UDC 519.17:004.421.2

LINEAR SEARCH ALGORITHM FOR STRONGLY CONNECTED COMPONENTS IN DIRECTED GRAPHS BASED ON THE ANALYSIS OF THE MATRIXS TRANSITIVE CLOSURE OF BINARY RELATIONS

Batsamut V. M. – Dr. Sc., Professor, Deputy Head of the Educational and Scientific Institute for State Security for Scientific Work of the National Academy of the National Guard of Ukraine, Kharkiv, Ukraine. ROR: https://ror.org/02p49q920. ORCID: https://orcid.org/0000-0003-2182-6891.

Bashkatov Y. H. – PhD, Associate Professor, Deputy Head of the Educational and Scientific Institute for State Security for Educational Work of the National Academy of the National Guard of Ukraine, Kharkiv, Ukraine. ROR: https://ror.org/02p49q920. ORCID: https://orcid.org/0000-0002-6078-886X.

Morkvin D. A. – PhD, Senior Researcher of the Research Laboratory of the Educational and Scientific Institute of Professional Education of the National Academy of the National Guard of Ukraine, Kharkiv, Ukraine. ROR: https://ror.org/02p49q920. ORCID: https://orcid.org/0000-0002-3651-6805.

Tolstonosov D. Yu. – PhD, Associate Professor, Head of the Department of Combat and Logistics Security, Faculty of Service and Combat Activities of the National Guard of Ukraine, Kyiv Institute of the National Guard of Ukraine, Kyiv, Ukraine. ROR: https://ror.org/01q3vqr85. ORCID: https://orcid.org/0000-0001-5181-7668.

Sakhnevych B. V. – PhD, Deputy Head of the Department of Combat and Logistics Security, Faculty of Service and Combat Activities of the National Guard of Ukraine, Kyiv Institute of the National Guard of Ukraine, Kyiv, Ukraine. ROR: https://ror.org/01q3vqr85. ORCID: https://orcid.org/0009-0003-7263-5689.

Solodun Y. M. – PhD, Associate Professor of the Department of Combat and Logistics Security, Faculty of Service and Combat Activities of the National Guard of Ukraine, Kyiv Institute of the National Guard of Ukraine, Kyiv, Ukraine. ROR: https://ror.org/01q3vqr85. ORCID: https://orcid.org/0009-0001-8390-9249.

ABSTRACT

Context. The relevance of the article is due to the need for further development of simple algorithms for analyzing the structural connectivity of network objects, in reducing the computational complexity and increasing the functional capabilities of such algorithms. The linear algorithm proposed in the article can be applied: in urban and rural transport route planning systems; in solving various types of problems related to carrying out restoration work on network objects after their destruction and the loss of certain connections between the elements of such objects (power supply systems, communication systems, transport networks, etc.); in developing options for effectively destroying key objects of the enemy's critical infrastructure; in analyzing social networks for grouping individuals into groups by interest, etc.

Objective. The goal of the work is to develop a linear algorithm for searching for strongly connected components in the structure of network objects, which is resistant to changes in the structure of objects and has a polynomial computational complexity.

Method. The physical object to be studied is modeled by a connected directed graph. To determine the structure of the model graph components of strong connectivity, the idea of analyzing the presence of direct and reverse transitive closures between all pairs of vertices of the model graph was applied. Since there is a route (or arc) connecting them between all pairs of vertices included in any component of strong connectivity, then in the matrix of transitive closures of binary relations of such a graph there will always be unit blocks (transitivity compactions). Individual elements of these blocks in the matrix of transitive closures are scattered in a certain way. By analyzing the matrix for the presence of identical rows and identical columns, it is possible to establish their belonging to certain unit blocks. After such analysis and grouping the corresponding rows and columns into unit blocks, it is possible to determine the number of components of strong connectivity, as well as the index (vertex) composition of each component.

To construct a matrix of transitive closures of binary relations, any of the known algorithms can be used, as discussed in the article. Therefore, the linear algorithm proposed in this article is conventionally divided into two main stages: the first is the search for transitive closures of binary relations of a model directed graph; the second is the determination of the matrix of inverse reachability of the graph, the determination of the matrix of mutual reachability, and the search for unit blocks in the latter by lining them up on the main diagonal of the matrix.

Results. 1) A theoretical basis for analyzing the matrix of transitive closures of binary relations of a model directed graph has been developed in order to determine the quantitative and index composition of strongly connected components in its composition; 2) In terms of graph theory, the search for strongly connected components in the structure of a network object is formalized; 3) The developed algorithm was verified for its ability to determine strongly connected components in the structure of a directed graph.

Conclusions. Theoretical studies and a number of experiments confirm the possibility of using the proposed algorithm in the tasks of structural analysis of network objects. Since the well-known provisions of graph theory were used in the theoretical studies, which are interpreted unambiguously in this theory, the absence of any probabilistic processes allows us to consider the developed algorithm accurate. Polynomial estimates of its computational complexity allow it to be used in real-time scale.

KEYWORDS: network object, object damage, directed graph, graph vertex, transitive closure, strongly connected component, matrix, main diagonal of the matrix, unit block, algorithm.

DOI 10.15588/1607-3274-2026-2-13

ABBREVIATIONS

DFS is a depth-first search;
SCC is an strongly connected component;
TC is a transitive closure.

NOMENCLATURE

G is an directed graph modeling an object;
V is a set of vertices of the model graph G;
E is a set of edges of a model graph G;
n is a dimension of the modeling graph;
G_k is a set of strong connectivity component;
k is a number of strongly connected components in the graph G;
M_{ij} is a route connecting two vertices i and j;
S_G is a connectivity matrix of the model graph G;
S_G^* is an auxiliary matrix;
R_G is a matrix of transitive closures of the graph G;
$m1$, $m2$ are first and second traversal of the matrix when determining the matrix R_G;
Q_G is an inverse reachability matrix of the graph G;
$(R_G \otimes Q_G)$ is a block-diagonal matrix of the graph G;
O is an estimation of the computational complexity of the algorithm.

