Multi-objective optimization of transport processes on complex networks

Jiexin Wu, Cunlai Pu, Shuxin Ding, *Member, IEEE,* Guo Cao, Chengyi Xia, *Member, IEEE,* and Panos M. Pardalos

***Abstract*—Transport processes are universal in real-world complex networks, such as communication and transportation networks. As the increase of transport demands in these complex networks, the problems of traffic congestion and transport delay become more and more serious, which call for a systematic network transport optimization. However, it is pretty challenging to improve transport capacity and efficiency simultaneously, since they are often contradictory in that improving one degenerates the other. In this paper, we formulate a multi-objective optimization problem including two objectives: maximizing the transport capacity and minimizing the average number of hops. In this problem, we explore the optimal edge weight assignments and the associated routing paths, corresponding to the optimal trade-off between the two objectives. To solve this problem, we provide a multi-objective evolutionary algorithm, namely network centrality guided multi-objective particle swarm optimization (NC-MOPSO). Specifically, within the framework of MOPSO, we propose a hybrid population initialization mechanism and a local search strategy by employing the network centrality theory to enhance the quality of initial solutions and strengthen the exploration of the search space, respectively. Simulation experiments performed on network models and real networks show that our algorithm has better performance than five state-of-the-art alternatives on several most-used metrics.**

***Index Terms*—Complex networks, network dynamics, network optimization and control.**

I. Introduction

In modern society, human life relies so much on various infrastructure networks, including transport, communication and power networks, to deliver either tangible quantities, such as goods and travelers, or intangible quantities, like information and electricity. The performance of transport processes on these infrastructure networks significantly affects the quality of our lives. Meanwhile, the traffic load on these networks increases exponentially owing to the rapid development of our society. For example, the latest Cisco traffic report [1] revealed that Internet traffic is experiencing an explosion with the growing number of applications. The increasing traffic poses a great challenge to the efficiency and scalability of these networks, and thus calls for systematic optimization to address these issues.

The performance of network transport can be measured from different aspects. On the one hand, we care about the maximum amount of traffic a network can transport per unit time, which is usually called network capacity [2]. Many factors affect the network capacity. For instance, generally the larger processing capability of nodes and links, the larger network capacity will be achieved [3]; homogeneous networks have larger transport capacity than heterogeneous ones [4]; the more diverse delivery paths, the larger network capacity [5]. On the other hand, the quantities transported through networks are always expected to reach their destinations as fast as possible, hence the average number of hops is also a critical metric in network transport [6]. This metric is also affected by the network structure, node capability, routing paths, etc. Therefore, we can optimize network transport performance by considering these factors.

Optimizing the "hard" factors such as network structure and node capability usually results in a large cost and is even impossible in many cases. A more feasible way is to optimize the routing paths in the network transport, which determine how quantities are transported from their sources to destinations. The routing paths are often selected to be the paths with the smallest sum of edge weight, named as the smallest-weight path (SWP) strategy, which offers the opportunity to tune the edge weight for routing optimization.

A. Our Motivation

For a network, we can give a form of edge weight based on the properties of the network. For example, the weight of an edge in computer networks can be defined according to its actual bandwidth [7]. In air traffic networks, the weight of an edge can be quantified by the number of passengers on the flights passing through it. Though we have the freedom to set edge weight, it is hard for us to provide the optimal edge weight assignment and the associated routing paths, leading to the optimal network transport performance.

Furthermore, when multiple transport performance metrics are under consideration, which is usually the case in practice, it becomes much harder to assign the edge weight, since these metrics may collide with each other and then a reasonable balance is needed. For instance, when the weight values of all edges are set to be equal, the SWP strategy is reduced

This work was supported in part by the National Natural Science Foundation of China under Grants 62203468 and 62173247, and in part by the Foundation of China Academy of Railway Sciences Corporation Limited under Grant 2021YJ043. P. M. Pardalos is partially supported by the Paul and Heidi Brown Preeminent Professorship at ISE (University of Florida, USA), and a Humboldt Research Award (Germany). (*Corresponding author: Cunlai Pu.)*

J. Wu, C. Pu and G. Cao are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: wujiexin, pucunlai, caoguo@njust.edu.cn).

S. Ding is with the Signal and Communication Research Institute, China Academy of Railway Sciences Corporation Limited, Beijing 100081, China (e-mail: dingshuxin@rails.cn).

C. Xia is with the School of Control Science and Engineering, Tiangong University, Tianjin 300387, China (e-mail: xialooking@163.com).

P. M. Pardalos is with the Center for Applied Optimization, Department of Industrial and Systems Engineering, University of Florida, Gainesville 32608, FL, USA. (e-mail: pardalos@ufl.edu).

to the shortest path (SP) strategy. Under the SP strategy, the traffic will be delivered with the least average number of hops. However, also under the SP strategy, highly connected nodes (hubs) are prone to become congested because they are usually the intersections of many routing paths, and the congestion will also spread to other nodes [8]. This cascading congestion problem significantly suppresses network capacity. To increase network capacity, the edge weight should be properly assigned that the routing paths bypass the hub nodes, which inevitably increases the average number of hops.

Our motivation is then to find a set of edge weight assignments and the associated routing paths leading to the optimal balance between the average number of hops and network capacity. It is however very challenging to achieve this optimal balance, since optimizing one objective, such as network capacity, is already NP-hard [9], [10]. Therefore, multi-objective evolutionary algorithms (MOEAs) are applicable to our problem, which can produce a bunch of nondominated solutions corresponding to the optimal trade-off among contradictory objectives.

A number of popular MOEAs have been developed in the field of evolutionary computation to deal with multi-objective optimization problems in the real world, such as the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [11] and multi-objective particle swarm optimization (MOPSO) [12]. The advantage of MOPSO mainly lies in that it requires very few assumptions or mathematical conditions about the optimization problem. It has however two main flaws. One is the early convergence to a local optimum, and the other is the loss of particle diversity during iterations. To counter these flaws of MOPSO, we can refer to the theories from the field of network science. Particularly, network centrality [13] has been found to have a great influence on both the structures and functions of complex networks. It also has the potential to be applied to enhance MOPSO, especially for network-related applications. There are many different metrics of network centrality. Adjusting the existing network centrality metrics for MOEAs in network applications is nontrivial, and more likely we have to give new forms of network centrality based on the specific optimization problems.

B. Contributions of This Paper

1) We find that in network transport the two critical performance metrics, network capacity and average number of hops, have a deep connection, i.e., they can both be mathematically expressed in terms of node routing betweenness. Based on this, we formulate a bi-objective optimization problem of transport process applied to a variety of real-world complex networks; one of the objectives is to maximize network capacity, and the other is to minimize the average number of hops. By solving this problem, we obtain the optimal edge weight assignments and the related routing paths, resulting in the best trade-off between network transport capacity and efficiency. To the best of our knowledge, our work is the first to provide a bi-objective optimization model for transport processes on complex networks.
2) We propose a MOEA called network centrality guided MOPSO (NC-MOPSO) to solve the network transport optimization problem. Specifically, within the framework of MOPSO, we propose an edge centrality based hybrid population initialization strategy and a node centrality inspired local search strategy to enhance respectively the quality of initial solutions and the exploration of the search space. We also analyze the space and time complexity of NC-MOPSO.
3) We conduct extensive comparative experiments on both network models and real-world networks to validate the performance of NC-MOPSO. Specifically, we compare NC-MOPSO with five state-of-the-art MOEAs in terms of three most popular metrics of solution quality. The experimental results show that NC-MOPSO outperforms these competitors in all cases. In addition, the convergence property and computational cost of NC-MOPSO are analyzed with simulation.

The remainder of this paper is organized as follows. Section II provides an overview of related works. In Section III, we present the traffic model and network transport optimization problem. In Section IV, we introduce the framework of NC-MOPSO. The experimental results and performance analysis are provided in Section V. Finally, this paper is concluded in Section VI.

II. Related works

The related works of our paper are presented in three parts. First, we give an introduction to network centrality. Then, we provide a review of network transport optimization in the field of network science. The last part is a brief introduction to the interaction between MOEAs and complex networks.

A. Network centrality

Network centrality is a key measurement in network science that can characterize the structural importance of nodes, edges, and even subgraphs in a network. This measurement helps to find the critical components that have a great influence on the dynamical processes, e.g., transport processes, happened in networks. There are many forms of network centrality in the literature [14]. The most representative one is betweenness centrality, which is thus employed in our work.

The node (or edge) betweenness is normally defined as the fraction of shortest paths between node pairs that pass through the node (or edge) of interest [15]. This definition can be adjusted for specific applications. For instance, to better quantify the importance of a node or edge in the network transport with SWP strategy, this metric should be extended to considering the routing paths instead of the shortest paths, which is called routing betweenness centrality [16].

It has been widely acknowledged that betweenness centrality facilitates optimizing the function of complex networks. For example, [17] presented an alternative topology control mechanism based on betweenness centrality, which improves network delivery efficiency and reduces node energy consumption in communication networks. Guan et. al. [18] proposed an improved routing strategy that achieves load balance in

scale-free networks by redistributing traffic load from nodes of large betweenness to nodes of small betweenness. In view of the proliferation of user-centric service instances across the Internet, [19] developed an effective heuristic approach to deal with the complexity and limitations of their distributed placement. This approach relies on node betweenness centrality to migrate service facilities towards near-optimal locations.

In our work, we utilize the routing betweenness centrality in the mathematical analysis of network transport performance. Moreover, we develop a simple edge centrality metric based on routing betweenness of nodes, and further employ this edge centrality to design an efficient initialization strategy in the optimization algorithm.

B. Transport Optimization on Complex Networks

In the past few decades, transport processes in various complex networks have aroused great attention. The biggest concern is the traffic congestion problem. Many methods have been put forward to alleviate traffic congestion and enhance the transport capacity of networks. Essentially, the transport capacity of a network depends mainly on its topological structure. The networks with scale-free topology tend to have an uneven distribution of traffic load, which limits their transport capacity. Many attempts have been made to optimize the network topological structures, such as removing some bottleneck nodes or edges and adding some edges between nodes with long distance [20].

