UDC 519.853: 658.52

V. Kozalietov **V. Batsamut**

METHOD OF ROUTING SEARCH GROUPS ON A FIXED NETWORK FOR EFFECTIVE SWEEPING OF POPULATED AREAS AND SEARCHING FOR ARMED CRIMINALS

The method proposed in this article addresses the problem of inefficient coverage of a given urban area by search teams, which is critical for the execution of search, rescue, and monitoring tasks in crisis-affected regions of various origins. The method employs the concept of dynamic programming, which consists in the successive decomposition of the initial structure of the urban transportation network into a set of substructures. Within each substructure, an edge-simple longest/shortest path (depending on the selected strategy) between a pair of specified vertices is determined. The collection of the identified extreme paths forms a Routing Plan for the search teams, which, in a certain sense, systematizes the sweep of the urban area.

The method significantly reduces the labor and computational effort required to generate a rational variant of search operations and is intended for use by the military command and control bodies of the National Guard of Ukraine.

Keywords: *area coverag, search teams, multi-agent system, routing, network-based object, weighted undirected (directed) graph, extreme paths, optimization criterion, method.*

Statement of the problem. One of the tasks defined in Article 1 of the Law of Ukraine "On the National Guard of Ukraine" [1] is "... the suppression of terrorist activities, as well as the activities of illegal paramilitary or armed formations (groups), terrorist organizations, organized groups, and criminal organizations". The fulfillment of this task is determined by the function of the National Guard of Ukraine (NGU) set out in Article 2 of the same Law, namely "... participation in special operations aimed at neutralizing armed offenders, terminating the activities of paramilitary or armed formations (groups), organized groups, and criminal organizations not provided for by law on the territory of Ukraine, as well as in measures related to the suppression of terrorist activities". It should be noted that the implementation of this function will constitute a particularly complex challenge for the NGU in the post-war period [2].

One of the operational methods employed by units (formations) of the NGU in the execution of these tasks is, and will remain, the conduct of search operations. Given the relatively high density of settlements across the territory of Ukraine, a substantial proportion of search operations will be carried out within populated areas. At the same time, the network-based spatial structure of settlements determines that the formation designated to conduct a search will necessarily be divided into several search groups, which will be required to advance independently along designated routes in order to cover the maximum possible number of facilities for inspection [3]. Accordingly, under urban and semi-urban conditions, the effectiveness of search operations will largely depend on the selected sweep (movement) routes of individual search groups. Moreover, these movement routes must comply with a set of specific requirements outlined in [3].

Accordingly, the development (justification) of movement routes for search groups within a given settlement constitutes a complex problem that has traditionally been, and continues to be, addressed through heuristic methods based on the personal experience, knowledge, and professional judgment of the commander (supervisor) responsible for decision-making. Within networked structures characterized by a high density of objects and interconnections, this problem, when approached heuristically, will inevitably be solved suboptimally and with inherent errors, which in some cases may be substantial.

Therefore, the scientific task of developing a routing method for search groups on a stationary network aimed at ensuring the effective sweeping

of settlements and the detection of armed offenders during stabilization operations in de-occupied territories is currently highly relevant. This issue constitutes the focus of the present article.

Analysis of recent research and publications. The theoretical foundation of the functioning of various multi-agent systems (MAS) is algorithmic theory. The development of algorithmic theory proceeds along two main directions: first, the expansion of the range of practical problems for which existing algorithms are applied; and second, the development and refinement of algorithms to address new problems that arise in the course of the design and operation of MAS. In matters of routing individual agents within MAS, the well-established theory of graphs plays a leading role.

If considered in a non-rigorous sense, the problem of routing search groups during the sweeping of a settlement by a military formation can be formulated as the task of identifying, within the structure of a modeled weighted graph $G = (P, E)$ all (or *k*, where *k >1*) the longest edge-simple paths between selected subsets of vertices located on opposite peripheries of the graph, such that the total sum of their weights (paths) maximizes the objective function F [3]. An additional and mandatory requirement for the resulting *Routing plan* is the absence of shared edges among the different paths.

A closely related and well-known problem in graph theory with numerous practical applications is the hamiltonian path problem, that is, determining whether a graph contains a simple path in which each vertex is visited exactly once. In cases where a graph does not admit a hamiltonian path, it is, in some applications, meaningful to seek a path of maximum length within the graph.

Thus, in [4], the authors demonstrate that even if a graph admits a Hamiltonian path, the problem of finding a path of maximum length remains computationally challenging $n - n^{\varepsilon}$ for some $\varepsilon < 1$ there are NP-full, where n denotes the number of vertices in the original graph. The authors assert that there exists no polynomial-time approximation algorithm with a constant-factor guarantee for the longest path problem, unless $P = NP$ [4]. Similar research findings are also reported in publications [5, 6, 7].

In contrast to the Hamiltonian path problem, for the longest path problem several polynomial-time algorithms are known that operate on trees and certain other classes of graphs. A linear-time algorithm for finding the longest path in a tree structure was proposed by Dijkstra in 1960, a formal description of which can be found in [8]. Subsequently, as a result of refining Dijkstra's algorithm for tree structures, the authors of [9] solved the longest path problem for weighted trees and block graphs in linear time, and for cactus graphs in polynomial time – $O(n^2)$, n is the number of vertices in the original graph. Recently, polynomial-time algorithms have been proposed to solve the longest path problem on bipartite graphs with a computational complexity $O(n)$ [10], on ptolemaic graphs with a computational complexity of $O(n^5)$ [11]. In [12], the authors present a polynomial-time algorithm for interval graphs, which is based on the idea of dynamic programming and has a computational complexity of $O(n^4)$. In [13], the authors propose a polynomial-time algorithm that likewise employs a dynamic programming approach but applies lexicographic depth-first search (the so-called LDFS graph traversal) to comparability graphs. The computational complexity of this algorithm is also bounded by $O(n^4)$. The conducted analysis of the literature indicates that the problem in the formal formulation presented above has not yet been explicitly posed or addressed in existing studies.

The purpose of the article is to develop a routing method for search groups on a stationary network to ensure the effective sweeping of settlements and the detection of armed offenders during stabilization operations in de-occupied territories.

