i'}, v_{i'+1})) = f(v_0, v_1, \cdots, v_{i'}, v_{i'+1}). \end{aligned} \tag{4}$$

On the other hand, for f is non-decreasing, we have

$$f(v_0, v_1, \cdots, v_{i'}, v_{i'+1}) \leq f(P) < f(P_T[v^*]). \tag{5}$$

From Equations (4) and (5), we can obtain $f(P_T[v_{i'}] + (v_{i'}, v_{i'+1})) < f(P_T[v^*])$. By Equation (1), $v_{i'+1} \neq v^*$. Noting $v_{i'} \in \{v(l)|l = 0, 1, \cdots, k\}$, on the basis of the process of EDA, we can derive $v(k + 1) \neq v^*$. This is in contradiction with $v(k + 1) = v^*$.

Combined with the above results of the three situations, we know that the assumption $m_f(v(k+1)) < f(P_T[v(k+1)])$ is incorrect. That is, $f(P_T[v(k+1)]) = m_f(v(k+1))$. Finally, by the induction principle, $f(P_T[v(l)]) = m_f(v(l))$ for $l = 0, 1, \cdots, |V(\mathcal{P})| - 1$. Hence (v) holds. □

Theorem 3. *For algorithm EMBFA, we have the following conclusions.*

(i) The algorithm EMBFA works well. (ii) The output T is a spanning tree of graph $\bar{G}$. (iii) T is an arborescence rooted at s. (iv) $\forall v \in V(\mathcal{P})$, the path from s to v on the tree T is $P_T[v]$, and $f(P_T[v]) = m_f(v)$. (v) The running time is $M(n)O(nm)$, provided $f(P)$ can be obtained in the $M(n)$ time of calculation for any $P \in \mathcal{P}$.

Here, $\bar{G} = (V(\mathcal{P}), E(\mathcal{P}), R(\mathcal{P}))$, which is the subgraph of G induced by $V(\mathcal{P})$; $n = |V|, m = |E|$; $M(n)$ is a constant related to n.

Proof. Note that V and R are all the finite sets. We can easily know (i) holds. Note that $f(P)$ can be obtained in the $M(n)$ time of calculation. Following the approach to analyse the running time of algorithm MBFA, we can easily know that (v) holds by the process of algorithm EMBFA. Next we focus to prove (ii), (iii) and (iv).

For they evidently hold when $|V(\mathcal{P})| \leq 2$, we prove them under the condition $|V(\mathcal{P})| > 2$.

Above all, we claim:

(a) $\forall v \in V(\mathcal{P}), f(P_T[v]) = m_f(v)$; further, once $f(P_T[v])$ attains $m_f(v)$, $P_T[v]$ will remain unchanged in the after process; and $P_T[v]$ is unique.

(b) $\forall v \in V(\mathcal{P})$, let $P_T[v] = (v_0, v_1, \cdots, v_k, v), 0 \leq k$, then

$$P_T[v_i] = (v_0, v_1, \cdots, v_i), i = 0, 1, 2, \cdots, k;$$

further, $P_T[v] \in \mathcal{P}_{nc}(v)$; and $k \leq |V(\mathcal{P})| - 2$.

(c) T has no cycle; further, T is directed graph; and

$$|\delta^-(s)| = 0, \forall v \in [V(\mathcal{P}) \setminus \{s\}], |\delta^-(v)| = 1.$$

Prove (a). When $v = s$, (a) clearly holds. For $\mathcal{P}$ is the complete path system on $[G, s]$, $\forall v \in [V(\mathcal{P}) \setminus \{s\}]$, from Corollary 2, there is a path $P = (v_0, v_1, \cdots, v_k, v) \in \mathcal{P}_{nc}(v), 0 \leq k \leq n-2$, such that $f(P) = m_f(v)$. From the process of EMBFA, this implies that $P_T[v]$ must become a path and $f(P_T[v])$ must attain $m_f(v)$ within $(k+2)$ iterations of step 2. Since also $f(P_T[v])$ never increases in the process of EMBFA, $f(P_T[v]) = m_f(v)$ in the end. Thus, the first statement is true. The correctness of the latter two statements is obvious. Hence (a) holds.