Another effective way to improve network capacity is to optimize resource allocation in complex networks. Network resources, including edge bandwidth and node processing capability, are usually constrained. Thus, it is necessary to optimize the usage of these limited resources. Wu et al. [21] provided an efficient allocation strategy of node processing capability based on the node usage probability. A global dynamic bandwidth allocation strategy was proposed in Ref. [22], which set the bandwidth of each link to be proportional to the length of real-time queue. Liu et al. [23] studied the joint optimization of traffic flow rate and node processing capability in complex communication networks.

The above strategies are however often costly or not practical in real applications since they are devoted to manipulating the hard factors, such as network topology and resource. A more practical direction is to optimize the routing strategy of the traffic. Traditional SP strategy is vulnerable to traffic congestion because it only considers path length. Other enhanced routing strategies consider more factors, such as node degree [24], node load [25], memory information [26], and next-nearest neighbors [27]. These routing strategies can achieve larger network capacity than the SP strategy since more information are used in their routing decisions, nevertheless, their computational cost is also higher. Moreover, [9] provided a simple heuristic algorithm for routing optimization on complex networks, which can suppress traffic congestion and achieve much higher network capacity.

The aforementioned works mainly focus on the optimization of network capacity, which may be at the cost of degenerating other important transport performance metrics, e.g., average number of hops. Therefore, how to balance the conflicting performance metrics when all of them are favored is an interesting problem. Our work tackles this problem with a multi-objective optimization framework. Specifically, we formulate a network transport optimization problem with two mutually exclusive objectives, network capacity and efficiency, and solve it with an improved evolutionary algorithm.

Note that [9] is the closest among others to our work, it is however about single-objective optimization and algorithm design. Specifically, the authors discussed how to improve network capacity only, which is a single-objective optimization problem. They did mention the average number of hops in their paper, nevertheless they neither gave an explicit expression of this quantity nor optimized it. In our work, we find mathematically that the two objectives, network capacity and average number of hops, have a deep connection, i.e., they can both be expressed in terms of node routing betweenness. Based on this, we present a bi-objective optimization model for the enhancement of network capacity and efficiency simultaneously. According to our analysis, these two objectives are conflicting, which further motivates us to design and implement efficient evolutionary algorithms to solve the bi-objective optimization problem.

C. The Interaction between MOEAs and Complex Networks

The multi-objective optimization framework is readily to deal with plenty of real-world problems involving multiple conflicting optimization objectives [28], [29], [30], [31]. When referring to the conflict of objectives, it means that no single solution exists that simultaneously optimizes each objective. A multi-objective optimization problem (MOP) is defined as (take the minimization problem as an example)

$$\min F(X) = (F_1(X), F_2(X), \ldots, F_m(X))^T, \qquad (1)$$

where m is the number of objective functions, and $X = (x_1, x_2, \ldots, x_D)^T$ is a D-dimensional decision vector, belonging to a D-dimensional decision space Ω. Given two decision vectors X_1 and X_2 ($\in \Omega$), we say X_1 dominates X_2 or X_2 is dominated by X_1 if the following conditions satisfied:

$$F_i(X_1) \leq F_i(X_2), \quad \forall i = 1, 2, \ldots, m. \qquad (2)$$

The solution set of a MOP is known to be a Pareto front (PF) [28], [29], which consists of a bunch of individual solutions that cannot dominate each other.

The multi-objective evolutionary algorithms (MOEAs), which can search the Pareto solutions, have been successfully applied to many MOPs in complex networks. In network vulnerability assessment, Zhang et al. [32] formulated a bi-objective optimization problem, which simultaneously maximizes the destructiveness and minimizes the cost of an attack on the network, and further solved this problem with an improved version of NSGA-II. Zhou et al. [33] investigated the problem of improving the robustness of scale-free networks against both node-based and link-based attacks, which is finally treated in a multi-objective framework, and they further proposed a two-phase MOEA to solve this problem. In the

network clustering problem, Gong et al. [34] provided a multi-objective discrete particle swarm optimization algorithm to minimize two evaluation objectives termed as kernel k-means and ratio cut. In addition, a problem-specific MOEA/D was developed to solve the so-called tradeoff barrier coverage problem in wireless sensor networks [35].

Network science can be in turn employed to improve evolutionary algorithms. Du et al. [36] systematically explored the evolutionary algorithms as networked interaction systems and analyzed the effects of population structure and information fusion strategies on the performance of the algorithms. Wu et al. [37] proposed a novel PSO algorithm, which allows particles to adaptively move in a static scale-free network for a more efficient search. Moreover, in [38] a network science inspired heterogeneous learning strategy is introduced into PSO, where high-degree particles utilize more information from neighbors for self-improving while low-degree particles tend to maintain the diversity by learning themselves.

Enlightened by the aforementioned works, we employ the theory of network centrality from network science to enhance a representative MOEA, which is MOPSOCD [39], so that it can efficiently solve the given multi-objective transport optimization problem in complex networks.

III. Problem Statements

In this section, we present the network transport optimization problem in detail. Firstly, we provide the widely used traffic model. Secondly, we introduce the metrics for evaluating network transport processes. Finally, we formulate a MOP in network transport.

A. Traffic Models

Network transport is a fundamental and ubiquitous process in reality, e.g., packet transmission in the Internet and passenger travel in transport networks. Despite the different nature of transport processes in various networks, they have four common essential elements: generation, storage, forwarding, and routing of various quantities. A traffic model is thus required to involve these critical elements to characterize the essence of network transport. For the sake of concreteness, we take packet transport in communication networks as an instance to elaborate the frequently used traffic model. The edges are assumed to have weights in communication networks, which can be associated with the bandwidth of edges in practice. The traffic model [9] of communication networks is given as follows:

Generation: Each node generates packets with a rate of λ, thus at each time step on average $N\lambda$ packets will be inserted into the network, where N represents the number of nodes. The source node of each packet is the one that generates it, while the destination node of each packet is randomly selected from the network excluding the source node. In this case, each node can be the source or destination of a packet.

Storage: Each node has an infinite queue for buffering packets abiding by the first-in-first-out (FIFO) rule, which is a representative way to schedule packets. When a node generates or receives a packet, it will append the packet to the tail of the queue.

Forwarding: The processing capability of a node is usually limited. At each time step, a node i is assumed to be able to deliver at most C_i packets. For simplicity, we assume that all nodes have the same processing capability, i.e., $C_i = 1$. Note that our optimization framework and algorithm can also be applied for other settings of node processing capability. When a node forwards a packet, it will first check the corresponding destination node; if the destination is one of its neighbors, the packet will be delivered directly to the destination and then be removed immediately; otherwise, the node will send the packet to the next hop determined by the given routing strategy.

Routing: There could be multiple paths between the source and the destination, and the routing strategy selects the optimal path for delivering packets. In our work, we use the SWP routing strategy in network transport.

Given an undirected weighted network $G = (V, E, \vec{w})$, where V is the node set with N nodes, E is the edge set of M edges, and $\vec{w} = (w_1, w_2, \ldots, w_M)^T$ is a M-dimensional vector, where w_e represents the weight of edge e. A path in the network is referred to a sequence of edges, where each pair of adjacent edges in the sequence share a node, and each such shared node can only occur once in the path. In the SWP strategy, the routing cost of a path is the sum of the weight values of all edges in the path. Let us denote an arbitrary path between nodes a and b by $Path(a \leftrightarrow b)$. The routing cost of $Path(a \leftrightarrow b)$ is calculated as

$$L(Path(a \leftrightarrow b)) = \sum_{e \in Path(a \leftrightarrow b)} w_e, \tag{3}$$

where e is an edge belonging to $Path(a \leftrightarrow b)$ and its weight is w_e, a component of $\vec{w}$. Among all the paths between nodes a and b, we select the one of the minimum routing cost as the optimal path between nodes a and b. Note that the optimal path from node a to node b and the reverse are the same in the undirected weighted network, and there may be multiple optimal paths between nodes a and b. Let $\pi^*_{ab}(\vec{w})$ be the set of optimal paths between nodes a and b for a given edge weight vector $\vec{w}$, i.e.,

$$\pi^*_{ab}(\vec{w}) = \mathop{\arg\min}_{Path(a \leftrightarrow b)} \sum_{e \in Path(a \leftrightarrow b)} w_e. \tag{4}$$

B. Metrics of Network Transport

We employ two representative metrics to evaluate the network transport processes: network capacity and average number of hops. We shall show here that these two metrics are both determined by the routing betweenness centrality, which is a straightforward extension of the normal betweenness centrality in network science.

Definition 1 (Routing Betweenness Centrality [16])*:* In an undirected weighted network $G = (V, E, \vec{w})$, as defined in the above, the routing betweenness centrality of node i is defined as

$$B_i(\vec{w}) = \sum_{a \neq i \neq b} \frac{n^i_{ab}(\vec{w})}{n_{ab}(\vec{w})} = \sum_{a \neq i \neq b} \frac{\sum_{\pi \in \pi^*_{ab}(\vec{w})} \delta_{i\pi}}{|\pi^*_{ab}(\vec{w})|}, \tag{5}$$

where $n_{ab}(\vec{w})$ is the number of optimal paths between nodes a and b given an edge weight vector $\vec{w}$, which is equal to $|\pi^*_{ab}(\vec{w})|$; $n^i_{ab}(\vec{w})$ is the number of optimal paths between nodes a and b that also pass through node i, i.e.,

$$n^i_{ab}(\vec{w}) = \sum_{\pi \in \pi^*_{ab}(\vec{w})} \delta_{i\pi}, \tag{6}$$

where

$$\delta_{i\pi} = \begin{cases} 1, & \text{if node } i \text{ is on path } \pi, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Note that $n^i_{ab}(\vec{w})$ and $n^i_{ba}(\vec{w})$ are equal, i.e., the network is symmetric.

According to this definition, we can infer that the nodes with large betweenness bear more traffic load than the nodes with small betweenness, which is why the onset of traffic congestion is often observed at nodes of large betweenness.

Generally, when the packet generation rate λ surpasses a critical value λ_c, traffic congestion begins to happen; hence, the critical packet generation rate is usually taken as a metric of network capacity.

Definition 2 (Critical Packet Generation Rate [40])*:* Suppose that each node can forward at most one packet per time step, the critical packet generation rate can be calculated as

$$\lambda_c = \frac{N-1}{\max\limits_{i \in N} B_i(\vec{w})}, \tag{8}$$

where $\max\limits_{i \in N} B_i(\vec{w})$ represents the largest routing betweenness of nodes given an edge weight vector $\vec{w}$, and N is the number of nodes in the network.