Summary of the main material. The NP-completeness issue in the search for longest paths (i.e., paths with the maximum total weight) within the structure of a networked object arises from the possible presence of cycles, which can lead to an unjustified artificial increase in path length (i.e., "cycling") and, consequently, to the inability to adequately identify a meaningful or realistic path. To eliminate this drawback in solving the maximization problem (i.e., the search for longest paths), the initial undirected graph is typically $G = (P, E)$ represented as a directed graph, which makes it possible to eliminate cycles. During this procedure, the edges of the graph are assigned a direction consistent with the overall traversal orientation of the graph (Figure 1). As a result of this transformation, the edges become arcs – i.e., they acquire directionality and the graph itself becomes directed and is denoted as $\vec{G} = (P, E)$.

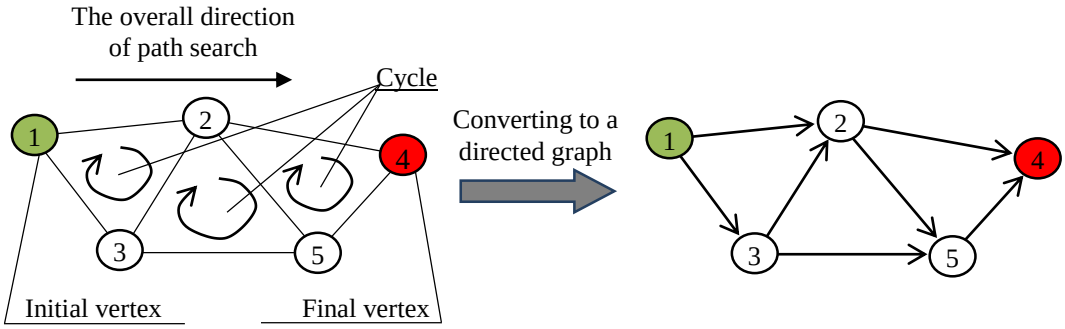
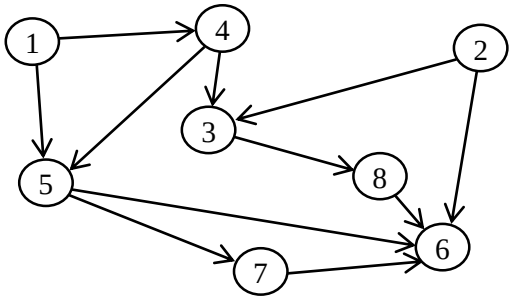


Figure 1 – Converting an undirected graph into a directed graph

Since the sweeping of a settlement is conducted from one periphery to the opposite one (which can be conceptualized as a conditional "wave" passing through the settlement), such an overall directionality is also present during the execution of search operations. As shown in Figure 1, the structure of the directed graph no longer contains cycles. Such a graph is referred to as acyclic.

To identify critical paths in an acyclic graph, it is necessary to establish an order for traversing its vertices. For this purpose, Kahn's topological sorting algorithm [14] is employed. The algorithm identifies vertices with no incoming arcs and removes all outgoing arcs from such a vertex, as well as the vertex itself, recording its index in an ordered list S. This process is iteratively repeated until all vertices have been examined and included in the list, thereby achieving a topological ordering. The algorithm is also based on the assertion that an acyclic graph always contains at least one vertex with no incoming arcs. Therefore, topological sorting in an acyclic graph can always be initiated and completed.

At the same time, Kahn's algorithm may yield multiple valid vertex orderings for the same acyclic graph (Figure 2), which renders the search for extreme paths ambiguous. In order to achieve a unique vertex ordering, it is proposed to refine Kahn's algorithm as follows. At each iteration of the algorithm through p_k^- denote the vertices of the graph that have only output arcs, where k is the number of such vertices at a certain iteration of the algorithm. During the analysis p_k^- in addition, it is recommended to consider the weight coefficients of the arcs and, in each iteration, to prioritize the selection of the vertex for which a given arc attains the maximum weight.



$S_1: p_1 \to p_4 \to p_2 \to p_5 \to p_3 \to p_8 \to p_7 \to p_6$
$S_2: p_2 \to p_1 \to p_4 \to p_3 \to p_5 \to p_7 \to p_8 \to p_6$
$S_3: p_1 \to p_2 \to p_4 \to p_5 \to p_7 \to p_3 \to p_8 \to p_6$

Figure 2 – Different variants of topological orderings of the graph vertices (the set of possible variants is not exhaustive)

Such a modification of the algorithm ensures not only the topological sorting of vertices but also their ordering according to the weights of the incident arcs. Vertices associated with higher-weight incident arcs are placed in the list earlier, and vice versa.

In formal terms, the vertex selection criterion during the sorting process can be expressed as follows:

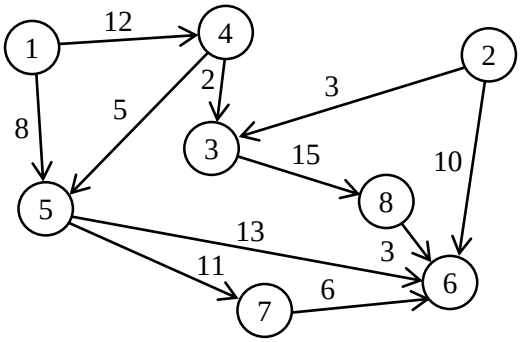
$$p_i^- \to S : \exists\, w(p_i^-, p_j) \to max, \qquad (1)$$

where p_i^- is the current vertex, which has only output arcs;

$w(p_i^-, p_j)$ is the weight of the arc emanating from the vertex p_i^-;

S is the list for storing graph vertices.

By accounting for the arc weights, the sorting of vertices becomes unique, which ensures an unambiguous traversal of the graph vertices for forming their current weighted estimates during the construction of the desired extreme paths (Figure 3).



$S_1: p_1 \to p_2 \to p_4 \to p_3 \to p_5 \to p_7 \to p_8 \to p_6$

Figure 3 – Topological sorting of the graph vertices using the modified Kahn's algorithm (unique ordering)

Thus, at this stage, two key issues have been addressed:

a) the negative impact of cycles in the graph structure during the search for maximum-length paths;

b) the order of traversing the vertices of an acyclic graph for forming the current weighted estimates of the graph vertices in the process of searching for maximum-length paths.

The resolution of these issues made it possible to proceed with the development of a routing method for search groups on a stationary network aimed at the effective sweeping of settlements and the detection of armed offenders.

Let the transportation structure of a given settlement be represented by a weighted graph G' (Figure 4).

