Inverse and Direct Maxflow Problem Study on the Free-Oriented ST-Planar Network Graph

Victor Tikhonov[1], Serhii Nesterenko[2], Abdullah Taher[3], Olena Tykhonova[1], Olexandra Tsyra[1], Olha Yavorska[1] and Kateryna Shulakova[1,4]

[1]*Department of Computer Engineering and Information Systems, State University of Intelligent Technologies and Telecommunications, Kuznechna Str. 1, Odesa, Ukraine*

[2]*Department of Computer Intellectual Systems and Networks, Odesa Polytechnic National University, Shevchenko Avenue 1, Odesa, Ukraine*

[3]*Department of Electronic and Communication Engineering, College of Engineering, Al-Qadisiyah University, Iraq*

[4]*Anhalt University of Applied Sciences, Bernburger Str. 55, Köthen, Germany*

{victor.tykhonov, elena.tykhonova}@suitt.edu.ua, sa_nesterenko@ukr.net, abdallahqays@gmail.com, aleksandra.tsyra@gmail.com, yavorskayao7@gmail.com, katejojo29@gmail.com

Keywords: Telecommunication Network, Maximal Flow, Free Oriented Planar Graph, SDN.

Abstract: The issues of data flow optimization in telecommunication networks are considered. The analyses of the problem state of art shows the primarily utilization of logistic Maxflow model on ST-planar directed network graph with predetermined fixed metric. Concluded, that conventional logistic Maxflow model is not adequate to modern telecoms with flexibly reconfigured channels. Introduced the concept of the free-oriented network graph as an enhanced math-model for digital flows simulation. The inverse and direct Maxflow tasks are formulated on the normalised free-oriented ST-planar network graph, and the properties of the graph obtained as functions of vertices number. The direct Maxflow task is studied in tensor form, and the algorithm of test-sequences generation for the inverse Maxflow task is constructed. The inverse Maxflow problem has been analyzed as a discrete optimization task on the Pontryagin maximum principle with two necessary extremum conditions. Related computation algorithm is built with polynomial complexity. Unlike the known approaches, proposed method is relevant to data flow optimization in the software defined networks with dynamically reconfigurable channels. Along with the maximal flow, the flow distribution over the network structure provided. The formalism of the direct Maxflow task can be used for testing the algorithms of inverse Maxflow task solutions, and generation the training sequences for machine learning in AI models.

1 INTRODUCTION

An efficient way to reduce the cost and improve the quality of telecommunication service is increasing the network productivity by optimal scheduling the digital flows to gain better equipment utilization. This task is known as the Maxflow optimization problem. Historically, the Maxflow problem (MFP) arose in logistic systems to deliver some unimodal product (e.g. gas, oil, water etc) from a producer site S to consumer target T. A feature of such systems is that products should not simultaneously be sent in opposite directions, and transportation channels from source S to target T should not intersect with each other; besides, there should be no internal flow generators or accumulators other than S and T.

The Maxflow problem conventionally involves searching a feasible flow on the given ST-directed weighted planar network graph with single-product flows and two unique vertices – the flow generating source S and flow accumulating sink T. Such a graph presumes that all the graph-arcs have fixed weights. Denote this type of graph ST-DWPG.

The ST-DWPG graph as a generic logistic-system presentation with fixed arcs-weights still remains to be a common model for digital flows optimization. However, it is not fully adequate to modern telecommunication data networks with dynamic reconfiguration of channel capacities, potentially enabling the overall network performance increase.

This work intends to advance the Maxflow study in data networks, using the feature of dynamic reconfiguration in digital communication channels.

Section 2 of this work considers the Maxflow problem state of the art. Section 3 formulates the objectives of the work. Section 4 introduces a dual formalization of the Maxflow problem as traditional (inverse) task of feasible flow finding and a coupled (direct) task of generating the testing sequences for MFP solution algorithms. Section 5 gives definition of the direct Maxflow task in tensor form. Section 6 provides a discrete analysis of the inverse Maxflow problem on the ST-planar free-oriented graph. Section 7 summarizes results of the work.

2 THE MAXFLOW PROBLEM STATE OF THE ART

The widely known solution of the Maxflow problem (MFP) is Ford-Fulkerson algorithm (FFA), also referred to as "method" [1] (1956). It uses the Depth-First path Searching technique (DFS), and works by iteratively finding an augmenting ST-path in residual graph (obtained by subtracting the current flow from all the arc-capacities along the flow-path). The optimum-criterion at any iteration is "maximal flow-increment along the path".

The FFA stops when no more paths exist. It has a pseudo-polynomial run-time complexity O(F*E), where F is the maximal ST-flow, E is the number of arcs in the ST-DWPG graph.

Based on FFA, alternative approaches have been developed. Among them, a very popular and strongly polynomial algorithm for Maxflow problem solution was proposed by E. Dinitz [2] (1970). In contrast to FFA, it provides the vertex-based Breadth-First path Searching (BFS) technique to find a single source shortest path in an unweighted graph. It has the $O(V^2*E)$ computational complexity; V, E – vertices and edges numbers.

Similar to Dinitz, another FFS-based MFP-algorithm was published by J. Edmonds and R. Karp in 1972 [3]; it has $O(V*E^2)$ computational complexity. For certain network topology it outperforms the Dinitz method in computation time. The theoretical foundations of the Maxflow problem in the context of general graph theory applications have been outlined in [4] (1976).