Prove (b). $\forall v \in V(\mathcal{P})$, let $P_T[v] = (v_0, v_1, \cdots, v_k, v)$. Assume that $P_T[v_i] = (v_0, v_1, \cdots, v_l, v_{l+1}, \cdots, v_i), l < i \leq k$ and $P_T[v_l] = (v_0, v'_1, \cdots, v'_{l'}, v_l) \neq (v_0, v_1, \cdots, v_l)$. Then it is implemented first that $P_T[v_l] = (v_0, v'_1, \cdots, v'_{l'}, v_l)$ and $f(P_T[v_l]) = m_f(v_l)$. By Corollary 2, f is ISP, so $f((v_0, v_1, \cdots, v_l)) = m_f(v_l) = f(P_T[v_l])$. This implies that $(v_0, v_1, \cdots, v_l, v_{l+1}, \cdots, v_k, v)$ can no longer be the path of T (element of $\mathcal{T}$) at any stage of EMBFA, which leads to contradictions. Thus the first statement is true. $\forall v \in V(\mathcal{P})$, assume that $P_T[v]$ has a cycle. Then we have the following representation,

$$P_T[v] = (v_0, v_1, \cdots, v_l, \bar{v}, v'_1, \cdots, v'_{l'}, \bar{v}, \cdots, v), 1 \leq l'.$$

In terms of the first statement, we have $P_T[\bar{v}] = (v_0, v_1, \cdots, v_l, \bar{v})$ and $P_T[\bar{v}] = (v_0, v_1, \cdots, v_l, \bar{v}, v'_1, \cdots, v'_{l'}, \bar{v})$, which means $P_T[v]$ is not unique, and contradicts the claim (a). So, $P_T[v]$ has no cycle, and the third statement is also true. Hence (b) holds.

Prove (c). According to the relation between T and $\mathcal{T}$, one of the following facts (1) and (2) holds if T has cycles. (1) There is a path of $\mathcal{T}$ such that it has cycles. (2) There are two paths $P_T[v], P_T[v'] \in \mathcal{T}$ such that

$$P_T[v] = (v_0, v_1, \cdots, v_l, v_{(l+1)}, \cdots, v_{(l+k)}, \cdots, v), 2 \leq k;$$

$$P_T[v'] = (v_0, v_1, \cdots, v_l, v'_1, \cdots, v'_{k'}, \cdots, v'), 1 \leq k';$$

$$v_{(l+k)} = v'_{k'} = \bar{v}, \{v'_1, \cdots, v'_{k'-1}\} \cap \{v_{(l+1)}, \cdots, v_{(l+k-1)}\} = \emptyset.$$

That is, the two paths intersect again after their separating. The fact (1) contradicts the second statement of claim (b). By the first statement of claim (b), the fact (2) results in $P_T[\bar{v}] = (v_0, v_1, \cdots, v_l, v_{(l+1)}, \cdots, v_{(l+k)})$ and $P_T[\bar{v}] = (v_0, v_1, \cdots, v_l, v'_1, \cdots, v'_{k'})$, which means that $P_T[\bar{v}]$ is not unique for

$$(v_0, v_1, \cdots, v_l, v_{(l+1)}, \cdots, v_{(l+k)}) \neq (v_0, v_1, \cdots, v_l, v'_1, \cdots, v'_{k'}),$$