The average number of hops is another main indicator for the performance of network transport. According to the SWP strategy, we can figure out the routing paths between any two nodes and thus the average number of hops of the network transport.

Definition 3 (Average Number of Hops)*:* The average number of hops of the transport process in a network is defined as

$$H_{avg} = \frac{\sum_{a \neq b, a \in V, b \in V} h_{ab}}{N(N-1)}, \tag{9}$$

where h_{ab} is the number of hops (intermediate nodes) in the routing path between node a and node b, N is the number of nodes in the network.

Theorem: In an undirected and weighted network, when the SWP strategy is employed in the routing decision, the average number of hops of the transport process can be calculated as

$$H_{avg} = \frac{\sum_{i=1}^{N} B_i(\vec{w})}{N(N-1)}, \tag{10}$$

where $B_i(\vec{w})$ represents the routing betweenness of node i given an edge weight vector $\vec{w}$.

Proof: For a pair of nodes a and b, its corresponding number of hops can be calculated as

$$\begin{aligned} h_{ab} &= \frac{n^1_{ab}(\vec{w})}{n_{ab}(\vec{w})} + \frac{n^2_{ab}(\vec{w})}{n_{ab}(\vec{w})} + ... + \frac{n^N_{ab}(\vec{w})}{n_{ab}(\vec{w})} \\ &= \frac{\sum_{i=1, a \neq i \neq b}^{N} n^i_{ab}(\vec{w})}{n_{ab}(\vec{w})}, \end{aligned} \tag{11}$$

where $n_{ab}(\vec{w})$ and $n^i_{ab}(\vec{w})$ are the same as in Eq. (5); $\frac{n^i_{ab}(\vec{w})}{n_{ab}(\vec{w})}$ provides the probability that node i is on the routing paths between nodes a and b. Then, the total number of hops of all node pairs is

$$\begin{aligned} H_{tot} &= \sum_{j=2}^{N} h_{1j} + \sum_{j=1, j \neq 2}^{N} h_{2j} + ... + \sum_{j=1, j \neq n}^{N} h_{nj} \\ &= \sum_{a \neq 1 \neq b} \frac{n^1_{ab}(\vec{w})}{n_{ab}(\vec{w})} + \sum_{a \neq 2 \neq b} \frac{n^2_{ab}(\vec{w})}{n_{ab}(\vec{w})} + ... + \sum_{a \neq N \neq b} \frac{n^N_{ab}(\vec{w})}{n_{ab}(\vec{w})}. \end{aligned} \tag{12}$$

According to Eq. (5), Eq. (12) can be further simplified as

$$H_{tot} = B_1(\vec{w}) + B_2(\vec{w}) + ... + B_N(\vec{w}) = \sum_{i=1}^{N} B_i(\vec{w}). \tag{13}$$

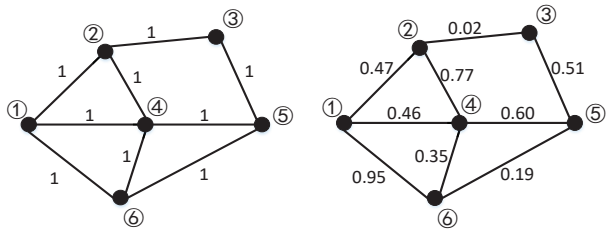
Thus, the average number of hops of the network transport can be calculated as

$$H_{avg} = \frac{H_{tot}}{N(N-1)} = \frac{\sum_{i=1}^{N} B_i(\vec{w})}{N(N-1)}. \tag{14}$$

■

C. Multi-objective Optimization Formulation

We consider the simultaneous optimization of network capacity and average number of hops in our work. The tunable parameter is the edge weight vector. For a specific edge weight vector, we can obtain the corresponding routing paths based on SWP strategy. Then, we can calculate the routing betweenness of all nodes through Eq. (5). When node routing betweenness is available, network capacity and average number of hops can be calculated through Eqs. (8) and (10). An immediate problem is: can we find the optimal edge weight vector for both network capacity and average number of hops?



(a) (b)

Fig. 1. A simple undirected weighted network for illustration purpose.

It is however very challenging to address this problem, since these two metrics are contradictory in that enhancing one suppresses the other. We take a simple instance to illustrate this conflict. Fig. 1 presents a simple undirected weighted network with two feasible solutions of edge weight assignment, which are shown in Fig. 1(a) and 1(b), respectively. For the first solution (Fig. 1(a)), the network capacity is $\lambda_c{}^{(a)} = 1.00$

and the average number of hops is ${H_{avg}}^{(a)} = 0.40$, while for the second solution (Fig. 1(b)), the network capacity is ${\lambda_c}^{(b)} = 1.25$ and the average number of hops is ${H_{avg}}^{(b)} = 0.53$. The network capacity of (b) is larger than that of (a), yet the average number of hops of (b) is also larger than that of (a). Note that network capacity and average number of hops have the larger-the-better and smaller-the-better characteristics, respectively. This instance indicates the two metrics are contradictory in the given network transport problem, which is consistent with the previous findings in [9]. Therefore, the ultimate goal is to find the optimal edge weight vector leading to the best trade-off between network capacity and average number of hops.

The multi-objective optimization framework is often adopted to solve the problems of multiple conflicting objectives. In our problem, we have two conflicting objectives. The first one is to maximize the network capacity, which is finally transformed into minimizing the inverse of the critical packet generation rate λ_c. The second one is to minimize the average number of hops H_{avg}. The edge weight vector $\vec{w}$ is the final variable determining the values of λ_c and H_{avg}. Our bi-objective optimization problem is formally given as follows,

$$\begin{cases} \min\limits_{\vec{w}} \quad F_1 = \dfrac{1}{\lambda_c}, \\ \min\limits_{\vec{w}} \quad F_2 = H_{avg}, \\ \text{s.t.} \quad 0 < w_e \leq 1, \quad \vec{w} = (w_1, w_2, \ldots, w_M)^T. \end{cases} \tag{15}$$

By substituting Eqs. (8) and (10) into Eq. (15) and replacing the intermediate variable $B_i(\vec{w})$ with Eq. (5), the complete mathematical programming formulation is given as

$$\begin{cases} \min\limits_{\vec{w}} F_1 = \dfrac{\max\limits_{i \in V} \sum_{a \neq i \neq b} \dfrac{\sum_{\pi \in \pi^*_{ab}(\vec{w})} \delta_{i\pi}}{|\pi^*_{ab}(\vec{w})|}}{N-1}, \\ \min\limits_{\vec{w}} F_2 = \dfrac{\sum_{i=1}^{N} \sum_{a \neq i \neq b} \dfrac{\sum_{\pi \in \pi^*_{ab}(\vec{w})} \delta_{i\pi}}{|\pi^*_{ab}(\vec{w})|}}{N(N-1)}, \\ \pi^*_{ab}(\vec{w}) = \arg\min\limits_{Path(a \leftrightarrow b)} \sum_{e \in Path(a \leftrightarrow b)} w_e, \\ \delta_{i\pi} = \begin{cases} 1, & \text{if node } i \text{ is on path } \pi, \\ 0, & \text{otherwise,} \end{cases} \\ \text{s.t.} \quad 0 < w_e \leq 1, \quad \vec{w} = (w_1, w_2, \ldots, w_M)^T. \end{cases} \tag{16}$$

Our optimization problem is essentially a constrained continuous MOP and the solution is a set of optimal compromises, the so-called Pareto set. The optimization of the single metric, i.e., network capacity, is already NP-hard [9], [10]. Therefore, we shall call for an evolutionary computation framework to solve our bi-objective optimization problem.

Remark: In reality, the weight of edges usually has different meanings in different networks. Even in the same network, we can have different definitions of edge weight. For instance, in computer networks, the edge weight can be related to the real cost, physical length, etc. Since our work considers network data from different domains, we constrain the edge weight in $(0, 1]$ consistently for the sake of convenience. It is worth to mention that our optimization framework is readily to be accommodated for a specific network when the actual constraints of edge weight are available.

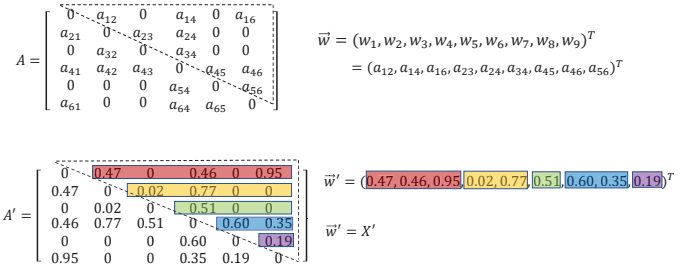


Fig. 2. An illustration of the coding schemes of our algorithm by using the network in Fig. 1 as an instance.

IV. Algorithm Specification and Implementation

In this part, a new algorithm called network centrality guided multi-objective PSO (NC-MOPSO) is presented to solve the given bi-objective transport optimization problem. Specifically, we first introduce the encoding and decoding schemes of NC-MOPSO. Then, we provide the framework of NC-MOPSO. Subsequently, we introduce in detail its two key mechanisms: hybrid initialization and local search. Finally, we discuss the computational complexity of NC-MOPSO.

A. Encoding and Decoding schemes

As mentioned above, the edge weight vector is the decision variable of our optimization problem, whose components are constrained in $(0, 1]$. Accordingly, our algorithm uses decimal encoding. In particular, an arbitrary solution X of our algorithm is set to be the same as edge weight vector $\vec{w}$, i.e., $X = \vec{w}$.

We use again the network in Fig. 1 as an example to show the relation between the solution of our algorithm and the optimization problem. Let us assume A is the weighted adjacency matrix for this weighted and undirected network, in which $a_{ij} = w_e$ if nodes i and j are connected by an edge e, and otherwise, $a_{ij} = 0$. Since the network is undirected, matrix A is symmetric. Then, matrix A is determined by the nonzero terms in the upper triangle, which store the weight values of all edges. In other words, the nonzero terms with natural order (from left to right row by row starting from the top) in the upper triangle of A constitute the edge weight vector $\vec{w}$.

