7-2012

Parallel Computing Solution for Capacity Expansion Network Flow Optimization Problems

Arun K. Somani
Iowa State University, arun@iastate.edu

Jinxu Ding
Iowa State University, jxding@iastate.edu

Parallel Computing Solution for Capacity Expansion Network Flow Optimization Problems

Abstract

In classical linear network flow (LNF) problems, a network consists of multiple source and sink nodes, where each node is a sink node or a source node, but not both. Usually, there is only one kind of commodity flow and the goal is to find flow schedules and routes such that all sink nodes' flow demands are satisfied and the total flow transmission cost is minimized. We develop a capacity expansion multicommodity network flow (CEMNF) problem, in which the total commodity supply is less than the total commodity demand. There are more than one kind of commodities and each node is a commodity flow generator, as well as a consumer. It is allowed to do expansion for commodity flow generation capacities at each node and also to do expansion for commodity flow capacities of each arc so that more flow can be transmitted among nodes. Thus, CEMNF is not only a commodity flow routing problem, but also a commodity generation and flow planning problem, in which the increasing commodity demands need to be satisfied by generation/transmission capacity expansions. The goal of CEMNF problems is to find the flow routes and capacity expansion plans such that all flow demands are satisfied and the total cost of routing and planning is minimized. High-performance distributed computing algorithms have been designed to solve classical linear network flow (LNF) problems have been proposed. Solving the general CEMNF problems by high-performance distributed computing algorithms is an open research question. The LNF problems can be formulated as linear programming models and algorithms have been proposed to solve them efficiently on distributed computing platforms. But, the constraints of the CEMNF problems do not allow them to solve using the same methodology. In this paper, we also develop a transformation method to transform CEMNF problems into LNF problems in polynomial time and space complexity to solve them efficiently on distributed computing platforms. The results show that we can solve CEMNF problems with high performance.

Keywords

Augmented network flow problems, Classical linear network flow problems, Flow generation/ transmission capacity expansions in network flow problems, High performance distributed computing solution to solve CEMNF problems

Disciplines

Electrical and Computer Engineering

Comments

This article is from *Journal of Computing* 4 (2012): 2151-9617. Posted with permission.

Parallel Computing Solution for Capacity Expansion Network Flow Optimization Problems

Arun Somani and Jinxu Ding

Abstract—In classical linear network flow (LNF) problems, a network consists of multiple source and sink nodes, where each node is a sink node or a source node, but not both. Usually, there is only one kind of commodity flow and the goal is to find flow schedules and routes such that all sink nodes' flow demands are satisfied and the total flow transmission cost is minimized. We develop a capacity expansion multicommodity network flow (CEMNF) problem, in which the total commodity supply is less than the total commodity demand. There are more than one kind of commodities and each node is a commodity flow generator, as well as a consumer. It is allowed to do expansion for commodity flow generation capacities at each node and also to do expansion for commodity flow capacities of each arc so that more flow can be transmitted among nodes. Thus, CEMNF is not only a commodity flow routing problem, but also a commodity generation and flow planning problem, in which the increasing commodity demands need to be satisfied by generation/transmission capacity expansions. The goal of CEMNF problems is to find the flow routes and capacity expansion plans such that all flow demands are satisfied and the total cost of routing and planning is minimized.

High-performance distributed computing algorithms have been designed to solve classical linear network flow (LNF) problems have been proposed. Solving the general CEMNF problems by high-performance distributed computing algorithms is an open research question. The LNF problems can be formulated as linear programming models and algorithms have been proposed to solve them efficiently on distributed computing platforms. But, the constraints of the CEMNF problems do not allow them to solve using the same methodology. In this paper, we also develop a transformation method to transform CEMNF problems into LNF problems in polynomial time and space complexity to solve them efficiently on distributed computing platforms. The results show that we can solve CEMNF problems with high performance.

Index Terms—Augmented network flow problems, Classical linear network flow problems, Flow generation/transmission capacity expansions in network flow problems, High performance distributed computing solution to solve CEMNF problems

1 Multicommodity Network Flow Problem

In classical linear network flow (LNF) problems, a network consists of multiple source and sink nodes, where each node is a sink node or a source node, but not both. Other nodes act as pass- through nodes where the inflow is equal to the outflow. A set of links, each with an associated cost and flow capacity, facilitates flow of the commodities through them. A source node generates a constant commodity flow and a sink node consumes a constant commodity flow. Usually, there is only one kind of commodity flow and the goal is to find flow schedules and routes such that all sink nodes' flow demands are satisfied and the total flow transmission cost is minimized.

The classical linear network flow (**LNF**) problems can be represented as a directed graph ($G(V, E)$). In the graph, two nodes $u, v \in V$ are connected by an arc ($e \in E$). The amount of flow on an arc is upperbounded by the arc's capacity. At each node ($v \in V$), the amount of flow into a node equals the amount of flow out of it and the amount that is consumed by the node. This kind of graph are used to model many real-world problems such as transportation problems, air traffic problems, telecommunication networks and electricity flow through electrical transmission & distribution systems.

When more than one kind of commodity flow can be generated and transmitted in the network, it is referred to as a multicommodity network flow problem. Each node can produce different commodities. For example, in a electrical power generation and distribution network, power may be generated using multiple resources such as fossil or renewable, and each may be considered as a separate commodity. Each node also consumes the power to satisfy its local demand. Each node also can sell power to other nodes to help them meet their local demands. Different commodities may share the same or different transmission systems. The flow on each arc cannot exceed the transmission line capacity. The commodity flows also have to follow the flow conservation at each node. Each node needs to maintain its commodity flow conservation ($inflow - outflow = flow\ demand$).

When total demand in the system is greater than the total supply for a commodity or a group of tradable commodities from the total current generation capacity, we need to plan commodity generation capacity expansions. In effect, we need to expand the classical multicommodity linear Network Flow problems into ca-

• *Both authors are with Department of Electrical and Computer Engineering, Iowa State University, Ames, IA, 50010, USA.*

pacity expansion multicommodity linear Network Flow (**CEMNF**) problems, because we need to consider not only flow routing but also new flow generation and flow transmission capacity expansions.

1.1 A LNF problem formulation

LNF problems considered in this paper are formulated as a directed graph with a set of nodes V and a set of arcs E with a_{ij} being the associated cost of arc (i,j) and f_{ij} being the flow of the arc (i,j). The primal problem for a classical linear network flow (**LNF**) problem [14] can be expressed by Eq. (1) and Figure 1. In Eq. (1), a_{ij}, b_{ij}, c_{ij} and s_i are integers. Constants b_{ij} and c_{ij} are lower and upper bounds of the flow on arc (i,j). $s_i > 0$ is the commodity flow supply of node i and $s_i < 0$ is the commodity flow demand of node i. Equations (1b) and (1c) represent the flow conservation and capacity constraints, respectively. It is assumed that $\sum_{i \in V} s_i = 0$ for problem feasibility. It is also assumed that there is at most one arc in each direction between any pair of nodes. The above **LNF** model is pictorially depicted in Figure 1.

$$\textbf{min} \sum_{(i,j)\in E} a_{ij} f_{ij} \tag{1a}$$

$$\textbf{s.t.}$$

$$\sum_{\{(i,j)\in E\}} f_{ij} - \sum_{\{(j,i)\in E\}} f_{ij} = s_i, \ \forall\ i \in V \tag{1b}$$

$$b_{ij} \le f_{ij} \le c_{ij}, \forall\ (i,j) \in E \tag{1c}$$

The corresponding multicommodity network flow model formulation can be expressed as in Eq. 2 and is pictorially depicted Figure 2. The model of Eq. (2) is different from the model of Eq. 1 only in that it has flow conservation constraints for each distinct kind of commodity flow. However, it assumes that all commodity flows share the arc transmission capacity in Figure 2.