and contradicts the claim (a). Thus T has no cycle. Assume T is not directed graph. Then there must be $u, v \in V(T)$ such that $(u,v),(v,u) \in R(T)$. From the approach that graph T is created by path system $\mathcal{T}$, there must be $u', v' \in V(\mathcal{T})$ such that $(u,v) \in P_T[u'], (v,u) \in P_T[v']$, namely, $P_T[u'] = (s,\cdots,u,v,\cdots,u'), P_T[v'] = (s,\cdots,v,u,\cdots,v')$. In terms of $P_T[u'] = (s,\cdots,u,v,\cdots,u')$ and (b), $P_T[u] = (s,\cdots,u)$ and $v \notin V(P_T[u])$ for $P_T[u']$ has no cycle. In terms of $P_T[v'] = (s,\cdots,v,u,\cdots,v')$ and (b), $P_T[u] = (s,\cdots,v,u)$ and $v \in V(P_T[u])$. For $v \notin V(P_T[u])$ contradicts $v \in V(P_T[u])$, the assumption is not true. That is, T is directed graph. For, in the beginning, $P_T[s] = (s,s), f(P_T[s]) = m_f(s)$, which remains unchanged in the after process, it is obvious that $|\delta^-(s)| = 0$. $\forall v \in [V(\mathcal{P}) \setminus \{s\}]$, we have clearly $|\delta^-(v)| \geq 1$. On the other hand, for all the paths of T have the same source point s and T has no cycle, we have $|\delta^-(v)| \leq 1$. So, $|\delta^-(v)| = 1$. To sum up, (c) holds.

Finally, we show that (ii), (iii) and (iv) hold basing on the above claims.

In terms of (a), $V(\mathcal{P}) \subseteq V(\mathcal{T})$. On the other hand, $V(\mathcal{T}) \subseteq V(\mathcal{P})$ is obvious. So, $V(\mathcal{P}) = V(\mathcal{T})$. Also, in terms of (a), T is connected. And, in terms of (c), T has no cycles. Hence, (ii) holds.

In terms of (ii) and (c), (iii) holds.

Note that the path system $\mathcal{T}$ is the complete path system on $[T,s]$. In terms of (iii) and (a), we can easily know that (iv) holds. □

6. Applications

This section shows the application of algorithm EDA and algorithm EMBFA by providing few instances.

Example 1. *Given a connected network with nonnegative weight (G,w) and source point $s \in V$. Let $\mathcal{P}$ be the complete path system on $[G,s]$. Define $d(P) = \sum_{i=1}^{k} w((v_{i-1},v_i)), \forall P = (v_0,v_1,\cdots,v_{k-1},v_k) \in \mathcal{P}, v_0 = s$, which is called the total weight path function of networks. It is obvious that d is a path functional on $\mathcal{P}$. The problem of finding a path $P^* \in \mathcal{P}(v)$ such that $d(P^*) = m_d(v), \forall v \in [V \setminus \{s\}]$, is called as the classical single-source shortest path problem with nonnegative weight (CSSSP-NW). See section 7.1 of the monograph [1]. Note that w is nonnegative. Then d is nondecreasing and OP. So, by Theorems 2 and 3, the problem CSSSP-NW can be effectively solved by the algorithm EDA and algorithm EMBFA, respectively.*

Remark 6. *From Example 1, we can know the following facts clearly. (i) EDA and EMBFA respectively reduces to DA and MBFA in the situation of the example. (ii) Let $P, P' \in \mathcal{P}$ and $S(P) = P + (u,v), S(P') = P' + (u,v) \in \mathcal{P}$. Then $d(S(P)) - d(P) = d(S(P')) - d(P') = w((u,v))$ for the function d. However, $f(S(P)) - f(P) = f(S(P')) - f(P')$ may not hold for a general path function f. That is, we may not find an edge weight of graph G such that $f(P) = \sum_{i=1}^{k} w((v_{i-1},v_i)), \forall P = (v_0,v_1,\cdots,v_{k-1},v_k) \in \mathcal{P}$ for a general path functional f. The two facts (i) and (ii) fully illustrate that EDA and EMBFA respectively extended DA and MBFA.*

Example 2. *For Example 1, change the nonnegative weight w as a conservative weight, e.g., see Definition 7.1 of the monograph [1]. Then the problem to find a path $P^* \in \mathcal{P}(v)$ such that $d(P^*) = m_d(v), \forall v \in [V \setminus \{s\}]$, is called as the classical single-source shortest path problem with conservative weight (CSSSP-CW). Clearly, d is conservative and OP. So, by Theorem 3, the problem CSSSP-CW can be effectively solved by Algorithm EMBFA.*