INTRODUCTION

Directed graphs are one of the key mathematical tools widely used in modern information technologies, analysis of complex systems and modeling of interdependencies in large data sets. In many problems related to the study of the structure of such graphs, the concept of strongly connected components (SCC) is of particular importance – a subgraph in which each vertex is reachable from any other. Identification of SCC allows us to understand the deep structure of the network, to isolate interdependent modules and to optimize further analytical processes, which makes this problem fundamental for graph theory and related fields [1].

The relevance of research into methods for finding strongly connected components is due to the continuous growth of data volumes and the development of large network systems, such as social platforms, information networks, logistics and transport systems, as well as engineering software structures. The graphs used in these areas can contain millions or even billions of vertices and edges, which places new demands on the computational efficiency of such algorithms. In such conditions, classical algorithmic approaches sometimes need to be adapted, modified, or completely rethought to work in scalable, distributed, or streaming environments.

Strongly connected components search plays an important role in many application problems. In compilers, it is the basis for analyzing cyclic dependencies between functions or modules [2]. In database theory, SCC allows us to identify groups of interdependent tables or transactions that share common constraints. In transportation networks, SCC analysis helps us to identify deeply connected routes or nodes that form cyclic interactions. In cybersecurity, analyzing highly connected substructures in network flows allows us to identify anomalies or groups of nodes that interact in a non-standard way [3]. In social networks, SCC search serves as a tool for identifying stable communities or structures in which interactions are two-way and intense.

Classical algorithms for finding strongly connected components, such as the Kosaraju [1], Tarjan [4], and Gabov [5] algorithms, provide linear time complexity and demonstrate high efficiency in a wide range of applications. However, with the growth of the scale of graphs, the emergence of streaming data, and the need to process dynamic structures, there is a need to improve existing methods. Modern information systems require not only high computational performance, but also the ability of algorithms to adapt to changes in the graph, work in real time, and effectively distribute the load in multiprocessor and cluster environments.

As the amount of data increases, the problem of efficiently processing graphs that are constantly changing becomes significant. In dynamic graphs, the addition or removal of edges can significantly affect the structure of strong connectivity, which makes it difficult to apply classical algorithms designed for static structures. In such cases, researchers create algorithms that allow updating the SCC without a complete recalculation [3]. Such approaches open up new possibilities in processing streaming data, network logs, and other real-time systems.

At the same time, a significant number of large graphs have the property of high sparsity and uneven degree distribution of such graphs, which is typical for social networks, web graphs and bioinformatics structures. Such graphs require adapted algorithms that can work effectively in conditions where some vertices have extremely high degree, and the majority have low. This creates additional difficulties for classical algorithms that rely on the DFS procedure, and is compensated by specialized methods for optimizing memory access and distributing computations.

Despite the significant development of the theory of algorithms for finding strongly connected components, a number of problems remain open. In particular, further work is required to optimize memory usage, reduce the number of auxiliary operations, increase the robustness to errors in input data, and ensure the possibility of working in partial graph update mode. Algorithms capable of working in real time without loss of accuracy, which is critical for monitoring systems, cybersecurity, and streaming analytics, are especially relevant.

In this context, the creation of a new or improved SCC search algorithm is an important scientific task. Such an algorithm should combine theoretical optimality with practical efficiency, provide stable operation on large, dynamic or uneven graphs, and have the ability to be effectively integrated into modern data analysis systems.

DOI 10.15588/1607-3274-2026-2-13

The aim of this work is to develop a new algorithm for searching for strongly connected components, which will provide increased performance, resistance to structural features of graphs and the ability to work under conditions of limited resources. It is planned to analyze existing methods, build a mathematical model of the new approach, estimate its computational complexity and conduct an experimental study on a specific graph.

The object of study is to determine the strong connectivity of network objects.

The subject of study is bidirectional connectivity between nodes of a network object**.**

The purpose of the work is to develop a linear algorithm for searching for strongly connected components in the structure of network objects, which is resistant to changes in the structure of objects and has a polynomial computational complexity.

1 PROBLEM STATEMENT

Problems of structural analysis of network objects are usually formalized and solved using models and methods of graph theory [6], since graphs best model the structure of such objects. So, we will model the structure of a network object by some directed graph $G=(V,E)$, where $V=\{i\}_{i\in I}$ – a set of graph vertex numbers that model key nodes in the object structure; $E=\{(i,j)\}_{(i,j)\in I^2}$ – a set of graph arcs that model linear elements (connections) between different nodes of an object; I – the set of graph vertex numbers. Each arc from the set E is characterized only by the fact of its existence.

Taking into account the peculiarities of establishing the fact of the presence of strongly connected components in the structure of graph G and determining their index composition, the formulation of the corresponding problem in general form will have the following form. In the structure of the model graph G, we need to find its substructures $G_1, G_2,..., G_k$, where $k=\overline{2,n}, n\geq 2$, such as $\{i\}\in G_1 \bigcap \{i\}\in G_2,..., \bigcap \{i\}\in G_k=\varnothing$. For $\forall G_k$, determine $\{i\}_{i\in I}\in G_k$. Also determine the parameter k.