$$\textbf{min} \sum_{p=1}^{M} \sum_{(i,j)\in E} a_{ij} f_{ij}^{p} \tag{2a}$$

$$\textbf{s.t.}$$

$$\sum_{\{(i,j)\in E\}} f_{ij}^{p} - \sum_{\{(j,i)\in E\}} f_{ij}^{p} = s_i^{p} \quad \forall\ p \in P, \forall\ i \in V \tag{2b}$$

$$b_{ij} \le \sum_{p=1}^{M} f_{ij}^{p} \le c_{ij}, \forall\ (i,j) \in E \tag{2c}$$

$P\ is\ the\ set\ of\ all\ commodities$

$M\ is\ the\ size\ of\ P$

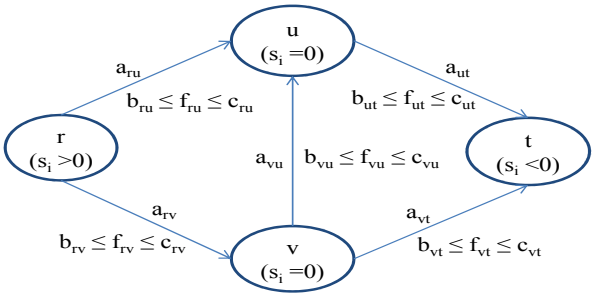

Fig. 1: A Classical Linear Flow Network (1)

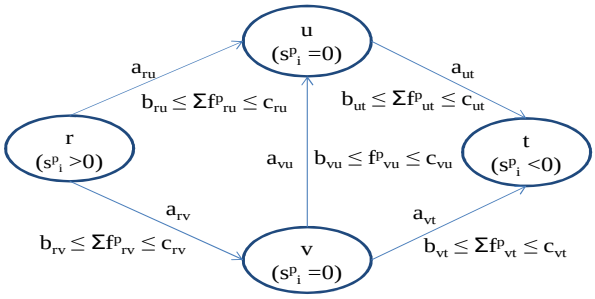

Fig. 2: A Classical Multicommodity Linear Flow Network

1.2 CEMNF problem formulation

CEMNF problems are different from classical **LNF** problems in that they are not only flow scheduling, but also supplemental flow generation planning problems. From the above observation, we obtain a new conceptual formulation for the capacity expansion multicommodity network flow problems as follows:

min $FlowGenCost + FlowTranCost$
such that
$Each\ node\ meets\ its\ local\ demand$
decision variables:
$Commodity\ generated\ by\ expanded\ capacity\ at\ each\ node$
$Commodity\ flow\ on\ each\ arc$
$Commodity\ flow\ transmission\ capacity\ expansion\ on\ each\ arc$

The above conceptual model is a new **capacity expansion multicommodity network flow problems** (**CEMNF**) formulation, which include new decision variables about the flow generation and transmission capacity expansions that are not considered in the classical multicommodity network flow problems. Another difference is that each node in the above model is a flow generator as well as a flow consumer. Nodes, other than "sources" and "sinks," do not consume any commodity flows and maintain flow conservation, i.e. $inflow\ commodity = outflow\ commodity$. The mathematical notations that are used are summarized in Table 1.

With the notations in Table 1, we describe the formulation for the above conceptual model as follows:

#	Name	Definitions
1	$G(V,E)$	network with a node set V and an arc set E
2	u,v	nodes in V, $u,v \in V$
3	uv	an arc from node u to v, uv or $(u,v) \in E$
4	X_{uv}^{p}	flow of commodity p from u to v
5	p	$p \in P$, P is set of commodities, $\|P\| = N$, $p = 1, 2, \cdots, N$
6	X_{v1}^{gp}	commodity p generated by existing capacity at node v
7	X_{v2}^{gp}	commodity p generated by expansion capacity at node v
8	C_{v}^{gp}	commodity p generation capacity at node v
9	ΔC_{v}^{gp}	commodity p generation capacity expansion at node v
10	T_{uv}	capacity of arc $uv \in E$
11	ΔT_{uv}	the arc $uv \in E$ capacity expansion
12	R_{v}^{gp}	Requirement for commodity p at node v in $[0,1]$
13	R_{G}^{gp}	Requirement for commodity p in graph G in $[0,1]$
14	D_v	demand for commodity p at node v
15	Z_p	objective cost function include $(X^p, X^{gp}, \Delta C^{gp}, \Delta T)$
16	$\widetilde{X^p}$	commodity p flow vector in G, $\{X_{uv}^{p} \| uv \in E\}$
17	$\widetilde{X^{gp}}$	commodity p generation vector in G, $\{X_{v}^{gp} \| v \in V\}$
18	$\widetilde{\Delta C^{gp}}$	generation capacity expansion vector for **commodity** $p \in P$ **in** G, $\{\Delta C_{v}^{gp} \| v \in V\}$
19	$\widetilde{\Delta T}$	arc capacity expansion vector in G, $\{\Delta T_{uv} \| uv \in E\}$
20	B_{v}^{gp}	upperbound for commodity p generation expansion at $v \in V$
21	B_{uv}^{T}	upperbound for arc $\{uv \in E\}$ capacity expansion

TABLE 1: Mathematical symbols used in the capacity expansion multicommodity network flow problems

$$\textbf{min } Z_p(\widetilde{X^p}, \widetilde{X^{gp}}, \widetilde{\Delta C^{gp}}, \widetilde{\Delta T}) \tag{3a}$$

$$\textbf{s.t.}$$

$$\sum_{p=1}^{|P|}(X_{v1}^{gp} + X_{v2}^{gp} + \sum_{uv \in E} X_{uv}^{p} - \sum_{vu \in E} X_{vu}^{p}) = D_v$$

$$(\forall\ v \in V, \forall\ p \in P) \tag{3b}$$

$$X_{v1}^{gp} \le C_{v}^{gp}, \forall\ p \in P,\ v \in V \tag{3c}$$

$$X_{v2}^{gp} \le \Delta C_{v}^{gp}\ , \forall\ p \in P,\ v \in V \tag{3d}$$

$$\sum_{p=1}^{N} X_{uv}^{p} \le T_{uv} + \Delta T_{uv}\ , \forall\ uv \in E \tag{3e}$$

$$\textbf{where :}$$

$$0 \le \Delta C_{v}^{gp} \le B_{v}^{gp},\ \forall\ v \in V \tag{3f}$$

$$0 \le \Delta T_{uv} \le B_{uv}^{T},\ \forall\ uv \in E \tag{3g}$$

$$\textbf{decisionvariables :}$$

$$X_{uv}^{p}, X_{v1}^{gp}, X_{v2}^{gp}, \Delta C_{v}^{gp}, \Delta T_{uv} \in R^{+} \tag{3h}$$

The explanations of the constraints of Model in Eq. (3) are summarized in Table 2. Model in Eq. (**3**) is a general optimization model. The model can be used for many real-world problems related to commodity generation and transmission scheduling. For example, long-term energy investment planning problems can be summarized using this model because energy demand of each location in a country or region needs to be satisfied and also the energy generation and transmission are constrained by local generation capacities and related transmission capacities.