Example 3. *For Example 1, define also $d(u,v) = \min\{d(P) | s(P) = u, t(P) = v, P \in \mathcal{P}_{nc}\}, \forall u,v \in V$, which is called as the distance from u to v on graph G. For given $(u',v') \in R(\mathcal{P}_{nc})$, define $d_{G\setminus(u',v')}(u,v)$ as the distance from u to v on graph $(V,[R \setminus \{(u',v')\}]), \forall u,v \in V$, which is called as the detour distance from u to v on the case that the road (u',v') is blocked. Here $G\setminus(u',v')$ denotes the graph $(V,[R \setminus \{(u',v')\}])$. In addition, $\forall (u',v') \in R(\mathcal{P}_{nc})$, assume that $G\setminus(u',v')$ is connected, namely $d_{G\setminus(u',v')}(u,v) < +\infty$ for each $(u,v) \in [R \setminus \{(u',v')\}]$.*

Define

$$\begin{aligned}
r(P) &= 0, P = (s,s);\\
r(P) &= \max\{d(P), d_{G\setminus(v_{i-1},v_i)}(s,v_i) + d(P_i), d_{G\setminus(v_{k-1},v_k)}(s,v_k) |\\
&\qquad P_i = (v_i,\cdots,v_{k-1},v_k), 1 \leq i \leq k-1\},\\
&\qquad \forall P = (v_0,v_1,\cdots,v_{k-1},v_k) \in \mathcal{P}_{nc}, k \geq 1;\\
r(P) &= \max\{d(P) | P \in \mathcal{P}_{nc}\} + 1, \forall P \in [\mathcal{P} \setminus \mathcal{P}_{nc}].
\end{aligned}$$

We call $r(P)$ *as the risk of* P. *It is obvious that* r *is a path functional on* $\mathcal{P}$. *The problem to find a path* $P \in \mathcal{P}(v)$ *such that* $r(P) = m_r(v)$ *for any* $v \in [V(\mathcal{P}) \setminus \{s\}]$ *is called the anti-risk path (ARP) problem, the purpose of which is to find a path such that it has minimum risk [10].*

Let $P = (v_0, v_1, \cdots, v_{k-1}, v_k) \in \mathcal{P}_{nc}$ *and* $S(P) = (v_0, v_1, \cdots, v_{k-1}, v_k, v_{k+1}) = P + (v_k, v_{k+1}) \in \mathcal{P}_{nc}, k \geq 1$. *Then we have*

$$\begin{aligned} & r(S(P)) \\ = \;& \max\{d(S(P)), d_{G\backslash(v_{i-1},v_i)}(s, v_i) + d(P_i), d_{G\backslash(v_k,v_{k+1})}(s, v_{k+1}) | \\ & \quad P_i = (v_i, \cdots, v_k, v_{k+1}), 1 \leq i \leq k\} \\ = \;& \max\{d(P) + w((v_k, v_{k+1})), d_{G\backslash(v_{i-1},v_i)}(s, v_i) + d(P_i) \\ & \quad + w((v_k, v_{k+1})), d_{G\backslash(v_{k-1},v_k)}(s, v_k) + w((v_k, v_{k+1})), \\ & \quad d_{G\backslash(v_k,v_{k+1})}(s, v_{k+1}) | P_i = (v_i, \cdots, v_{k-1}, v_k), 1 \leq i \leq k-1\} \\ = \;& \max\{d_{G\backslash(v_k,v_{k+1})}(s, v_{k+1}), w((v_k, v_{k+1})) + \max\{d(P), \\ & \quad d(P_i) + d_{G\backslash(v_{i-1},v_i)}(s, v_i), d_{G\backslash(v_{k-1},v_k)}(s, v_k) \\ & \quad | P_i = (v_i, \cdots, v_{k-1}, v_k), 1 \leq i \leq k-1\}\} \\ = \;& \max\{d_{G\backslash(v_k,v_{k+1})}(s, v_{k+1}), w((v_k, v_{k+1})) + r(P)\} \geq r(P). \end{aligned} \tag{1}$$

This shows r *is nondecreasing on* $\mathcal{P}_{nc}$.