A new push–relabel Goldberg-Tarjan algorithm for MFP solution proves that Dinitz blocking-flow method needs not more than $O(\min\{V^{2/3}, E^{1/2}\}$ augmentations in the unit capacity case [5] (1986). In particular case of capacity scaling and ingenious data structures, the run-time overcomes the O(V*E) barrier in Goldberg-Rao algorithm [6] (1998).

In 2006, an improved Goldberg-Rao algorithm for the maximum flow problem was exhibited by D. Papp [7]. It works not only for unit-capacity network graphs (as Dinitz algorithm does), but for more general case of binary weighted arks.

An original algorithm to approach the feasible flow in near linear time was proposed by P. Christiano et al for weighted undirected ST-graph. It uses the well-known Laplacian matrix method to solve a system of linear equations for electrical flow [8] (2010). The work claims developing the fastest known algorithm for computing approximately maximum S-T flows.

An enhanced formalism for combinatorial optimization problems like MFP is presented by M.A. Rajouh based on Pontryagin maximum principle. It formulates a set of necessary extremum-conditions kept along the algorithm iteration, that enables to curtail the initial diversity of iterations to a compact set of candidates for optimum. In some cases, the necessary conditions prove sufficient [9] (2013).

A solid overview of network flow algorithms on the closed bipolar ST-directed weighted graph reflects the book [10] (2019) by D.P. Williamson. The book extends the scope of network flow study for multicommodity task, and gives the detailed analysis of particular two-commodity case. Also, the notion of ST-flow in undirected graph is defined, where any edge has an arbitrary positive orientation and consist of two opposite-directed arcs of equal capacities.

Generalized MFP-statement to simulate the information flows in software defined network architecture is introduced in [11] (2019). In contrast to conventional logistic-network model in the form of bipolar closed ST-directed weighted planar graph (ST-DWPG), it defines the overall network capacity and its maximal flow on the open 3-pole non-planar free-oriented network graph. Related algorithm for the 6-vertex graph is presented in [12] (2019).

An important class of electric-power flows optimization (the Equal Maximum Flow Problem – EMFP) has been studied in [13] (2020). The EMFP aims for a maximum equal flow on all edges in a subset of the whole network graph edge-set. The paper extends the EMFP to Almost Equal Maximum Flow Problems (AEMFP) where electric-flow values differ according to a certain deviation-function.

A fast and near optimal algorithm for optimization of the end to end routing in SDN-network is proposed in [14] (2021), based on the max-flow/min-cut theorem. It shows that the maximal flow searching technique with minimal hops paths provides better performance compared to the traditional shortest path algorithms.

To optimize the complex multimodal flows in logistic systems, an application of genetic algorithm described in [15] (2022), which is generally used to solve the Vehicle Routing Problem (VRP). It simulates Darwin's natural evolution in transportation network model, and includes five recycling steps (task coding, population initialization, selective operations definition, mutation, selection process).

The overview of deterministic Maxflow and Min-cost flow logistic optimization problems for ST-directed weighted graph is given in [16] (2023) with emphasis on the worst-case of running time. The paper solely reflects the algorithms having a faster run-time than earlier published. It noticed, that Goldberg–Rao algorithm has a weakly polynomial run-time; the existence of a strongly polynomial O(V*E) algorithm was shown by Orlin in 2013, Orlin and Gong in 2021. Besides, interior-point methods (IPMs) for MFP-solutions considered, as well as the algorithms for minimal flow-cost.

A promising direction in solving complex combinatorial problems such as MFP is the use of artificial intelligence (AI) and machine learning (ML) technologies. A survey of the recent attempts in this area published in [17] (2021). Due to the hard nature of these problems, the surveyed algorithms rely on the handcrafted heuristics for making decisions, that otherwise are too expensive to compute or mathematically not well defined. Thus, machine learning looks like a natural candidate to make such decisions in a more principled and optimized way.

A set of single-machine scheduling problems with resource-dependent processing times is studied in [18] (2021). Heuristic algorithms for solving the reduced to unified model problems are presented. Three types of scheduling tasks are specified: due dates to jobs assignment, resources allocation to job operations and machines jobs scheduling.

Various researches on graph-based tasks deep learning are surveyed in [19] (2022). The paper focuses directed/undirected wired and wireless SDN networks as the most promising solution driving the networking industry to re-examine traditional network architecture. To track the follow-up research, a public GitHub repository is created, where the relevant papers will be updated continuously.

The ML-based approach to MFP solution presents the N. Orkun Baycik publication, where the tree-based learning method is applied. It includes decision tree and random forest regression (supervised learning algorithm and bagging technique, that uses an ensemble learning method). Both trees are built independently [20] (2022).

A network based intelligent node labelling (INL) algorithm for solving the Maxflow problem in directed network graph is developed in [21] (2023). It eliminates common augmenting paths technique to compute the maximal flow. Instead, it tries to balance input-to-output flow values for all the intermediate nodes, thus avoiding the excess or stagnant flow and reduction of the under-utilized outflow arcs. The algorithm needs at most two iterations to transform the initial N-nodes network into an equivalent network with O(V*E) worst-case complexity.