An arbitrary feasible solution X' of our algorithm gives an instance of edge weight vector $\vec{w}$, denoted by $\vec{w}'$, and further produces an instance of matrix A, denoted by A'. For a feasible solution $X' = (0.47, 0.46, 0.95, 0.02, 0.77, 0.51, 0.60, 0.35, 0.19)^T$, shown in Fig. 1(b), the corresponding matrix A' is given in Fig. 2. Based on matrix A', the routing paths between any two nodes can be calculated with the SWP strategy. Then, the routing betweenness, network capacity and average number of hops can be obtained through Eq. (5), Eq. (8) and Eq. (10), respectively. We can also obtain the values of objective functions directly based on the whole mathematical programming formulation (Eq. (16)), whose input is the network topology represented by an adjacency matrix and a solution of edge weight vector.

B. The framework of NC-MOPSO

The particle swarm optimization (PSO) [41] algorithm is a popular single-objective optimization algorithm with fast

convergence, simple framework and easy implementation. In PSO, a swarm of particles, initiated with random position and velocity, fly through the solution space for targeting the optimal solutions. In our problem, we let the position of a particle k be $p_k = (x_1, x_2, \ldots, x_M)^T$, where $x_e \in (0, 1]$ is the weight of the e-th edge and M is the number of edges in the network. The position and velocity of particle k are updated in the following way:

$$\begin{cases} v_k^{t+1} = \omega \times v_k^t + c_1 \times r_1 \times \left(pbest_k^t - p_k^t\right) \\ \qquad\quad + c_2 \times r_2 \times \left(gbest^t - p_k^t\right) \\ p_k^{t+1} = p_k^t + v_k^{t+1}, \end{cases} \tag{17}$$

where ${v_k}^t$ and ${p_k}^t$ are the velocity and position of particle k at iteration t, respectively; ω is an inertia weight; c_1 and c_2 are two acceleration coefficients; r_1 and r_2 are two learning coefficients, which are randomly and independently selected from interval $(0, 1)$; $pbest_k^t$ is the best personal solution of particle k until time t, while $gbest^t$ is the best global solution in the entire swarm until time t.

The PSO algorithm has been extended to find the optimal solutions in multi-objective optimization, which are non-dominated to each other but are superior to the rest of solutions in the search space. The algorithm named as multi-objective particle swarm optimization with crowding distance (MOPSOCD) [39] is a representative extension of PSO. In this algorithm, at each iteration, the personal best solutions of all particles are put into an external archive, where the comparisons between the solutions are conducted and the dominated ones are deleted accordingly. The density of solutions surrounding a solution in the archive is measured by the crowding distance, which was first given in [11]. The global best solution of particles is then randomly selected among the solutions of the largest crowding distances (Empirically, top 10% of the solutions are considered). The number of solutions in the archive grows with iterations. When the archive is full, the solutions with the minimum crowding distance will be discarded with priority. After the maximum number of iteration is reached, the solutions left in the archive are the final optimal solutions.

We employ the theory of network centrality to enhance MOPSOCD in terms of solution quality, and hence develop a new MOEA (Algorithm 1), named as network centrality guided multi-objective particle swarm optimization (NC-MOPSO). Concretely, we propose an edge-centrality guided hybrid initialization strategy (Algorithm 2) and a node-centrality guided local search strategy (Algorithm 3) to enhance the quality of initial solutions and strengthen the exploration of the search space, respectively. These two strategies are further integrated into MOPSOCD to solve the transport optimization problem. The effectiveness of these two strategies is justified through experimental comparison, the results of which are given in Table V and Fig. 3.

The framework of NC-MOPSO, as shown in Algorithm 1, consists of two parts: initialization (line 1 to line 5) and update (line 6 to line 17). First, an initial swarm is produced by the hybrid initialization strategy (Algorithm 2). Afterwards, each particle updates its position and velocity according to the given rules, and the personal best position will be renewed accordingly. Then, the local search (Algorithm 3) explores the neighborhood of the solution of the smallest crowding distance. Note that the local search is performed every g_{ls} iterations towards a compromise between the quality of solutions and the efficiency of NC-MOPSO. Finally, the global best solution is selected from the external archive. This cycle repeats until the stopping criterion is reached. The algorithm returns the external archive containing the final non-dominated solutions (PFs). The notations used in all the algorithms are summarized in Table I.

Algorithm 1 Framework of NC-MOPSO Algorithm.

Input: A: network adjacency matrix; pop: swarm size; HIR: the heuristic initialization ratio.

Output: PF solutions. Each solution corresponds to a weight assignment.

1: $S \leftarrow HybridInitialization(pop, A, HIR)$(Algorithm 2);
2: $V \leftarrow RandomInitialization(pop)$;
3: $pbest \leftarrow PersonalBestPosition(S, A)$;
4: $R \leftarrow SelectNondominatedSets(pbest)$;
5: $gbest \leftarrow GlobalBestPosition(R)$;
6: $t = 1$;
7: **while** termination condition is not satisfied **do**
8: $\quad V \leftarrow UpdateVelocity(V, pbest, gbest)$;
9: $\quad S \leftarrow UpdatePosition(S, V)$;
10: $\quad pbest \leftarrow PersonalBestPosition(S, A)$;
11: $\quad R \leftarrow SelectNondominatedSets(R, pbest)$;
12: $\quad$ **If** $t\%g_{ls} == 0$
13: $\quad\quad R \leftarrow LocalSearch(R)$(Algorithm 3);
14: $\quad$ **End**
15: $\quad gbest \leftarrow GlobalBestPosition(R)$;
16: $\quad t++$;
17: **end while**
18: **return** R

TABLE I
NOTATIONS IN THE ALGORITHMS.

A	adjacency matrix of a weighted undirected graph
M	number of edges in the graph
S	a swarm of particles
pop	number of particles in the swarm
p	a particle in the swarm
HIR	heuristic initialization ratio
V	set of particle velocities
$pbest$	set of personal best positions in S
$gbest$	global best position in S
R	external archive for storing non-dominated solutions
EC	set of edge centrality values of all edges
CD	set of crowding distances
nei_set	set of the neighbors of a particle
g_{ls}	interval for local search
t_{ls}	number of generated solutions in the local search

C. Edge-Centrality Guided Hybrid Initialization

A good initialization mechanism can reduce the search space and accelerate the algorithm for finding the global optimal solution. Instead of purely random initialization, we propose an edge-centrality guided hybrid initialization (ECHI) to provide better initial solutions. The pseudocode of ECHI is

shown in Algorithm 2.

In our optimization problem, defined in Eq. (15), the network capacity is a function of the largest node routing betweenness, while the average number of hops depends on the average node routing betweenness. Since both objectives are strongly correlated with node betweenness, we define the centrality of an edge as the normalized arithmetic mean of the routing betweenness of its two end nodes. We name it as node-betweenness based edge centrality (NBEC). Specifically, the NBEC of an edge $e(i,j)$ with two end nodes i and j is

$$EC_{e(i,j)} = \frac{B_i + B_j}{2\sum_{l=1}^{N} B_l}, \tag{18}$$

where $\sum_{l=1}^{N} B_l$ represents the sum of the routing betweenness of nodes in the network. Note that there exist several distinct definitions of edge centrality in network science. We have tested other definitions and found that Eq. (18) is the best form in our problem (see [42]).

Assume that the ratio of the heuristic solutions in all initial solutions of ECHI is HIR. We first generate a given number of random initial solutions. Then, we randomly select some of the solutions satisfying the ratio HIR. For each of these selected solutions, we redistribute the values of its elements according to the edge centrality, i.e., NBEC, to ensure the edges of larger centrality will have larger weights. The redistribution intentionally arranges large centrality edges, which are prone to be congested in the transport process, with large weights, so that the SWP routing strategy reduces the dependency of these edges. This is in favor of the transport process.

Algorithm 2 Edge-Centrality Guided Hybrid Initialization.

Input: A: network adjacency matrix; pop: swarm size; HIR: the heuristic initialization ratio.
Output: Initial swarm S.

$S \leftarrow RandomInitialization(pop)$;
$h \leftarrow pop * HIR$;
$count = 1$;
while $count \leq h$ **do**
    $p \leftarrow S(count)$;
    $EC \leftarrow CalculateEdgeCentrality(A, p)$;
    $[EC, ind_1] \leftarrow Sort(EC,' descend')$;
    $p \leftarrow Sort(p,' descend')$;
    for $i = 1$ to M **do**
        $p(ind_1(i)) \leftarrow p(i)$;
    end for
    $count++$;
end while
Return S

D. Node-Centrality Guided Local Search

The technique of local search has been increasingly used to hybridize evolutionary algorithms, which can find better PFs by exploring the neighborhood of current solutions. We propose a node-centrality guided local search (NCLS) strategy to strengthen the exploration of solution space. The pseudocode of NCLS is presented in Algorithm 3.

In the NCLS, we first select a solution with the smallest crowding distance from the current external archive. Based on the selected solution, we calculate the routing betweenness of all nodes, and pick the node with the largest routing betweenness. We then add a random value to the current weight of each edge connected to that node. In this way, we obtain a new solution, which is taken as a neighbor of the selected solution. Similarly, we can generate another neighbor based on the new solution and so on. For all generated solutions, we merge them into the external archive, which enhances the diversity of current solutions.

The idea of NCLS is to suppress the maximum node routing betweenness in an iterative way, so that the traffic becomes more evenly distributed on the nodes. This increases the threshold of traffic congestion and thus improves transport performance.

Algorithm 3 Node-Centrality Guided Local Search.

Input: A: network adjacency matrix; R: external archive; t_{ls}: number of generated solutions.
Output: The updated external archive R.

