South East Asian J. of Mathematics and Mathematical Sciences
Vol. 20, Proceedings (2022), pp. 177-188

ISSN (Online): 2582-0850

ISSN (Print): 0972-7752

ALGEBRAIC GRAPH JOIN OPERATION AND ITS APPLICATION

Gete Umbrey, Saifur Rahman* and Mahadevan Chandramouleeswaran**

Department of Mathematics,
Jawaharlal Nehru College, Pasighat,
East Siang - 791102, Arunachal Pradesh, INDIA

E-mail : gete.umbrey@rgu.ac.in

*Department of Mathematics,
Rajiv Gandhi University, Rono Hills,
Itanagar - 791112, Arunachal Pradesh, INDIA

E-mail : saifur.rahman@rgu.ac.in

**Department of Mathematics,
SBK College, Aruppukottai-626101, Tamil Nadu, INDIA

E-mail : moulee59@gmail.com

(**Received:** *Apr. 08, 2022* **Accepted:** *Aug. 20, 2022* **Published:** *Aug. 30, 2022)*

Special Issue
Proceedings of National Conference on
"Emerging Trends in Discrete Mathematics, NCETDM - 2022"

Abstract: In this article, we have investigated some algebraic structures of graphs and propose some non-conventional graph algorithms for dealing with network-like systems. The algebraic graph operation, namely *graph join* is used to find the optimal virtual networks and the shortest path with a sequence of vertices connecting source (say, least) and destination (say, greatest) vertices. We also represent graphs algebraically and propose related algorithm to simplify complicated network/decision problems using semiring axioms.

Keywords and Phrases: Graph Operations, Semiring, Graph Algorithms, Decision-Making, Joining Networks.

2020 Mathematics Subject Classification: 05C25, 05C76, 16Y60.

1. Introduction and Preliminaries

Graph theory has been a favorite platform for mathematicians and computer scientists for describing and analyzing the networks in a more abstract and general way. Some of the popularly known algorithms like the Dijkstra algorithm, Travelling Salesman Problem (TSP) are presented in the context of studying network analysis and routing problems [6]. A connection between interprocedural dataflow analysis and model checking of pushdown systems (PDSs) has been explored using semiring, and its related algebraic notions [14]. Working with graph operations is of great interest to a large scientific community. For instance, Azari et al. [2] present expressions for the eccentric connectivity coindex of several graph operations; Basavanagoud et al. [3] discuss the computation of the Hyper-Zagreb coindex of certain graph operations. Likewise, Gao et al. [7] present some exact expressions for the Hyper-Zagreb index of graph operations containing Cartesian product and join of n graphs, etc., and their consequent applications to chemical structures. The graph's *disjoint union* and the *join* operations are also used in investigating the neighborhood polynomial of the graphs [1]. In this article, we use three graph operations, namely graph *union* $\mathbin{\mathsf{U}}$[1], *join* ∇, and operation $\mathbf{\nabla}$ (which we call pseudo-graph *join*). We note that in [13], the structure $(S, \mathsf{U}, \nabla, (\emptyset, \emptyset))$ is a semiring, where S is the set of all simple undirected graphs and $(\emptyset, \emptyset)$ is an empty graph. This semiring is an example of a non-monosemiring in which the neutral element concerning addition and multiplication coincides. The empty graph $(\emptyset, \emptyset)$ is the neutral element concerning both operations. This semiring can be closely associated with max-plus semiring [8]. Many authors have studied various properties and usage of graph operations in algebraic settings. For instance, Mokhov [10] calls overlay $+$ and connect$\rightarrow$ to mean the graph *union* and *join*, respectively, which deals with directed graphs that satisfy various algebraic properties and are subsequently applied to working with graphs in Haskell. An approach to graph theory in an algebraic setting has also been found attempted by Bustamante [5], where the graph operation called the linking between two graphs G and G', which is akin to what we call *join* ∇ in this paper, and an algebraic structure called "Link Algebra" is analogous to the semiring $(S, \mathsf{U}, \nabla, (\emptyset, \emptyset))$. The study of graphs in algebraic settings has also been investigated by Umbrey and Rahman since 2020 [12, 16, 18]. Liang [9] formalizes dynamic programming algorithms using semiring and hypergraph frameworks, particularly the Viterbi-style fixed-order algorithms and the Dijkstra-style best-first algorithms.