Summarizing the spoken above methods, algorithms and approaches to Maxflow problem solution, we conclude the following:

1) The most of the surveyed works on the Maxflow problem explore the so called "logistic model" of transportation system in the form of directed bipolar ST-graph with a source node S and a target node T [1-7, 10, 14-16, 19, 21]; also, undirected graphs are used to simulate electrical circuits and networks ([8, 13]). The case of 'multi-source/multi-target' graph (aka bipartite graph) can be easily reduced to a bipolar ST-graph. The case of undirected graph is commonly understood as a symmetrically directed graph.
2) The conventional Maxflow model of logistic system is not really adequate to modern data networks with dynamic reconfiguration of optical and wireless communication channels, and therefore, prevents to benefit the digital channel reconfigurability for data network performance. A further development of logistic flow model towards the SDN networking architecture is given in [11-12] by the concept of the 'free-oriented weighted graph' along with related algorithm of the Maxflow computation over the 3-pole/6-node open graph.
3) Recently, artificial intelligence (AI) and machine learning (ML) techniques have been increasingly used to solve complex combinatorial problems on graphs [17-21]. An important inductive AI-method is supervised learning (input objects and desired output-values for machine learning an AI-model). In terms of MFP, a graph sample G is an input, and the maximal flow distribution F on the graph G is the output. So, the sequence {G, F} can be used for ML. Yet, finding F for G is a hard problem; instead, constructing the graph G(F) for a given F is a rather trivial task. In all, generating a complete set {G(F)} is equivalent to building a sufficient ML-sequence {G, F}.

Based on the surveyed publications analysis, the objectives of this work are formulated further on with emphasis on data flow study in bi-polar networks.

3 OBJECTIVES OF THE WORK

This work aims to take use of the dynamic reconfiguration ability of the modern telecommunication channels for data networks performance increase. To achieve this, the following objectives have been set:

1) Formalization of the inverse and direct Maxflow problem on the free-oriented bi-polar ST-planar network graph with multiple vertices. This includes normalization of the complete ST-planar graph in visual and matrix forms with the distinguished topology and metric, studying the principal properties of the normalized graph, setting the direct and inverse Maxflow tasks.
2) Definition of the direct Maxflow task in tensor form. The set of ST-paths in the graph is given as a system of vectors (tensor) in the N-dimensional orthonormal Euclidean space, where N is the full number of edges in ST-planar graph.
3) Discrete analysis of the inverse Maxflow task on the free-oriented ST-planar network graph as an NPC-combinatorial optimization task, aimed to find the feasible flow between the two open graph poles, along with the flow distribution tensor.

4 FORMALIZATION OF THE INVERSE AND DIRECT MAXFLOW PROBLEM ON THE FREE-ORIENTED ST-PLANAR NETWORK GRAPH

The concept of the free-oriented weighted graph (FWG) with reconfigurable edges was exhibited in 2019 to apply the known MFP-algorithms to digital flows simulation in telecoms [11]. In contrast to other methods, the weight of any FWG-graph edge is assumed to be equal the fixed capacity of a duplex digital channel, that is divided in any proportion between the two coupled simplex-channels, on order to fully utilize the overall channel capacity.

Besides, the solution of the FWG-based Maxflow task does not only involve finding the feasible flow F_{max}, but also the distribution of this flow $D(F_{max})$ over ST-paths of the graph. Related algorithm for calculation F_{max} and $D(F_{max})$ on the 3-pole/6 vertex FWG-graph is presented in [12].

In this regard, there is a need in adaptation of the conventional ST-DWPG model of a logistic system for digital flows simulation in data networks with flexible configurable channels.

In graph theory, graph G is a cortege of binary relations g(k, m) on the set of vertices V: G(V):= {g(k, m)}, k, m∈V. If g(k, m) ≡ g(m, k), then G is undirected graph; else G is directed graph [22]. That means: a) G is directed graph, if at least one g(k, m) ≠ g(m, k); b) undirected graph is equivalent to symmetrically directed graph; c) the Ford-Fulkerson algorithm FFA is valid for both directed and undirected planar graphs.

Consider conventional ST-planar directed weighted graph G(V) of 4 vertices {S, T, 2, 3} and 6 weighted arcs: $g_{S,2} = g_{3,T} = 1$; $g_{S,3} = g_{2,T} = 3$; $g_{2,3} = g_{3,2} = 1$. It is easy to see, that here $F_{max} = 3$, wherein residual graph RG has 3 arcs: $g_{S,3} = g_{2,T} = g_{2,3} = 1$; the arc $g_{2,3}$ can't be used to augment the flow.

Instead, the equivalent free-oriented weighted model FWG(V) of the same case has 5 edges $g_{S,2} = g_{T,3} = 1$; $g_{S,3} = g_{T,2} = 3$; $g_{2,3} = 2$. By this model, F_{max} is increased to 4, wherein residual graph RG falls down to null. This illustrates the privilege of coherent transition from the common logistics directed graphs to the free-oriented graph models in telecoms.

Definition of normalized ST-planar graph. Consider the visual and matrix forms of the ST-planar free-oriented weighted graph (FWG); the first one is less formal but intuitively clear; the latter is more convenient for digital processing. It is clear, that both forms of the graph are invariant towards the vertex's names. Let graph G(V) has digital vertex's indexes 0, 1, 2, ..., V-1 (V is the total vertex's quantity). Let indexes 0 and 1 reserved for S and T network nodes (we label them as S0 and T1 in visual form). In these terms, Figure 1 depicts normalized 6-vertex ST-planar visual graph; figure 2 shows this graph in matrix view; the graph edges have numeric names (1÷12).

