Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 21(2): 2626–2645.
DOI: 10.3934/mbe.2024116
Received: 21 December 2023
Revised: 04 January 2024
Accepted: 17 January 2024
Published: 19 January 2024

Research article

A novel approach for calculating single-source shortest paths of weighted digraphs based on rough sets theory

Mingfeng Hua[1], **Taihua Xu**[1,2,*], **Xibei Yang**[1], **Jianjun Chen**[1] **and Jie Yang**[1,3]

[1] School of Computer, Jiangsu University of Science and Technology, Zhenjiang 212100, China

[2] Key Laboratory of Oceanographic Big Data Mining & Application of Zhejiang Province, Zhejiang Ocean University, Zhoushan 316022, China

[3] School of Physics and Electronic Science, Zunyi Normal University, Zunyi 563002, China

* **Correspondence:** Email: xth19890410@163.com.

Abstract: Calculating single-source shortest paths (SSSPs) rapidly and precisely from weighted digraphs is a crucial problem in graph theory. As a mathematical model of processing uncertain tasks, rough sets theory (RST) has been proven to possess the ability of investigating graph theory problems. Recently, some efficient RST approaches for discovering different subgraphs (e.g. strongly connected components) have been presented. This work was devoted to discovering SSSPs of weighted digraphs by aid of RST. First, SSSPs problem was probed by RST, which aimed at supporting the fundamental theory for taking RST approach to calculate SSSPs from weighted digraphs. Second, a heuristic search strategy was designed. The weights of edges can be served as heuristic information to optimize the search way of k-step R-related set, which is an RST operator. By using heuristic search strategy, some invalid searches can be avoided, thereby the efficiency of discovering SSSPs was promoted. Finally, the W3SP@R algorithm based on RST was presented to calculate SSSPs of weighted digraphs. Related experiments were implemented to verify the W3SP@R algorithm. The result exhibited that W3SP@R can precisely calculate SSSPs with competitive efficiency.

Keywords: rough sets theory; graph theory; single-source shortest paths; weighted digraphs; heuristic search strategy

1. Introduction

In the area of graph theory, SSSPs is a crucial topic. From perspective of knowledge discovery, SSSPs is one sort of significant knowledge demanding to be extracted from specified graphs. SSSPs problem is to discover shortest paths from a specified source vertex to other vertices. Presently, SSSPs have been extended to many different research fields, including geographic information system [1], neural network [2–4], fuzzy network [5,6], transportation system [7,8], complex network [9,10], etc.

Up to now, numerous algorithms [11–14] have been devised for computing SSSPs of weighted digraphs, among which Dijkstra [11] is a well-known algorithm for handling SSSPs problem. If a digraph composes of n vertices and m edges, then the classic Dijkstra algorithm needs $O(n^2)$ calculations. Although the Dijkstra algorithm can exactly solve SSSPs problem, the quickly increasing computation time makes it unsuitable for large graphs. For this issue, several approaches have been presented accordingly. For instance, with the help of Fibonacci-heap, Fredman et al. [12] enhanced the efficiency of the Dijkstra algorithm to $O(m + nlogn)$. In addition, Sunita et al. [14] developed a method called D_Dij with time complexity of $O(nlogm)$ by means of retroactive priority queue. In spite of these approaches reducing computation time to some extent, the topic of developing more efficient approaches for calculating SSSPs of weighted digraphs also needs to be considered in future works.

RST [15] is an effective model to process uncertain tasks. So far, RST has been utilized in numerous research fields, including data mining [16–18], decision analysis [19–21], knowledge discovery [22–27], machine learning [28–32], etc. It is worthy to notice that RST has been successfully employed to extract knowledge from graphs. For example, Guan et al. [25] built the heuristic information of vertices by means of RST to calculate the minimum dominating set. Chen et al. [22] calculated connected components by k-step upper approximation, which is an RST operator. Xu et al. [23] first took RST to probe the strongly connected components (SCCs) problem. Inspired by granular computing, the vertex granules can be constructed by RST operators [24], then SCCs problem can be handled successfully in coarse granularity. Consequently, it is practicable to investigate graph theory problems by aid of RST.

Both SSSPs and SCCs, two different knowledge of graph theory, can be discovered through taking the graph theory procedure of breadth-first search (BFS). In [24], it has validated that forward BFS has an equivalent operator, k-step R-related set, in RST. This fact indicates that RST can provide another solution for solving graph theory problems. This perhaps can be the antecedent study of solving uncertain graph theory problems by aid of RST.

This work focuses on designing a novel solution based on RST for discovering SSSPs in weighted digraphs, a certain graph theory problem. First, in light of the theoretical connections between the notions of RST and graph theory, SSSPs problem is probed by means of RST. The fundamental theory of taking the RST approach to calculate SSSPs of weighted digraphs can then be gained. Second, a heuristic search strategy is devised to instruct the detailed implementation of k-step R-related set. Without the heuristic search strategy, the RST operator of k-step R-related set would be trapped in exhaustive search. However, from a global perspective, if a vertex x has the smallest weight to the vertex y, which belongs to the R-related set of x, then the two vertices x and y are more likely to be involved in the SSSPs. The above viewpoint is served as the heuristic information, which can avoid some invalid searches. Finally, combining the heuristic search strategy, a novel RST approach named W3SP@R is presented to calculate SSSPs of weighted digraphs efficiently.

The main contributions are as follows.

1. SSSPs problem in weighted digraphs is probed by means of RST, which can support the fundamental theory for taking the RST approach to calculate SSSPs from weighted digraphs.
2. A heuristic search strategy is introduced to optimize the way of the RST operator (k-step R-related set) in searching SSSPs.
3. The W3SP@R algorithm based on RST is put forward to calculate SSSPs of weighted digraphs.

This paper is planned as follows. In Section 2, the basic knowledge of SSSPs and RST is introduced. In Section 3, SSSPs problem is probed by means of RST. Moreover, a heuristic search strategy is designed. In Section 4, the W3SP@R algorithm based on RST is presented to calculate SSSPs of weighted digraphs. In Section 5, the result analysis is given. In Section 6, the work is summarized.

2. Preliminaries

2.1. SSSPs

SSSPs is a popular study branch of shortest paths problems, which plays a significant role in actual applications. For a digraph $D = (U, E)$ as Definition 1, SSSPs problem aims at calculating shortest paths from the specified source vertex a to all other vertices in D.

Definition 1. ([33]) A digraph $D = (U, E)$ consists of a nonempty finite set U of elements called vertices and a finite set E of ordered pairs of distinct vertices called directed edges. The size of U is denoted by $|U|$ or n. The size of E is denoted by $|E|$ or m.

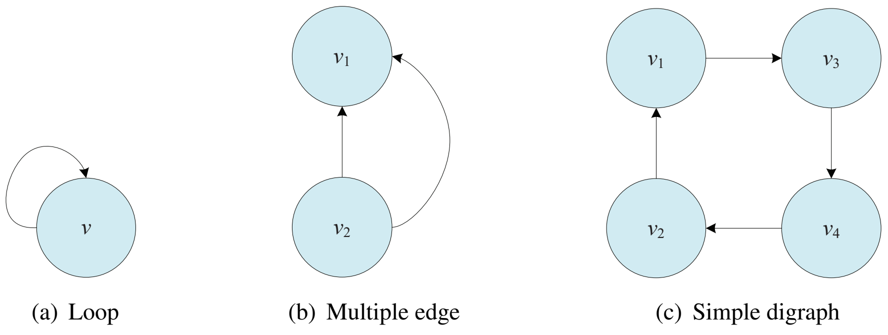

Figure 1. The display of loop, multiple edge and simple digraph.

In particular, a digraph with no multiple edges (or parallel edges) and loop is viewed as a simple digraph. Multiple edges refer to more than one directed edge associated with a pair of vertices, which possess the same direction. Loop refers to the directed edge, where source and target vertices are the same one. Figure 1 provides the examples of loop, multiple edge and simple digraph. In this paper, it should be stressed that the digraphs utilized are simple.

In addition, four fundamental notions in graph theory should be described here because they are commonly used to analyze SSSPs problem, as in Table 1. In special, it can be found: (1) $successors(x) \subseteq descendants(x)$; (2) $predecessors(x) \subseteq ancestors(x)$.

Table 1. Four fundamental notions in graph theory.