2 REVIEW OF THE LITERATURE

One of the first linear algorithms for finding SCCs is the Kosaraju algorithm, described in the work of Aho, Hopcroft and Ullman [2]. The method consists of two consecutive depth-first traversals (DFS) of the graph: the first traversal forms the order of completion of processing of vertices, while the second one – on the transposed graph – allows to isolate strongly connected components. The asymptotic complexity of the algorithm is $O(V+E)$. Although this algorithm is simple and efficient, its disadvantage is the need to construct a transposed graph, which increases memory usage. In the case where the initial graph is given by an adjacency matrix, the computational complexity increases to $O(n^2)$.

Tarjan's algorithm [2] significantly improves on Kosaraj's approach by performing only one DFS. Using some indicators and a stack, it is possible to determine when a search subtree forms an SCC. The algorithm has the same computational complexity $O(V+E)$, but is more memory-efficient and runs faster in practice. Due to its simplicity of implementation, it has become a standard in most computer applications, including compilers and dependency analysis systems.

Gabov's algorithm [3] is also based on DFS, but uses two stack structures to more accurately identify the desired strongly connected components. It is also linear in computational complexity, but shows improved performance on certain types of graphs. This makes it useful for applications where the specificity of the graph structure matters, such as network analysis and routing problems.

All three classical algorithms – Kosaraju, Tarjan, and Gabov – are linear in computational complexity, making them suitable for working with high-dimensional graphs. However, Tarjan and Gabov algorithms are more memory-efficient and efficient in practical implementations, as they do not require storing the transposed graph.

Current literature in this area and experimental studies [4, 7–10] indicate that Tarjan's algorithm often exhibits the most stable performance in real-world applications. Gabov's algorithm also shows high efficiency due to optimized stack management, making it competitive in practice.

However, the problem of developing such algorithms does not lose its relevance today, and in the field of traffic planning, social network analysis, and military affairs, it acquires new, more stringent requirements. Our article is devoted to solving this problem.

3 MATERIALS AND METHODS

For a clear understanding of the content of the proposed algorithm, the internal effects on which it is built, at the beginning of the article we will consider the essence of such concepts as: the component of strong connectivity of an arbitrary directed graph G and the transitive closure of the vertices of such a graph.

The concept of SCC reflects the fact that between any pair of vertices v_i and v_j belonging to this SCC, there is a route (a sequential set of arcs) that connects these vertices. It should be noted that such a connection exists both in direction $v_i \to v_j$ and in the opposite direction – $v_j \to v_i$, this is a mandatory condition [11]. It should also be said here that $M_{ij} \bigcap M_{ji}=\varnothing$, where M_{ij} and M_{ji} – are the corresponding bidirectional routes.

In the case where such a bidirectional connection between a pair of vertices is absent, vertices v_i and v_j belong to different SCCs. This fact indicates that there are several SCCs (at least two) in the graph G, see Fig. 1.

DOI 10.15588/1607-3274-2026-2-13

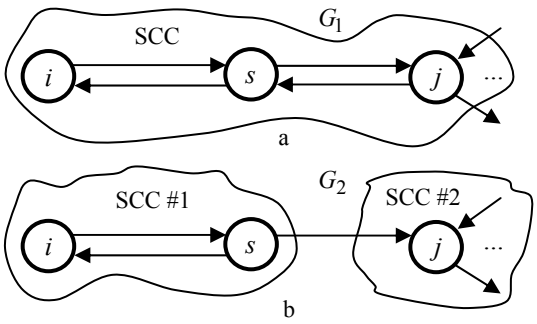

Figure 1 – SCC in graphs G_1 and G_2: a – vertices v_i and v_j belong to the same SCC; b – vertices v_i and v_j belong to different SCCs (since route $v_j \to v_i$ is missing from G_2)

A fundamental concept of graph theory is the notion of transitive closure (TC) of binary relations. Binary relations are the connectivity relations between all possible pairs of vertices of the model graph G.

If vertices v_i and v_s are connected by at least one route M_{is}, and vertices v_s and v_j are also connected by some route M_{sj}, then vertices v_i and v_j are considered transitively closed, since there is a route $M_{ij} = M_{is} \bigcup M_{sj}$ connecting them, Fig. 2.

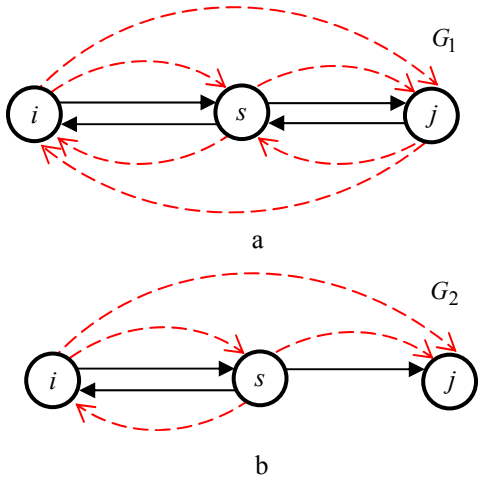

Figure 2 – Transitive closures of binary relations on directed graphs G_1 and G_2 (marked with red dotted lines): a – transitive closures are symmetric; b – transitive closures are asymmetric

It should be noted here that in this case, vertices v_i and v_j will be transitively closed precisely in direction $v_i \to v_j$, which does not at all ensure their transitivity in direction $v_j \to v_i$. This requires the existence of at least one route M_{ji} in the model graph G.

As can be seen from Fig. 2b, the graph G_2 has asymmetric transitive closures between vertices with respect to directions $v_i \to v_j$ and $v_j \to v_i$, since there is no arc (j,s) in the graph G_2. This fact leads to the division of the graph G_2 into two SCCs, see Fig. 1b.

The above provisions suggest the idea of conducting a special analysis of the transitive closure matrix (R_G) of the model graph G in order to determine the set of SCCs in its composition, as well as the index composition of each of them.