Let also $P' \in \mathcal{P}_{nc}(v_k), S(P') = P' + (v_k, v_{k+1}) \in \mathcal{P}_{nc}(v_{k+1})$. *Assume* $r(P) \geq r(P')$. *Then, from Equation (6), we have*

$$\begin{aligned} r(S(P)) &= \max\{d_{G\backslash(v_k,v_{k+1})}(s, v_{k+1}), w((v_k, v_{k+1})) + r(P)\} \\ &\geq \max\{d_{G\backslash(v_k,v_{k+1})}(s, v_{k+1}), w((v_k, v_{k+1})) + r(P')\} = r(S(P')). \end{aligned}$$

This shows r *is SOP on* $\mathcal{P}_{nc}$.

Further, basing on the above two conclusions, r *is nondecreasing and SOP on* $\mathcal{P}$. *So, by Theorem 2, the ARP problem can be effectively solved by algorithm EDA.*

Remark 7. *(i) Xiao et al. [10] introduce the definition of the risk of a path, and the anti-risk path (ARP) problem, to finding a path such that it has minimum risk. Suppose that at most one edge may be blocked, they also show that the ARP problem can be solved in* $O(mn + n^2 logn)$ *time. Mahadeokar and Saxena [11] propose a faster algorithm to solve the ARP problem, by which the ARP problem can be solved in* $O(n^2)$ *time. (ii) In example 3, for some technical reason, the risk is defined in a slightly different manner from that of Xiao et al. [10]. Due to the cause of symmetry, in order to conveniently understand the Example 3 and the next Example 4, the path* $P = (s, v_1, \cdots, v_{k-1}, v_k, v)$ *can be interpreted as the path from* v *to* s. *The ARP problem of Example 3, which is essentially similar to that of Xiao et al. [10], can be effectively solved by algorithm EDA. (iii) For* r *is not proved to be OR, the ARP problem of example 3 may not necessarily be solved by EMBFA. However, due that* r *is SOP, it can be solved by EDA. This fact fully demonstrates the advantages of SOP and EDA.*

Example 4. *For Example 1, assume*

$$d_{G\backslash(u',v')}(u, v) < +\infty, \forall (u', v') \in R(\mathcal{P}_{nc}), \forall u, v \in V,$$

where $d_{G\backslash(u',v')}(u, v)$ *is the detour distance in Example 3; assume also* $p \in (0, 1)$.

Define first $c((s, s)) = 0$. *Then,* $\forall P = (v_0, v_1, \cdots, v_k) \in \mathcal{P}_{nc}, k \geq 1$, *provided that* $c(F(P))$ *has been defined, define* $c(P) = pd_{G\backslash(v_{k-1},v_k)}(v_0, v_k) + w((v_{k-1}, v_k)) + c(F(P))$; $\forall P \in [\mathcal{P} \setminus \mathcal{P}_{nc}]$, *define* $c(P) = \max\{d(P) | P \in \mathcal{P}_{nc}\} + 1$. *Then* c *is a path functional on* $\mathcal{P}$.

Let $P = (v_0, v_1, \cdots, v_{k-1}, v_k) \in \mathcal{P}_{nc}$, *and* $S(P) = (v_0, \cdots, v_k, v_{k+1}) = P + (v_k, v_{k+1}) \in \mathcal{P}_{nc}$. *Then we have*

$$\begin{aligned} c(S(P)) &= pd_{G\backslash(v_k,v_{k+1})}(v_k, v_{k+1}) + w((v_k, v_{k+1})) + c(P) \\ &\geq c(P). \end{aligned}$$

This shows c *is nondecreasing on* $\mathcal{P}_{nc}$.