An example of usage of model (**3**) for the capacity expansion multicommodity network flow problems is demonstrated in Figure 3. Each arc is associated with a flow cost coefficient

#	Explanation
3a	total cost of all commodity flow, generation, and generation & transmission capacity expansion
3b	commodity flow, generation and consumption conservation at $v \in V$ i.e. commodity inflow - outflow + generation = commodity
3c	Upperbound constraint for commodity p generated by existing generation capacity at node $v \in V$
3d	Upperbound constraint for commodity p generated by expansion capacity at node $v \in V$
3e	Upperbound constraint for commodity flows on arc $uv \in E$
3f	Commodity generation capacity expansion upperbound
3g	Arc $uv \in E$ capacity expansion upperbound
3h	These are decision variables of the model

TABLE 2: The summary of explanation of constraints in model 3

(a_{ij}) from node i to j. The same definition also applies in Figure 1. At each node, the constraint in Eq. (3b), (3c) and (3d) need to be satisfied. Eq. (3b) is specified for commodity demand at the node, Eq. (3d) is for existing generation capacity at the node and Eq. (3d) is for commodity generation capacity expansion at the node. For each arc, a commodity flow cost is charged when the flow goes through the arc. It is the product of the arc-associated cost coefficient a_{ij} and the commodity flow amount (X_{ij}^{p}) on arc $ij \in E$. Each arc is also associated with the constraint Eq. (3e), which is for flow transmission capacity expansion on the arc.

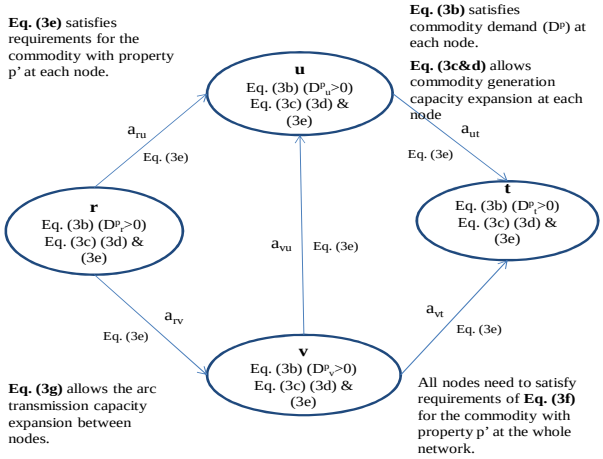

Fig. 3: An example of model (**3**) for the capacity expansion multicommodity network flow problems.

Figure 3 has the same network topology in Figure 1. Both of them have four nodes ($r,t,u,v \in V$) and five arcs ($cu, ut, rv, vt, vu \in E$). In Figure 1, s_i represents the commodity resources at node i. $s_i > 0$ means that the node supplies the commodity; $s_i < 0$ means that the node absorbs (consumes) commodity; $s_i = 0$ means that the node neither generates nor consumes any commodity and it maintain flow conservation. Each arc is associated with a flow cost, e.g. a_{ru} for arc ru. Each arc also has an upper bound (e.g. c_{ru}) and lower bound (e.g. b_{ru}). The flow on each arc is bounded by the upper and lower bounds of the arc e.g. $b_{ru} \le f_{ru} \le c_{ru}$.

In this paper, we propose a solution methodology to solve **CEMNF** problems. The paper is organized as follows. We present related works in the domain of network flow problems in Section 2. In Section 3, we present a comparison between the classical linear network flow (**LNF**) problems and the capacity expansion network flow (**CEMNF**) problems. In Section 3.1, we

compare and analyze the complexity of (**LNF**) and (**CEMNF**) problems. In Section 4, we present a method to transform **CEMNF** problems into classical **LNF** problems. In Section 5, we present a proof about the equivalence of the transformation from (**CEMNF**) formulation into (**LNF**) formulation on the basis of the transformation method proposed in the previous section. In Section 6, we present some computation results depicting both accuracy and time-saving. In Section 8, we present the conclusions.

2 Solving Multicommodity Network Flow Problems

Usually large-scale multicommodity network flow problems, modeled using linear programming, integer programming, or non-linear programming, are large in size, requiring longer run-times to solve. Parallel (or distributed) computing algorithms are often used to compute the solutions faster. Classical **LNF** problems can be solved using a distributed computing algorithms presented in [14] efficiently because the algorithm support decomposition of a linear network by nodes so that each node can be scheduled to a distinct processor which performs the computation associated with that node on a multi-processor system.

The authors of [7] proposed an ϵ-relaxation method to solve the problems. Since the problem has the flow conservation constraint at each node (the total input flow is equal to the total output flow), the constraint of each node is associated with a Lagrange multiplier and put into the objective function. The new problem is the dual of the original primal problem. The dual cost function has a separable form, which can be solved using Gauss-Seidel relaxation (or coordinate ascent) method. At each node, its Lagrange multiplier (also called dual price) is changed in a direction of improvement of the dual cost while keeping the other prices unchanged. Because the dual cost function is separable so that each node's dual problem can be solved in parallel.

The ϵ-relaxation method allows each single node's dual price to change even if these may worsen the dual cost. If the cost deterioration is small, the algorithm can finally approach the optimal solution. An exact solution can be obtained in a finite number of iterations because the arc costs are integer. The key point of the method is that each dual price change improves the dual cost of a perturbed problem, in which some of the arc cost coefficients are modified by a small amount ϵ. The ϵ-relaxation method can be implemented in a message passing system. A separate processor is assigned to each node, which runs relaxation iterations and communicates the results to adjacent processors. The nodes without arcs connected to each other are contained in the same subset. In each iteration of the algorithm, a subset is selected and each node in the same subset runs a relaxation iteration in parallel.

In [1], the authors summarized the message routing problem in data communication networks, and presented a convex multicommodity flow problem. They discussed several solution techniques proposed for solving such large-scale convex optimization problems. They found that the difficulty of solving convex multicommodity flow problems was caused by the coupling constraints (which link variables together) and also by the large size of the real-world problems. This issue can be addressed by decomposition methods. A large problem is decomposed into subproblems that find flows for the individual commodities and coordinate the flows between the commodities to satisfy the joint capacity constraint. The authors reviewed four algorithms that can solve nonlinear convex models with decomposition methods by using their special structures. The first algorithm is the Flow Deviation Method [8], which is used to solve transportation and telecommunication models. The algorithm can easily generate feasible solutions by shortest path calculations and shows fast convergence in early iterations. The second algorithm is Projected Newton Method [7] , which interchanges shortest path calculations with projected Newton steps. It also yields convergence if twice differentiable objective functions are available. The third algorithm is Analytic Center Cutting Plane Method, in which the problem (that needs to be solved) is transformed into a non-differentiable problem with much smaller size. A specialized algorithm (e.g. standard cutting plane method [5]) for non-differentiable optimization is used to solve the transformed problem. The fourth algorithm is Proximal Decomposition Method. It is a specialized version of the partial inverse method designed by [6] for constrained programs. Its good performance mainly depends on the arc-path formulation.

In the data communication network problems studied in [1], each commodity has a single source-sink pair and specified traffic quantity must be sent between the source and sink. This is different from our augmented commodity flow model, in which each node is a commodity source and also a sink. The work of [1] focuses on finding a min-flow-cost path between a pair of source and sink nodes in a network and that satisfies each arc's flow capacity limitations. Moreover, their model does not consider closely complicating constraints, such as percentage requirements for some commodities with some specified properties at node or network levels.

In [9], the authors implemented the basis-partitioning simplex method to solve multicommodity-flow benchmark problems, which are multi-time-period logistics models from NETLIB [10] library. In the basis-partitioning simplex method, the side constraints are dropped and the resulting network problems are solved for each commodity. The full problem is solved by using the network solution as part of an advanced basis using the partitioned basis, in which the working explicit inverse has fewer dimensions, less than or equal to the number of side constraints. Their results show that the extremely large multicommodity flow problems can be solved on workstations. The objective function is to minimize commodity costs. The network constraints require that the commodity supply, demand, and shipping-route requirements must be satisfied. The models also have joint capacity constraints for flow capacity. The network constraints matrix have diagonal block structure, which can be decomposed into submodels for each commodity and solved in parallel. The research work in [9] mainly focus on the problems from logics and the requirements about commodity properties are not considered.