The visual graph framework (Figure 1) contains the nested sub-graphs, regularly growing by adding new vertices 2, 3, ... to the initial primitive graph of two adjacent vertices (S0, T1) with two outer open edges. Vertices S0 and T1 simulate digital network border gateways for transfer the data flow F in both directions via the open edges. This framework has maximal number of edges (12) for a 6-vertices planar graph. The graph topology in Figure 1 is shown by edge-lines, while the metric is given by the edge values.

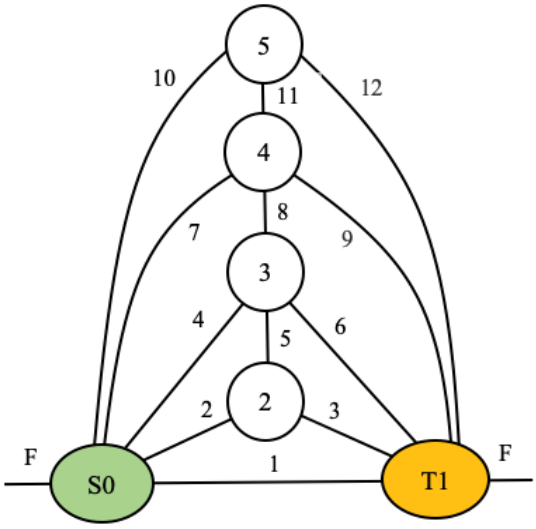

Figure 1: Normalized ST-planar visual graph.

The matrix graph (Figure 2) is divided in three sections: right-upper (topology), left-down (metric), diagonal (open edges). The yellow cells in Figure 2 indicate three eliminated edges of complete planar 6-vertex graph with 12 edges compare to complete non-planar graph with 15 edges. In digital presentation of matrix graph, names "S0", "T1" are to be substituted by 0 and 1 to adhere the Python array indexing [23].

	S0	T1	2	3	4	5
S0	1	1	1	1	1	1
T1	1	1	1	1	1	1
2	2	3		1	0	0
3	4	6	5		1	0
4	7	9	0	8		1
5	10	12	0	0	11	

Figure 2: Normalized ST-planar matrix graph.

In classic analysis, 'topology' T is a collection of subsets X` of the given set X, if: {∅, X∈T; ∀(U, V∈T) U∩V∈T; $\forall_I \cup(V_I \in T) \in T$}, where Ø is empty subset. The pair (X, T) is called 'topological space'. The subsets X`∈T are called 'open subsets' [24].

Considering the FWG-graph model, we interpret 'topology' as vertices connectivity in the free-oriented graph, where ∅ is open network environment; the 'open subsets' X` are bound sub-graphs that have finite paths to the network-outside via the open poles S and T; the latter simulate the network border gateways.

The visual graph in Figure 1 is convenient for studying core properties of ST-planar network graph as functions of the vertex quantity (V), which are important to formalize the inverse and direct MaxFlow problem in telecoms.

The number of inner edges. In case of non-planar complete graph, the number of inner edges equals $E_{NPG}(V)=V(V-1)/2$. Graph with $V \leq 4$ is always planar, so $E_{NPG}(3)=E_{PG}(3)=3$. In Figure 1, adding any next vertex to initial 3-vertex graph increments the total number of planar graph edges E_{PG} in 3 edges: $E_{PG}(3+1)=3+3=6$; $E_{PG}(4+1)=6+3=9$; $E_{PG}(5+1)=9+3=12$ and so on. In general, $E_{PG}(V)=3(V-2)$, $V \geq 3$.

The number of ST-paths. The ST-planar graph with 3 vertices (S0, T1, 2) in Figure 1 has two ST-paths: one 1-hop-path 'S0-T1' and one 2-hop-path 'S0-2-T1'. Each next vertex V adds two paths more: one 2-hop-path 'S0-V-T1' and one h-hop-path ($h \geq 3$) through the 'vertical' vertex (V-1, V), e.g. paths 'S0-3-T1' and 'S0-2-3-T1'.

Thus, the number of ST-paths in complete planar graph is $P(V)=2(V-2)$, $V \geq 3$. This includes: one 1-hop-path, (V-2) of 2-hop-paths, (V-3) of h-hop-paths with $h>2$; in sum: $1+(V-2)+(V-3)=2(V-2)$, $V \geq 3$. Any next path p in the list of counted above paths P(6)=8 in ST-planar graph (Figure 1) has at least one edge, that is new -added to the previously listed edges (yellow marked cells in Figure 4), and in bold-font here: {p}=(**1**), (2, **3**), (**4,** 6), (7**, 9**), (**10,** 12), (2, **5,** 6), (4, **8**, 9), (7, **11**, 12).

Distribution $P_V(h)$ of h-hop paths. We bring without proof the empirical formula: $P_V(h)=1$, if h=1; $P_V(h)=V-h$, if h=2, 3, ..., V-1; e.g. $P_6(h=1, 2, 3, 4, 5)=1, 4, 3, 2, 1$. The set of eight shortest paths includes: $P_6(h=1, 2, 3) = \{1, 4, 3\}$. The set of 8 max-diverse and short paths is {$P_6(1)=1$; $P_6(2)=4$; $P_6(3)=1$; $P_6(4)=1$; $P_6(5)=1$} formed by edges {(**1**); (2, **3**); (**4**, 6); (**7**, 9); (10, **12**); (2, **5**, 6); (2, 5, **8**, 9); (2, 5, 8, **11**, 12)} with 8 backward-unique edges in bold. It can be argued, that the number of paths P(V)={p} equals the total number of additive flows f(p) in complete ST-planar graph G(V). Figure 3 summarizes the spoken above properties of ST-planar network graph.

