A Mixed Integer Programming Approach for Economical Network Design Problem

Yuejiang Sun*, Qiuling Zhao, Hongjie Chen

***Abstract*—Recently, extensive real-world networks have emerged astounding structural properties, such as high-robustness, tree-hierarchies and low-cost, which bring deep insights into studying graph theory, combinatorics and topology, to name a few. In this paper, we investigate how to design an economical (or low-cost) networks with special graph structures as requirement. Our specific objective is to design a network with minimum construction cost while at least two basic but fundamental graph conditions are required as constraints: (i) network connectivity, implying no isolated subgraph exists in the network; (ii) minimum level of capacity requirement at each node must be satisfied. We propose a mixed-integer programming (MIP) method to efficiently construct optimal economical networks. Our computational results show that no more than n edges are required for designing a network satisfying the above two conditions. We also tackle a special network design case when network structure is a spanning tree. Such an economical spanning tree design problem (ESTDP) which has been proved to a NP-hard problem in S. Nakano's previous work. Three different MIP formulations are proposed and evaluated on extensive numerical studies. Our computational results demonstrate Martin's formation outperforms the *subtour elimination* formulation and *cutset* formulation due to their non-exponential increasing constraints in terms of computational time and optimality gap.**

***Index Terms*—NP-completeness, network design, mathematical programming, economical networks**

I. Introduction

GIVEN a set of nodes V=$\{1, 2, ..., n\}$, each node $i \in V$ requires a positive integer value of demand, denoted by ω_i. Initially, we have a empty network $G = (V, E)$ where edge set E is empty. Every added edge e_{ij} between node i and j will serve as a supply channel and provide a certain amount of goods to both node i and node j. In this paper, we aim to design an economical network satisfying the following fundamental network properties: (i) network connectivity or *reachability*; (ii) node demand satisfaction condition. Specifically, the connectivity condition requires any two nodes are connected among some path, which implies at least $n - 1$ edges are added to the network. Assume N_v is the set of neighbors connecting node v. To satisfy its node demand ω_v, every added edge e_{ij}, $i, j \in V$ to the network will be assigned a positive integer flow f_{ij} such that the total flows incident to node v $\sum_{i \in N_v} f_{vi}$ is at least the weight ω_v. Meanwhile, every new edge e_{ij} added to the network will cause cost c_{ij} to total budget. Our objective of interest is to minimize $\sum_{i,j \in V} c_{ij} f_{ij}$ while keeping the connectivity and demand satisfaction conditions satisfied. Without specific network structure requirements, such as spanning tree, minimum staring shape and clique, we name it as generic economical network design problem (ENDP).

From the perspective of real applications, let us clarify why understanding this problem may provide an abstract model of real-world networks and bring the underlying benefits for designing artificial networks with respect to high-efficiency, strong-robustness and low-cost. One intuitive realistic case arising in facility location network design can be categorized as ENDP. Suppose there are multiple factories, each of them requires a certain amount of goods from other factories to maintain their functioning of assembly lines [2]. Therefore, transportation network for these factories are needed constructed and its cost must be reduced as much as possible. From this perspective, how to design a network that captures properties of low-cost, high robustness, and high-effectiveness is of crucial importance. Meanwhile, the network should provide sufficient transport capacity for all factory with the traffic unimpeded. Another interesting application for this problem can be found in router data package deliver problem. Suppose one computer sends at most ω_{ij} data packets along edge e_{ij}. Meanwhile, the total amount of packages this computer needs to handle should be at least $\omega_v = \sum_{j \in N_v} \omega_{vj}$. If this total packets amount it handles is greater than a minimum service level, our network will satisfy our needs. Moreover, to ensure every node's network functionality, all nodes must be encompassed. Even though some node is not directly connected with another node, it can still communicate via various routes as long as the network is one complete component. As data is transmitting within network, cost minimization naturally becomes one of top priorities in our agenda. In general, these network design problems are increasingly extended and applied to fields as traffic management, supply chains [3], [4], and epidemiology.