[1]The operations $\cup$ is set union and the operations U is graph union.

Definition 1.1. *A semiring is an algebraic structure* $(S, \oplus, \otimes, \bar{0}, \bar{1})$ *equipped with addition* $\oplus$ *and multiplication* $\otimes$*, where* $\otimes$ *distributes over* $\oplus$*, and* $(S, \oplus, \bar{0})$ *and* $(S, \otimes)$ *are monoid and semigroup, respectively. Further,* $(S, \otimes, \bar{1})$ *is monoid if* $\bar{1}$ *exists. In general,* 0 *is multiplicatively absorbing element.*

Definition 1.2. *The term graph refers to a set of objects (called vertices or nodes) that are connected together. The connections between the vertices are called edges. A graph is said to be directed if the edges are directed by arrows, indicating that the relationship represented by the edge only applies from one vertex to the other, but not the other way around. On other hand, a graph whose edges are not directed is called an undirected graph. The graph* G *is denoted by* (V, E)*, where* V *is the set of vertices and* E *is the set of edges.*

Definition 1.3. *The union of the graphs* $G_1 = (V_1, E_1)$ *and* $G_2 = (V_2, E_2)$ *is defined as the graph* $G_1 \bigcup G_2 = (V_1 \cup V_2, E_1 \cup E_2)$*.*

Definition 1.4. *The join of two graphs* G *and* H *is a graph formed by connecting each vertex of* G *to each vertex of* H*, which we define and denote as* $G \nabla H = (V(G) \cup V(H), E(G) \cup E(H) \cup \{(u, v) : u \in V(G), v \in V(H)\} \setminus \{(a, a) : a \in V(G) \cap V(H)\})$*.*

Definition 1.5. *We call a graph to be an optimal graph if it represents the best possible decision of a problem.*

Definition 1.6. *The notation* $\boldsymbol{\nabla}$ *is a pseudo-graph join operation such that with two graphs* G *and* H*, we define,* $G \boldsymbol{\nabla} H = (V(G) \cup V(H), E(G) \cup E(H) \cup \{(u, v) : u \in V(G), v \in V(H);\ u \notin V(H), v \notin V(G)\})$*.*

An intuitive notion of this definition is that when we join two network systems, an individual network representing its original relations (or links) will be unaltered by adding a new relation between the two networks. Precisely, adding a new relation across two networks is possible only if there exists at least an uncommon node as such the uncommon nodes are linked by virtue of the definition of $\boldsymbol{\nabla}$.

2. Main Results

Property 2.1. *Let* G_1 *and* G_2 *have* m *and* n *vertices, respectively. Then the number of edges in* $G_1 \boldsymbol{\nabla} G_2$ *is given by*

$$|E(G_1 \boldsymbol{\nabla} G_2)| = \begin{cases} |E(G_1)| + |E(G_2)| + (m-k) \times (n-k), & \text{if } V(G_1) \cap V(G_2) \neq \emptyset; \\ |E(G_1)| + |E(G_2)| + m \times n, & \text{if } V(G_1) \cap V(G_2) = \emptyset, \end{cases}$$

where k *is the number of common vertices in* G_1 *and* G_2*.*

The following example illustrates its functional values in some realistic sense.

Example 2.1. Let the graphs G_1 and G_2 contain m and n vertices, respectively

having k common vertices. Let G_1 and G_2 represent two different networks of people, where people are denoted by vertices, and each edge represents a handshake. What is the maximum possible number of handshakes among the people of two networks combined?

The number of handshakes in the network will be maximum if the graph representing the network is a complete graph. Thus, the maximum number of handshakes in the network G_1 and G_2 are $\frac{m(m-1)}{2}$ and $\frac{n(n-1)}{2}$, respectively. Consequently, the maximum number of handshakes in G_1 and G_2 combined must be $\frac{m(m-1)}{2} + \frac{n(n-1)}{2} + (m-k) \times (n-1)$.