3 Comparison between LNF and CEMNF

The two problems, namely LNF and CEMNF, differ in significant ways as described in Table 3. The different goals of **LNF** and **CEMNF** problems show that **CEMNF** problems need to manage additional issues such as arc capacity expansions and node generation capacity expansions. Moreover, these requirements may have to meet specified properties at node levels and at network level. Other differences include the decision variables. In **CEMNF** problems, we need to consider the expansion capacity of arcs and the generation capacity of nodes in the network. In addition, all flow conservation constraints need to be satisfied.

3.1 Complexity of CEMNF

The complexity of **CEMNF** problems is higher due to the following factors.

1) The commodity generation capacity expansion (Eq. (3d)) makes the node flow conservation more complex than **LNF** problems. This can help satisfy commodity demand of the node and reduce the cost of buying the commodity from other nodes.

Problem Property	LNF Problems	CEMNF Problems
Goal	Find a minimum cost commodity flow in a convex network with convex objective function and constraints such that all nodes flow conservations are met	Find a minimum cost - generation/transmission and commodity capacity expansions - in a convex network with convex objective function and constraints such that (i) commodity demands of all nodes are met; (ii) the commodity generation requirements meet specified properties
Objective Function	Minimize the total cost of commodity transmission flow in the network G(V,E), ref. to (1a)	Minimize the total cost of commodity generation, and expansions in the network G(V,E), ref. to (Eq. 3a)
Decision Variables	the commodity flow on arc (i,j), defined as f_{ij}, in model of Eq. (1)	(i) X_{uv}^{p} - the commodity p flow on arc (uv) (ii) X_{v}^{gp} - the commodity p generation at node v (iii) ΔC_{v}^{gp} - generation capacity expansion at node v (iv) ΔT_{uv} - transmission capacity expansion on arc (uv)
Flow Conservation Constraint	Total inflow - total outflow at node j, (refer to Eq. (1b))	Commodity generation + inflow-outflow = commodity demand at node v, (refer to Eq. (3b))
Commodity Generation Capacity Constraint	None	Commodity generated at node v is upperbounded by existing generation and expansion for p (refer to Eq. (3c), (3d)
Commodity Transmission Capacity Constraint	Flow on arc (i,j) are bounded between b_{ij} and c_{ij}. (refer to Eq. (1c))	Commodity transmitted on arc (uv): upperbounded by existing + expansion capacity (refer to Eq. (3e))
others	$a_{ij}, b_{ij}, c_{ij} \in R$	Coefficients are real numbers

TABLE 3: A comparisons between LNF problems and CEMNF problems

2) The arc capacity expansion (Eq. (3e)) allows more flow transmitted on arcs among nodes. This can help reduce total flow cost if some commodities can be transmitted among nodes with cheaper transmission cost on some arcs.
3) **CEMNF** problems model has four kinds of decision variables (Eq. (3h)); the **LNF** problem model has only one kind of decision variables (Eq. (1b)). This greatly increases the problem dimensions and the difficulties of searching optimal solutions.

The above three points make **CEMNF** problems more complex in terms of run-time for searching optimal solutions because the solution space is much larger than that of **LNF** problems. In the **CEMNF** problem, new generated commodities expand the solution space where we need to search for the optimal solution. The more flow transmissions accommodated by arc capacity expansions also expand the possible solution spaces.

From the model of Eq. 1 of **LNF**, Fig. 1 and Fig. 2, we note that there are only one type of decision variables - flow transmissions on arc, which is referred to as f_{ij}. There is only one type of constraint (Eq. 1b), which keeps flow conservation at each node of **LNF**. According to the ϵ-relaxation algorithm in [14], the equality constraints allow that the model formulation of Eq. 1, which is a constrained optimization model formulation, can be transformed into an unconstrained optimization model formulation by multiplying a Lagrangian multiplier with each constraint equality of Eq. 1b and then add each product to the objective function Eq. 1a. In this way, each term in the new objective function can be decomposed into separate computing tasks, which can be performed by parallel computing framework on multi-computer systems.

In the model of Eq. (3) of **CEMNF**, the constraints of Eq. (3c), (3d) and Eq. (3e) are inequalities and a decision variables not only include flow transmission like the model of Eq. (3), but also include flow generation and transmission capacity expansions variables. Thus the ϵ-relaxation algorithm cannot be applied to **CEMNF** directly. In the next section we propose a method to transform **CEMNF** formulations into **LNF** formulations so that the ϵ-relaxation algorithm can be used to solve **CEMNF** using a parallel computing solution.

4 From CEMNF to LNF

In this section, we propose a transformation methodology, that transforms our **CEMNF** problems into **LNF** problems. We will show the transformation methodology (**Algorithm DecompositionTransform**) step by step in the following sections.

4.1 Algorithmic Formulation

Let $G(V,E)$ be the given network topology with $|V|$ nodes and $|E|$ arcs connecting these $|V|$ nodes in $G(V,E)$, which has $|P|$ commodities (P is the set of commodities with distinct properties) and the vector $\overline{CT}$ whose components are generation/transmission cost parameters for each node and edge in $G(V,E)$. We use the symbols in Table 4, and functions in Table 5, in addition to the symbols defined in Table 1. We also present explanations of the algorithm (**DecompositionTransform**) using graphs shown in Fig. 4, Fig. 5, Fig. 6, and Fig. 7.

Symbols	Definition
$CT_{S_p^v}$	the generation cost for commodity $p \in P$) in node v
$CT_{E_p^v}$	the expansion cost for commodity $p \in P$ in node v
$CT_{L_{vu}^b - T^u}$	the transmission cost of arc $vu \in E$
$CT_{L_{vu}^e - T^u}$	the transmission capacity expansion cost of arc vu

TABLE 4: The definitions of cost coefficients for generation and transmission capacities of Model in Eq. (9)

Syntax Symbol	Function
$G' \longleftarrow \Lambda_v$	Add a subnode in graph where $\Lambda_v \in \{D^v, S_p^v, E_p^v, T^v, L_{vu}^f \mid\}$ $p \in P, v \in V, vu$ is an arc between $v\ and\ u, f = \{b, e\}$ into G' ; Assign values to each subnode as follows: $D^v = D_v, S_p^v = C_{gp}^v$, and $E_p^v = B_v^{gp}$. The right hand side symbols are defined in model of Eq. 3 and Table 1
$((A,B)\|(0,UP,CT))$	Add an arc from subnode $A \in \Lambda_v$ to subnode $B \in \Lambda_v$ with the arc transmission capacity between 0 and UP and transmission cost as CT

TABLE 5: syntax of the algorithmic formulation of the transformation

Algorithm - DecompositionTransform
Input: $G(V, E)$ (e.g. Fig. 4)
Output: $G'(V', E')$ (e.g. Fig. 7)

$G'(V', E') = \emptyset$;

// add subnodes $S, E, D, T, \forall v \in V$ to G'

foreach *node* $v \in V$ **do**
  $V' \longleftarrow V' \cup \{D^v, T^v, S_p^v, E_p^v, p = 1, 2, ..., |P|\}$;
  //Assign flow demand/supply to each new subnode
  $D^v = D_v,\ S_p^v = C_{gp}^v,\ E_p^v = B_v^{gp},\ T^v = 0$; // Add new arcs between S, D, E and T
  $E' \longleftarrow E' \cup \{(S_p^v, D^v)|(0, +\infty, CT_{s_p}^v)$,
  $(E_p^v, D_p^v)|(0, +\infty, CT_{e_p}^v)$,
  $(T^v, D_p^v)|(0, +\infty, 0),\ p = 1, ..., |P|\}$;
end