In general, the elements of the matrix $R_G = [r_{ij}]_{n \times n}$ take one of the following values [11]:

$$r_{ij} = \begin{cases} 1, \exists\, v_i \to v_j \\ 0, \neg\exists\, v_i \to v_j \end{cases}, i, j = \overline{1,n}. \quad (1)$$

Taking into account expression (1), we formulate the following lemmas.

Lemma 1. If there is one SCC in the structure of the model graph G, then its transitive closure matrix R_G will be a unit square matrix, ($\forall r_{ij} = 1$).

Lemma 2. If there are several (k) SCCs in the structure of a model graph G, its transitive closure matrix R_G is not a unit matrix, i.e. ($\neg\forall r_{ij} = 1$).

Therefore, in the case of the existence of several (k) SCCs in the structure of the model graph and taking into account the fact that in any SCC all vertices are transitively closed, the unit blocks corresponding to them will be symmetric with respect to the main diagonal of the matrix. Hence, the idea comes to mind about the possibility of constructing a block-diagonal matrix on the main diagonal of which k unit blocks will be sequentially located. The index composition of each block will indicate the composition of the vertices that will be included in the corresponding SCC. The remaining elements of such a matrix will be equal to zero, which will indicate the absence of transitive closure between the corresponding pairs of vertices of the graph G.

Therefore, the linear logic of the algorithm being developed is represented by a set of matrices that need to be sequentially determined, namely:

– at stage I – using the adjacency matrix (S_G) of the model graph G, determine its transitive closure matrix R_G;

– at stage II – from the matrix R_G, determine the inverse reachability matrix – Q_G, and then the block-diagonal matrix ($R_G \otimes Q_G$).

Formally, this order can be written as a linear expression:

$$S_G \to R_G \to Q_G \to (R_G \otimes Q_G). \quad (2)$$

DOI 10.15588/1607-3274-2026-2-13

At the first stage, to determine the matrix R_G, one can apply such well-known combinatorial algorithms as the Warshall-Floyd algorithm [12–14], Bellman [15] or Shimbell's algorithm [16]. Instead, a more efficient algorithm for constructing transitive closures of network objects is the algorithm presented in [17], since it, due to the lack of functional redundancy, has lower computational complexity than the algorithms mentioned above.

This algorithm is based on the disjunctive embedding of the rows of the matrix S_G of the model graph G in the case where some $s_{ij} \neq 0$, and a double traversal of the rows of the current matrix, namely: $S_G \to R_G^{[1]} \to R_G^{[2]} = R_G$.

The operation of disjunctive nesting of the terms of the current matrix is written as the following two (by the number of matrix traversals) expressions [17]:

$$if \quad s_{ij} \neq 0 \Rightarrow \quad r_{ij}^{m1} := s_{ij} \vee s_{jk}, \; i,j = \overline{1,n}, \; k = \overline{1,n}, \qquad (3)$$

$$r_{ij}^{m2} := r_{ij}^{m1} \vee r_{jk}^{m1} \;, \; i,j = \overline{1,n} \;, \; k = \overline{1,n} \;. \qquad (4)$$

The described algorithm is implemented by the following program code:

```
procedure TForm1.Matrix_TC;
var i, j, jj, n: Integer;
                 m: Short;
begin
            for m:=1 to 2 do
            for i:=0 to n-1 do
            for j:=0 to n-1 do
                If (S[i, j]=1) and (i<>j) then
                begin
                for jj:=0 to n-1 do If
(S[i, jj]=0) and (S[j, jj]=1) then S[i, jj]:=1;
                end;
end.
```

Therefore, after the first stage, we will have a defined matrix (R_G) of the transitive closure of the model graph G.

Before presenting the content of the second stage of the developed algorithm, let us note some aspects that arise from the very definition of the SCC of a directed graph.

Axiom 1. If there is no transitive closure in direction $v_i \to v_j$, then this pair of vertices cannot be part of a certain SCC, even if there is a transitive closure in direction $v_j \to v_i$.

Axiom 2. The transitive closure matrix R_G of a directed graph, as well as its inverse reachability matrix Q_G, can be asymmetric matrices with respect to the main diagonal.

Based on these remarks, a special analysis of the transitive closure matrix (R_G) of the model graph G at the second stage of the algorithm will be as follows:

1. Transpose the matrix R_G to obtain the inverse reachability matrix between pairs of vertices, i.e. $Q_G = R_G^T$. Next, we perform an element-by-element product of matrices R_G and Q_G. As a result, we obtain a symmetric matrix ($R_G \otimes Q_G$) with respect to its main diagonal. This matrix contains information about pairs of vertices between which there is a bidirectional connection.

2. Simultaneous contraction of matrix ($R_G \otimes Q_G$) toward the north-west corner (toward the lower-order positions) by its identical rows and columns, respectively. The unit blocks thus formed on the main diagonal of the matrix will indicate the quantitative composition of the SCC, and the indices of the rows and columns will indicate the vertex composition of each component.

Taking into account the above provisions, we will conduct a computational experiment on a certain network object.

4 EXPERIMENTS

Let a real network object be modeled by a directed graph G, Fig. 3.