Symbols	Notions
$successors(x)$	The successors of x
$descendants(x)$	The descendants of x
$predecessors(x)$	The predecessors of x
$ancestors(x)$	The ancestors of x

Definition 2. ([34]) Let $D = (U, E)$ be a digraph. For $x, y \in U$, if there is a sequence of vertices and edges leading from x to y, that is $xe_1v_1e_2v_2 \cdots v_{k-1}e_ky$, where $e_1, e_2, \cdots, e_k \in E$ and $v_1, v_2, \cdots, v_{k-1} \in U$. The sequence can be regarded as a directed path, denoted by $p : x \overset{*}{\rightarrow} y$, where $\overset{*}{\rightarrow}$ is a binary relation on U and called reachable relation. x is called an ancestor of y, and y is called a descendant of x. Every vertex is an ancestor and a descendant of itself.

Definition 3. ([35]) Given a digraph $D = (U, E)$ with non-negative weights, $w(e)$ on $e \in E$. $P(u, v)$ represents the shortest path from u to v. $P^E(u, v)$ and $P^U(u, v)$ represent the set of edges and vertices, respectively. $|P(u, v)|$ represents the sum of the edge weights of the path from u to v, $|P(u, v)| = \sum_{e \in P^E(u,v)} w(e)$.

According to Definition 3, for any $x, y \in U$, if x has reachable relation to (or from) y, there would be $x \overset{*}{\rightarrow} y$ (or $y \overset{*}{\rightarrow} x$). Actually, there may be several directed paths from x to y (or from y to x). In these paths, the shortest path from x to y (or from y to x) is bound to exist, as Proposition 1.

Proposition 1. Let $D = (U, E)$ be a digraph. $\forall x, y \in U$, and if $x \overset{*}{\rightarrow} y$ or $y \overset{*}{\rightarrow} x$, then there must exist $P(x, y)$ or $P(y, x)$.

Proof. It is straightforward to obtain this conclusion. □

Conversely, if there exists $P(x, y)$ or $P(y, x)$, then $x \overset{*}{\rightarrow} y$ or $y \overset{*}{\rightarrow} x$ can be obtained.

Proposition 2. ([23]) Let $D = (U, E)$ be a digraph. If there are three vertices $x, y, z \in U$ such that $x \overset{*}{\rightarrow} y$ and $y \overset{*}{\rightarrow} z$, then $x \overset{*}{\rightarrow} z$.

Proposition 2 indicates that the reachable relation "$\overset{*}{\rightarrow}$" is transitive. Furthermore, the shortest path is also transitive, as in Proposition 3. For any vertices $x, y, z \in U$, if there exist the shortest paths from x to y and from y to z, respectively, then the shortest path from x to z must exist.

Proposition 3. Let $D = (U, E)$ be a digraph. $\forall x, y, z \in U$, and if there exist $P(x, y)$ and $P(y, z)$, then there must exist $P(x, z)$.

Proof. The existence of $P(x, y)$ means the shortest path from x to y exists, then it is straightforward that $x \overset{*}{\rightarrow} y$. Similarly, $y \overset{*}{\rightarrow} z$ can also be got by the existence of $P(y, z)$. According to Proposition 2,

$$\left.\begin{array}{r} x \overset{*}{\rightarrow} y \\ y \overset{*}{\rightarrow} z \end{array}\right\} \quad \Rightarrow \quad x \overset{*}{\rightarrow} z.$$

According to Proposition 1, $x \overset{*}{\rightarrow} z$ implies that there must exist $P(x, z)$. □

To describe the paths conveniently, the vertex sequence of $(x, v_1, v_2, \cdots, v_i, y)$ is used to represent the path from x to y, where $x, v_1, v_2, \cdots, v_i, y \in U$, x and y are source and target vertices, respectively. Moreover, for any $x, y \in U$, $w(x, y)$ is used to represent the weight of edge $(x, y) \in E$. Next, the SSSPs from a specified source vertex is illustrated in Example 1.

Example 1. *Given a weighted digraph $D = (U, E)$ exhibited in Figure 2, $U = \{a, b, c, d, e\}$, $E = \{(a, b), (a, c), (a, d), (c, b), (c, e), (d, e), (e, b)\}$. Let a be the source vertex.*

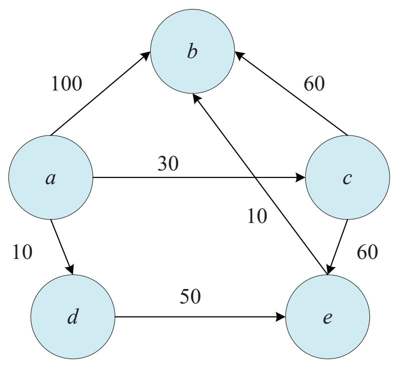

Figure 2. The weighted digraph D in Example 1.

(1) *There are 3 different paths from a to b:* (a, b)*,* (a, c, b)*, and* (a, d, e, b)*. The lengths of them are* 100*,* 90*, and* 70*, respectively. Hence,* (a, d, e, b) *is the shortest path, namely,* $P^U(a, b) = \{a, d, e, b\}$*,* $P^E(a, b) = \{(a, d), (d, e), (e, b)\}$*,* $|P(a, b)| = \sum_{e \in P^E(a,b)} w(e) = w(a, d) + w(d, e) + w(e, b) = 10+50+10 = 70$*;*

(2) *There is only one path from a to c:* (a, c)*. The length of* (a, c) *is 30 and it is the shortest path, namely,* $P^U(a, c) = \{a, c\}$*,* $P^E(a, c) = \{(a, c)\}$*,* $|P(a, c)| = \sum_{e \in P^E(a,c)} w(e) = w(a, c) = 30$*;*

(3) *There is only one path from a to d:* (a, d)*. The length of* (a, d) *is 10 and it is the shortest path, namely,* $P^U(a, d) = \{a, d\}$*,* $P^E(a, d) = \{(a, d)\}$*,* $|P(a, d)| = \sum_{e \in P^E(a,d)} w(e) = w(a, d) = 10$*;*

(4) *There are 2 different paths from a to e:* (a, c, e) *and* (a, d, e)*. The lengths of them are* 90 *and* 60*, respectively. Hence,* (a, d, e) *is the shortest path, namely,* $P^U(a, e) = \{a, d, e\}$*,* $P^E(a, e) = \{(a, d), (d, e)\}$*,* $|P(a, e)| = \sum_{e \in P^E(a,e)} w(e) = w(a, d) + w(d, e) = 10 + 50 = 60$.

In sum, a total of 4 shortest paths from a are listed above.

2.2. RST

Classical RST [15] is a mathematical model presented by Pawlak, which possesses the ability of processing uncertain tasks. With the continuous development of RST, it has been involved in many study areas. In the area of graph theory, there have been several related studies. For instance, Chen et al. [22] abstracted a simple undirected graph as a SP-relation, and an RST operator named k-step upper approximation (as Definition 7) was devised for calculating connected components. Further, Xu et al. [23] abstracted a simple digraph as an IR-relation, and an RST operator named k-step R-related set (as Definition 8) was devised for calculating SCCs.

Definition 4. ([22]) Let R be a relation on U. If R is serial, irreflexive and symmetric, we say R to be a serial preclusivity relation (SP-relation) on U. We call the approximation space (U, R) with R being SP-relation a SPA-space.

Definition 5. ([23]) Let R be a binary relation on U. If R is irreflexive, R is then named a irreflexive relation (IR-relation) on U. The corresponding approximation space (U, R) is here named IRA-space.

Three important notions of RST are introduced as Definition 6, that is, R-related set (as Eq (2.1)), upper approximation set (as Eq (2.2)), and lower approximation set (as Eq (2.3)).