Furthermore, it is also interesting to investigate ENDP from the viewpoint of optimization. In the field of complex graphs, small-world networks have been the focus of interest because of their potential as models for the interaction networks of complex systems [5], [6], [7], [8], [9], [10], [11], [12]. For example some networks arise from a natural random process where an "efficient" network could mean a relatively few edges lead to fairly small average distance. Recalling the context of ENDP, if we know all demands of nodes (potential factories or customers), can we design an optimal network in a practical time to minimize the cost while keeping the flow circulating through the whole net-

Manuscript received Feb. 28, 2019; revised Apr. 19, 2019. The work is supported by Research on Adaptive Learning Model of Artificial Intelligence – Take Intelligent Manufacturing Course Group as an Example (No. JG201926) and Project of Shandong Province Higher Educational Science and Technology Program (No. KJ2018ZBB021)

Y. Sun is with Qingdao Technical College, No. 369, Qiantangjiang Road, Qingdao West Coast New Area, Qingdao, Shandong 266555, China. e-mail: yuejiangsun2@163.com.

Q. Zhao and H. Chen are with Qingdao Technical College, No. 369, Qiantangjiang Road, Qingdao West Coast New Area, Qingdao, Shandong 266555, China.

work? The answer is intuitively positive if the following sub-problems are well-defined and being answerable: (i) what is the network structure and how to define it mathematically; (ii) how to assign the flow on the edges such that the total cost is minimized.

First of all, network connectivity is one of the most fundamental requirements in graph optimization problems [13], [14]. As a mandatory property in ENDP, its relaxation will simplify the problem as a linear program (LP), which can be solved trivially by using any current LP solvers. Previous work [1] has provided a specific algorithm for ENDP, which can be run in linear time. Apparently, any designed network (not tree) must contain at least n edges to ensure the connectivity. However, an interesting question arises as well: what if only $n-1$ edges are allowed to the network? Does it increase the complexity of solving? The authors [1] proved such induced problem is NP-hard. This problem becomes every harder when only integer flows are allowed to assign on edges. Although integer flows are not necessarily meaningful in reality, we know such integrity will make this problem more difficult enough. Relaxing integrity will lead to an easier case. Therefore, we focus on solving difficult scenario of this problem, thereby stronger results could be obtained.

In this study, we are interested in closing aforementioned gaps by introducing a mixed-integer programming (MIP) approach which can find the least cost networks while some conditions are satisfied. Our contributions are summarized as follows. (1) We first define and introduce the generic economical network design problem (ENDP). Later, a special NP-hard variant of this problem, called economical spanning tree design problem (ESTDP), is introduced as well. (2) To solve the general network design problem ENDP, a general MIP formulation is proposed as a methodology framework to solve such category of problems efficiently. Specifically, this model can be easily used to solve the simplified version of ENDP in S. Nakano's work [1]. (3) We develop three different formulations to tackle ESTDP. Generally speaking, depending on context and types of structural property that is used, it is very likely to extend our framework for cases where network structures vary, such as star and clique. (4) To solve these formulations efficiently, we initially relax exponential subtour elimination constraints from the formulations and use Gurobi callback method to generate them only if master problem solutions violate these constraints occurs, which saves computational time significantly. (5) We perform an extensive set of computational experiments to validate the efficiency and applicability of the proposed formulations.

II. Preliminaries

In previous work [1], the authors defined a simplified variant of ENDP, where no edge cost and the number of edges are specified, and proposed an efficient algorithm to find the optimal network. In this context, the possible "extra", "loss", or "overflow" capacity at node v, is defined as,

$$L_v = \sum_{j \in V} f_{vj} - \omega_v. \tag{1}$$

The total loss over the whole network is,

$$L = \sum_{v \in V} \Big(\sum_{j \in V} f_{vj} - \omega_v \Big) = 2 \sum_{i,j \in V} f_{ij} - \sum_{v \in V} \omega_v. \tag{2}$$

When solving $\min\{L\}$, it is actually equivalent to $\min_{i,j \in V} f_{ij}$ since constant terms or positive multiplier can be ignored.

Theorem 1: The time and space complexity for solving ENDP is $O(n)$ and $O(n)$, respectively and the number of edges in this network is no more than n, where n is number of nodes in the network [1].

Theorem 1 tells us that, when neither cost capacity c on edges nor node capability ω are specified, solving the ENDP is quite easy, which can be solved in linear time and space. However, when restricting the number of edges of network to $n-1$, this problem becomes computationally difficult to solve (NP-hard), not to mentioning assignment of positive integer flow cost on network edges.