// for each arc of G, add node L and arcs between
// L and T, S or D and L to G'

foreach *arc* $vu \in E$ *between node* v *and* u *with* $v \neq u$ **do**
  $V' \longleftarrow V' \cup \{(L_{vu}^b, L_{vu}^e)\}$; //Assign flow
  //demand/supply to each new subnode L
  $L_{vu}^b = 0, L_{vu}^e = 0$;
  $E' \longleftarrow E' \cup \{(L_{vu}^b, T^u)|(0, +\infty, 0)$,
  $(L_{vu}^e, T^u)|(0, +\infty, 0)\}$;
  $E' \longleftarrow E' \cup \{(S_p^v, L_{vu}^b)|(0, +\infty, CT_p^s)$,
  $(E_p^v, L_{vu}^b)|(0, +\infty, CT_{e_p}^v),\ p = 1, ..., |P|\}$;
  $E' \longleftarrow E' \cup \{(T^v, L_{vu}^b)|(0, +\infty, 0)$,
  $(T^v, L_{vu}^e)|(0, +\infty, 0)\}$;
end

4.2 Augmented single-commodity network flow problems decomposed into subnodes for ϵ-relaxation algorithm

Figure 4 shows an example of four connected nodes in **LNF** problems. Figure 5 shows the decomposition of the nodes and arcs in Figure 4 of **CEMNF** into subnodes. In Figure 5, each node is decomposed into subnodes. Each arc between two nodes in Figure 4 is decomposed into two subnodes and multiple arcs. The subnode L_{ij}^b represent the existing transmission capacity between node i and j. When the arc between subnode L_{xv}^b and T^v is saturated, the arc between subnode L_{xv}^e and T^v (transmission capacity expansion arc) is used. The arc cost of the latter is more expensive than the former since the transmission capacity expansion is usually more expansion than using the existing capacity. Figure 6 shows that each node in the **CEMNF** problem can also be decomposed into subnodes when there are multiple (P) commodities with existing and expansion capacities.

4.3 Augmented network flow problems reduction

Table 6 presents the symbols used in the section for analysis of **CEMNF** problems.

In Table 6, Figures 5, 6 and Fig. 7, S_i^v represents the existing flow generation variable flow i; E_i^v represents the flow generation capacity expansion variable of flow i; D is the demand that needs to be satisfied by the sum of flow i from S_i^v, E_i^v and flow transmitted from other nodes. However, we do not know the actual values of S_i^v and E_i^v, which are variables. This is a barrier in reduction from our augmented network flow problems (**CEMNF**) to classical network flow problems (**LNF**). In **CEMNF** problems, since we set S_i^v to be the maximal available generation capacity ($\overline{S_i^v}$). However, flows from other nodes' S_j^v or E_j^v can be transmitted to node i to satisfy part of node i's demand with the lower generation and transmission costs are the cost of S_i^v. Therefore, we need to create provision in the model to not expand S_i^v from its existing value if not desired for optimization.

name	meaning, all symbols are for each node v
S_i^v	the existing generation capacity of commodity i
E_i^v	the generation expansion capacity of commodity i
$\overline{S_i^v}$	the maximum existing generation capacity of commodity i
$\overline{E_i^v}$	the maximum generation expansion capacity of commodity i
DU^v	the local pseudo demand subnode
DUG	the global pseudo demand node in the network $G(V, E)$

TABLE 6: The symbols used for analysis of **CEMNF** problems

The value of E_i^v is determined by the maximum available generation capacity ($\overline{E_i^v}$) and the total demand of the network as follows.

$$E_i^v = min\{\overline{E_i^v}, \sum_{v \in V} D^v\} \tag{4}$$

By setting the values of S_i^v and E_i^v to the maximum possible generation capacity, we create excess capacity at many nodes in the network. We already have paths to allow flow transmission among nodes such that the demands of the "hungry" nodes, i.e. nodes for which $\sum_{i=1}^{|P|} \overline{S_i^v} + \sum_{i=1}^{|P|} \overline{E_i^v} < D^v$, can be satisfied with the flows from its neighbor nodes. After meeting the demands of the hungry nodes, the system may still have excess capacity. If it is not the case, then there is no feasible solutions as the total demand exceeds the total generation capacity. We assume that this is not the case and the system has a feasible solution. Therefore, in order to maintain flow conservation in the whole network, we create a global pseudo demand node (DUG). From each node v, we create a path to DUG using a flow-bypass subnode DU^v through which the "extra" flows from E_i^v and S_i^v can be absorbed by DUG. In Fig. 7 demonstrate the node DUG and the absorption paths to it. As shown, the global pseudo demand node (DUG) is connected to DU^v and it has no output arcs. The DU^v is a flow-bypass node and it does not have its own flow demand and supply. Its total flow input is equal to its total flow output. The value of DUG is determined as given below in Eq. 5.

$$DUG = \sum_{v \in V} (\sum_{p=1}^{|P|} (\overline{S_i^v} + \overline{E_i^v})) - \sum_{v \in V} D^v \tag{5}$$

We set the cost of the arcs between DU^vs and DUG as 0 and capacity as $+\infty$. The cost of the arcs between E_p^v and DU^v also is 0 and transmission capacity also is $+\infty$. The above set-up for the arc costs and capacities make it possible that the flows from redundant expanded capacities to be absorbed by DUG without any impacts on the total generation and transmission costs in the whole network. In this way, we guarantee that the global flow balance in the whole network in maintained. Our transformation allows any node that can economically generate a flow and transmit it to a needed node to do so. Also, notice that any flow from any nodes to DUG is in fact need not be generated. It is only a modeling convenience to allow flexibility in determining the most suitable generation points.

The LNF problem obtained this way is solved using an appropriate method. The solution will consist of all the flow

on different arcs. Since any flow to DUG is not to be even generated, the actual generation and expansion capacity S_i^v and E_i^v at each node $v \in V$ for each commodity $i \in P$ can be modified as follows.

$$S_i^v = \overline{S_i^v} - (flow\ from\ S_i^v\ to\ DU^v) \tag{6a}$$
$$E_i^v = \overline{E_i^v} - (flow\ from\ E_i^v\ to\ DU^v) \tag{6b}$$

In the following section, we present formal analysis and mathematical proof for **DecompositionTransform** (Section 4.1) by setting the equivalence between the two formulations and make some observations on the results.