Definition 6. ([23]) Let (U, R) be an IRA-space. For any $x \in U$ and $X \subseteq U$:

$$r(X) = \cup\{r(x) : x \in X\}, \tag{2.1}$$

$$\overline{R}(X) = \{x \in U | r(x) \cap X \neq \emptyset\}, \tag{2.2}$$

$$\underline{R}(X) = \{x \in U | \emptyset \neq r(x) \subseteq X\}. \tag{2.3}$$

Definition 7. ([22]) Let (U, R) be a SPA-space and k a positive number. For any $X \subseteq U$, a new notion of upper approximation $\overline{R}^{(k)}(X)$ called k-step upper approximation of X is introduced as follows:

$$\begin{aligned} &\overline{R}^{(1)}(X) = \overline{R}(X), \\ &\overline{R}^{(k)}(X) = \underbrace{\overline{R}(\overline{R}(\cdots(\overline{R}(X))))}_{k\ times}. \end{aligned} \tag{2.4}$$

Definition 8. ([23]) Let (U, R) be an IRA-space and k a positive number. For any subset $X \subseteq U$, $r^{(k)}(X)$ is named k-step R-related set of X. It is defined as follows:

$$\begin{aligned} &r^{(1)}(X) = r(X), \\ &r^{(k)}(X) = \underbrace{r(r(\cdots(r(X))))}_{k\ times}. \end{aligned} \tag{2.5}$$

The symbol of $\mathscr{A}(X)$ stands for the union of $\overline{R}^{(k)}(X)$, as Eq (2.6); the symbol of $\mathscr{D}(X)$ stands for the union of $r^{(k)}(X)$, as Eq (2.7).

$$\mathscr{A}(X) = \cup\{\overline{R}^{(k)}(X) : k \geq 1\}, \tag{2.6}$$

$$\mathscr{D}(X) = \cup\{r^{(k)}(X) : k \geq 1\}. \tag{2.7}$$

Specifically, $\forall X \subseteq U$, when it is only composed of element x, $\mathscr{A}(x)$ can be substituted for $\mathscr{A}(X)$. Similarly, $\mathscr{D}(x)$ can be substituted for $\mathscr{D}(X)$. In addition, for convenience of describing the intermediate results process of $r^{(k)}(X)$, a notion called the tth-step R-related set is introduced in Definition 9.

Definition 9. Let (U, R) be an IRA-space. $\forall X \subseteq U$, tth-step R-related set of X is denoted by $r^{(t)}(X)$, where t is an integer $(t > 0)$. It is described as follows:

$$r^{(t)}(X) = \underbrace{r(r(\cdots(r(X))))}_{t\ times} \quad (1 \leq t \leq k). \tag{2.8}$$

Correspondingly, the symbol of $\mathscr{D}_t(X)$ stands for the union of the previous t steps of $r^{(k)}(X)$, as Eq (2.9).

$$\mathscr{D}_t(X) = \cup\{r^{(i)}(X) : 1 \leq i \leq t\}. \tag{2.9}$$

Similarly, if X only contains x, then $\mathscr{D}_t(x)$ can be substituted for $\mathscr{D}_t(X)$. It is obvious that $r^{(t)}(x) \subseteq \mathscr{D}_t(x) \subseteq \mathscr{D}(x)$.

3. The theoretical study on SSSPs problem by RST and heuristic search strategy

In this section, SSSPs problem in weighted digraphs is probed by aid of RST, which can support the fundamental theory of taking the RST approach to calculate SSSPs of weighted digraphs. In addition, a heuristic search strategy is devised for improving the calculation efficiency of SSSPs.

3.1. The theoretical study of SSSPs in RST

The theoretical connections between the notions of RST and graph theory are described, which provide theoretical foundations for studying SSSPs by RST, as Theorems 3.1–3.3.

Theorem 3.1. *([23]) Let (U, R) be an IRA-space and $D = (U, E)$ the source directed graph of (U, R). Then, $\forall x \in U$, we have that $r(x) = successors(x)$.*

Theorem 3.2. *([23]) Let (U, R) be an IRA-space and $D = (U, E)$ the source directed graph of (U, R). Then, $\forall x \in U$, we have that $\overline{R}(x) = predecessors(x)$.*

Theorem 3.3. *([23]) Let (U, R) be an IRA-space and $D = (U, E)$ the source directed graph of (U, R). Then, $\forall x \in U$, we have that $\mathscr{D}(x) = descendants(x)$.*

For any vertex x belonging to U, if there is no vertex included in $\mathscr{D}(x)$, then it can be deduced that the SSSPs from x cannot be found. This conclusion is introduced in Theorem 3.4.

Theorem 3.4. *Given the IRA-space (U, R) and its source weighted digraph $D = (U, E)$, let x $(x \in U)$ be the source vertex. If $\mathscr{D}(x) = \emptyset$, then there does not exist SSSPs from x.*

Proof. For the source vertex x, if $\mathscr{D}(x) = \emptyset$, then we can get $descendants(x) = \emptyset$ according to the conclusion of $\mathscr{D}(x) = descendants(x)$ in Theorem 3.3. Further, $descendants(x) = \emptyset \Leftrightarrow \forall y$ $(y \in U \wedge y \notin descendants(x))$. In other words, there is no $y \in U$ so that $x \overset{*}{\nrightarrow} y$, since x cannot reach any other vertex y, let alone the existence of the shortest path $P(x, y)$. Overall, there does not exist SSSPs from x. □

According to above proof, if a vertex y belongs to $\mathscr{D}(x)$, then $x \overset{*}{\rightarrow} y$ can be got. By Proposition 1, there must exist $P(x, y)$. Thus, the SSSPs from x to y is bound to exist.

Moreover, if there is no vertex included in $r(x)$, then it also can be deduced that the SSSPs from x does not exist, as in Theorem 3.5.

Theorem 3.5. *Given the IRA-space (U, R) and its source weighted digraph $D = (U, E)$, let x $(x \in U)$ be the source vertex. If $r(x) = \emptyset$, then there does not exist SSSPs from x.*

Proof. In Eq (2.7), $\mathscr{D}(x) = \cup\{r^{(k)}(x) : k \geq 1\} = r(x) \cup \bigcup_{y \in r(x)} \mathscr{D}(y)$. It is obvious that $r(x) = \emptyset \Leftrightarrow \mathscr{D}(x) = \emptyset$. The result of $\mathscr{D}(x) = \emptyset$ indicates that there does not exist SSSPs from x by Theorem 3.4. □

Similarly, by $\mathscr{D}(x) = r(x) \cup \bigcup_{y \in r(x)} \mathscr{D}(y)$, if $r(x) \neq \emptyset$, then $\mathscr{D}(x) \neq \emptyset$ can be found, namely, $\mathscr{D}(x)$ contains a vertex y at least. Thus, the SSSPs from x must exist. Specially, if only y included in $r(x)$, then it can be obtained that (x, y) is the shortest path from x to y, as shown in Theorem 3.6.

Theorem 3.6. *Given the IRA-space (U, R) and its source weighted digraph $D = (U, E)$, let x $(x \in U)$ be the source vertex. $\forall y \in U$, if $r(x) = \{y\}$, then $P^U(x, y) = \{x, y\}$ and $|P(x, y)| = w(x, y)$.*

Proof. For the source vertex x, if $r(x) = \{y\}$, then we can get $successors(x) = \{y\}$ according to the conclusion of $r(x) = successors(x)$ in Theorem 3.1. In addition,

$$\left.\begin{array}{l} successors(x) \subseteq descendants(x) \\ successors(x) = \{y\} \end{array}\right\} \Rightarrow \quad y \in descendants(x) \quad \Leftrightarrow \quad x \overset{*}{\rightarrow} y.$$

By Proposition 1, there must exist $P(x, y)$. It is obvious that (x, y) is the only path from x to y because $successors(x) = \{y\}$, thus, it is definitely the shortest path $P(x, y)$. In conclusion, if $r(x) = \{y\}$, then $P(x, y)$ can be determined directly, that is, $P^U(x, y) = \{x, y\}$ and $|P(x, y)| = w(x, y)$. □

Use the RST notion of the upper approximation set to analyze the SSSPs. It can be concluded that if only the source vertex x included in $\overline{R}(y)$, then it can be obtained that (x, y) is the shortest path from x to y, as in Theorem 3.7.