III. Mathematical Models for Economical Network Design Problem

A. Economical Network Design Problem

As aforementioned in Section I, we consider decision variable f_{ij} for representing positive integer flow assigned on edge e_{ij}. By formulating minimum node weight requirement and network connectivity condition as constraints, we can represent ENDP as the following integer programming model,

$$[\mathcal{M}_1] \quad \min \quad \sum_{i,j \in V} c_{ij} f_{ij} \tag{3a}$$

$$\text{s. t.} \quad \sum_{j \in V} f_{ij} \geq \omega_i, i = \{1, 2, ..., n\}, \tag{3b}$$

$$\sum_{i \in S, j \in \bar{S}} f_{ij} > 0, \forall\ S \subset V, \bar{S} = V \setminus S, \tag{3c}$$

$$f_{ij} \in \mathbf{Z}^+, \forall\ i, j \in V. \tag{3d}$$

In model $\mathcal{M}_1$, the objective function is to minimize the total cost of shipping all positive flows on the network. Constraints $(3b)$ specify node demand requirements and constraints $(3c)$ says nodes in a subset $S \subset V$ of the resulting network must have positive flow with nodes in its complement set $\bar{S} = V \setminus S$, which enforces network connectivity. Note that the number of constraints $(3c)$ increase exponentially with n.

Before calling the-state-of-art solvers to solve $\mathcal{M}_1$ directly, let us first discuss its characteristic or formulation properties, in that model could be implemented more efficiently. If constraints (3c) are removed from the model, solving such relaxed problem is to solve a linear programming (LP) problem due to the unimodularity – integer variables f_{ij} can be relaxed as continuous variables such that solving IP is equivalent to solve LP, which can be solved in a fast way using one of the-state-of-art solvers. But the resulting network may comprise multiple disconnected subcomponents. According to Theorem 1, the resulting network has exactly n or $n-1$ edges if we set $c_{ij} = 1$, for all $i, j \in V$, which is confirmed by comparing results between this model

and the algorithm proposed in [1]. Furthermore, since model $\mathcal{M}_1$ does not restrict the number of added edges, the cost of solving "exponential" formulation is rather inexpensive. In Section IV, we will demonstrate the efficiency of this model by testing it on instances of 100 nodes and 200 nodes, respectively.

B. Economical Spanning Tree Design Problem

To model ESTDP, we propose a "two-stage" mathematical model. An additional set of binary variables $x_{ij} \in \{0,1\}, i,j \in V$ is introduced, where x_{ij} equals to one when there is an edge between node i and node j and $x_{ij} = 0$ means no such edge exists. The following *subtour elimination* formulation for ESTDP, denoted by $\mathcal{M}_2$, is given as,

$$[\mathcal{M}_2] \quad \min \quad \sum_{i,j\in V} c_{ij} f_{ij} \tag{4a}$$

$$\text{s.t.} \quad \sum_{i,j\in V} x_{ij} = n-1, \tag{4b}$$

$$\sum_{i,j\in S} x_{ij} \le |S| - 1, \forall\ S \subset V, S \neq \emptyset, \tag{4c}$$

$$f_{ij} \le K x_{ij}, \forall\ i,j \in V, \tag{4d}$$

$$\sum_{j\in V} f_{ij} \ge \omega_i, \forall\ i \in V, \tag{4e}$$

$$x_{ij} \in \{0,1\}, \forall\ i,j \in V, \tag{4f}$$

$$f_{ij} \in \mathbf{Z}^+, \forall\ i,j \in V. \tag{4g}$$

Constraints (4b) state there are exactly $n-1$ edges in network. Constraints (4c) are the subtour elimination constraints which means any subset of nodes S must have at most $|S|-1$ edges contained in that subset, which ensure no subtours. Constraints (4c) state that the weight assigned on edge e_{ij} is restricted by the existence of that edge. If edge e_{ij} exists in network, then the assigned weight should be less than a sufficient larger integer K. Otherwise, the assigned weight f_{ij} should be zero. Constraints (4e) have the same meaning as $(3b)$ in $\mathcal{M}_1$.