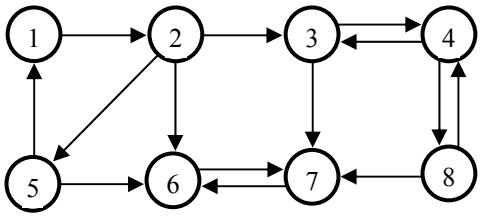

Figure 3 – Model directed graph G

The structure of the graph is represented by the initial adjacency matrix S_G. Let us write it:

$$S_G = \begin{array}{c|cccccccc} & v1 & v2 & v3 & v4 & v5 & v6 & v7 & v8 \\ \hline v1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ v2 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ v3 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ v4 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ v5 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ v6 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ v7 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ v8 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{array} . \qquad (5)$$

DOI 10.15588/1607-3274-2026-2-13

Stage I. We define the transitive closure matrix R_G according to expressions (3) and (4). The results of the first traversal of the matrix will be as follows:

$$R_G^{[1]} = \begin{array}{c|cccccccc} & v1 & v2 & v3 & v4 & v5 & v6 & v7 & v8 \\ v1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ v2 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ v3 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ v4 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ v5 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ v6 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ v7 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ v8 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{array}. \quad (6)$$

The results of the second and final pass through the matrix are as follows:

$$R_G = R_G^{[2]} = \begin{array}{c|cccccccc} & v1 & v2 & v3 & v4 & v5 & v6 & v7 & v8 \\ v1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ v2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ v3 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ v4 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ v5 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ v6 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ v7 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ v8 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{array}. \quad (7)$$

Stage II. Given matrix R_G, we define the block-diagonal matrix ($R_G \otimes Q_G$). To do this, first find matrix Q_G. We obtain:

$$Q_G = R_G^T = \begin{array}{c|cccccccc} & v1 & v2 & v3 & v4 & v5 & v6 & v7 & v8 \\ v1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ v2 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ v3 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ v4 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ v5 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ v6 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ v7 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ v8 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}. \quad (8)$$

Let's find the direct element-by-element product of matrices R_G and Q_G. The operation is correct, since the matrices have the same dimension. We get:

$$(R_G \otimes Q_G) = \begin{array}{c|cccccccc} & v1 & v2 & v3 & v4 & v5 & v6 & v7 & v8 \\ v1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ v2 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ v3 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ v4 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ v5 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ v6 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ v7 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ v8 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{array}. \quad (9)$$

As can be seen, the matrix ($R_G \otimes Q_G$) is symmetric.

By pulling the same rows and columns towards the lower digits of the matrix, we construct a block-diagonal matrix. We obtain:

$$(R_G \otimes Q_G) = \begin{array}{c|cccccccc} & v1 & v2 & v5 & v3 & v4 & v8 & v6 & v7 \\ v1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ v2 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ v5 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ v3 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ v4 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ v8 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ v6 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ v7 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}. \quad (10)$$

Therefore, the constructed block diagonal matrix shows that in the structure of the model graph G there are three components of strong connectivity (parameter $k = 3$), expression (10). Their vertex composition is as follows: $\{v_1, v_2, v_5\} \in G_1$; $\{v_3, v_4, v_8\} \in G_2$; $\{v_6, v_7\} \in G_3$. You can verify the correctness of the SCC definition by conducting a corresponding analysis of the graph presented in Fig. 4.

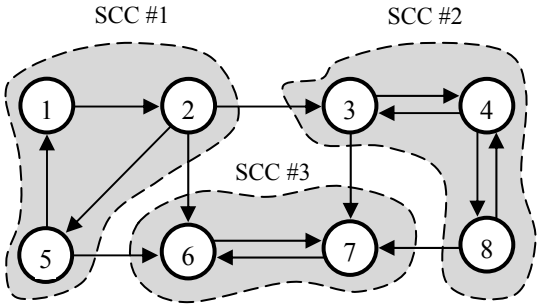

Figure 4 – The identified SCCs and their vertex composition in the directed model graph G

The next important issue in the structural analysis of network objects is the search for unidirectional connections (arcs) between SCCs, since it is their one-way directionality that is the reason for dividing the object into several SCCs. In this regard, they often strive to bring the network object to a connected structure - to one SCC (in the tasks of planning urban transport routes, when planning priority work to restore the connectivity of a destroyed network object, etc.). On small objects, such a search can be performed in the "manual" mode. On objects with a significant number of nodes and connections, such a task is laborious, and in many cases, one that will be solved with errors.

Let us show how such a problem is solved in matrix form, quickly and without errors.

DOI 10.15588/1607-3274-2026-2-13

In formalized form, the operation is written as the following two expressions with a condition:

$$S_G^* := S_G, \tag{11}$$

$$if \quad (r_G \otimes q_G)_{ij} \neq 0 \quad \Rightarrow s_{ij}^* := 0, \; i,j = \overline{1,n}. \tag{12}$$

After such an operation, matrix S_G^* of graph G, presented in Fig. 4, will have the form:

$$S_G^* = \begin{array}{c|cccccccc} & v1 & v2 & v3 & v4 & v5 & v6 & v7 & v8 \\ \hline v1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ v2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ v3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ v4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ v5 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ v6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ v7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ v8 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}. \tag{13}$$

Therefore, based on expression (13), the critical arcs from the point of view of connectivity in the graph G are: (v_2, v_3); (v_2, v_6); (v_3, v_7); (v_5, v_6); (v_8, v_7). It is these directions that must first be considered in order to form bidirectional connections between the corresponding vertices on their basis.

Conversely, if the graph presented in Fig. 4 models a certain network object of the enemy, then the above-mentioned connections should become the first targets for destruction in order to disable the object (disrupt the technological processes taking place there).

If we consider the nodal basis of the object as the target, then the first targets should be the nodes modeled by vertices v_2 and v_7, since they have two incident arcs each (the largest number of them), see the corresponding row and column of expression (13). Vertex v_2 is more critical in this respect, as its arcs connect different SCCs. The next target is the node modeled by vertex v_5. It is on the performance of these elements, first of all, that the functioning of the network object as a whole will depend.