$CD \leftarrow CalculateCrowdingDistance(R)$;
$R \leftarrow Sort(R,' CD',' descend')$;
$s \leftarrow size(R, 1)$;
$p \leftarrow R(s)$;
$count = 1$;
while $count \leq t_{ls}$ **do**
    $B \leftarrow CalculateNodeBetweeness(A, p)$;
    $[B_{max}, L_{max}] \leftarrow max(B)$;
    $ind_2 \leftarrow FindEdges(A, L_{max})$;
    $p \leftarrow p(ind_2) + rand()$;
    $p \leftarrow p(find(p > 1)) = 1$;
    $A \leftarrow Update(A, p)$;
    $nei\_set(count) \leftarrow p$;
    $count++$;
end while
$R \leftarrow SelectNondominatedSets(nei\_set, R)$;
Return R

E. Complexity Analysis

1) *Space complexity*: Three main memories are needed for NC-MOPSO. The first one is for storing the adjacency matrix of a network, which has a space complexity of $O(N^2)$, where N is the number of network nodes. The second one is to store particles; its space complexity is $O(M \cdot pop)$, where pop and M are the number and dimension of particles, respectively. The last one is for the external archive, which has $O(s \cdot M)$ space complexity, where s is the size of external archive. Note that the sizes of all these memories are given in the initialization and will not increase with iterations.
2) *Time complexity*: The time complexity mainly lies in the update process of NC-MOPSO, which includes the objective function computation, local search, crowding distance computation, and non-dominated comparison in the population and external archive. The objective function computation has $O(pop \cdot N^2)$ time complexity, where pop and N are the numbers of particles and nodes, respectively. The time complexity of the local search is

$O(t_{ls} \cdot N^2)$, where t_{ls} is the number of generated solutions in the local search. Sorting the solutions based on crowding distance in the external archive has $O(s \cdot \log s)$ time complexity, where s is the size of the archive. The time complexity for the non-dominated comparison is generally $O(pop \cdot s)$. Assuming that the size of external archive is proportional to the number of particles, the time complexity for the non-dominated comparison is $O(pop^2)$. Collectively, the overall time complexity for each iteration is $O(\max(pop^2, N^2))$.

V. Experimental Results

In this section, we assess the performance of NC-MOPSO in network transport optimization. First, we introduce the baseline algorithms and the network data. Then, we present three popular performance metrics and the parameter settings for MOEAs. Finally, the results of comparative experiments are provided.

A. Baseline Algorithms and Network Data

To validate the performance of NC-MOPSO, we compare it with the following MOEAs:

1) MOPSOCD [39], a popular MOEA, employs the crowding distance mechanism to deal with the global best solution selection and filter nondominated solutions when the external archive is full.
2) MOPSOCDELS [43], an improved version of MOPSOCD, employs the elitist learning strategy (ELS) to avoid early convergence and improve optimization efficiency. The ELS adopts a Gaussian perturbation to maintain the diversity of particles and is fairly effective for global optimization.
3) NSGA-II [11], one of the most popular MOEAs, selects individuals according to the Pareto dominance relation and propagates the offspring in an iterative way. The main feature of this algorithm is that it adopts the elitist non-dominated sorting by using the crowding distance as a ranking criterion.
4) GSPSO [44], one of the state-of-the-art MOEAs, enhances the PSO with the geometric structure of PF. In this MOEA, the inherent geometric structure of the current PF is exploited to directly guide the flight of particles.
5) CMOPSO [45] is a competitive mechanism based multi-objective particle swarm optimizer. The particles in CMOPSO are updated on the basis of the pairwise competitions performed in the current swarm at each iteration.

Note that these comparison algorithms can be directly applied to our network transport optimization problem without modification.

The comparative experiments are conducted on different types of networks. The dataset contains six undirected weighted networks; two of them are generated by network models and the others are real-world networks. Specifically, we employ two widely used network models, namely the Barabasi-Albert (BA) model [46] and Watts-Strogatz (WS) model [47], to generate scale-free networks and small-world networks, respectively. The scale-free networks exhibit the power-law degree distribution, in which a small percent of nodes have much larger degree than the others. While the small-world networks have approximately exponential degree distribution and high clustering, which are different from the scale-free networks. In addition, we select four different types of real-world networks, i.e., the power grid (118-bus) [48], email network (Email-enron-only) [48], road network (Barcelona) [48] and Internet (Uninett) [49]. Table II characterizes these networks with three properties: number of nodes N, number of edges M and average node degree $\langle D \rangle = 2M/N$.

TABLE II
Main Characteristics of Six Test Instances.

Instance	N	M	$\langle D \rangle$
BA300	300	597	3.98
WS300	300	600	4
118-bus	118	179	3.03
Email-enron-only	143	623	8.71
Uninett	74	101	2.73
Barcelona	1020	1798	3.52

B. Performance Metrics and Parameter Settings

Three popular metrics are employed to evaluate the performance of different algorithms, namely hypervolume (HV) [50], inverted generational distance (IGD) [51] and set coverage (C-metric) [50]. HV and IGD measure the quality of solutions in terms of convergence and diversity; C-metric compares two solution sets from the point of Pareto dominance.

The HV metric gives the total volume bounded by the points on the Pareto front P and a reference point z^* in the objective space, which is as follows:

$$HV\left(P, z^*\right) = \left\{ \bigcup_k a(z_k) | \forall z_k \in P \right\}. \tag{19}$$

In this equation, z_k is the k-th individual point in the Pareto front P, $a(z_k)$ is the rectangle area bounded by the reference point z^* and the point z_k. The reference point z^* is empirically set to be $(\max(F_1), \max(F_2))$ if the optimization problem has two minimization objectives, where F_1 and F_2 are the two objective functions. The larger value of HV means that the Pareto front is closer to the lower left, which indicates the algorithm produces better solutions.

The definition of IGD is provided in Eq. (20), where P^* denotes the true PF and P is an obtained solution set. $d(z, P)$ is the smallest Euclidean distance from $z \in P^*$ to all points in P. The smaller value of IGD, the better solution. Note that the true PF means the exact solution of PF, which are usually unknown for real-world optimization problems, however they can be approximated by selecting non-dominated solutions from all solutions obtained by different MOEAs [52].

$$IGD\left\{P, P^*\right\} = \frac{\sum_{z \in P^*} d\left(z, P\right)}{|P^*|}. \tag{20}$$

The C-metric $C(P, Q)$ is defined as the percentage of the solutions in Q dominated by at least one solution in P (see Eq. (21)), where P and Q are two PFs. Assuming P^* is the true PF, the lower the value of $C(P^*, P)$, the better P is.

$$C\left(P, Q\right) = \frac{\left| \left\{ q \in Q | \exists z \in P : z \quad dominates \quad q \right\} \right|}{|Q|}. \tag{21}$$

TABLE III
PARAMETER SETTINGS OF THE ALGORITHMS.

Parameters	pop	$maxgen$	c_1	c_2	r_1
Value	200	500	1.5	2	$(0, 1)$
Parameters	r_2	p_c	p_m	t_{ls}	g_{ls}
Value	$(0, 1)$	0.9	$\frac{1}{M}$	300	50

The parameter settings of all the concerned MOEAs are shown in Table III. The first and third rows present the parameters; the second and fourth rows provide their values. pop is the population size and $maxgen$ is the maximum number of iterations; c_1 and c_2 are two acceleration coefficients for PSO-based algorithms; r_1 and r_2 are two learning coefficients for PSO-based algorithms; p_c and p_m represent the crossover probability and the mutation probability in NSGA-II, respectively; t_{ls} is the number of generated solutions in the local search, which performs every g_{ls} iterations. For fair comparison, the common parameters are set be the same for the MOEAs.

C. Performance Evaluation

The results of comparative experiments are presented in this part. Specifically, we shall first show the performance of NC-MOPSO for different values of HIR. Next, we present the experimental results to show the effectiveness of the two improvement operators: ECHI and NCLS. Finally, we give the performance comparison between NC-MOPSO and the five representative MOEAs.

The initial population of NC-MOPSO is composed of heuristic and random solutions. The proportion of the heuristic solutions in all the initial solutions is controlled by HIR. We study how the value of HV changes with HIR to obtain an adequate HIR setting. The means and standard deviations of HV over ten independent runs of NC-MOPSO on the six instances are shown in Table IV, where HIR50 means 50% of initial solutions are generated heuristically and so on. We can see that HIR50 shows an advantage over HIR0 and HIR100. For totally randomly initialization, the particles have difficulties and are not efficient to find the best solution, while in the case of purely heuristic initialization, the particles are prone to fall into the local optimum. The hybrid initialization with both heuristic and random solutions, i.e., ECHI, makes a necessary compromise between efficiency and diversity. Therefore, we fix HIR value to be 50% in the following experiments.

TABLE IV
HV (MEAN(STD)) FOR DIFFERENT HIR (HIR0, HIR50 AND HIR100) ON EACH TEST INSTANCE.

Instance	HIR0	HIR50	HIR100
BA300	0.3563(0.0382)	**0.6047(0.0021)**	0.5832(0.0044)
WS300	0.4251(0.0175)	**0.4822(0.0269)**	0.3047(0.0432)
118-bus	0.3504(0.0105)	**0.3743(0.0085)**	0.3119(0.0261)
Email-enron-only	0.4431(0.0436)	**0.5807(0.0093)**	0.5135(0.0219)
Uninett	0.4101(0.0176)	**0.7970(0.0044)**	0.4578(0.0133)
Barcelona	0.3618(0.0348)	**0.5989(0.0062)**	0.4712(0.0396)

NC-MOPSO adopts two critical improvement operators, ECHI (see Section IV-C) and NCLS (see Section IV-D), to enhance the quality of solutions. We show the effectiveness of these two operators through comparing NC-MOPSO with MOPSOCD and MOPSOCD_in; MOPSOCD is the original algorithm without any improvement; MOPSOCD_in is an improved version of MOPSOCD by employing only ECHI; NC-MOPSO is also an improved version of MOPSOCD, which utilizes both ECHI and NCLS. Table V shows the means and standard deviations of IGD over ten independent runs of these three algorithms. We can observe that for all the instances, the IGD of MOPSOCD_in is smaller than that of MOPSOCD, which validates the effectiveness of ECHI. In other words, the performance of MOPSOCD with ECHI is better than that with random initialization. Furthermore, we find that the IGD of NC-MOPSO is smaller than that of MOPSOCD_in, which verifies the effectiveness of NBLS. Fig. 3 presents the distributions of non-dominated solutions of these three algorithms on the instance Uninett. It is obvious that NC-MOPSO achieves better solutions than MOPSOCD and MOPSOCD_in, which further confirms the effectiveness of these two proposed improvement operators.

TABLE V
IGD (MEAN(STD)) FOR MOPSOCD, MOPSOCD_IN AND NC-MOPSO ON EACH TEST INSTANCE.