To tackle ESTDP from various formulations, another formulation, called *cutset* formulation, is proposed as,

$$[\mathcal{M}_3] \quad \min \quad \sum_{i,j\in V} c_{ij} f_{ij} \tag{5a}$$

$$\text{s.t.} \quad \sum_{i,j\in V} x_{ij} = n-1, \tag{5b}$$

$$\sum_{i\in S, j\in \bar{S}} x_{ij} \ge 1, \forall\ S \subset V, \bar{S} = V - S, \tag{5c}$$

$$f_{ij} \le K x_{ij}, \forall\ i,j \in V, \tag{5d}$$

$$\sum_{j\in V} f_{ij} \ge \omega_i, \forall\ i \in V, \tag{5e}$$

$$x_{ij} \in \{0,1\}, \forall\ i,j \in V, \tag{5f}$$

$$f_{ij} \in \mathbf{Z}^+, \forall\ i,j \in V. \tag{5g}$$

Compared with formulation $\mathcal{M}_2$, the only difference is that we replace the subtour elimination constraints with cutset constraints (5c).

Since both model $\mathcal{M}_2$ and model $\mathcal{M}_3$ are exponentially increased in constraint size, the computation time could be inacceptable as the instance size becomes large. Hence, we give another formulation, called *Martin*'s formulation. Let y_{ij}^k denote the edge e_{ij} exists in the spanning tree and node k is on the side of node j.

$$[\mathcal{M}_4] \quad \min \quad \sum_{i,j\in V} c_{ij} f_{ij} \tag{6a}$$

$$\text{s.t.} \quad \sum_{i,j\in V} x_{ij} = n-1, \tag{6b}$$

$$y_{ij}^k + y_{ji}^k = x_{ij}, \forall\ i,j,k \in V, \tag{6c}$$

$$\sum_{k\in V} y_{ik}^j + x_{ij} = 1, \forall\ i,j,k \in V, \tag{6d}$$

$$f_{ij} \le K x_{ij}, \forall\ i,j \in V, \tag{6e}$$

$$\sum_{j\in V} f_{ij} \ge \omega_i, \forall\ i \in V, \tag{6f}$$

$$x_{ij}, y_{ij}^k, y_{ji}^k \in \{0,1\}, \forall\ i,j,k \in V, \tag{6g}$$

$$f_{ij} \in \mathbf{Z}^+, \forall\ i,j \in V. \tag{6h}$$

Constraints $(6c)$ guarantee that if e_{ij} is selected into the tree ($x_{ij} = 1$), any node $k \in V$ must be either on the side of node j ($y_{ij}^k = 1$) or on the side of node i ($y_{ji}^k = 1$). If e_{ij} is not in the tree ($x_{ij} = 0$), any node k cannot be on the side of node j nor node i ($y_{ij}^k = y_{ji}^k = 0$). Constraints $(6d)$ ensure that if e_{ij} is in the tree, edge e_{jk} which connects node i are on the side of node i. Otherwise, there must be an edge e_{ik} such that node j is on the side of node k ($y_{ik}^j = 1$ for some node k).

IV. Simulation Experiment

In this section, we will examine the efficiency of the four proposed formulations testing on different sizes of instances. All instances in this paper are randomly generated according to the following recipe. Given n nodes, each of them is assigned with a positive integer generated randomly from ω_{min} to ω_{max}, where ω_{min} and ω_{max} are two pre-specified positive integers ($\omega_{max} > \omega_{min}$). Nodes in every instance are uniformly distributed among a panel $10n \times 10n$. The euclidean distance between two nodes actually indicates the cost of that edge. The computational experiments are performed on a PC with Intel Core I7-8850HQ 6 Core Processor with $vPro^{tm}$ and 32 GB of memory, running Windows 10. All formulations are implemented in C++ programming language under Visual Studio 2018 and Gurobi Optimizer 8.1. A 1,000-second limit is imposed on the computational time of all following instances and formulations.

In section IV-A, the efficiency of model $\mathcal{M}_1$ will be tested on instances of size 100 and 200 nodes. Section IV-B will further examine the effectiveness among formulations $\mathcal{M}_2$, $\mathcal{M}_3$ and $\mathcal{M}_4$ from the perspectives of CPU time and optimality gap which is mathematically defined as subtraction of lower bound from upper bound divided by lower bound.