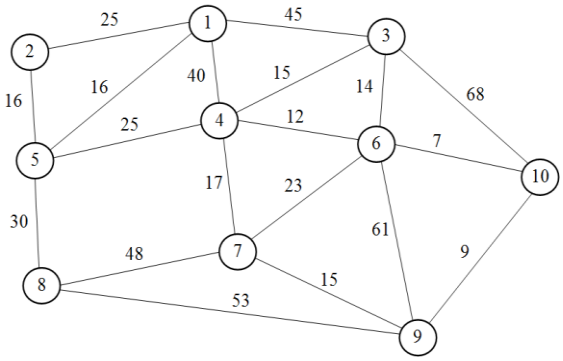


Figure 4 – Initial weighted graph modeling the transportation network of a settlement (edge weights indicate the number of inspection objects on the corresponding street of the settlement)

It is required to find all edge-simple longest paths from vertices p_2 and p_5 vertices p_9 and p_{10}, that is, the sweeping operation during the search for armed offenders is assumed to be conducted from the western to the eastern periphery of the settlement. Thus, the subset A will include vertices p_2 and p_5, therefore ($A = \{p_2, p_5\}$), and the subset B will include vertices p_9 and p_{10}, ($B = \{p_9, p_{10}\}$).

Stage I. Let a general direction be assigned to the edges of the graph from vertices p_2 and p_5 toward the opposite periphery of the graph (vertices p_9 and p_{10}). The result of these steps is a directed weighted graph $\vec{G}'$, as shown in Figure 5. Vertices belonging to subsets A and B are indicated by the corresponding colors (green – peaks $p_i \in A$, red – vertices $p_i \in B$).

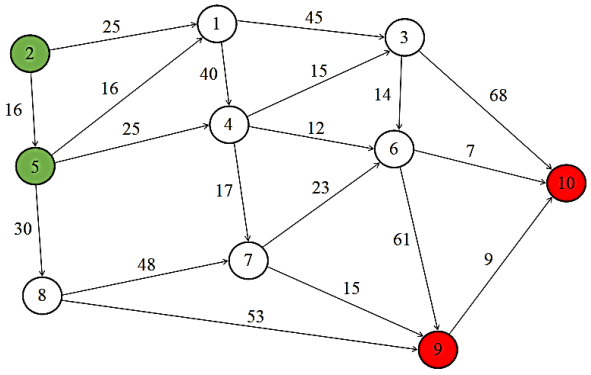


Figure 5 – Directed weighted graph $\vec{G}'$

Stage II. Let us determine the longest path between any pair of vertices belonging to subsets A and B, respectively. To this end, a topological sorting of the graph $\vec{G}'$ vertices is performed according to criterion (1). The result is the following unique sequence of vertices: $p_2 \to p_5 \to \to p_8 \to p_1 \to p_4 \to p_3 \to p_7 \to p_6 \to p_9 \to p_{10}$.

By traversing the vertices of the graph according to this sequence, each vertex p_j is assigned an individual estimate of the path length to it from a given source $p_i \in A$.

At the same time, the weighted estimates are proposed to be calculated according to the following expression

$$d(p_j) := \max\{d(p_j), d(p_i^-) + w(p_i^-, p_j)\}, (2)$$

where $d(p_j)$ is the current estimate of the path length to a vertex that is adjacent to a given vertex p_i^-;

$d(p_i^-)$ is the current estimate of the path length to the vertex selected from the topological ordering list S;

$w(p_i^-, p_j)$ is the weight of the arc between adjacent vertices.

It should be noted that, as a result of applying expression (2) at each iteration, the vertices of the graph are assigned the maximum estimates of the lengths of the paths leading to them. The identification of the paths themselves should be carried out by backtracking based on the obtained estimates.

Thus, the algorithm for traversing the vertices of a directed weighted graph with the formation of current estimates of the longest path for each vertex $p_i \notin A$ consists of the following steps.

Step 1. Assign the current estimate $p_i \in A$ to all vertices $d(p_i) = 0$.

Step 2. Assign an estimate of $d(p_i) = -\infty$ to the remaining vertices, including vertices $p_i \in B$.

Step 3. For the vertex whose index appears first in the list S, which is topologically ordered using the modified Kahn's algorithm, recompute the current estimates of its adjacent vertices according to expression (2). Remove the index of this vertex from the list S.

Step 4. If no vertex indices remain in the list S, consider all vertices to have been processed and assigned the maximum path-length estimates, terminate the algorithm, and proceed to the identification of the paths. Otherwise, return to Step 3.

The current estimates of path lengths obtained during the execution of the developed algorithm, as well as the identified longest path in the structure of the initial graph, are shown in Figure 6.

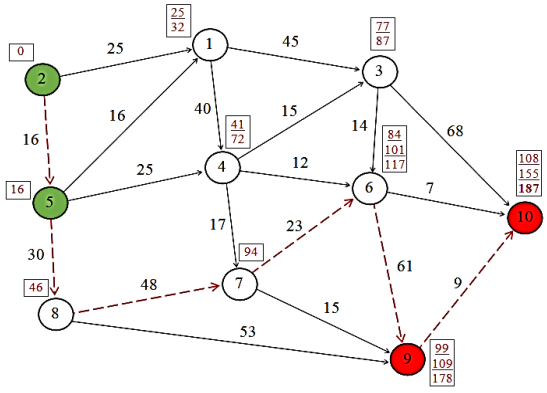


Figure 6 – Initial weighted graph and the longest path within its structure (indicated by a dashed line)

As a result of the algorithm's execution, a path (route) to vertices p_9 and p_{10}, which belong to subset B, is obtained, namely:

$M_1 = \{p_2, p_5, p_8, p_7, p_6, p_9\}$;

$M_2 = \{p_2, p_5, p_8, p_7, p_6, p_9, p_{10}\}$.

The absolute weight of route $w(M_1) = 178$, while that of route $w(M_2) = 187$. Therefore, the longer route M_2 is selected from this pair, assigned the conditional index M_1 (according to the numbering sequence), and recorded in the *Routing plan.*

Stage III. All arcs that constitute route M_1 are removed from the structure $\vec{G}'$. Any pendant (isolated) vertices that arise as a result of removing these arcs are also eliminated. As a result, the structure $\vec{G}''$ is obtained (Figure 7), on which the next longest path is identified. To this end, a topological sorting of the vertices of graph $\vec{G}''$ is performed. The result is the following sequence of vertices: $p_2 \to p_5 \to p_8 \to p_1 \to p_4 \to p_3 \to p_7 \to \to p_6 \to p_{10} \to p_9$. The vertices of the graph are then traversed, and an individual weighted estimate is formed for each of them.