V	3	4	5	6	Function
E_{PG}	3	6	9	12	3(V-2)
P(V)	2	4	6	8	2(V-2)
$P_V(h=1)$	1	1	1	1	1
$P_V(h=2)$	1	2	3	4	V-h
$P_V(h=3)$	0	1	2	3	V-h
$P_V(h=4)$	0	0	1	2	V-h
$P_V(h=5)$	0	0	0	1	V-h
$P_V(h<V)$					V-h

Figure 3: ST-planar free-oriented graph properties.

The known MFP-algorithms suppose predefined fixed arcs-weights, and thus, do not correctly work on the FWG-model with flexible scheduled arcs. We propose here an alternative approach to MFP study in the form of the direct and inverse Maxflow task.

The ordinal direct Maxflow task (O-DMT) we set as 'how to construct the free-oriented ST-planar graph G(F) with minimal edges` capacities to provide the given total flow distribution F over the set of ST-paths. The inverse Maxflow task (IMT) we set as 'how to find the maximal feasible ST-flow distribution F(G) over the set of ST-paths on the given free-oriented ST-planar graph G.

Generic inverse maxflow task IMT implies scanning all the ST-paths on the network graph topology. The total number of paths P(V) as function of vertices quantity V has a faster than polynomial growth even for planar graphs. Such computational tasks are referred to as nondeterministic polynomial complete problems (NPC) [13].

To solve complex combinatorial optimization problems of NPC-type, heuristic approaches are often used, which do not guarantee the best results, but may have less complexity, calculation time and produce adequate outcomes [18].

One of the famous and widely used heuristic method is Pontryagin maximum principle. This is choosing a set of necessary extreme conditions kept at any point of the object phase-track or the algorithm iteration, aimed to curtail the initial solutions diversity to a compact set of candidates for optimum. In some cases, the necessary conditions prove sufficient [9].

Solving complex NPC problem by heuristic method needs reliable proof the result obtained. The rigorous proof not always feasible, and empirical verification often necessary using various testing tools. Here, we propose an idea 'Verify complex inverse task algorithm by simple direct task samples generation'.

The spoken above ordinal direct Maxflow task O-DMT is rather simple for particular given flow F-distribution over ST-graph G. In general, the direct Maxflow task (DMT) turns into non-trivial case of 'generation a comprehensive sequence {F, G} with input F and output G for IMT-testing with inversed input/output {G, F}'.

Today, more and more human tasks addressed to artificial intelligence (AI) with digital neural networks, that capable to be taught by machine learning (ML) techniques. The matter is, how to construct training samples sequences (TSS) for AI-models comprehensive teaching.

Let S = (G, F) a distinct DMT-task output, which can be used for an IMT-case testing. In these terms, we define {S}= {(G, F)} as the testing/training samples sequence (TSS) for testing the inverse Maxflow task algorithms on the free-oriented ST-planar graph.

Towards the Maxflow problem in the context of AI, we propose the DMT-output generator of IMT-tests {G, F} to be used as a training samples sequence (TSS) for machine learning of an AI-model. The core issue of the direct Maxflow task (DMT) is, how many samples {S} needed for complete testing the IMT-algorithms or exhaustive MFP machine learning.

5 DEFINITION OF THE DIRECT MAXFLOW TASK IN TENSOR FORM

We formulate a particular case of direct Maxflow task (DMT) on the bi-pole planar FWG-graph as following: 'Find metric M of network graph G(V) with arbitrary number of vertices V, which is relevant to given flow distribution F(P) over the set P(V)={p} of paths $p \in P(V)$ in the given ST-planar free-oriented network graph G with complete topology T'. The 'complete topology' means, that all the possible edges E_{PG} of the ST-planar graph G(V) are included (Figure 3).

Consider the complete planar graph G(V) with V=6 vertices (Figure 1). The graph topology T is presented by the set of 12 edges E_{PG}={e} identified by their digital values 1÷12 in matrix graph (Figure 2). All the edges $e \in E_{PG}$ are mutually independent; let each of them be unitary (topological) vector **e**, and the set **E**:={**e**} be the Euclid vector basis. Graph G(6) has 8 paths P(6)=8, and each path $p \in P(V)$ possess at least one backward-unique edge e_f to carry augment flow f(p). The set of 8 shortest paths is a system of vectors (tensor) **P**={**p**} in matrix view (Figure 4); here, empty cells are zeros, **p**(1)=**e**(1); **p**(2)=**e**(2)+**e**(3) and so on. In this case, **P** includes: a single 1-hop path, four 2-hop paths and three 3-hop paths.

The matrix of scalar products {**p**(k)×**p**(m)} is *paths-metric tensor* M_P of **P** (Figure 5). It is evident, that the first five vectors of **P** are mutually orthogonal, as well as vectors number 6 and 8. The system **P** with $M_P>0$ we call the *path's tensor*. It is easy to show, that M_P is positive matrix [25].

Let $\{f_P(k)\}$ be the distribution of the entire ST-flow F_Σ over the paths p(k) of the graph G(V),

Figure 4; F_Σ=sum{$f_P(k)$}; **f**:={$\sqrt{f_P(k)}$} the *flow row-vector* with scalar product (**f**×**f**)=F_Σ; **F**:=diag(**f**) the diagonal matrix denoted as *flow-tensor*.

p(k)	e(n)											
n / k	1	2	3	4	5	6	7	8	9	10	11	12
1	1											
2		1	1									
3				1		1						
4							1		1			
5										1		1
6		1			1	1						
7						1	1	1				
8							1				1	1

Figure 4: System P of path' vectors in ST-planar graph.