A. Economical Network Design Problem

We start our computational study by showing the performance of model $\mathcal{M}_1$ on random instances in which node demands are randomly generated among different interval. For consistent comparison with the algorithm in paper [1],

all coefficients in the objective function are set to be one. ω_{min} and ω_{max} are two integers randomly selected from the interval [1, 300]. The computational results of model $\mathcal{M}_1$ over two sets of instances are reported in Table I and II, respectively. The information for instances are reported in first three columns. The column "Conn. cuts" reports the number of connectivity cuts are generated in this model. The column "Time(s)" reports the computational time and the column "Gap" reports the gap between the lower bound and upper bound when running on Gurobi. We observe that although the weight intervals vary significantly, the gap is always zero for instances of 100 nodes. We also observe that there is positive correlation between the "Conn. cuts" and "Time(s)". The larger the number of cuts is, the more CPU time it requires. This is because every violated subtour elimination constraint found by CALLBACK at the beginning are required to be added consequently to the model. In Table II, the model $\mathcal{M}_1$ is not capable of reaching optimal on all instances within 1,000 secs limit since generating model-required connectivity cuts consumes much more time.

B. Economical Spanning Tree Design

In this section, we compare difference of formulations $\mathcal{M}_2$, $\mathcal{M}_3$ and $\mathcal{M}_4$ with respect to computational time and optimality gap. The computational results over instances of 100 nodes and 150 nodes are reported in tables III and IV, respectively. Overall speaking, model $\mathcal{M}_2$ can always find the optimal solutions in both 100-nodes instances and 150-nodes instances. As different node demand ranges and node distance costs are assigned, the computational time varies significantly from different instances. However, we observe that the average time of $\mathcal{M}_2$ is generally larger than $\mathcal{M}_3$. Although $\mathcal{M}_4$ is a model of non-exponential size, the computational time is significantly larger than both $\mathcal{M}_2$ and $\mathcal{M}_3$. In addition, All three models reach optimality without no gap on these ten instances.

TABLE I: The computational performance of testing model $\mathcal{M}_1$ on 100-node instances

Instance	ω_{min}	ω_{max}	Conn. cuts	Time(s)	Gap(%)
G-100-01	12	30	341	22.7	0.0
G-100-02	3	70	80	21.3	0.0
G-100-03	5	100	209	17.4	0.0
G-100-04	10	50	110	16.6	0.0
G-100-05	14	79	168	20.3	0.0
G-100-06	31	200	228	33.1	0.0
G-100-07	34	150	173	19.5	0.0
G-100-08	22	88	149	18.7	0.0
G-100-09	32	112	263	32.3	0.0
G-100-10	62	260	133	14.9	0.0

To make comparison on different size of instances, we respectively generated instances with size 50, 130, 180, and 200, reported Table V. For each size, we generate 5 different instances. All the weight interval are same, which is [1, 500]. As we can see, $\mathcal{M}_4$ outperforms the other two models in both performance metrics used. All models reach the optimality quickly. On 130-nodes instances, all models reach optimality. However, $\mathcal{M}_4$ becomes slower in CPU time than $\mathcal{M}_2$ and $\mathcal{M}_3$. When running on 180-nodes instances, the CPU time for all three models increases dramatically. We keep increasing the size of instances to 200, $\mathcal{M}_4$ reaches the time limit (1,000s) on all five 200-nodes instances.

TABLE II: The computational performance of testing model $\mathcal{M}_1$ on 200-node instances

Instance	ω_{min}	ω_{max}	Conn. cuts	Time(s)	Gap(%)
G-200-01	41	95	542	145.6	0.0
G-200-02	31	80	356	21.2	0.0
G-200-03	22	76	675	57.1	0.0
G-200-04	141	222	345	11.2	0.0
G-200-05	114	245	351	40.3	0.0
G-200-06	45	121	672	893.1	7.9
G-200-07	36	134	324	133.8	0.0
G-200-08	75	186	562	666.1	0.0
G-200-09	72	131	769	1000.5	9.2
G-200-10	12	140	676	1000.8	12.3

TABLE III: The computational performance of models $\mathcal{M}_2$, $\mathcal{M}_3$ and $\mathcal{M}_4$ on 100-node instances