Instance	MOPSOCD	MOPSOCD_in	NC-MOPSO
BA300	0.2303(0.0421)	0.0116(0.0008)	**0.0026(0.0001)**
WS300	0.0982(0.0108)	0.0621(0.0157)	**0.0012(0.0004)**
118-bus	0.0121(0.0071)	0.0061(0.0019)	**0.0012(0.0004)**
Email-enron-only	0.1249(0.0484)	0.0163(0.0044)	**0.0021(0.0008)**
Uninett	0.0128(0.0028)	0.0055(0.0005)	**0.0008(0.0002)**
Barcelona	0.2046(0.0121)	0.0749(0.0017)	**0.0017(0.0007)**

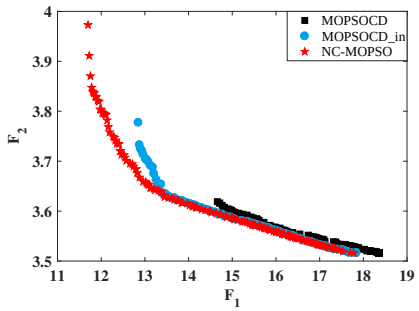


Fig. 3. Non-dominated solutions (PFs) of MOPSOCD, MOPSOCD_in and NC-MOPSO for the Uninett instance.

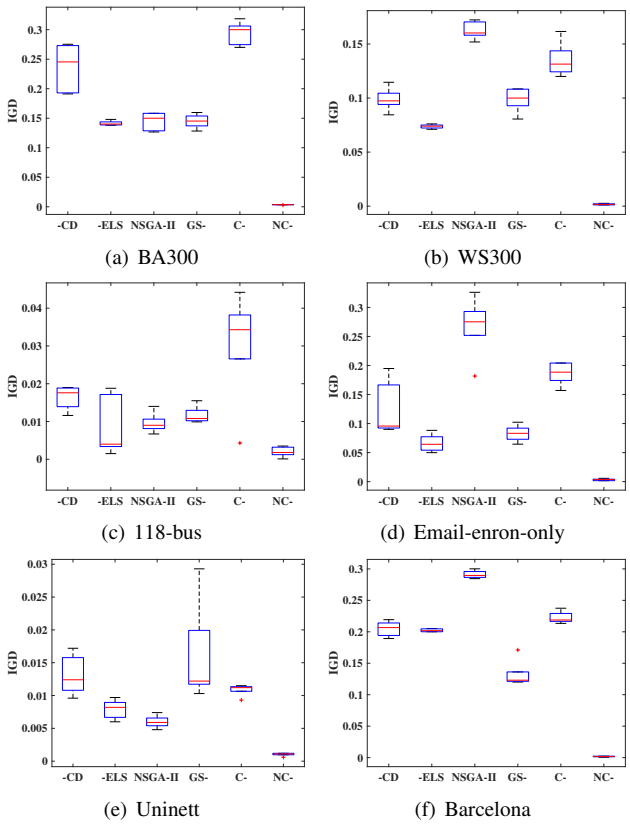
Next, we compare NC-MOPSO with the five baseline algorithms on the three popular metrics: IGD, C-metric and HV. The results of each algorithm are obtained by averaging over ten independent runs. The statistical results of IGD and HV are presented in Tables VI and VII, respectively. For a network instance, the performance ranking of each algorithm is given in square brackets, while the standard deviation of metric values is in parentheses. We conduct the Wilcoxon rank sum test at a 5% significance level to check whether the differences between the metric values yielded by NC-MOPSO and the other compared algorithms are significant. The symbols $\dagger$, $\S$ and $\approx$ indicate that the performance of NC-MOPSO is better than, worse than and similar to the baseline algorithms, respectively. Table VIII shows the overall performance of all algorithms via the Wilcoxon rank sum test. Figs. 4, 5 and 6 present the box plots of IGD, C-metric and HV of each algorithm for the six test instances, where "-CD", "-CDELS", "GS-", "C-"and "NC-" represent MOPSOCD, MOPSOCDELS, GSPSO, CMOPSO and NC-MOPSO, respectively.

TABLE VI
PERFORMANCE COMPARISON AMONG MOPSOCD, MOPSOCDELS, NSGA-II, GSPSO, CMOPSO AND NC-MOPSO IN TERMS OF IGD (MEAN[RANKING](STD)).

Instance	MOPSOCD	MOPSOCDELS	NSGA-II	GSPSO	CMPSO	NC-MOPSO
BA300	0.2303†[5](0.0421)	0.1358†[3](0.0041)	0.1215†[2](0.0181)	0.1384†[4](0.0122)	0.2891†[6](0.0167)	**0.0026[1](0.0001)**
WS300	0.0982†[3](0.0108)	0.0707†[2](0.0043)	0.1412†[6](0.0101)	0.0998†[4](0.0231)	0.1334†[5](0.0162)	**0.0012[1](0.0004)**
118-bus	0.0121†[5](0.0071)	0.0072≈[4](0.0065)	0.0024≈[2](0.0012)	0.0047†[3](0.0016)	0.0319†[6](0.0166)	**0.0012[1](0.0004)**
Email-enron-only	0.1249†[4](0.0484)	0.0633†[2](0.0171)	0.2313†[6](0.0581)	0.0663†[3](0.0128)	0.1941†[5](0.0311)	**0.0021[1](0.0008)**
Uninett	0.0128†[5](0.0028)	0.0075†[3](0.0013)	0.0046†[2](0.0015)	0.0144†[6](0.0071)	0.0108†[4](0.0009)	**0.0008[1](0.0002)**
Barcelona	0.2046†[4](0.0121)	0.2023†[3](0.0024)	0.2911†[6](0.0061)	0.1321†[2](0.0219)	0.2276†[5](0.0095)	**0.0017[1](0.0007)**
†/§/ ≈	6/0/0	5/0/1	5/0/1	6/0/0	6/0/0	

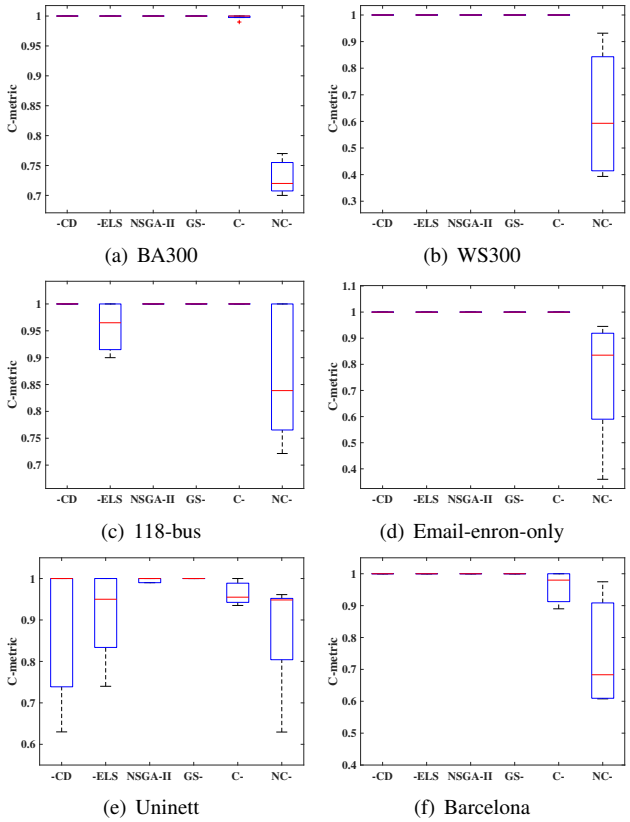
TABLE VII
PERFORMANCE COMPARISON AMONG MOPSOCD, MOPSOCDELS, NSGA-II, GSPSO, CMOPSO AND NC-MOPSO IN TERMS OF HV (MEAN[RANKING](STD)).

Instance	MOPSOCD	MOPSOCDELS	NSGA-II	GSPSO	CMOPSO	NC-MOPSO
BA300	0.3563†[5](0.0382)	0.4367†[3](0.0139)	0.4349†[4](0.0193)	0.4492†[2](0.0102)	0.2957†[6](0.0158)	**0.6381[1](0.0001)**
WS300	0.4251†[3](0.0175)	0.4684†[2](0.0051)	0.1791†[6](0.0108)	0.3424†[5](0.0374)	0.3611†[4](0.0198)	**0.6297[1](0.0021)**
118-bus	0.3503†[5](0.0105)	0.3695†[4](0.0199)	0.3947†[2](0.0194)	0.3701†[3](0.0117)	0.3255†[6](0.0369)	**0.4273[1](0.0038)**
Email-enron-only	0.4431†[4](0.0436)	0.5081†[2](0.0215)	0.3111†[6](0.0448)	0.4859†[3](0.0184)	0.3834†[5](0.0172)	**0.5740[1](0.0042)**
Uninett	0.4101†[6](0.0176)	0.4532†[3](0.0141)	0.4811†[2](0.0107)	0.4425†[4](0.0118)	0.4247†[5](0.0074)	**0.5868[1](0.0036)**
Barcelona	0.3618†[3](0.0348)	0.3465†[4](0.0164)	0.2348†[6](0.0184)	0.4229†[2](0.0132)	0.3367†[5](0.0167)	**0.5989[1](0.0062)**
†/§/ ≈	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0	



(a) BA300 (b) WS300 (c) 118-bus (d) Email-enron-only (e) Uninett (f) Barcelona

Fig. 4. The box-plot of the metric: IGD for all the test instances.



(a) BA300 (b) WS300 (c) 118-bus (d) Email-enron-only (e) Uninett (f) Barcelona

Fig. 5. The box-plot of the metric: C-metric for all the test instances.

TABLE VIII
OVERALL PERFORMANCE COMPARISON OF THE SIX ALGORITHMS ON THE SIX INSTANCES.