Theorem 2.1. *If S_G is the set of all subgraphs of a graph G, then $(S_G, \bigcup, \nabla)$ is a semiring if and only if G is a complete graph.*

Proof. Let us consider that $(S_G, \bigcup, \nabla)$ is a semiring. Then the operation ∇ must be closed in S_G, which implies that for any two graphs $G_1, G_2 \in S_G$, we have $G_1 \nabla G_2 \in S_G$ i.e., $G_1 \nabla G_2 \subseteq G$. This is true only if $E(G_1) \cup E(G_2) \cup \{(u,v) : u \in V(G_1), v \in V(G_2)\} \setminus \{(a,a) : a \in V(G_1) \cap V(G_2)\} \subseteq E(G)$ or, $\{(u,v) : u \in V(G_1), v \in V(G_2)\} \setminus \{(a,a) : a \in V(G_1) \cap V(G_2)\} \subseteq E(G)$. Equivalently, $V(G_1) \times V(G_2) \subseteq E(G)$ (ignoring the self loops). Since G_1 and G_2 are arbitrarily taken and so are $V(G_1)$ and $V(G_2)$, hence we can conclude that for every pair of vertices $v_i, v_j \in V(G)$, the edge (v_i, v_j) belongs to $E(G)$, which is true only if G is a complete graph.

The proof of the converse part is easy, hence omitted.

Remark 2.1. *In the above theorem, the graph G is absorbing element for both the operations of S_G.*

Corollary 2.1. *If (S_G, ∇) is magma and G is absorbing element in S_G, then $(S_G, \bigcup, \nabla)$ is a semiring.*

Proof. Since G is absorbing element, $G_i \nabla G = G = G \nabla G_i \; \forall \; G_i \in S_G$. This implies that $E(G_i) \cup E(G) \cup \{(u,v) : u \in V(G_i), v \in V(G)\} \setminus \{(a,a) : a \in V(G_i) \cap V(G)\} \subseteq E(G)$ or, $\{(u,v) : u \in V(G_i), v \in V(G)\} \setminus \{(a,a) : a \in V(G_i) \cap V(G)\} \subseteq E(G)$. Equivalently, $V(G_i) \times V(G) \subseteq E(G)$ (ignoring the self loops). Since G_i is arbitrarily taken and so is $V(G_i)$, hence we can conclude that every vertex of G is connected to every other vertex of itself. That is, G is a complete graph, hence by Theorem 2.1, $(S_G, \bigcup, \nabla)$ is a semiring.

Proposition 2.1. *Let S be the set of all simple undirected graphs. For all $G_1, G_2 \in S$, $G_1 \subseteq G_2$ if and only if there exists $G \in S$ such that $G_1 = G_2 \bigcup G$.*

Proof. Let $G_1 \subseteq G_2$ or, $(V_1, E_1) \subseteq (V_2, E_2)$, which implies that $V_1 \subseteq V_2$ and $E_1 \subseteq E_2$. Without loss of generality, we can write $V_1 = V_2 \bigcup V$ for some vertex set

V and $E_1 = E_2 \bigcup E$ for some edge set E. Consequently, $(V_1, E_1) = (V_2 \cup V, E_2 \cup E) = (V_2, E_2) \bigcup (V, E)$ i.e., $G_1 = G_2 \bigcup G$. The converse part is obvious, hence omitted.

In view of the Proposition 2.1, the semiring $(S, \bigcup, \nabla, (\emptyset, \emptyset))$ can be closely associated with path algebra [4], which is an important tool for graph (path) algorithms, especially when it comes to computer science.

In the sequel, we will introduce some graph algorithms in algebraic settings which can be used in non-conventional decision-making problems. For a given set of vertices of a graph, the best possible graph we can have to have the optimal path between the given source and destination vertices seems to be a complicated problem. For instance, with a given set of vertices, viz., $V = \{2, 6, 7, 4, 10\}$, what could be the best possible graph containing the optimal path between 2 and 10? Such open-ended questions may sound no practical sense, but they can also be great questions under specific contexts. The following approach is an attempt to address such problems.