Instance	$\mathcal{M}_2$		$\mathcal{M}_3$		$\mathcal{M}_4$	
	Time(s)	Gap(%)	Time(s)	Gap(%)	Time(s)	Gap(%)
G-100-01	92.3	0.0	62.2	0.0	143.1	0.0
G-100-02	97.1	0.0	81.2	0.0	153.2	0.0
G-100-03	63.2	0.0	35.5	0.0	144.3	0.0
G-100-04	73.7	0.0	66.2	0.0	132.3	0.0
G-100-05	56.7	0.0	25.6	0.0	136.3	0.0
G-100-06	194.3	0.0	51.8	0.0	183.5	0.0
G-100-07	52.3	0.0	23.4	0.0	167.3	0.0
G-100-08	89.2	0.0	38.2	0.0	123.5	0.0
G-100-09	193.6	0.0	25.3	0.0	134.3	0.0
G-100-10	143.3	0.0	93.1	0.0	154.0	0.0

TABLE IV: The computational performance of models $\mathcal{M}_2$, $\mathcal{M}_3$ and $\mathcal{M}_4$ on 150-node instances

Instance	$\mathcal{M}_2$		$\mathcal{M}_3$		$\mathcal{M}_4$	
	Time(s)	Gap(%)	Time(s)	Gap(%)	Time(s)	Gap(%)
G-150-01	132.3	0.0	52.5	0.0	243.1	0.0
G-150-02	195.3	0.0	101.7	0.0	253.2	0.0
G-150-03	163.2	0.0	85.5	0.0	234.3	0.0
G-150-04	173.7	0.0	76.3	0.0	212.6	0.0
G-150-05	156.6	0.0	95.1	0.0	216.6	0.0
G-150-06	153.5	0.0	91.1	0.0	283.7	0.0
G-150-07	252.7	0.0	63.4	0.0	237.1	0.0
G-150-08	155.2	0.0	118.2	0.0	223.5	0.0
G-150-09	160.3	0.0	125.3	0.0	264.2	0.0
G-150-10	203.9	0.0	142.6	0.0	234.0	0.0

V. CONCLUSIONS

In this article, we first proposed a MIP formulation to solve ENDP on random instances of 100-nodes and 200-nodes, respectively. This model can efficiently solve 100-nodes instances with no optimality gap. For larger size instances, our model can reach less than 12.3% gap within 1,000 seconds limitation. To conquer more difficult variants of this problem where the constructed network is a spanning tree, three distinct MIP models are proposed to tackle this problem from perspective of various constraints. Five sets of random instances where the instance size varies from 50 to 200 are used to compare their performance.

TABLE V: The computational performance of models $\mathcal{M}_2$, $\mathcal{M}_3$ and $\mathcal{M}_4$ on instances of various sizes

Instance	$\mathcal{M}_2$		$\mathcal{M}_3$		$\mathcal{M}_4$	
	Time(s)	Gap(%)	Time(s)	Gap(%)	Time(s)	Gap(%)
G-50-01	8.3	0.0	8.2	0.0	3.1	0.0
G-50-02	8.1	0.0	9.2	0.0	3.2	0.0
G-50-03	9.2	0.0	8.5	0.0	4.3	0.0
G-50-04	9.7	0.0	7.2	0.0	2.3	0.0
G-50-05	8.7	0.0	9.6	0.0	6.3	0.0
G-130-06	294.4	0.0	151.8	0.0	283.5	0.0
G-130-07	152.2	0.0	123.4	0.0	267.3	0.0
G-130-08	189.8	0.0	138.7	0.0	232.5	0.0
G-130-09	153.6	0.0	183.3	0.0	264.3	0.0
G-130-10	143.1	0.0	193.1	0.0	234.1	0.0
G-180-06	594.3	0.0	651.8	0.0	883.5	0.0
G-180-07	252.3	0.0	572.4	0.0	718.3	0.0
G-180-08	679.2	0.0	638.7	0.0	833.7	0.0
G-180-09	593.6	0.0	623.3	0.0	661.3	0.0
G-180-10	843.3	0.0	634.5	0.0	936.1	0.0
G-200-06	1000.3	10.0	1000.8	12.6	1000.3	12.5
G-200-07	973.3	0.0	923.4	0.0	1000.3	23.0
G-200-08	799.2	0.0	688.2	0.0	1000.3	8.70
G-200-09	993.6	0.0	927.7	0.0	1000.3	13.1
G-200-10	1000.3	9.0	963.2	0.0	1000.3	10.0