5 The transformation from CEMNF problems to LNF problems

In Section 3.1, we have shown a method for transforming a node and an arc in **CEMNF** network into multiple subnodes such that the flow conservation can be maintained at each node. This makes it possible to transform **CEMNF** into **LNF** problems, which can be solved using the algorithm in [14] in parallel computing environment. To be able to compare **CEMNF** and **LNF** models side by side, we reproduce Model in Eq. (3) and Model in Eq. 1 below for easy reading.

$$\mathbf{min}\ Z_p(\widetilde{X^p}, \widetilde{X^{gp}}, \widetilde{\Delta C^{gp}}, \widetilde{\Delta T}) \tag{7a}$$
$$\mathbf{s.t.}$$
$$\sum_{p=1}^{|P|}(X_{v1}^{gp} + X_{v2}^{gp} + \sum_{uv\in E} X_{uv}^p - \sum_{vu\in E} X_{vu}^p) = D_v \quad \forall u, v \in V, uv, vu \in E \tag{7b}$$
$$X_{v1}^{gp} \le C_v^{gp},\ \forall\ p \in P,\ \forall v \in V \tag{7c}$$
$$X_{v2}^{gp} \le \Delta C_v^{gp}\ ,\forall\ p \in P,\ \forall v \in V \tag{7d}$$
$$\sum_{p=1}^{|P|} X_{uv}^p \le T_{uv} + \Delta T_{uv}\ ,\forall\ uv \in E \tag{7e}$$
$$\mathbf{where}:$$
$$0 \le \Delta C_v^{gp} \le B_v^{gp},\ \forall\ v \in V \tag{7f}$$
$$0 \le \Delta T_{uv} \le B_{uv}^T,\ \forall\ v \in V \tag{7g}$$
$$\mathbf{decision\ variables}:$$
$$X_{uv}^p, X_{v1}^{gp}, X_{v2}^{gp}, \Delta C_v^{gp}, \Delta T_{uv} \in R^+ \tag{7h}$$

The **LNF** model formulation :

$$\mathbf{min} \sum_{(i,j)\in E} a_{ij} f_{ij} \tag{8a}$$
$$\mathbf{s.t.}$$
$$\sum_{\{(i,j)\in E\}} f_{ij} - \sum_{\{(j,i)\in E\}} f_{ij} = s_i,\ \forall\ i \in V \tag{8b}$$
$$b_{ij} \le f_{ij} \le c_{ij}, \forall\ (i,j) \in E \tag{8c}$$

Given a **CEMNF** problem in Eq. 7, we transform it into an **LNF** model of Eq. (8). In model of Eq. (7), there are four decision variables of Model in Eq. (8), there is one decision variable. Constraints of Eq. (7c) and Eq. (7e) specify the flow generation and transmission capacity expansions. We transform them to node flow conservation constraints like in Eq. (8b). After transformation, the decision variable in model of Eq. 7 are reduced to only one type - the flow on arc type, which is the only type of decision variable in model of Eq. 8.

Symbol	Meaning
$S_p^v - D^v$	an arc between subnode S_p^v to subnode D^v in node $v \in V$.
$E_p^v - D^v$	an arc between subnode E_p^v to subnode D^v in node $v \in V$.
DUG	the global pseudo demand node, absorbs extra flow
DU^v	a flow-bypass subnode, total inflow = total outflow
$X_{S_p^v - L_{vu}^b}$	the flow on the arc from subnode S_p^v to subnode L_{vu}^b
$X_{S_p^v - L_{vu}^e}$	the flow on the arc from subnode S_p^v to subnode L_{vu}^e
$X_{E_p^v - L_{vu}^b}$	the flow on the arc from subnode E_p^v to subnode L_{vu}^b
$X_{E_p^v - L_{vu}^e}$	the flow on the arc from subnode E_p^v to subnode L_{vu}^e
$X_{T^v - L_{uv}^b}$	the flow on the arc from subnode T^v to subnode L_{uv}^b
$X_{T^v - L_{uv}^e}$	the flow on the arc from subnode T^v to subnode L_{uv}^e
$X_{T^v - D^v}$	the flow from subnode T^v to subnode D^v in node v

TABLE 7: The sym bols used in node transformation

5.1 Flow Conservation at a node

The transformed CEMNF model.

$$\mathbf{min}\ Z(\widetilde{X}) \tag{9a}$$
$$\mathbf{s.t.}$$
$$\sum_{p=1}^{|P|}(X_{S_p^v - D^v} + X_{E_p^v - D^v} + X_{T^v - D^v}) = D_v, \forall v \in V \tag{9b}$$
$$\sum_{p=1}^{|P|} X_{S_p^v - DU^v} + \sum_{p=1}^{|P|} X_{E_p^v - DU^v} = X_{DU^v - DUG}, \quad \forall v \in V \tag{9c}$$
$$C_v^{gp} = X_{S_p^v - D^v} + X_{S_p^v - DU^v} + \sum_{vu\in E}(X_{S_p^v - L_{vu}^b} + X_{S_p^v - L_{vu}^e}) \tag{9d}$$
$$B_v^{gp} = X_{E_p^v - D^v} + X_{E_p^v - DU^v} + \sum_{vu\in E}(X_{E_p^v - L_{vu}^b} + X_{E_p^v - L_{vu}^e}) \tag{9e}$$
$$\sum_{uv\in E}(X_{L_{uv}^b - T^v} + X_{L_{uv}^e - T^v}) = X_{T^v - D^v} + \sum_{vk\in E}(X_{T^v - L_{vk}^b} + X_{T^v - L_{vk}^e}) \tag{9f}$$
$$\sum_{p=1}^{|P|}(X_{S_p^v - L_{vk}^b} + X_{E_p^v - L_{vk}^b} + X_{T^v - L_{vk}^b}) = X_{L_{vk}^b} \tag{9g}$$
$$\sum_{p=1}^{|P|}(X_{S_p^v - L_{vk}^e} + X_{E_p^v - L_{vk}^e} + X_{T^v - L_{vk}^e}) = X_{L_{vk}^e} \tag{9h}$$
$$\sum_{v=1}^{|V|} X_{DU^v - DUG} = DUG \tag{9i}$$
$$\mathbf{decision\ variables}:$$
$$\widetilde{X} = \{X_{S_p^v - D^v}, X_{E_p^v - D^v}, X_{S_p^v - DU^v}, X_{E_p^v - DU^v}, X_{S_p^v - L_{vu}^b}, X_{S_p^v - L_{vu}^e}, X_{E_p^v - L_{vu}^b}, X_{S_p^v - L_{vu}^e}, X_{T^v - D^v}, X_{T^v - L_{vu}^b}, X_{T^v - L_{vu}^e}, X_{L_{uv}^e - T^v}, X_{L_{uv}^b - T^v}, X_{DU^v - DUG} \mid \forall\ p \in P,\ \forall\ u, v, k \in V,\ \forall\ uv, vu, vk \in E\} \tag{9j}$$
$$X_{L_{vk}^b - T^k} \le T_{vk}, X_{L_{vk}^e - T^k} \le B_{vk}^T \tag{9k}$$
$\mathbf{Eqs}.\ (9\mathrm{d}, 9\mathrm{e}, 9\mathrm{f}, 9\mathrm{g}, 9\mathrm{h})$ hold $\forall v \in V$