Let also $P' \in \mathcal{P}_{nc}(v_k), S(P') = P' + (v_k, v_{k+1}) \in \mathcal{P}_{nc}(v_{k+1})$. *Assume* $c(P) \geq c(P')$. *Then, we have*

$$\begin{aligned} c(S(P)) &= pd_{G\backslash(v_k,v_{k+1})}(v_k, v_{k+1}) + w((v_k, v_{k+1})) + c(P) \\ &\geq pd_{G\backslash(v_k,v_{k+1})}(v_k, v_{k+1}) + w((v_k, v_{k+1})) + c(P') \\ &= c(S(P')). \end{aligned}$$

This shows c is SOP on $\mathcal{P}_{nc}$.

Further, we can easily show that c is nondecreasing and SOP. So, by Theorem 2, the problem GSSSP with path functional c can be effectively solved by algorithm EDA.

Remark 8. *(i) The path functional c can be interpret as follows. Suppose that at most one edge may be blocked.* $\forall P \in \mathcal{P}_{nc}$, $c(P) = pd_{G\backslash(v_{k-1},v_k)}(v_{k-1}, v_k) + w((v_{k-1}, v_k)) + c(F(P))$ *denotes the cost that one goes to the point s from the point* v_k *by train (or ship, or plane), among which, the term* $w((v_{k-1}, v_k)) + c(F(P))$ *is the normal cost, while the term* $pd_{G\backslash(v_{k-1},v_k)}(v_{k-1}, v_k)$ *is the additional cost, which is paid out due that one needs to change route when the road* (v_{k-1}, v_k) *is blocked. To some extent, p is the probability that the road* (v_{k-1}, v_k) *may be blocked. (ii) It is shown by Examples 3 and 4 that path functionals is very useful in practice. Especially, because the path functional is rich in content, it may find applications in various fields in the future, for example, in Alzheimer's Disease, see Razavi et al. [32]; in resource allocation, see Saghezchi et al. [33]; in uncertain environments; in dynamic networks; so on.*

Remark 9. *The Examples 3 and 4 fully demonstrate potential applications of either EDA or EMBFA. With the development of science and technology, new networks continue to appear, such as the internet networks, social networks, biological networks, so the problems that need EDA and EMBFA to solve will be more and more. That is, the two algorithms have broad application prospects. On the other hand, the two algorithms have significant limitations. For example, EDA requires the path functional to satisfy non-decreasing and SOP. It is a promising direction to develope algorithms with weaker requirements for solving GSSSP.*

Remark 10. *Finally, we hope the following points will draw the attention of scholars. (i) The present work expands and deepens our understanding of the network and spatial structure, which may be valuable to recent studies, such as the research of Alzheimer's Disease [32], and the research of resource allocation [33]. (ii) The present work promotes and enhances the level of digitization with the relevant content, which may be valuable to the future development of artificial intelligences. (iii) The ideas that are mined from the process of the present work could improve and advance the development of artificial intelligence for the Human Mind is the foundation of artificial intelligence.*

7. Concluding Remarks

Due to the quick and extensive development in the research field of graph theories, and due to the quick and extensive development in the research fields of networks and functionals, especially with the emergence of the Anti-risk Path Problem and the related research, the need to extend the total weight path function of networks and the shortest path problems is gradually increasing. Motivated by this trend, in the present article, we have mainly done the following three aspects of work.

(1) We proposed the definition of the path functional and several definitions regarding the characteristics of the defined path functional, especially the characteristic in orders, such as increasing, order-preserving, so on; see Definitions 1–3. Further, based on the proposed definitions, we made some related discussions on the properties of the path functional, see Propositions 1–6.

(2) We introduced a kind of general single-source shortest path problem (GSSSP) and designed two algorithms EDA and EMBFA to solve it under certain conditions. Further, based on the discussions on the properties of the path functional, we studied respectively the attributes of EDA and EMBFA; see Theorems 2 and 3.

(3) We further explained the significance of the defined path functional and the two designed algorithms by several examples.

What we have done not only extends the Dijkstra algorithm and the Moore-Bellman-Ford algorithm, but also more profoundly reveals their mechanism, which will greatly promote our understanding and applying of the two algorithms. The discussions on the properties of the path functional, not only support our designing and studying the two extended algorithms, but also more profoundly reveals that the contents of path functionals, in particular the content about the order, are quite rich, depth and interest, which further shows that there are many other research problems about path functionals. What we have done can also promote the development of the researches and the applications of other combinatorial optimization problems, promote the development of the algorithm theory and promote the development of the functional analysis.