5 RESULTS

In the article, during the development of the algorithm, the following results were obtained:

– a theoretical basis for analyzing the matrix of transitive closures of binary relations of a model directed graph has been developed in order to determine the quantitative and index composition of strongly connected components in its composition;

– in terms of graph theory, the search for strongly connected components in the structure of a network object is formalized;

– the developed algorithm was verified for its ability to determine strongly connected components in the structure of a directed graph.

6 DISCUSSION

General ideas about the transitivity properties of binary relations, knowledge about the internal properties of strongly connected components, and mathematical manipulation of these properties in a matrix representation made it possible to develop a new matrix algorithm for searching for strongly connected components in the structure of network objects.

Understanding the fact that any SCC is a symmetric and unitary block (a submatrix of the original) led to the idea that such blocks can (and should) be collected on the main diagonal of the so-called block-diagonal matrix ($R_G \otimes Q_G$).

The operation of the element-wise product of matrices R_G and Q_G ensured the symmetry of the resulting matrix in which the unit elements denote pairs of vertices between which a bidirectional (duplex) connection has been established. The number of unit blocks will determine the number of existing SCCs, and the index composition of the corresponding rows and columns is the vertex composition of each SCC.

The use in the article of well-known provisions of graph theory, which are interpreted unambiguously in this theory, and the absence of any probabilistic processes, allow us to consider the developed algorithm as accurate.

An important characteristic of combinatorial algorithms, including algorithms for structural analysis of network objects, is their computational complexity. It is clear that on real structures, and therefore on structures with a larger number of nodes and denser connections, the number of operations for searching for SCC will be significantly greater than in the example given in the article. It should be noted here that the computational complexity of the developed linear algorithm will be determined by the computational complexity of its "basic element" – the algorithm for determining the transitive closure matrix (stage I), which is estimated by $O(n^3)$ [17].

The obtained polynomial estimate of the computational complexity of the algorithm is quite acceptable for its use in real time.

CONCLUSIONS

The article solves a relevant scientific and applied problem – the development of an SCC search tool, which can be used to justify appropriate options for the development/destruction of network objects.

The scientific novelty of the developed algorithm is as follows:

1) in the representation of SCC in the form of unit blocks on the main diagonal of a block-diagonal matrix, which is constructed in a special way.

DOI 10.15588/1607-3274-2026-2-13

The practical significance of the algorithm lies in the fact that its application allows:

1) it is reasonable to plan the development of a network object or the restoration of its structural connectivity after external influences;

2) In military affairs, the algorithm can be applied when developing an appropriate option for inflicting fire damage on critical enemy targets that have a network (distributed) structure.

The computational complexity of the proposed algorithm has a polynomial dependence on the dimension and density of the model graph, which allows using such an algorithm in real time.

Prospects for further research are to development of similar algorithms for determining SCCs, the elements of which correspond to a set of certain criteria – multi-level SCCs.

ACKNOWLEDGMENTS

This article presents one of the results obtained by the authors in 2023–2024 during the implementation of an initiative project at the Research Center of the National Academy of the National Guard of Ukraine. The authors express their gratitude to their colleagues for their support during the research and their active participation in discussing the results.

DECLARATIONS

Conflict of interest: The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship, or otherwise, that could affect the research and its results presented in this paper.

Authors' contributions: Vladimir Batsamut: the algorithm for searching for transitive closures and strongly connected components in directed graphs; Yevhen Bashkatov: completed the problem statement; Dmytro Morkvin: conducted an analysis of existing algorithms for searching for strongly connected components in network structures; Dymytrii Tolstonosov: the algorithm for searching strongly connected components in directed graphs; Borys Sakhnevych: experimental study of the developed algorithm; Yevhen Solodun: experimental study of the developed algorithm.

Data availability: The manuscript has associated data in a data repository https://computingonline.net/computing/article/view/2444/.

Software availability: The manuscript has no associated software.

Use of artificial intelligence tools: The authors confirm that they did not use artificial intelligence technologies in creating the submitted work.

REFERENCES

1. Cormen T. H. et al. Introduction to algorithms. Cambridge, MIT Press, 2009, 1292 p.
2. Aho A. V., Hopcroft J. E., Ullman Jeffrey D. The Design and Analysis of Computer Algorithms. [S. l.]. Addison-Wesley, 1974, 482 p.
3. Demetrescu C., Italiano G. F. Fully Dynamic Transitive Closure: Breaking Through the $O(n^2)$ Barrier. *Proceedings of FOCS 2000*, [S. l.], 2000, pp. 381–389.
4. Tarjan R. E. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1972, Vol. 1, No. 2, pp. 146–160.
5. Gabow H. N. Path-based depth-first search for strong and biconnected components. *Information Processing Letters*, 2000, Vol. 74, № 3–4, pp. 107–114.
6. Christofides N. Theory of graphs : algorithmic approach. Moscow, Mir, 1978, 432 p.
7. Alshomrani S., Iqbal G. An extended experimental evaluation of SCC (Gabow's vs Kosaraju's) based on adjacency list. *Global Journal of Computer Science and Technology. Series E, Network, Web & Security*, 2013, Vol. 13, № 11, pp. 68–74. ISSN 0975-4350
8. Tarjan R. E., Zwick U. Finding strong components using depth-first search. *European Journal of Combinatorics*, 2024, Vol. 119, № 2, Art. 103815. DOI: 10.1016/j.ejc.2023.103815
9. Dhingra S., Dodwad P. S., Madan M. Finding strongly connected components in a social network graph. *International Journal of Computer Applications*, 2016, Vol. 136, № 7, pp. 1–5. DOI: 10.5120/ijca2016908481
10. Nuutila E., Soisalon-Soininen E. O. On finding the strongly connected components in a directed graph. *Information Processing Letters*, 1994, Vol. 49, № 1, pp. 9–14. DOI: 10.1016/0020-0190(94)90047-7.
11. Lipskyy V. Combinatory for programmers. Moscow, Mir Press, 1988, 213 p.
12. Warshall S. Algorithm on Boolean matrices. *Journal of the ACM*, 1962, Vol. 9, № 1, pp. 11–12. DOI: 10.1145/321105.321107
13. Floyd R. Algorithm 97: Shortest path. *Communications of the ACM*, 1961, Vol. 5, № 6, P. 345. DOI: 10.1145/367766.368168
14. Warren H. A modification of Warshall's algorithm for the transitive closure of binary relations, *Communications of the ACM*, 1975, Vol. 18, № 4, pp. 218–220.
15. Bellman R. On a Routing Problem. *Quarterly of Applied Mathematics*, 1958, Vol. 16, № 1, pp. 87–90.
16. Shimbel A. Structural parameters of communication networks. *Bulletin of Mathematical Biophysics*, 1953, Vol. 15, № 4, pp. 501–507. DOI: 10.1007/BF02476438
17. Batsamut V., Manzura S., Kosiak O. et al. Fast Algorithm for Calculating Transitive Closures of Binary Relations in the Structure of a Network Object. *International Journal of Computing*, 2021, Vol. 20, № 4, pp. 560–566. DOI: 10.47839/ijc.20.4.2444