The metric tensor M_F=(**F**×**F**) is distribution of the total flow F_Σ over the paths **P**={**p**}; F_Σ = trace(M_F)= (**f**×**f**). On this premise, we define the *flow-paths tensor* $\mathbf{F_P}$:=**F**×**P**. For the graph in Figure 1, the $\mathbf{F_P}$ is (8×12) matrix, where each k-row element is multiplied by the correspondent $\sqrt{f_P(k)}$ value.

n / k	1	2	3	4	5	6	7	8
1	1	0	0	0	0	0	0	0
2	0	2	0	0	0	1	0	0
3	0	0	2	0	0	1	1	0
4	0	0	0	2	0	0	1	0
5	0	0	0	0	2	0	0	1
6	0	1	1	0	0	3	1	0
7	0	0	1	1	0	1	3	1
8	0	0	0	0	1	0	1	3

Figure 5: Metric tensor M_P of path's P vector system.

The convolution g:=**f**×$\mathbf{F_P}$ we define as network *flow-load metric*. For the graph G in Figure 1, g is the row-set with 12 flow-loads of G-edges. If $f_P(k)\equiv 1$ and **P** in Figure 4, then **g**=(1, 2, 1, 1, 1, 3, 3, 1, 1, 1, 1, 2).

Now, the direct Maxflow task for ST-planar free-oriented graph G with complete topology T lies in generating a relevant manifold of path's tensors **P** on T, along with the metric tensors M_F. Here, the shortest-paths criteria or the maximal paths-diversity can be applied. This results in calculation the tensor $\mathbf{F_P}$=**F**×**P**=diag(**f**)×**P** and row-set g=**f**×$\mathbf{F_P}$:

$$\text{DMT: } (\mathbf{P}, M_F)_T \rightarrow (g)_T.$$

The graph G(T, g) with topology T and metric g is the input for the inverse Maxflow task, while the output is (M_F, **P**). Thereby, a comprehensive set {G(T,g)/(M_F,**P**)}, obtained by the routine direct Maxflow task solution, can be used for testing the non-trivial algorithms of the hard inverse Maxflow task, as well as a training sample sequence (TSS) for machine learning of an AI-model:

$$\text{IMT: } (g)_T \rightarrow (\mathbf{P}, M_F)_T.$$

Ultimately, the direct Maxflow task (DMT) as a first part of the whole Maxflow problem (MFP) for data flows simulation in telecommunication networks can be given by the following formalism.

1) Bring the network topology to the normalized view of the ST-planar graph G(V) with V vertices (see Figures 1 and 2).
2) Generate the path's tensor **P**(G) and flow-tensor **F** = diag(**f**), **f** = $\sqrt{f_P(k)}$ (see Figure 4).
3) Calculate the flow metric-tensor M_F = (**F**×**F**).
4) Get flow-paths tensor $\mathbf{F_P}$ = **F**×**P**(G).
5) Count the graph G(V) metric g = **f**×$\mathbf{F_P}$.
6) Fix (**P**, $M_F)_T \rightarrow (g)_T$ as a DMT result sample.
7) Use {$(g)_T \rightarrow$ (**P**, $M_F)_T$} as IMT/TSS sequence.

6 DISCRETE ANALYSIS OF THE INVERSE MAXFLOW TASK ON THE FREE-ORIENTED ST-PLANAR NETWORK GRAPH

Following the definitions in Section 5, here we introduce the inverse Maxflow task (IMT) on the free-oriented ST-planar weighted network graph G(V) with V vertices, complete topology T and given metric **g**, as the objective: "Find the feasible total flow value F_Σ between the two open poles S, T, along with the path's tensor **P**(G) and flow-path tensor $\mathbf{F_P}$".

This type of discrete optimization tasks belongs to the known class of NPC combinatorial problems. An obvious but worst strategy for exact solving the formulated above IMT task is as follows.

Let P_i={p_k}$_i$ the set of ST-paths on the complete ST-planar graph G, where each P_i is unique in paths p_k ordering, e.g.,

$$\{P_i\}=\{\{p_1, p_2, p_3\}, \{p_1, p_3, p_2\}, \{p_2, p_1, p_3\}, \{p_2, p_3, p_1\}, \{p_3, p_1, p_2\}, \{p_3, p_2, p_1\}\}.$$

On any iteration P_i, the path p_k is consequently examined for feasible flow F_i=sum{$f(p_k)$}$_i$ until no more paths p_k exist. Among the iterations {F_i}, one or more best results {F_{max}}⊂{F_i} are to be taken as final product. The power of the set {F_i} steamily grows with the increase of vertex's quantity V.

To reduce this task, apply the known heuristic approach on the base of Pontryagin maximum

principle. Let two extremum necessary conditions for each step ‘k’ of the current iteration ‘i’:

1) “max-flow-try”, i.e. f(pk)=max;
2) “shortest-path-first”, i.e. pk≤ pk+1 in hops.

The first condition is used in common FFA-based algorithms of MFP solution on the ST-planar directed graphs; this ensures convergence of the solution process.

The second condition is used in FFA-based Edmonds-Karp algorithm of MFP to reduce computational complexity. Figure 6 helps to see the necessity of the ‘shortest-path-first' condition to achieve the maximum flow in ST-planar free-oriented graph.