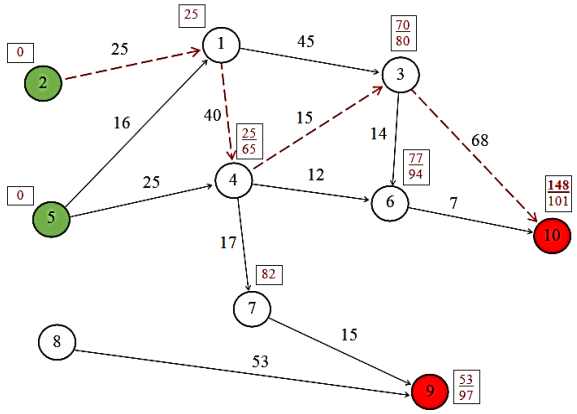


Figure 7 – Weighted graph and the longest path within its structure (indicated by a dashed line)

Vertices p_9 and p_{10}, which belong to subset B, have been assigned their respective estimates. The maximum route length to vertex p_9 is 97 units, while to vertex p_{10} it is 148 units. Thus, the longest route is selected, namely $M_2 = \{p_2, p_1, p_4, p_3, p_{10}\}$. Its absolute weight is $w(M_2) = 148$. The route is recorded in the *Routing plan.*

Stage IV, V. The actions performed at these stages are presented in a condensed form in Figure 8.

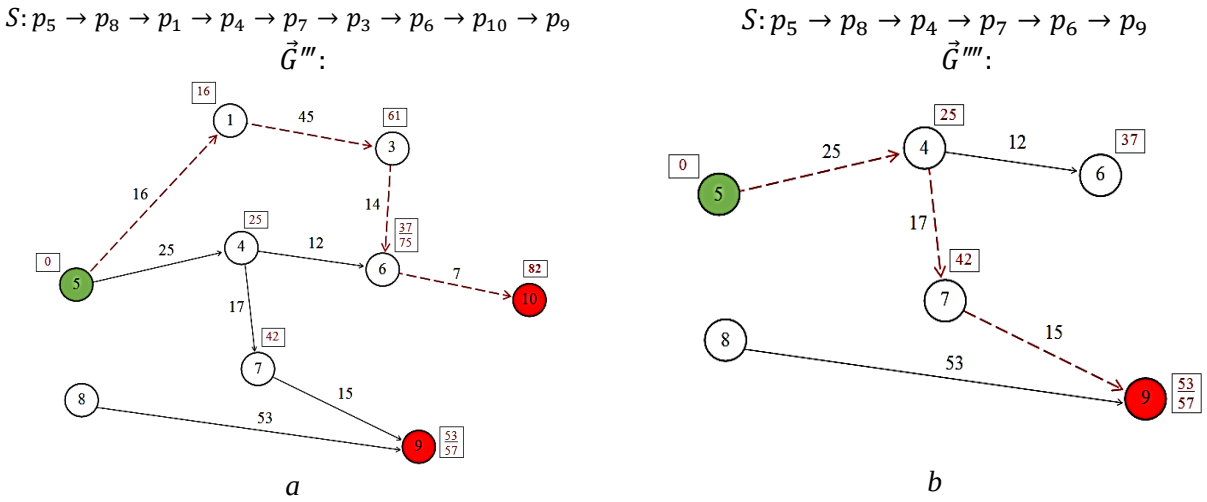


Figure 8 – Edge-simple longest paths (indicated by dashed lines): *a* – within the structure $\vec{G}'''$; *b* – within the structure $\vec{G}''''$

As a result of performing these stages, the following routes are obtained: $M_3 = \{p_5, p_1, p_3, p_6, p_{10}\}$ with a route weight of $w(M_3) = 82$ and $M_4 = \{p_5, p_4, p_7, p_9\}$ with a route weight of $w(M_4) = 57$.

After removing route M_4 from the structure $\vec{G}''''$, it can be verified that no paths remain between the vertices of subsets A and B (the transitive closure between them has been lost). This indicates that all edge-simple longest paths between the specified vertex sets in the structure of the original network object $\vec{G}'$ have been identified. Such paths here are referred to as zero-level paths, as they directly connect vertices from subsets A and B.

Thus, as a result of the successive decomposition of the initial graph $\vec{G}'$, namely $\vec{G}' \to \vec{G}'' \to \vec{G}''' \to \vec{G}''''$, four edge-simple longest paths (i.e., paths that do not share any arcs) have been identified within its structure, namely:

$M_1 = \{p_2, p_5, p_8, p_7, p_6, p_9, p_{10}\}$;
$M_2 = \{p_2, p_1, p_4, p_3, p_{10}\}$;
$M_3 = \{p_5, p_1, p_3, p_6, p_{10}\}$;
$M_4 = \{p_5, p_4, p_7, p_9\}$.

A careful examination of Figures 6, 7 and 8 leads to the conclusion that it is advisable to re-engineer the identified paths in order to reduce the dispersion in their length values with respect to weight. Under real operational and service-combat conditions, such path re-engineering is intended to achieve a certain balance in workload (i.e., the number of inspection objects along each route) among different search groups and to reduce the overall time required to conduct search operations.

The essence of the re-engineering procedure is as follows. If, between any pair of vertices from subsets A and/or B, there exists an arc connecting them, and this arc belongs to an extreme path that has the maximum (or relatively larger) length estimate compared to other paths, then it is advisable to reassign this arc to the path that has the minimum (or relatively smaller) length.

According to Figure 6 such arcs are (p_2, p_5) and (p_9, p_{10}). They belong to path $M_1 = \{p_2, p_5, p_8, p_7, p_6, p_9, p_{10}\}$, the total length of which is the largest among the other identified paths and equals $w(M_1) = 187$. At the same time, the length of the path $M_4 = \{p_5, p_4, p_7, p_9\}$ amounts to $w(M_4) = 57$. So, the difference in estimates between these paths will be $\Delta_{1-4} = 130$ units. Therefore, the arcs (p_2, p_5) and (p_9, p_{10}) should be removed from route M_1 and incorporated into route M_4.

After such re-engineering, the corresponding routes and their lengths will change as follows:

$M_1 = \{p_5, p_8, p_7, p_6, p_9\}$, $w(M_1) = 162$,
$M_4 = \{p_2, p_5, p_4, p_7, p_9, p_{10}\}$, $w(M_4) = 82$ and the difference in length between these paths will be $\Delta_{1-4} = 80$ units. Thus, the longest and shortest paths have been partially balanced by 50 units. At the same time, the total coverage of the paths in the initial structure $\vec{G}'$ remained unchanged and stayed at the level of 474 units. The set of paths determined after the re-engineering procedure is shown in Figure 9.