3. Algorithms and Examples

Rahman [11] in 2016 have shown that any two elements belonging to the same sets equipped with a relation exhibit a certain degree of relationship between them in a practical sense even if their relationship is not visible crisply or discretely. He has introduced this notion by defining membership and non-membership functions to determine the degrees of divisibility and non-divisibility for each natural number of a set of all-natural numbers less than or equal to 20. For example, the number 8 is neither a factor of 12 nor coprime. So, they have some common factors implying a certain degree of divisibility. Hence 8 and 12 are related by weak divisibility. These motivate us to extend such an idea to the graph theory, where a graph represents a vague decision problem. In this context, we will consider that any pair of vertices of a connected graph G (where edges represent a relationship between the vertices) can always be connected by a direct edge. Suppose the vertices v_1 and v_2 are adjacent in G. In that case, the edge (v_1, v_2) represents a strong relation. At the same time, if v_1 and v_2 are non-adjacent but connected by a path, then these vertices will be connected by a dotted edge showing a certain degree of relationship between the two vertices. All these can be done with the help of algebraic graph/vertex join operations.

The join of two vertices v_1 and v_2 of a graph G is an edge (v_1, v_2) denoted by $v_1 \nabla v_2$. The join of the vertices $v_1, v_2, ..., v_n$ is denoted by $v_1 \nabla v_2 \nabla ... \nabla v_n$.

In this section, we will discuss some non-conventional algorithms based on algebraic join operations ∇. Let the labels of vertices are from an ordered idempotent semiring $(R, \oplus, \otimes, \leq_R)$ such that for every u and v being vertices of the graphs in S implies that $u \oplus v \in R$ and $u \otimes v \in R$, such graphs are called semiring-valued

graphs [14]. Choosing the algebraic expressions as either $u \oplus v \in R$ or $u \otimes v \in R$ according to the context of applications, we propose the following Algorithms.

Algorithm 3.1. Let s and t be the source and the destination vertices of a semiring valued graph G, where s is the least element, and t is the greatest element. All the vertices of G are from an ordered idempotent semiring R.

1. Starting from s, we first visit the least vertex v_1 (say), which is adjacent to s. Mark the edge (s, v_1) by red color, showing that it is traversed.

2. Starting from v_1, we first visit the least unvisited vertex v_2 (say). If v_1 and v_2 are adjacent in G, then mark the edge (v_1, v_2) by red color. If v_1 and v_2 are non-adjacent in G, then join them by a dotted line.

3. Taking v_2 as the starting point, we will repeat the process in Step 2. This process continues up till all the vertices which are adjacent to the vertex t are visited.

4. Join v_i and t with the red color, where (v_i, t) is an edge of G and $v_i < t$ such that there exist no edge (u, t) in G such that $v_i < u < t$.

5. Merge the vertices connected by dotted line using addition (max) operation or multiplication (min) operations of R. Thus, we will get a reduced graph with optimal path indicated by red color.

We will consider the weight of an edge of G as the max or min of end vertices for the above algorithm.

If we use the multiplication (min) operation of R in Step 5 of the above algorithm, then we will get the shortest path with minimum path weight. On the other hand, if we use the addition (max) operation of R instead of the multiplication (min) operation, we will get the shortest path with maximum path weight. Sometimes, the maximum and the minimum path weights for the required shortest paths may coincide. She shortest paths, as obtained in both cases, will be chains. That is, the set of vertices $s, v_1, v_2, ..., t$ on the the path connecting s and t will be in the sequence $s < v_1 < v_2 < ... < t$.

We illustrate the following example with detailed steps.

Example 3.1. Let us consider the following graph G with the vertex set $R = \{1, 2, 3, 4, 5, , 6, 7\}$, then $(R, \max, \min, \leq_R)$ is an idempotent semiring. In the following graph, we let $s = 1$ and $t = 7$.

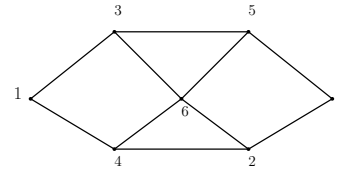

Figure 1: G

Following the Steps $1-4$ of the above algorithm, we have transformed the graph G into the following graph G'.

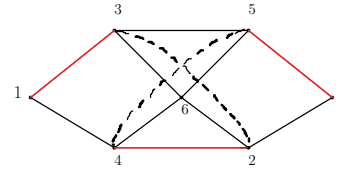

Figure 2: G'

Detailed steps for obtaining Figure 2 from Figure 1:

1. The least vertex adjacent to source vertex 1 is 3, so visit 3 from 1 and mark the edge $(1, 3)$ as visited by coloring it red.