Theorem 3.7. *Given the IRA-space (U, R) and its source weighted digraph $D = (U, E)$. Let x $(x \in U)$ be the source vertex. $\forall y \in r(x)$, and if $\overline{R}(y) = \{x\}$, then $P^U(x, y) = \{x, y\}$ and $|P(x, y)| = w(x, y)$.*

Proof. If $\overline{R}(y) = \{x\}$, then we can get $predecessors(y) = \{x\}$ according to the conclusion of $\overline{R}(x) = predecessors(x)$ in Theorem 3.2. In addition,

$$\left.\begin{array}{l} predecessors(y) \subseteq ancestors(y) \\ predecessors(y) = \{x\} \end{array}\right\} \Rightarrow \quad x \in ancestors(y) \quad \Leftrightarrow \quad x \overset{*}{\rightarrow} y.$$

By Proposition 1, there must exist $P(x, y)$. It is obvious that (x, y) is the only path from x to y because $predecessors(y) = \{x\}$, thus, it is definitely the shortest path $P(x, y)$. In conclusion, if $\overline{R}(y) = \{x\}$, then $P(x, y)$ can be determined directly, that is, $P^U(x, y) = \{x, y\}$ and $|P(x, y)| = w(x, y)$. □

Example 2. *The weighted digraph D in Figure 2 is utilized to interpret above theories. The binary relation $R = \{(a, b), (a, c), (a, d), (c, b), (c, e), (d, e), (e, b)\}$ is constructed from D. Table 2 displays the R-related set and upper approximation of each vertex in D.*

Table 2. R-related set and upper approximation of each vertex.

x	a	b	c	d	e
$r(x)$	$\{b, c, d\}$	$\emptyset$	$\{b, e\}$	$\{e\}$	$\{b\}$
$\overline{R}(x)$	$\emptyset$	$\{a, c, e\}$	$\{a\}$	$\{a\}$	$\{c, d\}$

Let a be the source vertex. Since $\overline{R}(c) = \overline{R}(d) = \{a\}$, $P^U(a, c) = \{a, c\}$, $|P(a, c)| = w(a, c) = 30$, $P^U(a, d) = \{a, d\}$ and $|P(a, d)| = w(a, d) = 10$ according to Theorem 3.7.

Let b be the source vertex. Since $r(b) = \emptyset$, the SSSPs from b does not exist according to Theorem 3.5.

Let d be the source vertex. Since $r(d) = \{e\}$, $P^U(d, e) = \{d, e\}$ and $|P(d, e)| = w(d, e) = 50$ can be found according to Theorem 3.6.

Next, the instance of $r(x)$ containing multiple elements needs to be analyzed, as in Theorem 3.8.

Theorem 3.8. *Given the IRA-space (U, R) and its source digraph $D = (U, E)$ with positive weights. Let x $(x \in U)$ be the source vertex, $r(x) = \{y_1, y_2, \cdots, y_i, \cdots, y_n\}$ $(1 \leq i \leq n)$. If $\forall y_i \exists y_j$ $(w(x, y_j) \leq w(x, y_i))$, then $P^U(x, y_j) = \{x, y_j\}$ and $|P(x, y_j)| = w(x, y_j)$.*

Proof. For the source vertex $x \in U$, if $r(x) = \{y_1, y_2, \cdots, y_i, \cdots, y_n\}$, then we can get $successors(x) = \{y_1, y_2, \cdots, y_i, \cdots, y_n\}$ according to the conclusion of $r(x) = successors(x)$ in Theorem 3.1. In addition,

$$\left.\begin{array}{l} successors(x) = \{y_1, y_2, \cdots, y_i, \cdots, y_n\} \\ successors(x) \subseteq descendants(x) \end{array}\right\} \Rightarrow \quad \{y_1, y_2, \cdots, y_i, \cdots, y_n\} \subseteq descendants(x).$$

According to Definition 2, for any $y_i \in descendants(x) \Leftrightarrow x \overset{*}{\rightarrow} y_i$ $(1 \leq i \leq n)$. The expression of $\forall y_i \exists y_j\ (w(x, y_j) \leq w(x, y_i))$ means that $w(x, y_j)$ is the shortest distance between x and y_i $(1 \leq i \leq n)$. Except for the path of (x, y_j), x may reach y_j by passing through the other y_i. Obviously, the path of passing through the other y_i cannot be shorter than the path of (x, y_j) because that $w(x, y_j)$ is the shortest distance between x and y_i $(1 \leq i \leq n)$. Hence, we have $P^U(x, y_j) = \{x, y_j\}$ and $|P(x, y_j)| = w(x, y_j)$. □

For convenience of computing SSSPs from source vertex x, $Dist(y)$ is used to denote the distance from x to y. In the beginning of computing SSSPs, for each vertex $y \in U$, the distance from x to y is initialized as infinite ($Dist(y) = \infty$). Specifically, if there exists $y \in U$ so that x cannot reach y, then $Dist(y) = \infty$. During the process of finding SSSPs from x, if any vertex $y \in U$ is visited, then $Dist(y)$ will be updated by the weights of corresponding edges. Let $\mathscr{D}_t(x) = \{y_1, y_2, \cdots, y_i, \cdots, y_t\}$ $(1 \leq i \leq t)$. If there exists $y_j \in \mathscr{D}_t(x)$ (obviously, $Dist(y_j) \neq \infty$) so that $Dist(y_j)$ is always less than or equal to $Dist(y_i)$ $(1 \leq i \leq t)$, then $Dist(y_j)$ can be determined as the shortest distance from x to y_j $(\forall y_i \exists y_j (Dist(y_j) \leq Dist(y_i)\ (1 \leq i \leq t)))$, as exhibited in Theorem 3.9.

Theorem 3.9. *Given the IRA-space (U, R) and its source digraph $D = (U, E)$ with positive weights. Let x $(x \in U)$ be the source vertex, $\mathscr{D}_t(x) = \{y_1, y_2, \cdots, y_i, \cdots, y_t\}$ $(1 \leq i \leq t)$. If $\forall y_i \exists y_j\ (Dist(y_j) \leq Dist(y_i))$, then $|P(x, y_j)| = Dist(y_j)$.*

Proof. For the source vertex $x \in U$, if $\mathscr{D}_t(x) = \{y_1, y_2, \cdots, y_i, \cdots, y_t\}$, then we can get $\{y_1, y_2, \cdots, y_i, \cdots, y_t\} \subseteq \mathscr{D}(x)$ by $\mathscr{D}_t(x) \subseteq \mathscr{D}(x)$. In addition, $\{y_1, y_2, \cdots, y_i, \cdots, y_t\} \subseteq descendants(x)$ can be found according to the conclusion of $\mathscr{D}(x) = descendants(x)$ in Theorem 3.3. According to Definition 2, for any $y_i \in descendants(x) \Leftrightarrow x \overset{*}{\rightarrow} y_i$ $(1 \leq i \leq t)$. The expression of $\forall y_i \exists y_j\ (Dist(y_j) \leq Dist(y_i))$ means that $Dist(y_j)$ is the shortest distance between x and y_i $(1 \leq i \leq t)$. Although x may reach y_j by passing through the other y_i, the length of the path by passing through the other y_i cannot be shorter than $Dist(y_j)$ because that $Dist(y_j)$ is the shortest distance between x and y_i $(1 \leq i \leq t)$. Finally, we have $|P(x, y_j)| = Dist(y_j)$. □

Corollary 1. Given the IRA-space (U, R) and its source digraph $D = (U, E)$ with positive weights. Let x $(x \in U)$ be the source vertex, $\mathscr{D}(x) = \{y_1, y_2, \cdots, y_i, \cdots, y_n\}$ $(1 \leq i \leq n)$. If $\forall y_i \exists y_j\ (Dist(y_j) \leq Dist(y_i))$, then $|P(x, y_j)| = Dist(y_j)$.

Proof. Actually, $\mathscr{D}(x)$ is a special case of $\mathscr{D}_t(x)$. Hence, the conclusion of $|P(x, y_j)| = Dist(y_j)$ is straightforward according to Theorem 3.9. □

3.2. Heuristic search strategy

The RST operator of the k-step R-related set can be employed to traverse reachable vertices of specified vertex x. That is to say, any vertex whom x has reachable relation "$\overset{*}{\rightarrow}$" to can be found by k-step R-related set. However, the implementation of the k-step R-related set is an exhaustive search process, which affects calculational efficiency of SSSPs. In response to this issue, a heuristic search strategy is devised to instruct the search way of the k-step R-related set.