In this paper, optimizing the cost is the goal we have only considered when constructing networks. Observe that it may be not optimization goal (at least the not only one goal) for some other networks, such as Facebook, Twitter and co-author networks, which are constructed or developed for more complex reasons. It is highly possible that multiple objective functions to optimize and much more requirements to keep satisfied. In the future, we will put more effort on these remaining challenging topics.

ACKNOWLEDGMENT

Y. Sun and H. Chen conceived of the presented idea. Y. Sun and Q. Zhao designed the mathematical models, carried out the experiment and wrote the manuscript. Y. Sun and H. Chen took the lead in writing the second revision of the manuscript. The work is supported by Research on Adaptive Learning Model of Artificial Intelligence – Take Intelligent Manufacturing Course Group as an Example (No. JG201926) and Project of Shandong Province Higher Educational Science and Technology Program (No. KJ2018ZBB021).

REFERENCES

[1] S. Nakano, R. Uehara and T. Uno, "Efficient Algorithms for a Simple Network Design Problem," *Networks*, vol. 62, No. 2, pp. 95–104, 2013

[2] M. T. Melo, S. Nickel and F. Saldanha-Da-Gama, "Facility Location and Supply Chain Management–A Review," *European Journal of Operational Research*, vol. 196, no. 2, pp. 401–412, 2009

[3] D. Zhang, "A Network Economic Model for Supply Chain Versus Supply Chain Competition," *Omega*, vol. 34, no. 3, pp. 283–295, 2006

[4] S. Sang, "The Cost Sharing Contract of Greening Level and Pricing Policies in a Green Supply Chain," *IAENG International Journal of Applied Mathematics*, vol. 49, no. 3, pp. 299–306, 2019

[5] D. J. Watts and S. H. Strogatz, "Collective Dynamics of Small-World Networks," *Nature*, vol. 393, no. 6684, pp. 440, 1998

[6] M. Dziubiński and S. Goyal, "Network Design and Defence," *Games and Economic Behavior*, vol. 79, pp. 30–43, 2013

[7] V. Latora and M. Marchiori, "Efficient Behavior of Small-World Networks," *Physical Review Letters*, vol. 87, no. 19, pp. 198701, 2001

[8] D. Kim, Y. H. Kim, C. Park and K. I. Kim, "KREONET-S: Software-Defined Wide Area Network Design and Deployment on KREONE," *IAENG International Journal of Computer Science*, vol. 45, no. 1, pp. 27–33, 2018

[9] W. Du, Y. Li, J. Zhang and J. Yu, "Stochastic Synchronization between Different Networks and Its Application in Bilayer Coupled Public Traffic Network," *IAENG International Journal of Computer Science*, vol. 46, no. 1, pp. 102–108, 2019

[10] S. Li, K. Yuan, Y. Zhang and W. Sun, "A Novel Bandwidth Allocation Scheme for Elastic and Inelastic Services in Peer-to-peer Networks," *IAENG International Journal of Computer Science*, vol. 46, no. 2, pp. 163–169, 2019

[11] W. R Simpson and K. E. Foltz, "Secure Enterprise Mobile Ad-hoc Networks," *IAENG International Journal of Computer Science*, vol. 46, no. 2, pp. 243–256, 2019

[12] C. Gao, D. Wei, Y. Hu, S. Mahadevan and Y. Deng, "A Modified Evidential Methodology of Identifying Influential Nodes in Weighted Networks," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 21, pp. 5490-5500, 2013

[13] R.-W. Hung, F. Keshavarz-Kohjerdi, C.-B. Lin, and J.-S. Chen, "The Hamiltonian Connectivity of Alphabet Supergrid Graphs," *IAENG International Journal of Applied Mathematics*, vol. 49, no. 1, pp. 69–85, 2019

[14] Q. Liu, "On Maximal Incidence Energy of Graphs with Given Connectivity," *IAENG International Journal of Applied Mathematics*, vol. 48, no. 4, pp. 429–433, 2018