2. From 3, we will visit the smallest unvisited vertex. In this case, 2 is the smallest vertex which unvisited, so visit 2 from 3 by drawing a dotted edge between the vertices. Note that the dotted edge is used when the end vertices are non-adjacent in G.

3. From 2, we visit 4, the smallest unvisited vertex, marking the edge $(2, 4)$ red.

4. Similarly, from 4, we visit 5 using the dotted edge $(4, 5)$.

5. Since the vertices 2 and 5 being the only adjacent to the destination vertex 7 have been visited, we stop the process by visiting 7 from 5, and mark the edge $(5, 7)$ red.

On applying the max operation in Step 5 of the above algorithm, we have the following reduced graph with the shortest path having maximum weight denoted by red color.

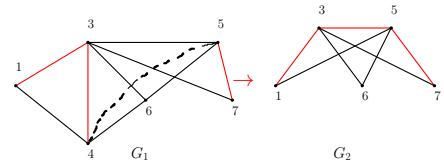

Figure 3

Detailed steps for obtaining Figure 3 from Figure 2:

1. Merge the vertices connected by dotted edges. Due to the associativity of the max operation, the sequence of merging vertices doesn't alter the algorithm. That is, in this case, any one of the pairs $(2, 3)$ or $(4, 5)$ can be merged first. Here, pair $(2, 3)$ is merged first to get the graph G_1.

2. Since $\max\{2, 3\} = 3$, the merged vertex is 3. Now, all the edges incident on 2 will also be incident on 3. For instance, the vertices 6 and 7 are adjacent to 2; hence they will now be adjacent to 3 (note that 6 is already adjacent to 3 in G). Similarly, 4 is adjacent to 2 and the edge $(2, 4)$ is marked red; hence 4 is adjacent to 3 and marked the edge $(3, 4)$ red in G_1. Note that if there exist multiple edges between any two vertices v_i and v_j such that at least one of them is red, then the edge (v_i, v_j) are marked red.

3. Similarly, the vertices 4 and 5 are merged. Thus, we get the required reduced graph G_2 in which the shortest path with maximum weight is denoted by red color. Taking the edge weight as the minimum of end vertices, we get the weight of the shortest path/chain as $1 + 3 + 5 = 9$.

Similarly, on applying the min operation in Step 5 of the above algorithm, we have the following reduced graph with the shortest path having minimum weight denoted by red color.

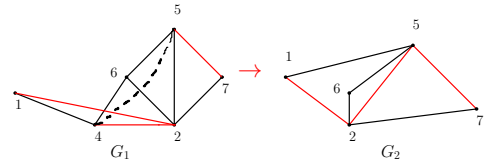

Figure 4

Again, taking the edge weight as the minimum of end vertices, the weight of the required path/chain denoted by red color is $1+2+5 = 8$. Note that by the shortest path, we mean the path with the minimum number of edges.

4. Algebraic Simplification of Complex Network Problems

Zykov [19] first talked about expressing a graph algebraically, calling it *linear complexes*. We will express a complicated graph/network algebraically, where the approach is different from previously discussed methods. Here, each vertex of a graph will be considered as a graph again, called *vertex graph*, and study of such networks have been initiated by Umbrey and Rahman in 2020 [17]. For the sake of simplicity, we will consider a particular network problem G in which every pair of *vertex graphs* is related under subgraph relation $\subseteq$. For such networks, we will denote *vertex graphs* of G by a sequence of numbers, implying that a *vertex graph* labeled with a smaller number is a subgraph of the *vertex graph* with a greater number. For example, if G_1 and G_2 are any two *vertex graph* of G such that $G_1 \subseteq G_2$, then G_1 and G_2 can be replaced by a sequence of numbers, namely 1 and 2, respectively. That means $G_1 \cup G_2 = G_2$, which is equivalent to $1 \oplus 2 = max\{1, 2\} = 2$. Henceforth, we will use $\oplus$ instead of $\cup$ for the ease of notation while simplifying an algebraic graph. We propose the following algorithm.