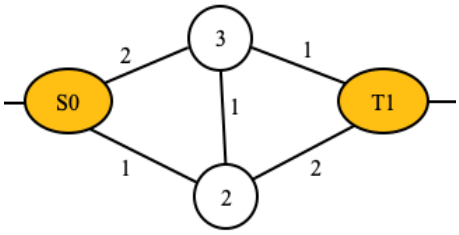

Figure 6: ST-planar graph case with critical first path.

It is clear, that the feasible flow from the source S0 to the target T1 in Figure 6 is F_{max}= f_1(S0-2-T1) + f_2(S0-3-T1) + f_3(S0-3-2-T1) = (1+1+1) = 3, whereas the residual graph is null. However, in case of violation the second necessary condition (‘shortest-path-first’), e.g. if begin with the 3-hop path p_1=(S0-2-3-T1) instead 2-hop path (S0-2-T1), we obtain less feasible flow F=1< F_{max}=3, and non-null residual graph G_R, whereas no more paths from S0 to T1 exist:

$$F=f1(S0\text{-}2\text{-}3\text{-}T1)=1<F_{max}=3;$$

$$GR=\{(3, S0)=2; (3, T1)= 2\};$$

vertices S0 and T1 are isolated.

Thus, violating the second necessary condition (‘shortest-path-first’) may result in false output product (at least one case shown above); therefore, it is always needed, i.e. is necessary condition.

Let estimate the complexity reduction of the inverse MFP-task by the ‘shortest-path-first’ condition provision. The total number of different h-hop paths in ST-planar graph G(V) with V vertices can be calculated with Figure 3:

$$NPV=1+sum(V-h),\ h\in 2, 3,...,V-1.$$

In particular case of V=6, it is

$$NP6=1+(6-2)+(6-3)+(6-4)+(6-5)=1=11.$$

It is easy to show, that in general case, the number of h-hop paths in ST-graph with V vertices is polynomial function

$$N_{PV} = 1+sum(V-2,V-3, ...,1) = 1+0.5(V-2)\ (V-1).$$

So, each set $P_i=\{p_k\}_i$ has N_{PV} paths number.

Let $\{P_i\}$ be the class of all the sets P_i. According to the known combinatorial formulas, the number of sets P_i in the class $\{P_i\}$ equals the number of permutations of the paths p_k:

$$|\{Pi\}|=1\cdot 2\cdot 3\cdot ....\cdot NPV=(NPV)!.$$

For instance, if V=6, then $|\{P_i\}|$=11!=39`916`800.

It is rather clear, that among all the P_i sets within the full $\{P_i\}$ collection, the only one set $P\in\{P_i\}$ may satisfy the "shortest-path-first" condition.

Thus, by applying Pontryagin maximum principle along with two heuristic necessary conditions (“max-flow-try” and “shortest-path-first”), the inverse combinatorial Maxflow task (IMT) on the bi-polar free-oriented ST-planar network graph can be reduced from the none-deterministic polynomial complexity NPC, with $\{(N_{PV})!\}$ iterations for scanning sets of paths $\{P_i\}$, to PC-task with computation the flow distribution over a single set of paths $P=\{p_k\}$, that includes the polynomial number of paths N_{PV} = 1+0.5(V-2) (V-1).

Now, we get the answer, what ML training sequence TSS is sufficient for testing the IMT-algorithms on the FWG network graph. It is the set of paths N_{PV}, constructed by Pontryagin maximum principle with two extremum necessary conditions.

7 CONCLUSIONS

The main scientific result of the work is formalization of the inverse and direct Maxflow tasks on the free-oriented weighted ST-planar network graph. This allows to use the dynamic reconfiguration of the modern telecommunication channels in digital network-flow optimization, in order to increase the overall networks performance. Within the scope of this result, the following is obtained.

The Maxflow problem state of the art is analyzed in Section 2. It shows, that most publications on the Maxflow problem explore the so called “logistic models” of transportation system in the form of directed bipolar ST-graph. Among them, recent researches on artificial intelligence and machine learning techniques have been increasingly used to approach the Maxflow problem.

It is concluded, that conventional logistic Maxflow models are not enough adequate to modern

telecoms, and therefore, hamper to fully benefit the SDN-reconfigurability. Known algorithms on multi-pole free-oriented graphs are solely limited by the 6-node graph case. Because of that, further researches on Maxflow methods in telecoms on the free-oriented graph model needed.

To advance the Maxflow problem study in data networks with reconfigurable channels, related objectives have been set in Section 3. Section 4 formalizes the inverse and direct Maxflow tasks on the free-oriented bi-polar ST-planar network graph in terms of inverse Maxflow task testing and machine learning of artificial intelligence models. Section 5 studies the direct Maxflow task as a first part of the whole Maxflow problem in tensor form; an algorithm of testing samples set calculation is constructed.

The inverse Maxflow problem has been analyzed in Section 6 as a discrete optimization task on the Pontryagin maximum principle with two necessary extremum conditions: ‘max-flow-try’ for the flow-path scanning, and “shortest-path-first” for augmenting paths searching. The related algorithm is reduced to a single iteration of paths tensor analysis with polynomial paths number.

In general, unlike the known approaches to product flow maximization on logistic system model, a novel method introduced for digital flow optimization in software defined networks with dynamically reconfigurable channels. Along with the total maximal flow, this method provides the maximal flow distribution over the network structure. The direct Maxflow formalism also enables the Maxflow algorithms testing and machine learning of artificial intelligence models.