In addition, what we have done can effectively promote the development of intelligent technology for algorithms are the important foundation of technologies of AI, big Data, machine Learning, etc.

It is an interesting topic for further research in the future to explore other efficient algorithms and the applications of the problem GSSSP. Moreover, to study other questions about path functionals is also an interesting topic for further research in the future. Finally, cordially hope that the present work can improve the development of the researches and applications of shortest path problems as well as other problems of combinatorial optimization.

Funding

This work received no external funding.

Institutional Review Board Statement

Not applicable.

Informed Consent Statement

Not applicable.

Data Availability Statement

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Acknowledgments

The author cordially thanks the editor and the anonymous referees for their valuable comments and suggestions which lead to the improvement of this paper.

Conficts of Interest

The author declares no conflict of interest.

References

1. Korte, B.; Vygen, J. *Combinatorial Optimization: Theory and Algorithms*; Springer-Verlag: Berlin, Germany, 2000.
2. Bellman, R.E. On a Routing Problem. *Q. Appl. Math.* **1958**, *16*, 87–90.
3. Dijkstra, E.W. A Note on Two Problems in Connexion with Graphs. *Numer. Math.* **1959**, *1*, 269–271.
4. Ford Jr., L.R. *Network Flow Theory*; The Rand Corporation: Santa Monica, CA, USA, 1956.
5. Moore, E.F. The Shortest Path through a Maze. In *Proceedings of the International Symposium on the Theory of Switching, Part II*; Harvard University Press: Cambridge, MA, USA, 1959; pp. 285–292.
6. Hershberger, J.; Suri, S. Vickrey Prices and Shortest Paths: What Is an Edge Worth? In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*; Computer Society: Las Vegas, NV, USA, 2001; pp. 252–259.
7. Hershberger, J.; Suri, S.; Bhosle, A. On the Difficulty of Some Shortest Path Problems. In *Lecture Notes in Computer Science*; Springer: Berlin, Germany, 2003; pp. 343–354.
8. Cho, H.J.; Lan, C.L.; Simos, T.E.; et al. A Hybrid Shortest Path Algorithm for Navigation System. *AIP Conf. Proc.* **2007**, *963*, 1073–1076. [CrossRef]
9. Du, D.Z.; Graham, R.L.; Pardalos, P.M.; et al. Analysis of Greedy Approximations with Nonsubmodular Potential Functions. In Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, CA, USA, 20–22 January 2008; pp. 167–175.
10. Xiao, P.; Xu, Y.; Su, B. Finding an Anti-Risk Path between Two Nodes in Undirected Graphs. *J. Comb. Optim.* **2009**, *17*, 235–246.