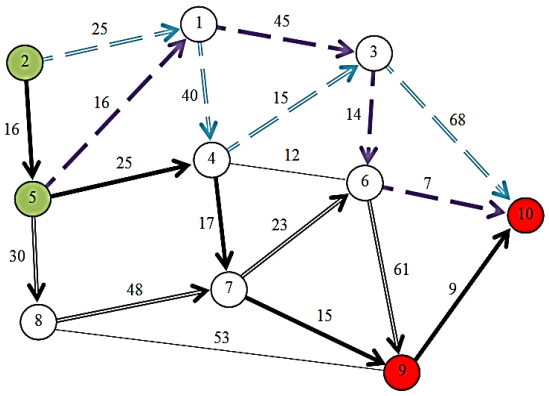


Figure 9 – Routing plan for the simultaneous traversal of the initial weighted graph between the selected vertex groups according to the criterion of maximizing the number of inspection objects (after path re-engineering)

The paths presented in Figure 9 are optimized (maximized) in terms of length and constitute the basis of the *Routing plan* with respect to the order in which the settlement is swept by the search groups.

The scheme presented in the figure provides the commander (supervisor) with the following information.

1. To conduct search operations within the settlement modeled by structure $\vec{G}'$, at least four search groups are required.

2. For each search group, the initial entry point (line of entry into the settlement) and the final exit point (line of withdrawal from the search) are clearly defined.

3. Each search group is assigned a specific search (movement) route within the settlement. As a result, a defined number of inspection objects (i.e., the "workload") are established for each group. The routes of different search groups do not intersect by arcs (streets), which enables such a MAS to perform the task efficiently.

4. Collectively, the search routes are optimized both in terms of movement direction and the number of inspection objects located along each route, which indirectly optimizes the overall time required to conduct the search within the settlement as a whole.

The summary evaluations of the developed routing plan based on a set of selected indicators are presented in Table 1.

Table 1 – Evaluations of the developed routing plan based on a set of indicators

No.	Parameter (Indicator)	Indicator Value
1	Number of search groups conducting operations within the settlement, N_{sg}, (u.)	4
2	Search routes, M_i	$M_1 = \{p_5, p_8, p_7, p_6, p_9\}$, $M_2 = \{p_2, p_1, p_4, p_3, p_{10}\}$, $M_3 = \{p_5, p_1, p_3, p_6, p_{10}\}$, $M_4 = \{p_2, p_5, p_4, p_7, p_9, p_{10}\}$
3	Number of inspection objects along the route, O_i, (u.)	$O_1 = 162$; $O_2 = 148$; $O_3 = 82$; $O_4 = 82$
4	Number of objects to be inspected under the implementation of a given *Routing plan*, O_{PL}, (u.)	474
5	Estimated time required to conduct the search for armed offenders within the settlement, T_{cs}, (hrs.)	27
6	Probability of detecting offenders under the implementation of a given *Routing plan*, P_{dc}	0.88
7	Capability of the formation to conduct search operations within the directive time frame, P_{sa}	0.58

Note. Initial data: initial weighted graph $\vec{G}'$; subsets of vertices $A = \{p_2, p_5\}$ and $B = \{p_9, p_{10}\}$; Optimization criterion for the routing plan: maximization of the number of inspected objects within the directive time. Direct time: 25 h (1500 min). Average inspection time per object along a search route: 10 min. Time for NGU forces to deploy to the operational area: 1 h. Area blocking time: 0.7 h. Time for search groups to move to the initial positions: 0.3 h.

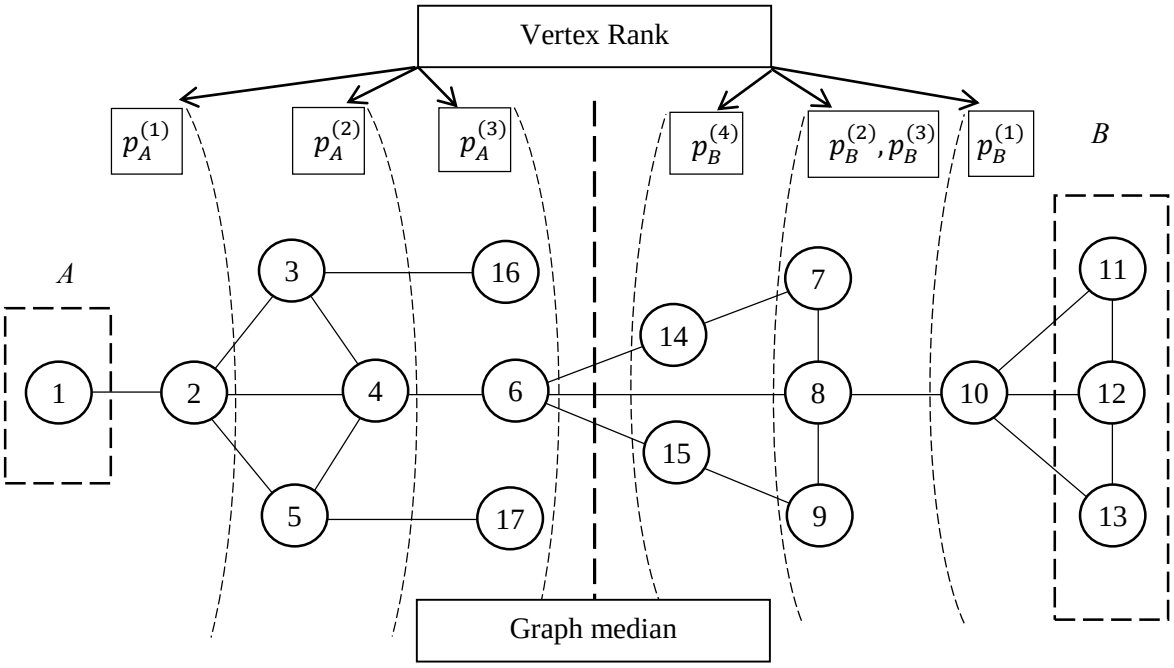


Figure 10 – Ranks of vertices that do not belong to subsets A and B(graph variant)

Let us introduce the following concept, namely the rank of a vertex p_i, which does not belong to subsets A and B. The rank of vertex p_i is defined as its edge (arc) distance from the vertices belonging to subsets A or B (Figure 10).