	Mean ranking	Total †/§/ ≈
MOPSOCD	4.33	12/0/0
MOPSOCDELS	2.92	11/0/1
NSGA-II	4.17	11/0/1
GSPSO	3.42	12/0/0
CMOPSO	5.16	12/0/0
NC-MOPSO	1.00	-

We note that smaller values are better for IGD and C-metric, while the opposite is true for HV. According to the results in Tables VI-VIII and Figs. 4-6, we can conclude that NC-MOPSO is superior to the other five competitors for almost all the test instances. Specifically, in terms of IGD (Table VI and Fig. 4), NC-MOPSO has the smallest value on all the test instances, which means it has the best performance among all the algorithms. The poor IGD results of MOPSOCD and CMOPSO indicate that their solutions have

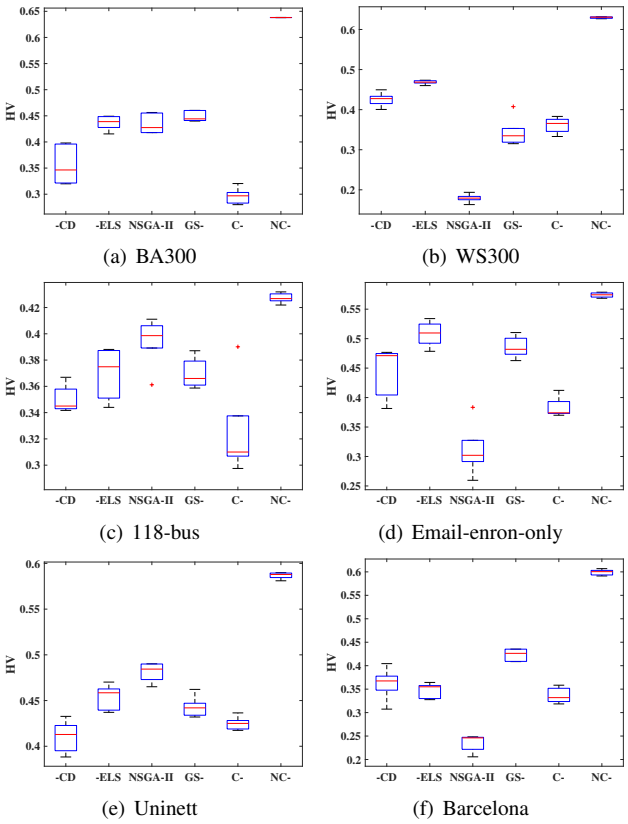


Fig. 6. The box-plot of the metric: HV for all the test instances.

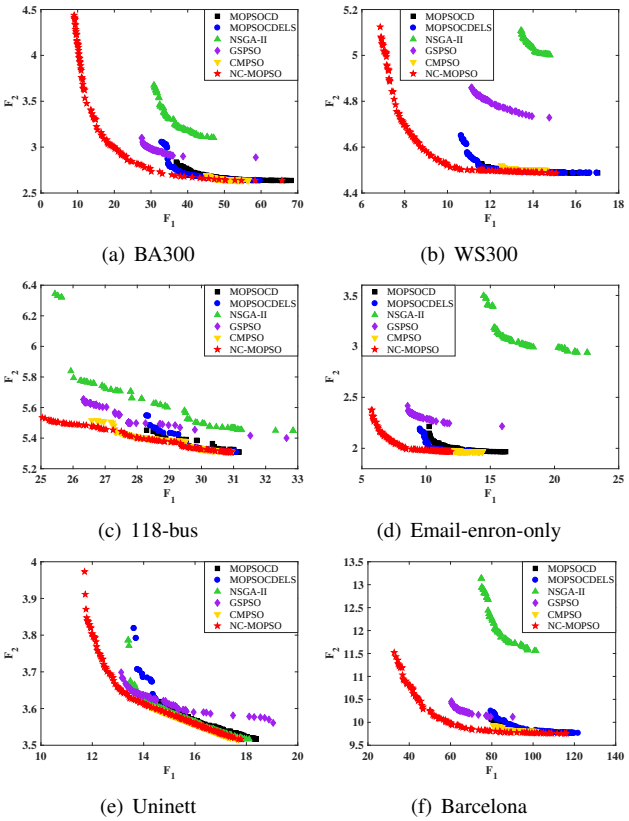


Fig. 7. Non-dominated solutions (PFs) of all the algorithms for each instance.

small diversity. Note that for the 118-bus network, NSGA-II and MOPSOCDELS have similar IGD values with NC-MOPSO. For C-metric (Fig. 5), the value of NC-MOPSO is less than one for all the test instances, which means that part of the solutions obtained by NC-MOPSO cannot be dominated by the true PF. The other algorithms all have poor performance in terms of C-metric. Especially for GSPSO, the C-metric values are always one for all the instances, which means all the solutions are dominated by the true PF. With regard to the HV metric (Table VII and Fig. 6), NC-MOPSO is also superior to the other five algorithms. The mean ranking and the total number of †/§/≈ (Table VIII) show that overall NC-MOPSO outperforms all the other comparison algorithms.

Moreover, we present the non-dominated solutions of all algorithms for each instance graphically in Fig. 7 to highlight the advantage of NC-MOPSO. We can clearly see from the distributions of non-dominated solutions that NC-MOPSO obtains the higher-quality solutions, i.e., better PFs, compared with the other algorithms.

Finally, we analyze the convergence of all MOEAs along with their actual CPU time. The results of convergence are presented in Fig. 8, where the HV metric is considered. We see that all the MOEAs generally have good convergence, some of which have occasionally slight fluctuations due to the local minima problem. In addition, we compare the actual run time of all the MOEAs on instance BA300 in Table IX. We can find that given the same experimental setting, the CPU time of NC-MOPSO is slightly larger than MOPSOCD and MOPSOCDELS, yet much smaller than the other comparison algorithms. Overall, the comparative experiments demonstrate

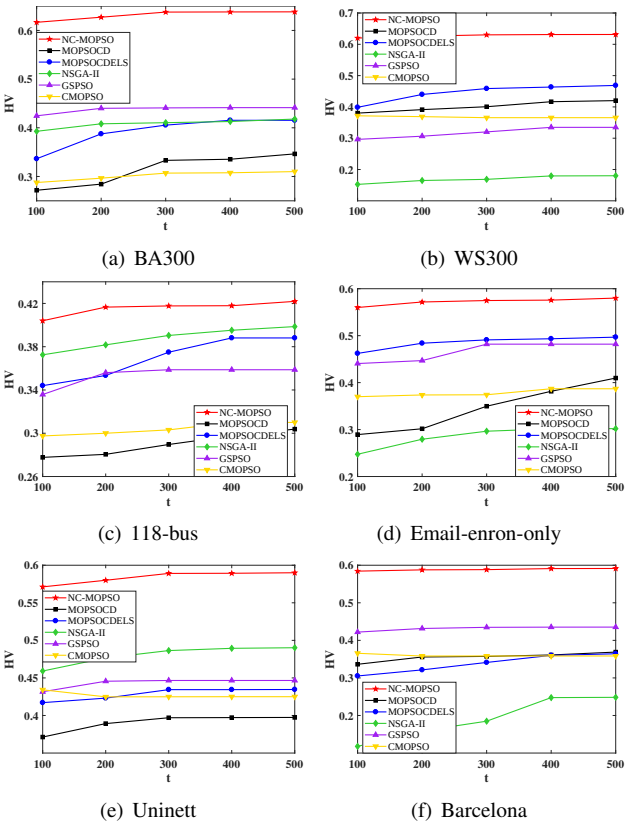


Fig. 8. Convergence analysis on the six MOEAs for each instance.

that NC-MOPSO not only obtains high-quality PFs, but also has good convergence property and relatively low computational cost. The good performance of NC-MOPSO is mainly owing to the following two facts:

1) An edge-centrality guided hybrid initialization, ECHI, is employed to generate high-quality initial solutions.
2) A node-centrality guided local search, NCLS, is adopted to expand the search space and improve the quality of solutions.

TABLE IX
THE CPU TIME OF EACH ALGORITHM ON INSTANCE BA300. THE ITERATION NUMBER IS $t = 500$. EXPERIMENTAL ENVIRONMENT: INTEL CORE I5 PROCESSOR (2.3 GHZ, 4 CORES) AND 8 GB RAM.

Algorithms	CPU time(s)
MOPSOCD	3074.39
MOPSOCDELS	3027.81
NSGA-II	12570.31
GSPSO	13790.21
CMOPSO	6311.51
NC-MOPSO	3215.91

VI. CONCLUSION

In this paper, transport optimization on complex networks is transformed as a bi-objective optimization problem by simultaneously maximizing the network capacity and minimizing the average number of hops. The main advantage of this transformation is that it can provide decision makers with a holistic view for optimizing the network transport performance. Furthermore, a network centrality based MOEA named NC-MOPSO is proposed to solve the optimization problem. In this algorithm, an edge-centrality guided hybrid initialization is proposed to provide high-quality initial solutions, and a node-centrality guided local search is developed to enhance the exploration of search space. The comparative experimental results on both network models and real networks demonstrate the high quality and efficiency of the proposed algorithm. Note that in our work we only consider static complex networks. Transport optimization on dynamic and multi-layer networks needs to be further explored. Moreover, how to apply our optimization framework and algorithm for the network flow involving multiple source-sink paths is an open problem.

REFERENCES

[1] Cisco, "Cisco annual internet report (2018-2023)," *White Paper*, 2020.

[2] R. Guimerà, A. Díaz-Guilera, F. Vega-Redondo, A. Cabrales and A. Arenas, "Optimal network topologies for local search with congestion," *Physical Review Letters*, vol. 89, no. 24, pp. 248701, 2002.

[3] Y. Jiang, Y. Zou, H. Guo, T. A. Tsiftsis, M. R. Bhatnagar, R. C. de Lamare and Y. D. Yao, "Joint power and bandwidth allocation for energy-efficient heterogeneous cellular networks," *IEEE Transactions on Communications*, vol. 67, no. 9, pp. 6168–6178, 2019.

[4] G. Yan, T. Zhou, B. Hu, Z. Q. Fu, "Efficient routing on complex networks," *Physical Review E*, vol. 73, no. 4, pp. 046108, 2006.

[5] Y. Lin, J. Zhang, B. Yang, H. Liu and L. Zhao, "An optimal routing strategy for transport networks with minimal transmission cost and high network capacity," *Physica A: Statistical Mechanics and its Applications*, vol. 521, pp. 551–561, 2019.

[6] R. Guerin and A. Orda, "Computing shortest paths for any number of hops," *IEEE/ACM transactions on networking*, vol. 10, no. 5, pp. 613–620, 2002.

[7] M. Ericsson, M. G. C. Resende and P. M. Pardalos, "A genetic algorithm for the weight setting problem in ospf routing," *Journal of Combinatorial Optimization*, vol. 6, no. 3, pp. 299–333, 2002.

[8] L. Buzna and R. Carvalho, "Controlling congestion on complex networks: fairness, efficiency and network structure," *Scientific Reports*, vol. 7, no. 1, pp. 1–15, 2017.

[9] B. Danila, Y. Yu, J. A. Marsh and K. E. Bassler, "Optimal transport on complex networks," *Physical Review E*, vol. 74, no. 4, pp. 046106, 2006.