Received 05.01.2026.
Accepted 20.04.2026.
Published 26.06.2026.

DOI 10.15588/1607-3274-2026-2-13

УДК 519.17:004.421.2

ЛІНІЙНИЙ АЛГОРИТМ ПОШУКУ КОМПОНЕНТ СИЛЬНОЇ ЗВ'ЯЗНОСТІ В ОРІЄНТОВАНИХ ГРАФАХ, ЗАСНОВАНИЙ НА АНАЛІЗІ МАТРИЦЬ ТРАНЗИТИВНОГО ЗАМИКАННЯ БІНАРНИХ ВІДНОШЕНЬ

Бацамут В. М. – д-р військ. наук, професор, заступник начальника навчально-наукового інституту забезпечення державної безпеки з наукової роботи Національної академії Національної гвардії України, Харків, Україна. ROR: https://ror.org/02p49q920. ORCID: https://orcid.org/0000-0003-2182-6891.

Башкатов Є. Г. – канд. військ. наук, доцент, заступник начальника навчально-наукового інституту забезпечення державної безпеки з навчальної роботи Національної академії Національної гвардії України, Харків, Україна. ROR: https://ror.org/02p49q920. ORCID: https://orcid.org/0000-0002-6078-886X.

Морквін Д. А. – д-р філософ., старший науковий співробітник науково-дослідної лабораторії навчально-наукового інституту професійної освіти Національної академії Національної гвардії України, Харків, Україна. ROR: https://ror.org/02p49q920. ORCID: https://orcid.org/0000-0002-3651-6805.

Толстоносов Д. Ю. – канд. юрид. наук, доцент, начальник кафедри бойового та логістичного забезпечення факультету службово-бойової діяльності Національної гвардії України Київського інституту Національної гвардії України, Київ, Україна. ROR: https://ror.org/01q3vqr85. ORCID: https://orcid.org/0000-0001-5181-7668.

Сахневич Б. В. – канд. екон. наук, заступник начальника кафедри бойового та логістичного забезпечення факультету службово-бойової діяльності Національної гвардії України Київського інституту Національної гвардії України, Київ, Україна. ROR: https://ror.org/01q3vqr85. ORCID: https://orcid.org/0009-0003-7263-5689.

Солодун Є. М. – канд. техн. наук, доцент кафедри бойового та логістичного забезпечення факультету службово-бойової діяльності Національної гвардії України Київського інституту Національної гвардії України, Київ, Україна. ROR: https://ror.org/01q3vqr85. ORCID: https://orcid.org/0009-0001-8390-9249.

АНОТАЦІЯ

Актуальність. Актуальність статті обумовлюється потребою у подальшому розвитку простих алгоритмів аналізу структурної зв'язності мережевих об'єктів, у зменшенні обчислювальної складності і збільшенні функціональних можливостей таких алгоритмів. Запропонований у статті лінійний алгоритм може бути застосований: у системах планування маршрутів руху міського та позаміського транспорту; у ході розв'язання різного роду задач, пов'язаних із проведенням відновлювальних робіт на мережевих об'єктах після їх руйнування і втрати певних зв'язків між елементами таких об'єктів (системи енергопостачання, системи зв'язку, транспортні мережі тощо); у ході вироблення варіантів щодо ефективного ураження ключових об'єктів критичної інфраструктури противника; під час аналізу соціальних мереж на предмет групування індивідів у групи за інтересами тощо.

Мета роботи – розроблення лінійного алгоритму для пошуку в структурі мережевих об'єктів компонент сильної зв'язності, стійкого до змін структури об'єктів та такого, що має поліноміальну обчислювальну складність.

Метод. Фізичний об'єкт, що підлягає дослідженню моделюється зв'язним орієнтованим графом. Для визначення в структурі модельного графа компонент сильної зв'язності застосовано ідею аналізу наявності прямих і зворотних транзитивних замкнень між всіма парами вершин модельного графа. Оскільки між всіма парами вершин, що входять в будь-яку компоненту сильної зв'язності, існує маршрут (або дуга), що їх зв'язує, то в матриці транзитивних замкнень бінарних відношень такого графу завжди існуватимуть одиничні блоки (ущільнення транзитивності). Окремі елементи цих блоків в матриці транзитивних замкнень розкидані певним чином. Аналізуючи матрицю на наявність однакових за складом рядків та однакових стовпців, можна встановити їх приналежність до певних одиничних блоків. Після такого аналізу і групування відповідних рядків і стовпців в одиничні блоки можна визначити кількість компонент сильної зв'язності, а також індексний (вершинний) склад кожної компоненти.