Thus, vertex p_2 has a rank equal to one, since it is located at a distance of one edge from vertex $p_1 \in A$. Vertices p_3, p_4, and p_5 have a rank equal to two, as they are located at a distance of two edges from $p_1 \in A$. Vertices p_{14} and p_{15} have a rank equal to four, as they are located at a distance of four edges from vertices $\{p_{11}, p_{12}, p_{13}\} \in B$ and so forth.

If, during the development of the *Routing plan*, the transitive closure between vertices $p_i \in A$ and vertices $p_i \in B$ is lost, then it is advisable to examine vertices with ranks 1, 2,…, k, that are located at a certain distance from vertices $p_i \in A$ and $p_i \in B$ for the presence of a transitive closure between them. If such a transitive closure exists, extreme paths between the specified vertices should be identified and added to the routing plan. In this case, the initial and terminal vertices of these paths will serve, respectively, as additional entry and exit points for the search groups. Such paths will be referred to as 1, 2, …, k paths, respectively. The upper bound of the rank k for additional vertices will be determined by the relevant commander (supervisor), who will make the operational decision to conduct the search activities on site.

Taking into account the features of searching for edge-simple longest paths in the structure of a networked object discussed above, Figure 11 presents, in a formalized manner, the structural scheme of the developed routing method for search groups on a stationary network aimed at the effective sweeping of settlements and the detection of armed offenders during stabilization operations in de-occupied territories.

Since, under certain conditions, the sweeping process may be conducted according to the criterion of minimizing the time required to carry out such operations, the developed routing method for search groups is also capable of identifying edge-simple shortest paths. For this purpose, the method employs an appropriate tool – Dijkstra's algorithm [15] (Figure 11).

Using the proposed method, the military command authority is able to develop rational *Routing plans* for the search for armed offenders in a settlement by NGU search groups. The resulting decision alternatives are optimized either according to the criterion of maximizing the number of inspected objects or according to the criterion of minimizing the time required to conduct search operations. The choice of the applied criterion is made by the commander (supervisor) based on the operational situation in the mission area and/or the area of responsibility, the available forces and assets, and the tasking received from the higher command authority.

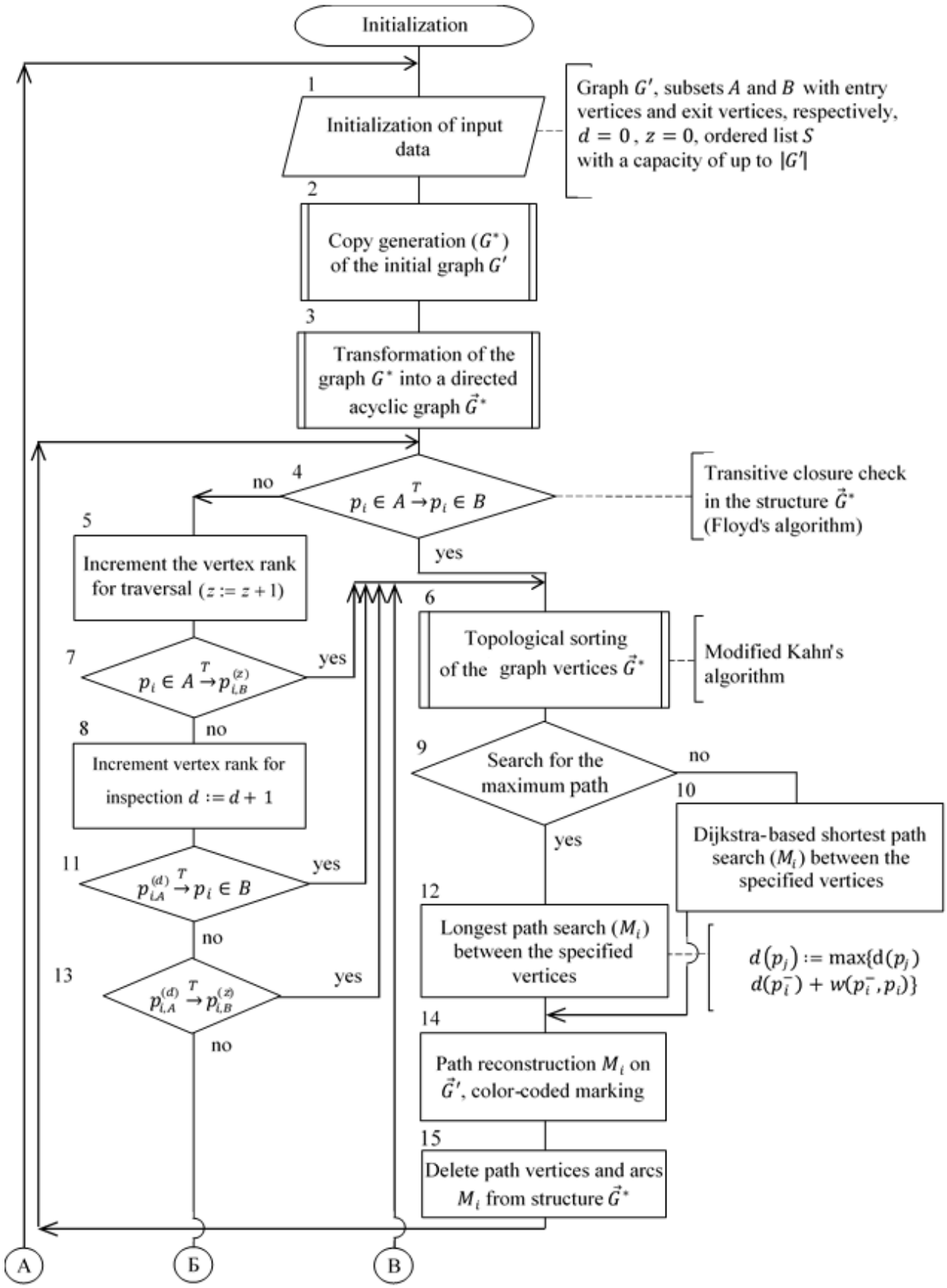


Figure 11 – Structural scheme of the proposed routing method for search groups on a stationary network