Given a weighted digraph $D = (U, E)$ and a source vertex x, two sets are used, Q and T. Initially, $Q = U - \{x\}$ and $T = \{x\}$, the distances from x to all vertices in U are initialized as infinite. The shortest paths of vertices in Q have not been determined. If the shortest path of a vertex is determined, then it will be added into T. The details of the heuristic search strategy are as follows:

- Based on Theorem 3.6, if $r(x) = \{u\}$, then $P(x, u) = (x, u)$. The corresponding heuristic operation is adding u into T;
- Based on Theorem 3.7, if $\overline{R}(u) = \{x\}$, then $P(x, u) = (x, u)$. The corresponding heuristic operation is adding u into T;
- Based on Theorems 3.8, if $r(x) = \{u_1, u_2, \cdots, u_i, \cdots, u_n\}$, $\forall u_i \exists u$ $(w(x, u) \leq w(x, u_i)$ $(1 \leq i \leq n))$, then $|P(x, u)| = Dist(u) = w(x, u)$. The corresponding heuristic operations are adding u into T and removing u from Q;
- Based on Theorems 3.9, if $\mathscr{D}_t(x) = \{u_1, u_2, \cdots, u_i, \cdots, u_t\}$, $\forall u_i \exists u$ $(Dist(u) \leq Dist(u_i)$ $(1 \leq i \leq t))$, then $|P(x, u)| = Dist(u)$. The corresponding heuristic operation is using the found $|P(x, u)|$ to instruct the computation of k-step R-related set, that is $r^{(t+1)}(x) = r^{(t)}(x) \cup r(u)$. As a result, $\mathscr{D}_{t+1}(x) = \mathscr{D}_t(x) \cup r(u)$, then remove u from Q and add u into T. Meanwhile, $\forall v \in r(u)$, if $v \notin T$ and $Dist(v) > Dist(u) + w(u, v)$, then update $Dist(v)$ to $Dist(u) + w(u, v)$.

In short, the weights of edges can be served as heuristic information to optimize the search way of the k-step R-related set, which can avoid some invalid searches. The specific process of calculating SSSPs by heuristic search strategy will be demonstrated in Section 4.

4. Proposed algorithm and the case study

In this section, an RST approach is presented to discover SSSPs of weighted digraphs, then a case study is taken to introduce the presented approach.

4.1. An algorithm for calculating SSSPs of weighted digraphs based on RST

Combining the heuristic search strategy, a novel approach based on RST named W3SP@R is presented to calculate SSSPs of weighted digraphs, as Algorithm 1.

Algorithm 1 An algorithm for calculating SSSPs of weighted digraphs based on RST (W3SP@R).

```
Input: A weighted digraph $D = (U, E)$ and a source vertex $s$.
Output: $Pre(x)$ and $Dist(x)$; // The predecessor of each vertex $x$ in SSSPs and the distance of each
    vertex $x$ in $U$ from $s$.
1: $Pre(x) \leftarrow -1$; $Dist(x) \leftarrow \infty$; $Q \leftarrow U \setminus \{s\}$; $T \leftarrow \{s\}$; $k \leftarrow 1$;
2: if $r(s) = \emptyset$ then
3:     return; // The SSSPs from $s$ do not exist.
4: end if
5: if $r(s) = \{x\}$ then
6:     $Pre(x) \leftarrow s$; $Dist(x) \leftarrow w(s, x)$; $T \leftarrow T \cup \{x\}$;
7: end if
```

```
8: if $|r(s)| > 1$ then
9:     for $x \in r(s)$ do
10:        if $\overline{R}\{x\} = \{s\}$ then
11:            $Pre(x) \leftarrow s$; $Dist(x) \leftarrow w(s,x)$; $T \leftarrow T \cup \{x\}$; continue;
12:        end if
13:        $Pre(x) \leftarrow s$; $Dist(x) \leftarrow w(s,x)$;
14:    end for
15:    Pick out the $x$ with the minimal $Dist(x)$;
16: end if
17: while $Q \neq \emptyset$ do
18:     if $Dist(x) = \infty$ then
19:         break; // All remaining vertices belong to $Q$ are inaccessible from $s$.
20:     end if
21:     $Q \leftarrow Q \setminus \{x\}$; $T \leftarrow T \cup \{x\}$;
22:     for $y \in r(x)$ do
23:         if $y \notin T$ and $Dist(y) > Dist(x) + w(x,y)$ then
24:             $Dist(y) \leftarrow Dist(x) + w(x,y)$; $Pre(y) \leftarrow x$;
25:         end if
26:     end for
27:     $k \leftarrow k + 1$; Pick out the $x$ with the minimal $Dist(x)$;
28: end while
```

Phase 1 of W3SP@R (steps 2–16): Based on Theorem 3.5, determining whether SSSPs from s exists through calculating R-related set of s, that is, $r(s)$. If $r(s)$ is empty, then the procedure can be ended. If $r(s)$ is not empty, then several vertices are picked out quickly to determine the shortest distance from s according to Theorems 3.6 and 3.7, meanwhile, they will be added into T. By Theorem 3.8, the vertex with the minimal distance $Dist(x)$ is picked out.

Phase 2 of W3SP@R (steps 17–28): For the $Dist(x)$ picked out in Phase 1, if $Dist(x)$ is infinite, then all remaining vertices belong to Q are inaccessible from s and the procedure can be ended. Otherwise, remove x from Q and adding x into T. Subsequently, the k-step R-related set of x is performed heuristically. For each vertex y in $r(x)$, the distance from s to y ($Dist(y)$) and predecessor of y in SSSPs ($Pre(y)$) are updated. After that, the vertex with the minimal distance $Dist(x)$ is picked out and conducts the next iteration of steps 17–28. With the iterations, Q will eventually become empty, which implies that all SSSPs from s have been calculated, then the procedure can be ended.

The core of W3SP@R algorithm is Phase 2. Assuming that weighted digraph $D = (U, E)$ is the input of W3SP@R, where $|U| = n$, $|E| = m$. Because only a vertex x with the minimal distance $Dist(x)$ from s is extracted to calculate the R-related set in each iteration, and it will not be extracted again. Then total m edges will be visited at most. Meanwhile, only the vertices belonging to the R-related set of x are found, then n vertices will be visited in each iteration at most. If total k iterations are performed to calculate SSSPs, then the worst calculational time of the W3SP@R algorithm is $O(m) + O\left(\sum_{i=1}^{k} n\right) = O(m + kn)$.

4.2. A case study of the W3SP@R algorithm

The specific process of the W3SP@R algorithm in computing the SSSPs of an example digraph is demonstrated as Example 3.

Example 3. *Given a weighted digraph $D = (U, E)$ in Figure 3, the binary relation $R = \{(a, b), (a, c), (a, g), (a, h), (b, a), (c, d), (c, e), (c, f), (d, b), (d, f), (h, c), (h, e), (i, e)\}$ is constructed from D. Let a be the source vertex.*

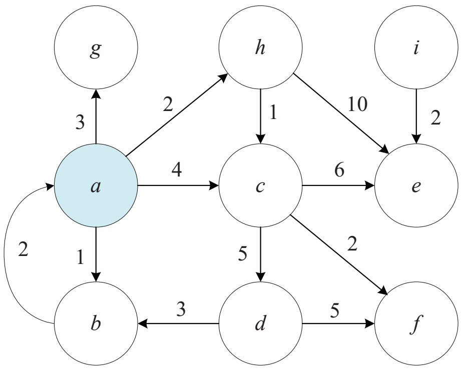

Figure 3. The weighted digraph D of Example 3.

Table 3. The initial status of two outputs of W3SP@R.

x	a	b	c	d	e	f	g	h	i
$Pre(x)$	–1	–1	–1	–1	–1	–1	–1	–1	–1
$Dist(x)$	∞	∞	∞	∞	∞	∞	∞	∞	∞

Now, W3SP@R is taken to discover SSSPs of D: In the beginning, Q is initialized to $\{b, c, d, e, f, g, h, i\}$, T is initialized to $\{a\}$, the predecessor of each vertex in SSSPs is initialized to –1, and the distance of each vertex in U from s is initialized to infinite. The initial status of two outputs of W3SP@R is shown as Table 3. Starting from the source vertex a, $r(a) = \{b, c, g, h\} \neq \emptyset$. By steps 10–12, because $\overline{R}\{g\} = \{a\}$, then $Pre(g) = a$, $Dist(g) = w(a, g) = 3$, and $T = T \cup \{g\} = \{a, g\}$. Similarly, $Pre(h) = a$, $Dist(h) = w(a, h) = 2$, and $T = T \cup \{h\} = \{a, g, h\}$ by reason of $\overline{R}\{h\} = \{a\}$. The remaining b and c are treated by step 13, then $Pre(b) = a$, $Dist(b) = w(a, b) = 1$, $Pre(c) = a$, and $Dist(c) = w(a, c) = 4$. After Phase 1, because the path information of b, c, g, and h are revised, then the status of two outputs of W3SP@R is updated, as shown in Table 4. Next, according to Table 4, for the vertices in Q, because vertex b has the minimal distance ($Dist(b) = 1$) from a, it is picked out by step 15 to induce the first iteration of Phase 2.