REFERENCES

[1] L.R. Ford and D.R. Fulkerson, "Maximal flow through a network," Canadian Journal of Mathematics, vol. 8, 1956, pp. 399-404. [Online]. Available: https://www.semanticscholar.org/paper/Maximal-Flow-Through-a-Network-Ford-Fulkerson/794b01b2513f3610cb151ecfd07e9dc0eae7e1f3.

[2] Y. Dinitz, "Algorithm for solution of a problem of maximum flow in a network with power estimation," 1970. [Online]. Available: https://www.researchgate.net/publication/228057696_Algorithm_for_Solution_of_a_Problem_of_Maximum_Flow_in_Networks_with_Power_Estimation.

[3] J. Edmonds and R. Karp, "Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems," Journal of the Association for Computing Machinery, vol. 19, no. 2, 1972, pp. 248-264.

[4] J.A. Bondy, et al., "Graph theory with applications," Macmillan Press Ltd, GB, 1976. [Online]. Available: https://www.iro.umontreal.ca/~hahn/ift3545/gtwa.pdf.

[5] A.V. Goldberg and R.E. Tarjan, "A new approach to the maximum flow problem," Proceedings of the eighteenth annual ACM symposium on Theory of computing – STOC '86, 1986, pp. 136-146.

[6] A. V. Goldberg and S. Rao, "Beyond the flow decomposition barrier," J. ACM, vol. 45, no. 5, 1998, pp. 783-797.

[7] D. Papp, "The Goldberg-Rao algorithm for the maximum flow problem," COS 528 class notes, 2006, 5 p. [Online]. Available: https://www.cs.princeton.edu/courses/archive/fall06/cos528/handouts/Goldberg-Rao.pdf.

[8] P. Christiano, et al., "Electrical Flows, Laplacian Systems, and Faster Approximation of Maximum Flow in Undirected Graphs," Computer Science, Data Structures and Algorithms, 2010. [Online]. Available: https://www.semanticscholar.org/reader/af1afe809c106ed2618fc2ae14b696f672fc4bc1.

[9] M.A. Rajouh, "Osobenosti primenenia princip maximuma Pontryagin," BNTU, 2013. [Online]. Available: https://rep.bntu.by/bitstream/handle/data/5527/Osobennosti_primenenie.pdf?sequence=1&isAllowed=y.

[10] D.P. Williamson, "Network flow algorithms," Cornell University, 2019. [Online]. Available: https://www.networkflowalgs.com/book.pdf.

[11] O.V. Tykhonova, et al., "The Max-Flow Problem Statement on the Three-Pole Open Network Graph," 2019 3rd International Conference on Advanced Information and Communications Technologies (AICT), Lviv, Ukraine, 2019, pp. 209-212, doi: 10.1109/AIACT.2019.8847745. [Online]. Available: https://ieeexplore.ieee.org/document/8847745/metrics#metrics.

[12] O.V. Tykhonova, "Conveyor-modular method of multimedia flows integration with delay control in packet based telecommunication network," PhD Dissertation, O.S.Popov ONAT, Odessa, 2019.

[13] R. Haese et al., "Algorithms and Complexity for the Almost Equal Maximum Flow Problem," Operations Research Proceedings 2019. Springer, Cham, 2020, pp. 323-329.

[14] A. Alzaben, et al., "End-to-End Routing in SDN Controllers Using Max-Flow Min-Cut Route Selection Algorithm," 23rd International Conference on Advanced Communication Technology (ICACT), 2021, pp. 461-467.

[15] H. Liu, et al., "Optimization of a logistics transportation network based on a genetic algorithm," Hindawi Mobile Information Systems, vol. 2022, 2022, pp. 167-176. [Online]. Available: https://doi.org/10.1155/2022/1271488.

[16] O. Cruz-Mejía and A.N. Letchford, "A survey on exact algorithms for the maximum flow and minimum-cost flow problems," Networks, vol. 82, no. 2, 2023. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/net.22169.

[17] Y. Bengio, et al., "Machine learning for combinatorial optimization: A methodological tour d'horizon," European Journal of Operational Research, vol. 290, no. 2, 2021, pp. 405-421.

[18] B. Mor, et al., "Heuristic algorithms for solving a set of NP-hard single-machine scheduling problems with resource-dependent processing times," Computers & Industrial Engineering, vol. 153, 2021.

[19] W. Jiang, "Graph-based Deep Learning for Communication Networks: A Survey," Computer Communications, vol. 185, 2022, pp. 40-54. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0140366421004874?via%3Dihub.

[20] N. Orkun Baycik, "Machine learning based approaches to solve the maximum flow network interdiction problem," Computers & Industrial Engineering, vol. 167, 2022.

[21] T. Tawanda, et al., "An intelligent node labelling maximum flow algorithm," Int J Syst Assur Eng Manag 14, pp. 1276-1284, 2023. [Online]. Available: https://doi.org/10.1007/s13198-023-01930-3.

[22] "Graph theory," Britannica, 2023. [Online]. Available: https://www.britannica.com/topic/graph-theory.

[23] L. Wagner, "Complete Python Tutorial for Absolute Beginners," 2023.

[24] A. Kuronya, "Introduction to topology," 2010, 102 p.

[25] Matrix calculator. [Online]. Available: https://matrixcalc.org.