[10] T. N. Bui and C. Jones, "Finding good approximate vertex and edge partitions is NP-hard," *Information Processing Letters*, vol. 42, no. 3, pp. 153–159, 1992.

[11] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[12] C. A. C. Coello, G. T. Pulido and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 256–279, 2004.

[13] B. Liu, Z. Li, X. Chen, Y. Huang and X. Liu, "Recognition and vulnerability analysis of key nodes in power grid based on complex network centrality," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 3, pp. 346–350, 2017.

[14] F. A. Rodrigues, "Network centrality: an introduction," *A mathematical modeling approach from nonlinear dynamics to complex systems*, pp. 177–196, Springer, Cham, 2019.

[15] L. C. Freeman, "A Set of Measures of Centrality Based on Betweenness." *Sociometry*, vol. 40, no. 1, pp. 35–41, 1977.

[16] S. Dolev, Y. Elovici and R. Puzis, "Routing betweenness centrality," *Journal of the ACM*, vol. 57, no. 4, pp. 1–27, 2010.

[17] A. Vàzquez-Rodas and J. Luis, "A centrality-based topology control protocol for wireless mesh networks," *Ad Hoc Networks*, vol. 24, pp. 34–54, 2015.

[18] Z. H. Guan, L. Chen and T. H. Qian, "Routing in scale-free networks based on expanding betweenness centrality," *Physica A: Statistical mechanics and its applications*, vol. 390, no. 6 pp. 1131–1138, 2011.

[19] P. Pantazopoulos, M. Karaliopoulos and I. Stavrakakis, "Distributed placement of autonomic internet services," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1702–1712, 2013.

[20] Z. Zhang, S. Liu, Y. Yang and Y. Bai, "A link-adding strategy for improving robustness and traffic capacity in large-scale wireless sensor networks," *Cluster Computing*, vol. 22, no. 3, pp. 7687–7694, 2019.

[21] J. Wu, K. T. Chi, F. C. Lau and I. W. Ho, "Analysis of communication network performance from a complex network perspective," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 12, pp. 3303–3316, 2013.

[22] Z. Y. Jiang, M. G. Liang, Q. Li and D. C. Guo, "Optimal dynamic bandwidth allocation for complex networks," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 5, pp. 1256–1262, 2013.

[23] H. Liu and Y. Xia, "Optimal resource allocation in complex communication networks," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 7, pp. 706–710, 2015.

[24] C. L. Pu, S. Y. Zhou, K. Wang, Y. F. Zhang and W. J. Pei, "Efficient and robust routing on scale-free networks," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 3, pp. 866–871, 2012.

[25] P. Li, L. Guo and F. Wang, "A multipath routing protocol with load balancing and energy constraining based on AOMDV in ad hoc network," *Mobile Networks and Applications*, vol. 26, pp. 1871–1880, 2019.

[26] C. Kirst, M. Timme and D. Battaglia, "Dynamic information routing in complex networks," *Nature Communications*, vol. 7, no. 1, pp. 1–9, 2016.

[27] S. J. Yang, "Exploring complex networks by walking on them," *Physical Review E*, vol. 71, no. 1, pp. 016107, 2005.

[28] P. M. Pardalos, A. Zilinskas and J. Zilinskas, "Non-convex multi-objective optimization," *New York: Springer International Publishing*, 2017.

[29] P. M. Pardalos, A. Migdalas and L. eds. Pitsoulis, "Pareto optimality, game theory and equilibria," *Springer Science and Business Media*, 2008.

[30] S. Wang and J. Liu, "A multi-objective evolutionary algorithm for promoting the emergence of cooperation and controllable robustness on directed networks," *IEEE Transactions on Network Science and Engineering*, vol. 5, no. 2, pp. 92–100, 2018.

[31] J. H. Roach, R. J. Marks and B. B. Thompson, "Recovery from sensor failure in an evolving multiobjective swarm," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 1, pp. 170–174, 2015.

[32] L. Zhang, J. Xia, F. Cheng, J. Qiu and X. Zhang, "Multi-objective optimization of critical node detection based on cascade model in complex networks," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 3, pp. 2052–2066, 2020.

[33] M. Zhou and J. Liu, "A two-phase multiobjective evolutionary algorithm for enhancing the robustness of scale-free networks against multiple malicious attacks," *IEEE Transactions on Cybernetics*, vol. 47, no. 2, pp. 539–552, 2017.

[34] M. Gong, Q. Cai, X. Chen and L. Ma, "Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 1, pp. 82–97, 2014.

[35] X. Zhang, Y. Zhou, Q. Zhang, V. C. Lee and M. Li, "Problem specific MOEA/D for barrier coverage with wireless sensors," *IEEE Transactions on Cybernetics*, vol. 47, no. 11, pp. 3854–3865, 2017.

[36] W. Du, M. Zhang, W. Ying, M. Perc, K. Tang, X. Cao and D. Wu, "The networked evolutionary algorithm: A network science perspective," *Applied Mathematics and Computation*, vol. 338, pp. 33–43, 2018.

[37] D. Wu, N. Jiang, W. Du, K. Tang and X. Cao, "Particle swarm optimization with moving particles on scale-free networks," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 497–506, 2020.

[38] W. Du, W. Ying, P. Yang, X. Cao, G. Yan, K. Tang and D. Wu, "Network-based heterogeneous particle swarm optimization and its application in UAV communication coverage," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 3, pp. 312–323, 2020.

[39] C. R. Raquel and P. C. Naval Jr, "An effective use of crowding distance in multiobjective particle swarm optimization," in *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, pp. 257–264, 2005.

[40] L. Zhao, Y. C. Lai, K. Park, "Onset of traffic congestion in complex networks," *Physical Review E*, vol. 71, no. 2, pp. 026125, 2005.

[41] R. Poli, J. Kennedy and T. Blackwell, "Particle swarm optimization," *Swarm Intelligence,* vol. 1, no. 1, pp. 33–57, 2007.

[42] J. Wu, et. al., "Supplemental material: Multi-objective optimization of transport processes on complex networks," [Online]. Available: https://github.com/pangzixin/NCMOPSO/blob/main/appendix.pdf

[43] S. Ding, C. Chen, B. Xin and P. M. Pardalos, "A bi-objective load balancing model in a distributed simulation system using NSGA-II and mopso approaches," *Applied Soft Computing*, vol. 63, pp. 249–267, 2018.

[44] W. Yuan, Y. Liu, H. Wang and Y. Cao, "A geometric structure-based particle swarm optimization algorithm for multiobjective problems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 9, pp. 2516–2537, 2017.

[45] X. Zhang, X. Zheng, R. Cheng, J. Qiu and Y. Jin, "A competitive mechanism based multi-objective particle swarm optimizer with fast convergence," *Information Sciences*, vol. 427, pp. 63–76, 2018.

[46] A. L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[47] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[48] R. Rossi and N. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[49] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden and M. Roughan, "The internet topology zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011.

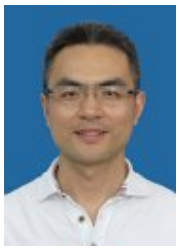
[50] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms–a comparative case study," in *International Conference on Parallel Problem Solving from Nature*, pp. 292–301, 1998.

[51] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[52] H. Ishibuchi, H. Masuda, Y. Tanigaki and Y. Nojima, "Difficulties in specifying reference points to calculate the inverted generational distance for many-objective optimization problems," in *2014 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making (MCDM)*, pp. 170–177, 2014.

Jiexin Wu received the B.E. degree in network engineering from Nanjing University of Science and Technology, Nanjing, China, in 2016. She is currently pursuing the Ph.D. degree in computer science and technology at the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China.

Her research interests include network optimization and network science.



Cunlai Pu received the Ph.D. degree in Information and Communication Engineering from Southeast University, Nanjing, China, in 2012.

He is currently an Associate Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. His current research interests include network science, communication systems and network optimization.

Shuxin Ding (Member IEEE) received the B.E. degree in automation and the Ph.D. degree in control science and engineering from the Beijing Institute of Technology, Beijing, China, in 2012 and 2019, respectively.

From 2016 to 2017, he was a Visiting Scholar of Industrial and Systems Engineering at the University of Florida, Gainesville, FL, USA. He is currently an Associate Researcher with the Signal and Communication Research Institute, China Academy of Railway Sciences Corporation Limited. His current research interests include railway scheduling, evolutionary computation, multiobjective optimization, and optimization under uncertainty.

Guo Cao received the Ph.D. degree in pattern recognition and intelligent system from Shanghai Jiaotong University, Shanghai, China, in 2006.

He is a Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. His current research interests include pattern recognition, machine learning and intelligent systems.

Chengyi Xia (Member IEEE) received the B.S. degree in mechanical engineering from the Hefei University of Technology, Hefei, China, in 1998, the M.Sc. degree in nuclear energy science and engineering from the Institute of Plasma Physics, Chinese Academy of Science, Hefei, in 2001, and the Ph.D. degree in control theory and control engineering from Nankai University, Tianjin, China, in 2008.

Since 2022, he is a full professor with the School of Control Science and Engineering, Tiangong University, Tianjin, China. His research interests include complex system modeling and analysis, complex networks, epidemic propagation, and evolutionary game theory.

Panos M. Pardalos received the B.S. degree in mathematics from the Athens University, Athens, Greece, in 1977, the M.S. degree in mathematics and computer science from the Clarkson University, Potsdam, NY, USA, in 1978, and the Ph.D. degree in computer and information sciences from the University of Minnesota, Minneapolis, MN, USA, in 1985.

He is a Distinguished Professor and the Paul and Heidi Brown Preeminent Professor of Industrial and Systems Engineering at the University of Florida, and a World Renowned Leader in Global Optimization, Mathematical Modeling, and Data Sciences. He has published over 500 papers, edited/authored more than 200 books, and organized over 80 conferences.

Prof. Pardalos was a recipient of the 2018 Humboldt Research Award, the 2013 Constantin Caratheodory Prize of the International Society of Global Optimization, and the 2013 EURO Gold Medal Prize bestowed by the Association for European Operational Research Societies. He is the Founding Editor of Optimization Letters and Energy Systems, and the Co-Founder of Journal of Global Optimization. He is also a Foreign Member of the Lithuanian Academy of Sciences, the Royal Academy of Spain, and the National Academy of Sciences of Ukraine. He is a fellow of AAAS, AIMBE, and INFORMS.