Table 4. The status of two outputs of W3SP@R after Phase 1.

x	a	b	c	d	e	f	g	h	i
$Pre(x)$	–1	a	a	–1	–1	–1	a	a	–1
$Dist(x)$	∞	1	4	∞	∞	∞	3	2	∞

$Q = \{b, c, d, e, f, g, h, i\} \neq \emptyset$. 1st iteration of Phase 2: Obviously, $Dist(b) = 1 \neq \infty$, then $Q = Q - \{b\} = \{c, d, e, f, g, h, i\}$ and $T = T \cup \{b\} = \{a, b, g, h\}$ by step 21. Because $r(b) = \{a\}$ and $a \in T$, there is no update operations on $Pre(a)$ and $Dist(a)$ by step 23. After the first iteration, because the path information of any vertex has not been revised, then the status of two outputs of W3SP@R is also shown as Table 4. Next, according to Table 4, for the vertices in Q, because vertex h has the minimal distance ($Dist(h) = 2$)

from a, it is picked out by step 27 to induce the second iteration of Phase 2.

$Q = \{c, d, e, f, g, h, i\} \neq \emptyset$. *2nd iteration of Phase 2: Obviously,* $Dist(h) = 2 \neq \infty$, *then* $Q = Q - \{h\} = \{c, d, e, f, g, i\}$ *by step 21. By steps 22–26,* $r(h) = \{c, e\}$, *then there is some update operations, which are* $Dist(c) = 3$, $Pre(c) = h$, $Dist(e) = 12$, *and* $Pre(e) = h$. *After the second iteration, because the path information of c and e are revised, the status of two outputs of W3SP@R is updated, as shown in Table 5. Next, according to Table 5, for the vertices in Q, because vertex c has the minimal distance* ($Dist(c) = 3$) *from a, it is picked out by step 27 to induce the third iteration of Phase 2.*

Table 5. The status of two outputs of W3SP@R after second iteration.

x	a	b	c	d	e	f	g	h	i
$Pre(x)$	–1	a	h	–1	h	–1	a	a	–1
$Dist(x)$	∞	1	3	∞	12	∞	3	2	∞

$Q = \{c, d, e, f, g, i\} \neq \emptyset$. *3rd iteration of Phase 2: Obviously,* $Dist(c) = 3 \neq \infty$, *then* $Q = Q - \{c\} = \{d, e, f, g, i\}$ *and* $T = T \cup \{c\} = \{a, b, c, g, h\}$ *by step 21. By steps 22-26,* $r(c) = \{d, e, f\}$, *then there is some update operations, which are* $Dist(d) = 8$, $Pre(d) = c$, $Dist(e) = 9$, $Pre(e) = c$, $Dist(f) = 5$, *and* $Pre(f) = c$. *After the third iteration, because the path information of d, e, and f are revised, the status of two outputs of W3SP@R is updated, as shown in Table 6. Next, according to Table 6, for the vertices in Q, because vertex g has the minimal distance* ($Dist(g) = 3$) *from a, it is picked out by step 27 to induce the fourth iteration of Phase 2.*

Table 6. The status of two outputs of W3SP@R after third iteration.

x	a	b	c	d	e	f	g	h	i
$Pre(x)$	–1	a	h	c	c	c	a	a	–1
$Dist(x)$	∞	1	3	8	9	5	3	2	∞

$Q = \{d, e, f, g, i\} \neq \emptyset$. *4th iteration of Phase 2: Obviously,* $Dist(g) = 3 \neq \infty$, *then* $Q = Q - \{g\} = \{d, e, f, i\}$ *by step 21. Because* $r(g) = \emptyset$, *there is no update operation by step 22. After the fourth iteration, because the path information of any vertex has not been revised, the status of two outputs of W3SP@R is also shown as Table 6. Next, according to Table 6, for the vertices in Q, because vertex f has the minimal distance* ($Dist(f) = 5$) *from a, it is picked out by step 27 to induce the fifth iteration of Phase 2.*

$Q = \{d, e, f, i\} \neq \emptyset$. *5th iteration of Phase 2: Obviously,* $Dist(f) = 5 \neq \infty$, *then* $Q = Q - \{f\} = \{d, e, i\}$ *and* $T = T \cup \{f\} = \{b, c, e, g, h, f\}$ *by step 21. Because* $r(f) = \emptyset$, *there is no update operation by step 22. After the fifth iteration, because the path information of any vertex has not been revised, the status of two outputs of W3SP@R is also shown as Table 6. Next, according to Table 6, for the vertices in Q, because vertex d has the minimal distance* ($Dist(d) = 8$) *from a, it is picked out by step 27 to induce the sixth iteration of Phase 2.*

$Q = \{d, e, i\} \neq \emptyset$. *6th iteration of Phase 2: Obviously,* $Dist(d) = 8 \neq \infty$, *then* $Q = Q - \{d\} = \{e, i\}$ *and* $T = T \cup \{d\} = \{b, c, d, e, g, h, f\}$ *by step 21. Because* $r(d) = \{b, f\}$ *and* $\{b, f\} \subseteq T$, *there is no update operations on* $Pre(b)$, $Dist(b)$, $Pre(f)$, *and* $Dist(f)$ *by step 23. After the sixth iteration, because the path information of any vertex has not been revised, the status of two outputs of W3SP@R is also shown as Table 6. Next, according to Table 6, for the vertices in Q, because vertex e has the minimal distance* ($Dist(e) = 9$) *from a, it is picked out by step 27 to induce the seventh iteration of Phase 2.*

$Q = \{e, i\} \neq \emptyset$. 7th iteration of Phase 2: Obviously, $Dist(e) = 9 \neq \infty$, then $Q = Q - \{e\} = \{i\}$ by step 21. Because $r(e) = \emptyset$, there is no update operation by step 22. After the seventh iteration, because the path information of any vertex has not been revised, the status of two outputs of W3SP@R is also shown as Table 6. Next, according to Table 6, for the vertices in Q, because vertex i has the minimal distance ($Dist(i) = \infty$) from a, it is picked out by step 27 to induce the eighth iteration of Phase 2.

$Q = \{i\} \neq \emptyset$. 8th iteration of Phase 2: By $Dist(i) = \infty$, the procedure is ended by steps 18–20. Because the path information of any vertex has not been revised, the status of two outputs of W3SP@R is also shown as Table 6.

Ultimately, the final status of two outputs of W3SP@R is recorded in Table 6. Actually, the distance from a to itself is 0 ($Dist(a) = 0$). To ensure the correct execution of the W3SP@R algorithm, $Dist(a)$ is considered as infinite here for being neglected by the selection of minimal distance. According to the obtained $Pre(x)$ and $Dist(x)$, SSSPs can be found. In detail, total 7 shortest paths from a are discovered: (1) $P^U(a, b) = \{a, b\}$, $P^E(a, b) = \{(a, b)\}$, $|P(a, b)| = 1$; (2) $P^U(a, c) = \{a, h, c\}$, $P^E(a, c) = \{(a, h), (h, c)\}$, $|P(a, c)| = 3$; (3) $P^U(a, d) = \{a, h, c, d\}$, $P^E(a, d) = \{(a, h), (h, c), (c, d)\}$, $|P(a, d)| = 8$; (4) $P^U(a, e) = \{a, h, c, e\}$, $P^E(a, e) = \{(a, h), (h, c), (c, e)\}$, $|P(a, e)| = 9$; (5) $P^U(a, f) = \{a, h, c, f\}$, $P^E(a, f) = \{(a, h), (h, c), (c, f)\}$, $|P(a, f)| = 5$; (6) $P^U(a, g) = \{a, g\}$, $P^E(a, g) = \{(a, g)\}$, $|P(a, g)| = 3$; (7) $P^U(a, h) = \{a, h\}$, $P^E(a, h) = \{(a, h)\}$, $|P(a, h)| = 2$.

In the process of the W3SP@R algorithm, by means of the heuristic search strategy, the exhaustive search can be optimized. For instance, in the 2nd iteration of Phase 2, $r(h) = \{c, e\}$, then $Dist(c)$ and $Dist(e)$ are updated. According to heuristic search strategy, vertex c has the minimal distance from s, hence, only c is picked out to induce the 3rd iteration of Phase 2. Meanwhile, the search on $r(e)$ can be avoided. As a result, there will be an improvement on efficiency of calculating SSSPs, which will be verified in the experimental section.