Для побудови матриці транзитивних замкнень бінарних відношень можна застосовувати будь-який з відомих алгоритмів, про що йдеться в статті. Отже, лінійний алгоритм, що пропонується в цій статті, умовно ділиться на два основні етапи: перший – пошук транзитивних замкнень бінарних відношень модельного орієнтованого графа; другий – визначення матриці зворотних досяжностей графа, визначення матриці взаємних досяжностей, і пошук в складі останньої одиничних блоків шляхом вишиковування їх на головній діагоналі матриці.

Результати. 1) Розроблено теоретичну основу аналізу матриці транзитивних замкнень бінарних відношень модельного орієнтованого графа з метою визначення в його складі кількісного і індексного складу компонент сильної зв'язності; 2) В термінах теорії графів формалізовано пошук компонент сильної зв'язності в структурі мережевого об'єкта; 3) Виконано верифікацію розробленого алгоритму на його здатність визначати в структурі орієнтованого графа компоненти сильної зв'язності.

Висновки. Проведені теоретичні дослідження та низка проведених експериментів підтверджують можливість використання пропонованого алгоритму в задачах структурного аналізу мережевих об'єктів. Оскільки у ході теоретичних досліджень використані добре відомі положення теорії графів, які трактуються в цій теорії однозначно, відсутність будь-яких ймовірнісних процесів, дозволяють вважати розроблений алгоритм точним. Поліноміальні оцінки його обчислювальної складності дозволяють його використовувати в масштабі реального часу.

КЛЮЧОВІ СЛОВА: мережевий об'єкт, ураження об'єкта, орієнтований граф, вершина графа, транзитивне замкнення, компонента сильної зв'язності, матриця, головна діагональ матриці, одиничний блок, алгоритм.

DOI 10.15588/1607-3274-2026-2-13

ЛІТЕРАТУРА

1. Introduction to algorithms / Thomas H. Cormen [et al.]. – Cambridge : MIT Press, 2009. – 1292 p.
2. Aho A. V. The Design and Analysis of Computer Algorithms / Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman. – [S. l.] : Addison-Wesley, 1974. – 482 p.
3. Demetrescu C. Fully Dynamic Transitive Closure: Breaking Through the O(n²) Barrier / C. Demetrescu, G. F. Italiano // Proceedings of FOCS 2000. – [S. l.], 2000. – P. 381–389.
4. Tarjan R. E. Depth-first search and linear graph algorithms / R. E. Tarjan // SIAM Journal on Computing. – 1972. – Vol. 1, no. 2. – P. 146–160.
5. Gabow H. N. Path-based depth-first search for strong and biconnected components / H. N. Gabow // Information Processing Letters. – 2000. – Vol. 74, № 3–4. – P. 107–114.
6. Кристофидес Н. Теория графов. Алгоритмический подход / Н. Кристофидес. – М.: Мир, 1978. – 432 с.
7. Alshomrani S. An extended experimental evaluation of SCC (Gabow's vs Kosaraju's) based on adjacency list / S. Alshomrani, G. Iqbal // Global Journal of Computer Science and Technology. Series E, Network, Web & Security. – 2013. – Vol. 13, № 11. – P. 68–74. – ISSN 0975-4350
8. Tarjan R. E. Finding strong components using depth-first search / R. E. Tarjan, U. Zwick // European Journal of Combinatorics. – 2024. – Vol. 119, № 2. – Art. 103815. DOI: 10.1016/j.ejc.2023.103815
9. Dhingra S. Finding strongly connected components in a social network graph / S. Dhingra, P. S. Dodwad, M. Madan // International Journal of Computer Applications. – 2016. – Vol. 136, № 7. – P. 1–5. DOI: 10.5120/ijca2016908481
10. Nuutila E. On finding the strongly connected components in a directed graph / E. Nuutila, E. O. Soisalon-Soininen // Information Processing Letters. – 1994. – Vol. 49, № 1. – P. 9–14. DOI: 10.1016/0020-0190(94)90047-7.
11. Lipskyy V. Combinatory for programmers / V. Lipskyy. – M.: Mir Press, 1988. – 213 p.
12. Warshall S. Algorithm on Boolean matrices / S. Warshall // Journal of the ACM. – 1962. – Vol. 9, № 1. – P. 11–12. DOI: 10.1145/321105.321107
13. Floyd R. Algorithm 97 : Shortest path / R. Floyd // Communications of the ACM. – 1961. – Vol. 5, № 6. – P. 345. DOI: 10.1145/367766.368168
14. Warren H. A modification of Warshall's algorithm for the transitive closure of binary relations / H. Warren // Communications of the ACM. – 1975. – Vol. 18, № 4. – P. 218–220.
15. Bellman R. On a Routing Problem / R. Bellman // Quarterly of Applied Mathematics. – 1958. – Vol. 16, № 1. – P. 87–90.
16. Shimbel A. Structural parameters of communication networks / A. Shimbel // Bulletin of Mathematical Biophysics. – 1953. – Vol. 15, № 4. – P. 501–507. DOI: 10.1007/BF02476438
17. Batsamut V. Fast Algorithm for Calculating Transitive Closures of Binary Relations in the Structure of a Network Object / V. Batsamut, S. Manzura, O. Kosiak et al. // International Journal of Computing. – 2021. – Vol. 20, № 4. – P. 560–566. DOI: 10.47839/ijc.20.4.2444

DOI 10.15588/1607-3274-2026-2-13