Algorithm 4.1. We will consider a problem network G of *vertex graphs*. That is, each vertex of G is a graph again. Reduced the graph into a simpler form called simplified network.

1. Express $G = \bigoplus_{i,j}^{m,n}(v_i \nabla v_j)$, where v_i and v_j are *vertex graphs* of G.
2. Simplify the graph G algebraically to obtain its corresponding simplified network.

Example 4.1. Consider the following network problem.

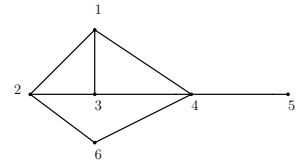

Figure 5: G

$$\begin{aligned} G &= (1\nabla 2) \oplus (1\nabla 3) \oplus (1\nabla 4) \oplus (2\nabla 3) \oplus (2\nabla 6) \oplus (3\nabla 4) \oplus (4\nabla 5) \oplus (4\nabla 6) \\ &= [1\nabla(2 \oplus 3 \oplus 4)] \oplus [2\nabla(3 \oplus 6)] \oplus [4\nabla(3 \oplus 5 \oplus 6)] \\ &= (1\nabla 4) \oplus (2\nabla 6) \oplus (4\nabla 6) \end{aligned}$$

$$= [(1\nabla 4) \oplus (4\nabla 6)] \oplus [(2\nabla 6) \oplus (4\nabla 6)] \text{ (idempotency)}$$
$$= 4\nabla(1 \oplus 6) \oplus 6\nabla(2 \oplus 4)$$
$$= (4\nabla 6) \oplus (6\nabla 4) = 4\nabla 6, \text{ which is in simplified form.}$$

This algorithm will be elegant in dealing with complicated networks because the networks/graphs we represent here seem to be simple, but they are not as simple as they appear because each vertex is again a graph. In the physical world, such types of graphs are very realistic. Such graphs could be social networks (like Facebook), computer networks, a complicated decision network problem, etc. Moreover, such a simplified subgraph must represent the original graph. That is, all the properties of the original graph will be reflected in the simplified subgraph. The chromatic number, energy, etc., of the original and simplified graphs, should be identical. But in the above examples, we know only a part of the original graphs; we don't know what the *vertex graphs* 1, 2,, etc. are, but we only know that they are all related by the subgraph relation. We even don't know how the edges of the original graphs are defined. Once knowing what the given networks/graphs are, we can study all their properties in terms its reduced simplified subgraphs.

Let us present a concrete example of this type of network.

Example 4.2. Let us consider $V = \{1, 2, 3, 4\}$; $[1] = \{0\}$; $[2] = \{0, 1\}$; $[3] = \{0, 1, 2\}$; $[4] = \{0, 1, 2, 3\}$ and $[5] = \{0, 1, 2, 3, 4\}$. On the set V, we define a set $E = \{(a, b) : a \text{ divides } b \text{ or } b \text{ divides } a \ \forall \ a, b \in V; a \neq b\}$.
Now, we have the following graphs: $G_1 = ([1], \emptyset)$; $G_2 = ([2], \{(0, 1)\})$; $G_3 = ([3], \{(0, 1), (0, 2), (1, 2)\})$; $G_4 = ([4], \{(0, 1), (0, 2), (0, 3), (1, 2), (1, 3)\})$ and $G_5 = ([5], \{(0, 1), (0, 2), (0, 3), (0, 4), (1, 2), (1, 3), (1, 4), (2, 4)\})$.

Now, we consider the following network G with the set of *vertex graphs* $V(G) = \{G_1, G_2, G_3, G_4, G_5\}$.

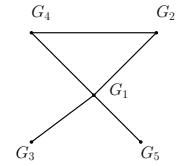

Figure 6: G

The edge set of G is defined as
$E(G) = \{(G_i, G_j) : \ | V(G_i) |$ divides $| V(G_j) |$ or $| V(G_j) |$ divides $| V(G_i) |; | V(G_i) | \neq | V(G_j) | \}$. Next, we have

$$G = (G_1 \nabla G_2) \oplus (G_1 \nabla G_3) \oplus (G_1 \nabla G_4) \oplus (G_1 \nabla G_5) \oplus (G_2 \nabla G_4) \qquad (4.1)$$

$$= (G_2 \nabla G_4) \oplus (G_1 \nabla G_5). \tag{4.2}$$

Here, we can easily verify that the Expressions (4.1) and (4.2) of the right side of the above equation give exactly the same graph, which is equal to G_5.