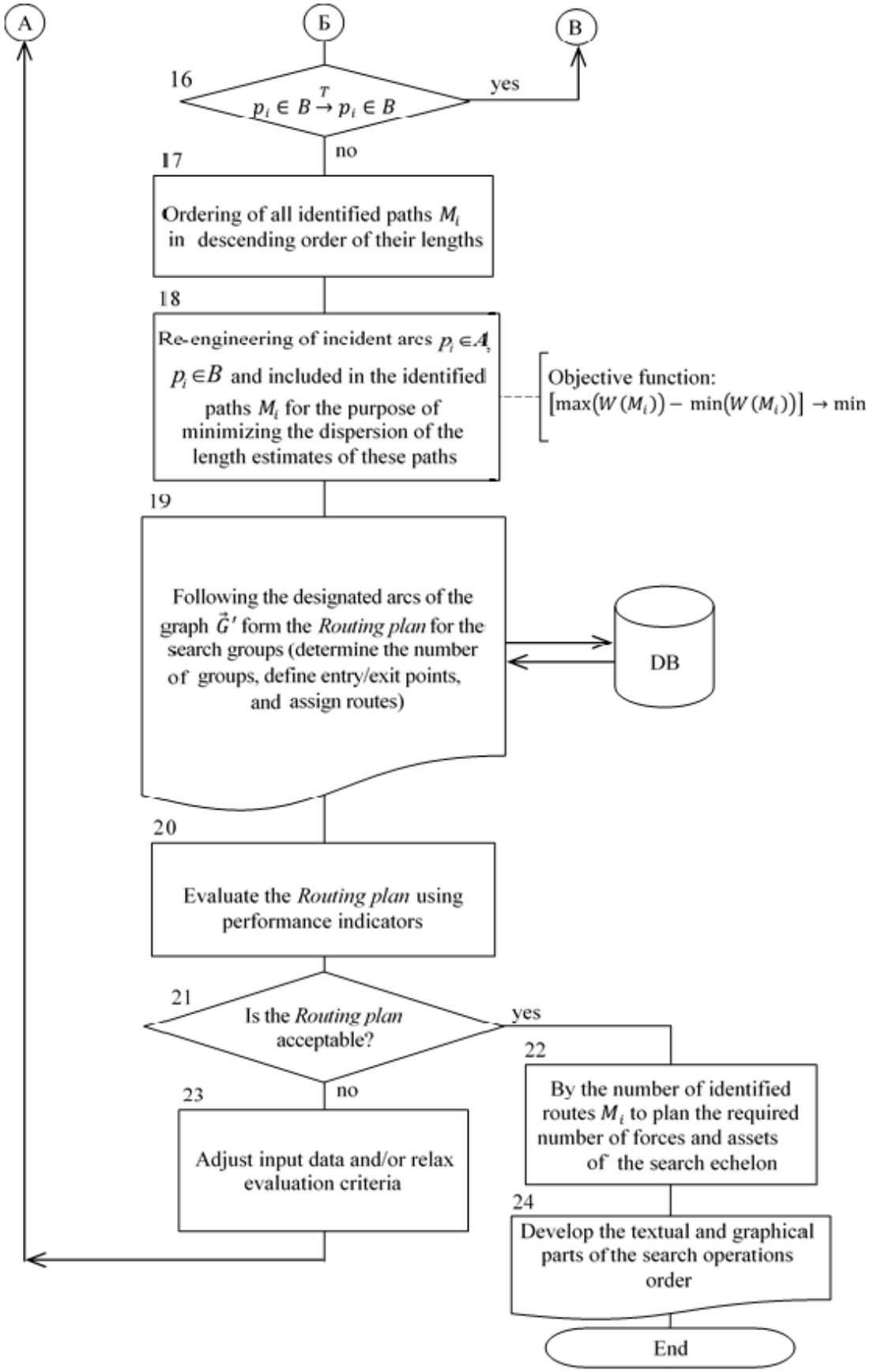


Figure 11, Sheet 2

The computational complexity of the algorithm implementing the developed method for solving the problem of identifying all longest edge-simple paths between specified pairs of vertices in the structure of a networked object exhibits a polynomial dependence and, in asymptotic terms, is estimated as $O(P \cdot \log P)$, where P – denotes the size (dimension) of the modeled graph. This computational complexity of the algorithm is due to the fact that at each iteration the number of vertices of the graph decreases due to the removal of the found extreme paths together with their corresponding vertices. Consequently, the algorithm is capable of generating decision-making alternatives in real time, which constitutes an important property for military applications.

Conclusions

The developed method is advisable for use by the military command authorities of the National Guard of Ukraine in the planning of search operations conducted as part of stabilization operations in de-occupied territories. The decision alternatives generated by the method will serve as the basis for the commander's decision on the conduct of search actions.

The development of the method is based on a comprehensive combination of several well-established tools. In particular, it is shown that solving the problem of identifying all edge-simple longest paths should be carried out by combining the dynamic programming approach with a longest-path search method. The identification of all edge-simple shortest paths is performed by integrating the dynamic programming approach with the well-known Dijkstra's algorithm.

The problem of finding longest paths in an arbitrary graph is NP-complete and currently admits no method for obtaining an optimal solution in real time. By its properties, the developed method is quasi-optimal, as its optimality is not formally proven in this article. The algorithm implementing the proposed method is characterized by polynomial-time complexity, which enables the rapid generation of decision-making alternatives. This capability is crucial for application in the context of military command and control.

Since the method is formulated in terms of graph theory, it employs combinatorial algorithms and is intended to operate on networked objects of sufficiently large scale and high connectivity density; therefore, its practical application without automation tools is not feasible. For effective use, the method requires the development of appropriate software within the framework of a geographic information system.

A further course of research within the scope of the problem addressed in this article should be considered the development of a routing method for search groups on a non-stationary network.

References

1. *Zakon Ukrainy "Pro Natsionalnu hvardiiu Ukrainy" № 876-VII* [Law of Ukraine about the National Guard of Ukraine activity no. 876-VII]. (2014, March 13). Retrieved from: https://surl.li/mzyjbr (accessed 23 January 2025) [in Ukrainian].

2. Morkvin D. A., Batsamut V. M., Hokh I. M. (2024). *Prohnozovani vyklyky, shcho stoiatymut pered derzhavoiu u pisliavoiennyi period: zavdannia NHU iz zabezpechennia derzhavnoi bezpeky ta vykonannia pravookhoronnykh funktsii* [Predicted challenges facing the state in the post-war period: tasks of the National Security Service to ensure state security and perform law enforcement functions]. *Bezpeka derzhavy*, no. 1, pp. 90–98 DOI: https://doi.org/10.33405/2786-8613/2024/1/3/309959 [in Ukrainian].