5. Experiments

In this section, a range of datasets are prepared for experiments. Furthermore, the analysis of test results are drawn.

5.1. Experimental configuration

The experiments are conducted on a 2.80 GHz Intel(R) Core(TM) i7-7700HQ PC (Win10) with 8 GB RAM. The testing platform is MATLAB with version R2017b. A total of 12 different datasets are utilized for experiments, which are gained from a UFMC database [36], as exhibited in Table 7. Concretely, m and n stand for the number of edges and vertices, respectively. The comparison algorithms are Dijkstra [11] and D_Dij [14]. Dijkstra is selected because it is a well-known approach to exactly discover SSSPs from digraphs. As for the reason of selecting the D_Dij algorithm, in addition to guaranteeing the correctness of the results, it also has less calculation time than Dijkstra.

Table 7. The details of datasets.

Number	Datasets	n	m
1	gre-216b	216	660
2	Harvard500	500	2563
3	Roget	1010	5074
4	CollegeMsg	1899	20296
5	ODLIS	2900	18241
6	Kohonen	3772	12729
7	Pd	5098	4509
8	California	6175	16150
9	wiki-Vote	8297	103689
10	coater2	9540	206968
11	foldoc	13356	120238
12	poli	15575	17458

5.2. Comparative study

The validity of the presented W3SP@R algorithm demands to be tested at first. The SSSPs calculated by the Dijkstra algorithm are utilized as the comparative reference for the standard. Specifically, MATLAB provides a method named *isequal*(), which can be used for data comparison. *isequal*() is taken to compare the SSSPs calculated by W3SP@R with that of the Dijkstra algorithm. The final return value obtained is 1, which means that the SSSPs calculated by W3SP@R are identical to the Dijkstra algorithm. Therefore, it is proved that SSSPs can be exactly calculated by the W3SP@R algorithm.

The calculation efficiency of the W3SP@R algorithm also needs to be considered. Concretely, the calculation efficiency of the Dijkstra (column 2), D_Dij (column 3), and W3SP@R (column 4) algorithms are exhibited in Table 8. The symbol of $S_{w3s\backslash di}$ in column 5 stands for the acceleration ratio of W3SP@R relative to Dijkstra. The symbol of $S_{w3s\backslash d\_di}$ in column 6 stands for the acceleration ratio of W3SP@R relative to D_Dij.

By observing the performances of three algorithms on each dataset, it is apparent that W3SP@R and D_Dij are superior to the Dijkstra algorithm in terms of calculation time. Moreover, W3SP@R also has comparable efficiency to the D_Dij algorithm. The efficiency advantage of the W3SP@R algorithm can be reflected through the acceleration ratio; the scope of $S_{w3s\backslash di}$ is 2.79 × –6.99×, and the scope of $S_{w3s\backslash d\_di}$ is 1.13 × –2.09×.

Furthermore, for reflecting the efficiency of the W3SP@R algorithm more intuitively, the changing trends of calculation time of the Dijkstra, D_Dij, and W3SP@R algorithms are drawn as Figure 4. The x-axis stands for the number of vertices and the y-axis stands for the execution time. Obviously, the efficiency supplied by the W3SP@R algorithm is better than that of the other two algorithms.

Table 8. The calculation efficiency of three algorithms on 12 datasets.

Datasets	Execution time (s)			$S_{w3s\backslash di}$	$S_{w3s\backslash d\_di}$
	Dijkstra [11]	D_Dij [14]	W3SP@R		
gre-216b	0.0052	0.0022	**0.0013**	3.89	1.60
Harvard500	0.0130	0.0057	**0.0042**	3.09	1.36
Roget	0.0433	0.0235	**0.0155**	2.79	1.51
CollegeMsg	0.1451	0.0949	**0.0454**	3.20	2.09
ODLIS	0.3457	0.1238	**0.0856**	4.04	1.45
Kohonen	0.5770	0.1330	**0.1146**	5.03	1.16
Pd	1.1785	0.2171	**0.1783**	6.61	1.22
California	1.5373	0.3220	**0.2675**	5.75	1.20
wiki-Vote	2.7442	0.6081	**0.5066**	5.42	1.20
coater2	4.5368	0.7332	**0.6492**	6.99	1.13
foldoc	8.0022	4.7078	**2.4553**	3.26	1.92
poli	8.7143	4.3446	**3.0129**	2.89	1.45

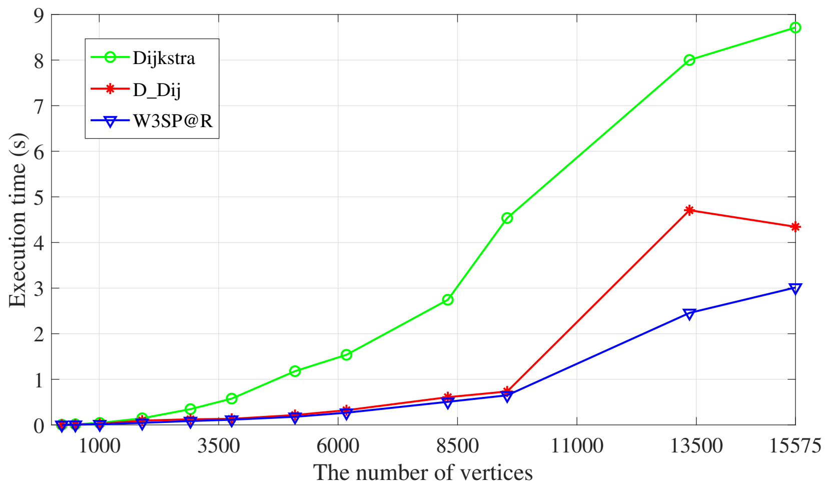

Figure 4. The changing trends of calculation time of three algorithms on 12 datasets.

5.3. Related discussion

A range of experiments display that the presented W3SP@R algorithm based on RST is effective to extract SSSPs from weighted digraphs. In terms of calculation time, the W3SP@R algorithm performs better than the Dijkstra and D_Dij algorithms. The W3SP@R algorithm combines with a heuristic search strategy to change the way of the k-step R-related set in searching SSSPs, which can optimize the exhaustive search. Overall, the utilization of the heuristic search strategy can lead to an improvement on efficiency of calculating SSSPs.

6. Conclusions

As a crucial knowledge of graph theory, SSSPs is popularly utilized to actual research. In this paper, a novel solution based on RST for calculating SSSPs of weighted digraphs was presented. First, in order to acquire the fundamental theory of taking the RST approach to calculate SSSPs of weighted digraphs, SSSPs problem was probed by aid of RST. Second, a heuristic search strategy was introduced to optimize the search way of the k-step R-related set, which is an RST operator. By using this strategy, some invalid searches can be avoided. Finally, through combining the heuristic search strategy, an approach named W3SP@R based on RST for calculating SSSPs of weighted digraphs was proposed. The testing result exhibited that the SSSPs in weighted digraphs can be exactly calculated by the W3SP@R algorithm. Moreover, the performance of W3SP@R was competitive to that of the Dijkstra and D_Dij algorithms in terms of calculation time. In conclusion, SSSPs is successfully extracted from weighted digraphs through using the RST approach. This work can be the antecedent study of handling uncertain graph theory problems by means of RST.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos.62006099, 62076111) and the Key Laboratory of Oceanographic Big Data Mining & Application of Zhejiang Province (No.OBDMA202104). The authors would like to thank the anonymous reviewers for their insightful and constructive comments, which greatly improved the quality of this paper.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. R. M. Murekatete, T. Shirabe, An experimental analysis of least-cost path models on ordinal-scaled raster surfaces, *Int. J. Geogr. Inf. Sci.*, **35** (2021), 1545–1569. https://doi.org/10.1080/13658816.2020.1753204
2. J. Y. Dong, J. Y. Zhang, Z. Chen, Autowave-competition neural network and its application to the single source shortest paths problem, *Acta Phys. Sin.*, **56** (2007), 5013–5020. https://doi.org/10.7498/aps.56.5013
3. M. Eshaghnezhad, F. Rahbarnia, S. Effati, A. Mansoori, An artificial neural network model to solve the fuzzy shortest path problem, *Neural Process. Lett.*, **50** (2019), 1527–1548. https://doi.org/10.1007/s11063-018-9945-y

4. Z. L. Xu, W. Huang, J. S. Wang, A wave time-varying neural network for solving the time-varying shortest path problem, *Appl. Intell.*, **52** (2022), 8018–8037. https://doi.org/10.1007/s10489-021-02866-6