11. Mahadeokar, J.; Saxena, S. Faster Algorithm to Find Anti-Risk Path between Two Nodes of an Undirected Graph. *J. Comb. Optim.* **2014**, *27*, 798–807. [CrossRef]
12. Srivastava, K.; Tyagi, R. Shortest Path Algorithm for Satellite Network. *Int. J. Innov. Res. Dev.* **2013**, *5*, 438–445.
13. Murota, K.; Shioura, A. Dijkstra's Algorithm and L-Concave Function Maximization. *Math. Program. Ser. A* **2014**, *145*, 163–177. [CrossRef]
14. Feng, G. Finding k Shortest Simple Paths in Directed Graphs: A Node Classification Algorithm. *Networks* **2014**, *63*, 6–17. [CrossRef]
15. Zhang, H.L.; Xu, Y.F.; Wen, X.G. Optimal Shortest Path Set Problem in Undirected Graphs. *J. Comb. Optim.* **2015**, *29*, 511–530. [CrossRef]
16. Nip, K.; Wang, Z.; Nobibon, F.T.; et al. A Combination of Flow Shop Scheduling and the Shortest Path Problem. *J. Comb. Optim.* **2015**, *29*, 36–52. [CrossRef]
17. Meng, S.C.Y.; Adnan, N.; Sukri, S.S.; et al. An Experiment on the Performance of Shortest Path Algorithm. In Proceedings of the Knowledge Management International Conference, Chiang Mai, Thailand, 29–30 August 2016; pp. 7–12.
18. Dinitz, Y.; Itzhak, R. Hybrid Bellman–Ford–Dijkstra Algorithm. *J. Discrete Algorithms* **2017**, *42*, 35–44. [CrossRef]
19. AbuSalim, S.W.G.; Ibrahim, R.; Saringat, M.Z.; et al. Comparative Analysis between Dijkstra and Bellman–Ford Algorithms in Shortest Path Optimization. In Proceedings of the IOP Conference Series: Materials Science and Engineering, Penang, Malaysia, 17–18 April 2020; pp. 1–11. [CrossRef]
20. Xia, D.W.; Shen, B.Q.; Zheng, Y.L.; et al. A Bidirectional A-Star-Based Ant Colony Optimization Algorithm for Big-Data-Driven Taxi Route Recommendation. *Multimed. Tools Appl.* **2024**, *83*, 16313–16335. [CrossRef]
21. Gajjar, K.; Jha, A.V.; Kumar, M.; et al. Reconfiguring Shortest Paths in Graphs. *Algorithmica* **2024**, *86*, 3309–3338. [CrossRef]
22. Wu, H.C. Bernstein Approximations for Fuzzy-Valued Functions. *Mathematics* **2025**, *13*, 2424. [CrossRef]
23. Xu, D.K.; Tian, R.Q.; Lu, Y. Bayesian Adaptive Lasso for the Partial Functional Linear Spatial Autoregressive Model. *J. Math.* **2022**, 1616068. [CrossRef]
24. Bäusing, C.; John, D.; Mathwieser, C. Precedence-Constrained Shortest Path. *Networks* **2025**, *86*, 282–295. [CrossRef]
25. Deen, M.R.Z.E.; Aboamer, W.A.; El-Sherbiny, H.M. Explicit Formulas for the Complexity of Networks Produced by New Duplicating Corona and Cartesian Product. *J. Math.* **2024**, 9131329. [CrossRef]
26. Henke, D.; Wulf, L. On the Complexity of the Bilevel Shortest Path Problem. *Networks* **2025**, *86*, 428–445. [CrossRef]
27. Hooshmand, M.K.; Huchaiah, M.D. Network Intrusion Detection with 1D Convolutional Neural Networks. *Digit. Technol. Res. Appl.* **2022**, *1*, 66–75.
28. Kovács, L. Classification Improvement with Integration of Radial Basis Function and Multilayer Perceptron Network Architectures. *Mathematics* **2025**, *13*, 1471. [CrossRef]
29. Acuña, E.G.A.; Doriano, S.C.; Salgado, F. Quantum-Enhanced Cognitive Modeling for Advanced Logistics Route Optimization. *Digit. Technol. Res. Appl.* **2025**, *4*, 61–84.
30. Mao, Z.; Yang, Z.; Luo, D.; et al. A Multi-Strategy Enhanced Dung Beetle Algorithm for Solving Real-World Engineering Problems. *Artif. Intell. Rev.* **2025**, *58*, 253. [CrossRef]
31. Jahanmanesh, P.; Farhadi, A.; Zamanifar, A. Mobility Management with AI. *IEEE Access* **2025**, *13*, 83022–83060. [CrossRef]
32. Razavi, S.M.; Tran, T.M.; Wilson, S.E.; et al. Brain State Transition Disruptions in Alzheimer's Disease: Insights from EEG State Dynamics. In Proceedings of the IEEE International Symposium on Biomedical Imaging, Seoul, Republic of Korea, 2 December 2025.
33. Saghezchi, A.; Kashani, V.G.; Ghodratizadeh, F. A Comprehensive Optimization Approach on Financial Resource Allocation in Scale-Ups. *J. Bus. Manag. Stud.* **2024**, *6*, 62–75. [CrossRef]