3. Kozalietov V. V., Batsamut V. M. (2025). *Problemni pytannia u diialnosti orhaniv upravlinnia Syl bezpeky pid chas planuvannia poshukovykh dii u naselenykh punktakh na deokupovanii terytorii Ukrainy: shliakhy vyrishennia* [Problematic issues in the activities of security forces management bodies in planning search operations in settlements on the deoccupied areas of Ukraine: solutions]. *Bezpeka derzhavy*, vol. 1, no. 5, pp. 47–55 DOI: https://doi.org/10.33405/2786-8613/2025/1/5/336712 [in Ukrainian].

4. Karger D., Motwani R., Ramkumar G.D.S. (1997). On approximating the longest path in a graph. *Algorithmica*, 18, pp. 82–98. DOI: https://doi.org/10.1007/BF02523689 [in English].

5. Feder T., Motwani R. (2005). Finding large cycles in Hamiltonian graphs. *Proc. of the 16th annual ACM-SIAM Symp. on Discrete Algorithms (SODA), ACM*, pp. 166–175. DOI: https://doi.org/10.1016/j.dam.2009.12.006 [in English].

6. Gabow H., Nie S. (2008). Finding long paths, cycles and circuits. *Proc. of the 19th annual International Symp. on Algorithms and Computation (ISAAC)*. LNCS 5369, pp. 752–763. DOI: https://doi.org/10.1007/978-3-540-92182-0_66 [in English].

7. Zhang Z., Li H. (2007). Algorithms for long paths in graphs. *Theoret. Comput. Sci*, 377, pp. 25–34. DOI: https://doi.org/10.1016/j.tcs.2007.02.012 [in English].

8. Bulterman R., van der Sommen F., Zwaan G., Verhoeff T. et al. (2002). On computing a longest path in a tree. *Inform. Proc. Lett*, 81, pp. 93–96. DOI: https://doi.org/10.1016/S0020-0190(01)00198-3 [in English].

9. Uehara R., Uno Y. (2004). Efficient algorithms for the longest path problem. *Proc. of the 15th annual International Symp. on Algorithms and Computation (ISAAC)*. LNCS 3341, pp. 871–883. DOI: https://doi.org/10.1007/978-3-540-30551-4_74 [in English].

10. Uehara R., Valiente G. (2007). Linear structure of bipartite permutation graphs and the longest path problem. *Inform. Proc. Lett*, 103, pp. 71–77. DOI: https://doi.org/10.1016/j.ipl.2007.02.010 [in English].

11. Takahara Y., Teramoto S., Uehara R. (2008). Longest path problems on ptolemaic graphs. *IEICE Trans. Inf. and Syst*, 91-D, pp. 170–177. DOI: https://doi.org/10.1093/ietisy/e91-d.2.170 [in English].

12. Ioannidou K., Mertzios G., Nikolopoulos S. (2011). The Longest Path Problem has a Polynomial Solution on Interval Graphs. *Algorithmica*, vol. 61, pp. 320–341. DOI: https://doi.org/10.1007/s00453-010-9411-3 [in English].

13. Mertzios G. B., Corneil D. G. (2012). A Simple Polynomial Algorithm for the Longest Path Problem on Coco Graphs. *SIAM Journal on Discrete Mathematics*, pp. 940–963. DOI: https://doi.org/10.1137/100793529 [in English].

14. Arthur B. Kahn (1962). Topological sorting of large networks. *Communications of the ACM*, 5 (11), pp. 558–562. DOI: https://doi.org/10.1145/368996.369025 [in English].

15. Dijkstra E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, vol. 1, iss. 1, pp. 269–271. DOI: https://doi.org/10.1007/BF01386390 [in English].

The article was submitted to the editorial office 19.12.2025
Accepted for publication after peer review 16.01.2026
Publication date 29.05.2026

УДК 519.853: 658.52

В. В. Козалєтов, В. М. Бацамут

МЕТОД МАРШРУТИЗАЦІЇ ПОШУКОВИХ ГРУП НА СТАЦІОНАРНІЙ МЕРЕЖІ ДЛЯ ЕФЕКТИВНОГО ПРОЧІСУВАННЯ НАСЕЛЕНИХ ПУНКТІВ І ПОШУКУ ОЗБРОЄНИХ ЗЛОЧИНЦІВ

Актуальність статті зумовлена потребою у подальшому розвитку моделей колективної поведінки систем із мультиагентною організацією структури, у наділенні таких систем інтелектом, який забезпечує синхронізацію спільних зусиль різних агентів у ході досягнення поставлених перед системою цілей. З використанням запропонованого методу можливо вирішити проблему неефективного обходу пошуковими групами (агентами мультиагентної системи) населеного пункту, що є важливим у ході виконання пошукових, рятувальних, моніторингових завдань у кризових районах різного характеру походження.

У методі застосовано ідею динамічного програмування, яка полягає у послідовному поділі початкової структури транспортної мережі населеного пункту на низку підструктур. У межах кожної підструктури відшукується реберно-простий найдовший/найкоротший шлях між парою визначених вершин. Поділ структури продовжується доти, доки не буде втрачено транзитивну зв'язність між точками входу і точками виходу із пошуку, визначених у структурі населеного пункту. Сукупність окреслених екстремальних шляхів складає Маршрутний план для пошукових груп у ході проведення пошукових заходів, що, в певному розумінні, упорядковує прочісування населеного пункту. За своїм характером розв'язувана задача є NP-повною, а розроблений метод належить до класу квазіоптимальних. Обґрунтовано поліноміальну оцінку обчислювальної складності комбінаторного алгоритму, що реалізує цей метод.

Метод розроблено й викладено в термінах теорії графів і теорії множин та подано у статті у формалізованому вигляді. Порядок вироблення Маршрутного плану пошуковим групам за цим методом показано на практичному спрощеному прикладі.

Метод значно зменшує трудовитрати на вироблення раціонального варіанта пошукових дій і призначений для використання в органах військового управління Національної гвардії України під час планування пошукових заходів у певному населеному пункті на деокупованій території країни.

Ключові слова: *прочісування, пошукові групи, мультиагентна система, маршрутизація, мережевий об'єкт, зважений неорієнтований (орієнтований) граф, екстремальні шляхи, критерій оптимізації, метод.*

Kozalietov Vitalii – Adjunct, National Academy of the National Guard of Ukraine
https://orcid.org/0009-0007-9540-8277

Batsamut Volodymyr – Doctor of Military Sciences, Professor, Deputy Head of the Institute for Scientific Work – Head of the Research Laboratory of Service and Combat Application of the NGU of the Educational and Scientific Institute for Ensuring State Security, National Academy of the National Guard of Ukraine
https://orcid.org/0000-0003-2182-6891