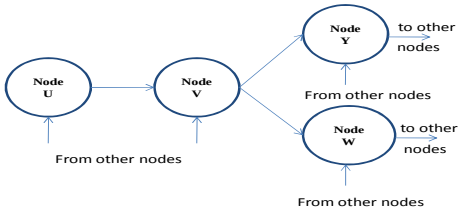

Fig. 4: An example of four connected nodes in **CEMNF**

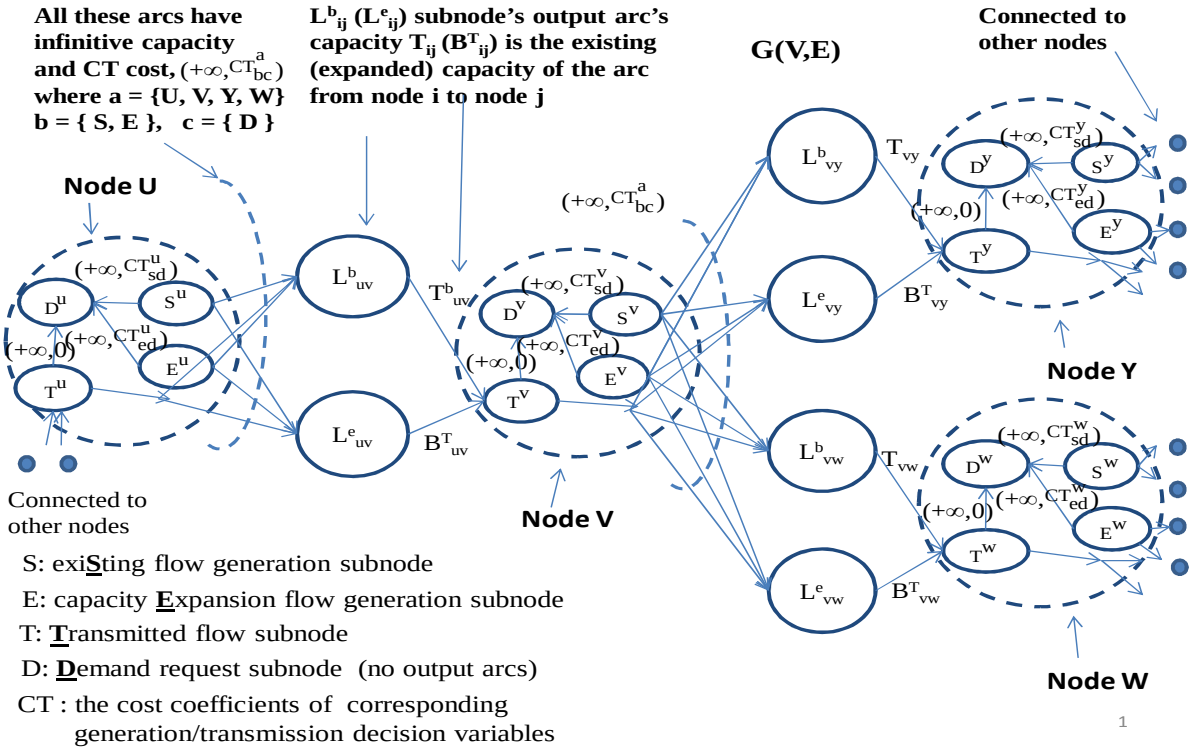

Fig. 5: The nodes in single-commodity **CEMNF** decomposed into subnodes

In this section, we propose the model in Eq. (9) that describe LNF problems transfermed from CEMNF problems. We use node v in Figure 7 as an example to show the flow conservation equations for each subnode. The symbols used in the this section and following sections are presented in Table 7. We notice that we need to maintain the following flow conservations.

(1) For subnode D, the sum of all input flows to subnode D^v is equal to the local demand of node v, i.e. Eq. (9b).

(2) For subnode DU, the sum of all input flows from existing and expanded flow generation capacity is equal to the pseudo demand, i.e. Eq. (9c).

(3) For subnode $S_p, p \in P$, the sum of all output flows is equal to the existing generation capacity of flows $p \in P$, i.e Eq. (9d).

(4) For subnode $E_p, p \in P$, the sum of all output flows is equal to its flow to subnodes D and DU and transmission subnodes L^b and L^e, which connect node v to its output nodes, i.e. Eq. (9e).

(5) For subnode T, the sum of all input flows from others nodes is equal to the flow going to its local demand node D and the sum of its output flows to other nodes, i.e. Eq. (9f).

(6) For node L^b_{ij}, the sum of all its input flows is equal to its output flow, i.e. Eq. (9g).

(7) For node L^e_{ij}, the sum of all its input flows is equal to its output flow, i.e. Eq. (9h).

These constraints lead to Model in Eq. (9), where the constraints are held for all $v \in V$.

The constraints of Model in Eq. (9) are systematically equivalent to the constraints of Model in Eq. (7). The visualization of the transformation is expressed in Figures 4 and 7. Next, we show that the two models are equivalent.

Equivalence 1: *The constraints of Eq. (7b), Eq. (7c) and Eq. (7d) in the model of Eq. (7) are equivalently transformed into the constraints of Eq. (9b), Eq. (9c), Eq. (9d), Eq. (9e) and Eq. (9f).*

Proof:

In the constraint of Eq. (7b), $\sum_{p=1}^{|P|}(\sum_{uv \in E} X^p_{uv} - \sum_{vu \in E} X^p_{vu})$ is the net inflow to node v, which is equivalent to $X_{T^v - D^v}$ in constraint Eq. (9b). It is the difference between the inflow from all nodes connected to node v and the outflow to all nodes connected to node v.

X^{gp}_v is the flow generated by the existing generation capacity and expanded capacity. This is expressed by the constraint Eq.