5. L. H. Lin, C. Z. Wu, L. Ma, A genetic algorithm for the fuzzy shortest path problem in a fuzzy network, *Complex. Intell. Syst.*, **7** (2020), 225–234. https://doi.org/10.1007/s40747-020-00195-8

6. Y. L. Yang, D. D. Yan, J. H. Zhao, Optimal path selection approach for fuzzy reliable shortest path problem, *J. Intell. Fuzzy Syst.*, **32** (2017), 197–205. https://doi.org/10.3233/JIFS-151393

7. L. Zhang, J. M. Liu, B. Yu, G. Chen, A dynamic traffic assignment method based on connected transportation system, *IEEE Access.*, **7** (2019), 65679–65692. https://doi.org/10.1109/ACCESS.2019.2915993

8. J. P. Liu, X. C. Kang, C. Dong, F. H. Zhang, Simulation of real-time path planning for large-scale transportation network using parallel computation, *Intell. Autom. Soft Comput.*, **25** (2019), 65–77. https://doi.org/10.31209/2018.100000013

9. Q. Wei, G. M. Hu, C. Shen, Y. F. Yin, A fast method for shortest path cover identification in large complex networks, *CMC-Comput. Mat. Contin.*, **63** (2020), 705–724.

10. G. Chen, G. H. Wen, X. H. Yu, Performance analysis of distributed short-path set based routing in complex networks, *IEEE Trans. Circuits Syst. II-Express Briefs.*, **66** (2019), 1426–1430. https://doi.org/10.1109/TCSII.2018.2882515

11. E. W. Dijkstra, A note on two problems in connexion with graphs, *Numer. Math.*, **1** (1959), 269–271. https://doi.org/10.1007/BF01386390

12. M. L. Fredman, R. E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, in *Proceedings of the 25th Annual Symposium on Foundations of Computer Science*, (1984), 338–346. https://doi.org/10.1109/SFCS.1984.715934

13. H. Arslan, M. Manguoglu, A hybrid single-source shortest path algorithm, *Turk. J. Electr. Eng. Comput. Sci.*, **27** (2019), 2636–2647. https://doi.org/10.3906/elk-1901-23

14. Sunita, D. Garg, Dynamizing Dijkstra: a solution to dynamic shortest path problem through retroactive priority queue, *J. King Saud Univ.-Comput. Inf. Sci.*, **33** (2021), 364–373. https://doi.org/10.1016/j.jksuci.2018.03.003

15. Z. Pawlak, Rough sets, *Int. J. Comput. Inf. Sci.*, **11** (1982), 341–356. https://doi.org/10.1007/BF01001956

16. G. Q. Wang, T. R. Li, P. F. Zhang, Q. Q. Huang, H. M. Chen, Double-local rough sets for efficient data mining, *Inf. Sci.*, **571** (2021), 475–498. https://doi.org/10.1016/j.ins.2021.05.007

17. X. B. Yang, S. C. Liang, H. L. Yu, S. Gao, Y. H. Qian, Pseudo-label neighborhood rough set: measures and attribute reductions, *Int. J. Approx. Reason.*, **105** (2019), 112–129. https://doi.org/10.1016/j.ijar.2018.11.010

18. W. W. Yan, J. Ba, T. H. Xu, H. L. Yu, J. L. Shi, B. Han, Beam-influenced attribute selector for producing stable reduct, *Mathematics*, **10** (2022), 553. https://doi.org/10.3390/math10040553

19. J. Yang, G. Y. Wang, Q. H. Zhang, Y. H. Chen, T. H. Xu, Optimal granularity selection based on cost-sensitive sequential three-way decisions with rough fuzzy sets, *Knowl.-Based Syst.*, **163** (2019), 131–144. https://doi.org/10.1016/j.knosys.2018.08.019

20. J. L. Yang, X. Y. Zhang, K. Y. Qin, Constructing robust fuzzy rough set models based on three-way decisions, *Cogn. Comput.*, **14** (2022), 1955–1977. https://doi.org/10.1007/s12559-021-09863-4

21. J. Qian, X. Han, Y. Yu, C. H. Liu, Multi-granularity decision-theoretic rough sets based on the fuzzy T-equivalence relation with new strategies, *J. Intell. Fuzzy Syst.*, **44** (2023), 5617–5631. https://doi.org/10.3233/JIFS-222910

22. J. K. Chen, J. J. Li, Y. J. Lin, Computing connected components of simple undirected graphs based on generalized rough sets, *Knowl.-Based Syst.*, **37** (2013), 80–85. https://doi.org/10.1016/j.knosys.2012.07.013

23. T. H. Xu, G. Y. Wang, Finding strongly connected components of simple digraphs based on generalized rough sets theory, *Knowl.-Based Syst.*, **149** (2018), 88–98. https://doi.org/10.1016/j.knosys.2018.02.038

24. T. H. Xu, G. Y. Wang, J. Yang, Finding strongly connected components of simple digraphs based on granulation strategy, *Int. J. Approx. Reason.*, **118** (2019), 64–78. https://doi.org/10.1016/j.ijar.2019.12.001

25. L. H. Guan, H. Wang, A heuristic approximation algorithm of minimum dominating set based on rough set theory, *J. Comb. Optim.*, **44** (2022), 752–769. https://doi.org/10.1007/s10878-021-00834-x

26. Q. Chen, T. H. Xu, J. J. Chen, Attribute reduction based on Lift and random sampling, *Symmetry*, **14** (2022), 1828. https://doi.org/10.3390/sym14091828

27. Z. C. Gong, Y. X. Liu, T. H. Xu, P. X. Wang, X. B. Yang, Unsupervised attribute reduction: improving effectiveness and efficiency, *Int. J. Mach. Learn. Cybern.*, **13** (2022), 3645–3662. https://doi.org/10.1007/s13042-022-01618-3

28. L. M. Fang, X. Y. Yun, C. C. Yin, W. P. Ding, L. Zhou, Z. Liu, et al., ANCS: automatic NXDomain classification system based on incremental fuzzy rough sets machine learning, *IEEE Trans. Fuzzy Syst.*, **29** (2021), 742–756. https://doi.org/10.1109/TFUZZ.2020.2965872

29. S. Vluymans, P. N. Mac, C. Cornelis, Y. Saeys, Weight selection strategies for ordered weighted average based fuzzy rough sets, *Inf. Sci.*, **501** (2019), 155–171. https://doi.org/10.1016/j.ins.2019.05.085

30. S. S. Zhang, K. Y. Liu, T. H. Xu, X. B. Yang, A. Zhang, A meta-heuristic feature selection algorithm combining random sampling accelerator and ensemble using data perturbation, *Appl. Intell.*, **53** (2023), 29781–29798. https://doi.org/10.1007/s10489-023-05123-0

31. T. Y. Yin, H. M. Chen, Z. Yuan, J. H. Wan, K. Y. Liu, S. J. Horng, et al., A robust multilabel feature selection approach based on graph structure considering fuzzy dependency and feature interaction, *IEEE Trans. Fuzzy Syst.*, **31** (2023), 4516–4528. https://doi.org/10.1109/TFUZZ.2023.3287193

32. T. Y. Yin, H. M. Chen, J. H. Wan, P. F. Zhang, S. J. Horng, T. R. Li, Exploiting feature multi-correlations for multilabel feature selection in robust multi-neighborhood fuzzy β covering space, *Inf. Fusion.*, **104** (2024), 102150. https://doi.org/10.1016/j.inffus.2023.102150

33. J. Bang-Jensen, G. Z. Gutin, *Digraphs: Theory, Algorithms and Applications*, 2nd edition, Springer Science & Business Media, London, 2009. https://doi.org/10.1007/978-1-84800-998-1

34. R. Tarjan, Depth-first search and linear graph algorithms, *SIAM J. Comput.*, **1** (1972), 146–160. https://doi.org/10.1109/SWAT.1971.10

35. Z. Gotthilf, M. Lewenstein, Improved algorithms for the k simple shortest paths and the replacement paths problems, *Inf. Process. Lett.*, **109** (2009), 352–355. https://doi.org/10.1016/j.ipl.2008.12.015

36. T. A. Davis, Y. F. Hu, University of florida sparse matrix collection, *ACM Trans. Math. Softw.*, **38** (2011), 734–747. https://doi.org/10.1145/2049662.2049663