5. Conclusion

This work is a new approach towards the fusion of graph and algebra to deal with some real-time problems that are preferably non-conventional bit vaguer due to various interdependent factors in a decision problem. It is very convenient to represent a decision problem by a graph. The use of algebraic axioms in graphs (or, set of graphs) will be a great potential to simplify and generalize complicated or vague decision problems. The Algorithms have potential implementations in networking issues of social analysis, internet and computer, etc. Developing software systems\programs for such graph algorithms will provide practical computational tools to make it easier to study complicated graph-theoretic problems like communication network design, traffic optimization or network visualization, and other decision-making problems. However, such development is a challenge for future research.

References

[1] Alipour, M. and Tittmann, P., Graph Operations and Neighborhood Polynomials, Discussiones Mathematicae Graph Theory, 41 (2021), 697-711.

[2] Azari, M., Eccentric Connectivity Coindex under Graph Operations, Journal of Applied Mathematics and Computing, 62 (2020), 23-35.

[3] Basavanagoud, B. and Patil, S., A note on Hyper-Zagreb coindex of Graph Operations, Journal of Applied Mathematics and Computing, 53 (2017), 647-655.

[4] Boulmakoul, A., Generalized path-finding algorithms on semirings and fuzzy shortest path problem, Journal of Computational and Applied Mathematics, 162 (2004), 263-272.

[5] Bustamante, A., Link Algebra: A New Approach to Graph Theory (2011), CoRR, abs/1103.3539, 2011.

[6] Flitter, H. and Grossmann, T., Accessibility Network Analysis, Geographic Information Technology Training Alliance (GITTA), 2016.

[7] Gao, W., Jamil, M. K. and Farahani, M. R., The Hyper-Zagreb index and some graph operations, Journal of Applied Mathematics and Computing, 54 (2017), 263-275.

[8] Golan, J. S., Some Recent Applications of Semiring Theory, International Conference of Algebra in memory Kostia Beidar, CiteSeerX, 2005.

[9] Huang, L., Dynamic Programming Algorithms in Semiring and Hypergraph Frameworks, CiteSeerX, 2006.

[10] Mokhov, A., Algebraic Graphs with Class(Functional Pearl), In: Proceedings of 10^{th} ACM SIGPLAN International Haskell Symposium, 2017, *https* : *//doi.org*/10.1145/3122955.3122956, (2017).

[11] Rahman, S., On cuts of Atanassov's intuitionistic fuzzy sets with respect to fuzzy connectives, Information Sciences, 340-341 (2016), 262-278.

[12] Rahman, S. and Umbrey, G., On Some properties of Semirings of Graphs, Southeast Asian Bulletin of Mathematics, 46 (4) (2022), 553-563.

[13] Rahman, S. and Umbrey, G., Semirings of Graphs: Homomorphisms and Applications in Network Problems, Proyeccines Journal of Mathematics, 41 (6), (2022), 1273-1296.

[14] Rajkumar, M., Jeyalakshmi, S. and Chanmdramouleeswaran, M., Semiring-Valued Graphs, International Journal of Mathematical Sciences and Engineering Applications, 9 (III) (2015), 141-152.

[15] Reps, T., Schwoon, S., Jhaa, S. and Melski, D., Weighted Pushdown Systems and their Application to Interprocedural Dataflow Analysis, Science of Computer Programming, Elsevier, 58 (2005), 206-263.

[16] Umbrey, G. and Rahman, S., Journal of Advance Research in Dynamical and Control Systems, 12 (Special Issue-2) (2020), 35-45.

[17] Umbrey, G. and Rahman, S., Application of Graph Semirings in Decision Networks, Mathematical Forum, 28 (1) (2020), 40-51.

[18] Umbrey, G. and Rahman, S., Determining Paths Energy of a complex network, Advances in Mathematics: Scientific Journal, 9 (10) (2020), 8761-8770.

[19] Zykov, A. A., On Some Properties of Linear Complexes, Matematicheskii Sbornik, 24 (66) (1949), 163-188.