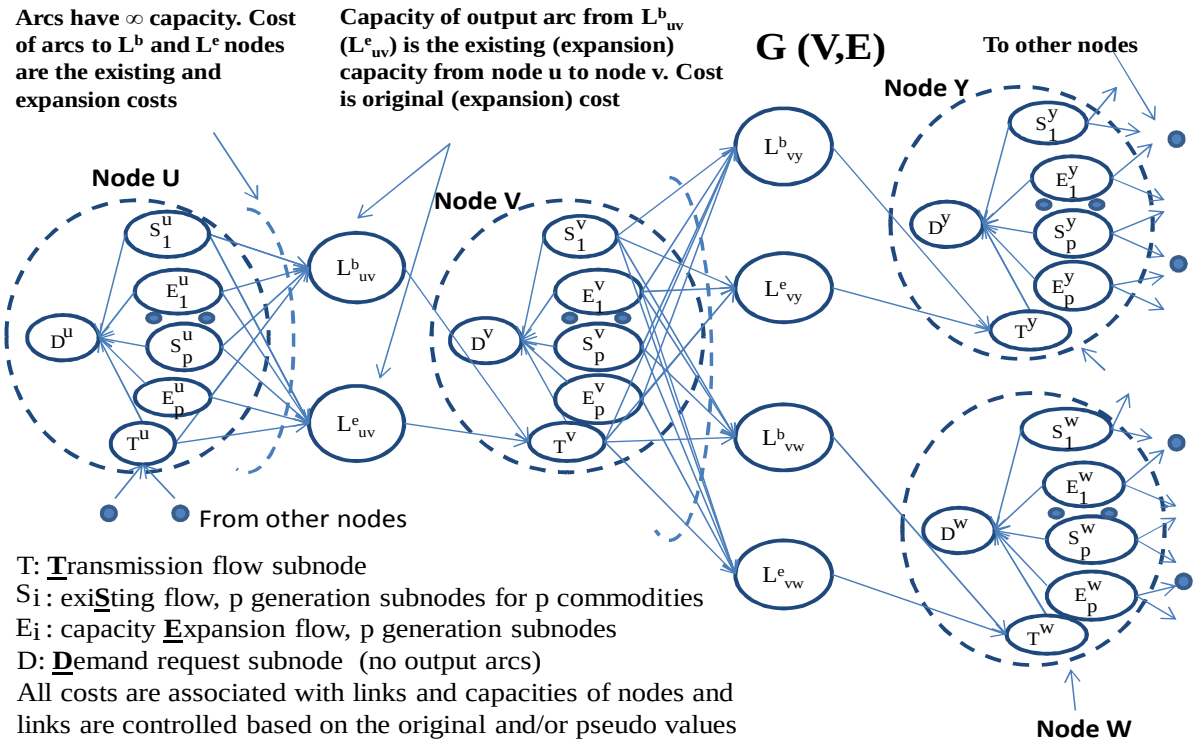

Fig. 6: The nodes in multi-commodity **CEMNF** decomposed into subnodes

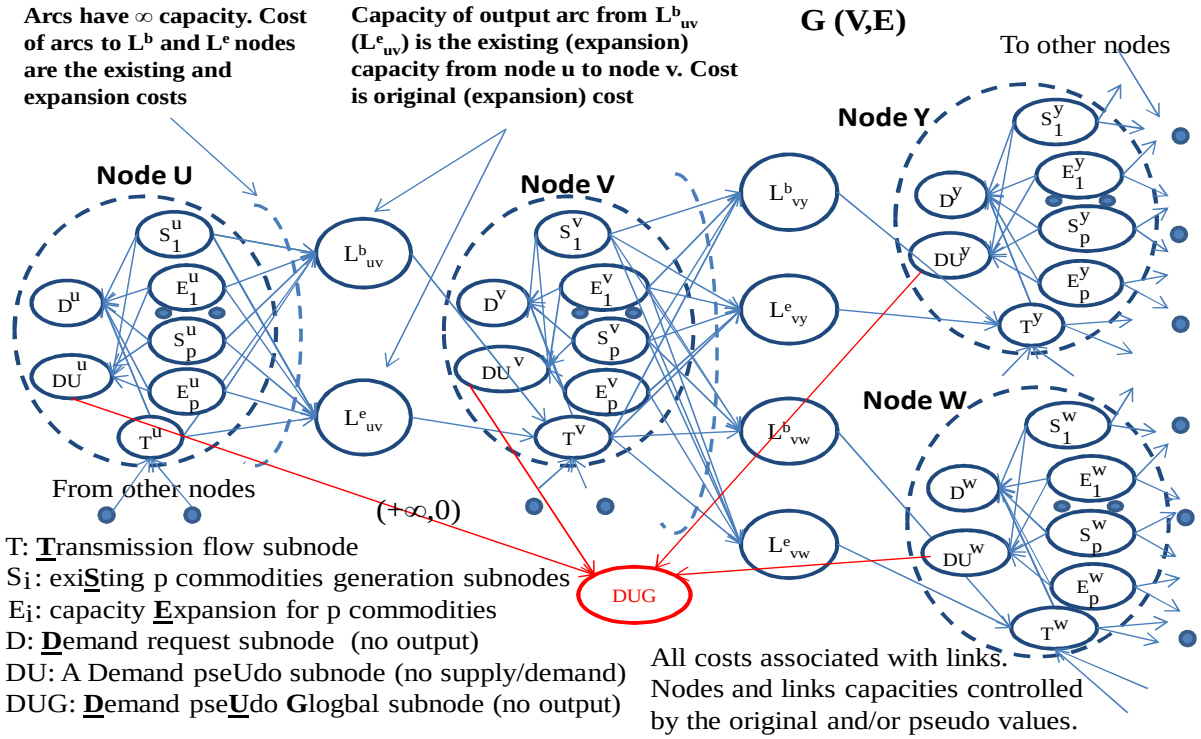

Fig. 7: The nodes in **CEMNF** decomposed into subnodes with global pseudo demand node added

(7c) and Eq. (7d). Here, X_{v1}^{gp} is generated by C_v^{gp} and X_{v2}^{gp} is generated by ΔC_v^{gp}. The first is generated by the existing capacity in node v, which is $X_{S_p^v-D^v}$ in Eq. (9b). The second is generated by the expanded capacity in node v, which is $X_{E_p^v-D^v}$ in Eq. (9b).

In order to transform Eq. (7c) into an equality, we add a slack variable such that :

$$X_{v1}^{gp} + Slack\_generation = C_v^{gp} \forall\ p \in P,\ v \in V \qquad (10)$$

The $Slack\_generation$ can be decomposed into three parts: (1) the flow going to L_{vu}^b; (2) the flow going to L_{vu}^e; and (3) the flow going to DU^v. The first part is the flow that is transmitted to another node u through the existing transmission capacity of the arc vu. The second part is the flow that is transmitted to another node u through the expanded transmission capacity of the arc vu. The third part is the flow that cannot be consumed by the local demand D^v and the other nodes. This "extra" flow needs to be consumed by the local pseudo node DU^v.

We can do the same analysis for X_{v2}^{gp}. We add a slack variable to Eq. (7d) and also replace ΔC_v^{gp} with B_v^{gp} such that:

$$X_{v2}^{gp} + Slack\_generation = B_v^{gp} \forall\ p \in P,\ v \in V \qquad (11)$$

The $Slack\_generation$ can also be decomposed into three parts: (1) the flow going to L_{vu}^b; (2) the flow going to L_{vu}^e; and (3) the flow going to DU^v. The first part is the flow that is transmitted to another node u through the existing transmission capacity of the arc vu to node u. The second part is the flow that is transmitted to another node u through the expanded transmission capacity of the arc vu to node u. The third part is the flow that cannot be consumed by the local demand D^v and its connected neighboring nodes. This "extra" flow needs to be consumed by the local pseudo node DU^v.

The sum of flow from S_p^v to DU^v and from E_p^v to DU^v are absorbed by DUG, which is responsible for absorbing all these kinds of "extra" flows from all nodes $v \in V$. Thus, we have:

$$\sum_{p=1}^{|P|} X_{S_p^v-DU^v} + \sum_{p=1}^{|P|} X_{E_p^v-DU^v} = X_{DU^v-DUG} \qquad \forall\ p \in P,\ v \in V$$

The above analysis shows that Eq. (7b) can be transformed into Eq. (9b); Eq. (7c) can be transformed into Eq. (9d) and Eq. (7e) can be transformed into Eq. (9e). They are summarized in Table 8.

□

Next, we prove the transformation for flow transmission part by proposing **Equivalence 2**.

Equivalence 2: *The constraints of Eq. (7e) in model of Eq. (7) is equivalent to the constraints of Eq. (9f), Eq. (9g), and Eq. (9h) of Model in Eq. (9).*

Proof: The left-hand side term of Eq. (7e) is the flow on the arcs between any connected node u and v in Figure 4. It can be decomposed into two parts. Part 1 is the flow through the existing transmission capacity between any connected two node pairs. Part 2 is the flow through the expended transmission capacity between the connected node pairs. In Figure 7, one arc connecting node u and v is decomposed into two arcs. One arc (its capacity upperbounded by T_{uv}^b) goes to node T^v and another arc (its capacity upperbounded by B_{uv}^T) goes to node T^v.

Because we add the subnodes S_p^v, E_p^v and T_d^v inside a node v in Figure 4, the output arcs from these nodes go to the transmission nodes L_{vx}^b and L_{vx}^e for any connected node pairs v and x in Figure 4. Thus, the left-hand side term of Eq. (7e) can be decompose into three parts in Figure 7. Part 1 is the flow from the subnode S_p^v; Part 2 is the flow from subnode E_p^v; and Part 3 is the flow from subnode T^v. The sum of the three parts (the left-hand side term of Eq. (9g)) is equal to the output flow $X_{L_{vk}^b-T^k}$ (the right-hand side term of Eq. (9g)). The sum of the three parts of the flow can also go through the expanded transmission capacity. Similarly, we have Eq. (9h)). Thus, Eq. (7e) is equivalent to Eq. (9g) and Eq. (9h).

For new subnode T^v in Figure 7, we have Eq. (9f) to keep the flow conservation hold at the transmission subnode. The left hand side of Eq. (9f) is the sum of input flows from node v's input nodes. The first term at the right-hand side $X_{T^v-D^v}$ is the flow from T^v to D^v. The second term is the flow from T^v to L_{vx}^b and the third term is the flow from T^v to L_{vx}^e, where $x = y, w$ in Figure 7. Their transformations are summarized in Table 8 in the last two multi-rows. □

Thus the constraints in Model in Eq. (7) are equivalent to Model in Eq. (9).

Transformation	Related variables	Function
(7b) → (9b)	$X_{S_p^v-D^v}$ $X_{E_p^v-D^v}$ $X_{T^v-D^v}$	The flows from existing generation capacity, expanded capacity and transmissions from other nodes are used to satisfy local demand D^v.
(7b) → (9c)	$X_{E_p^v-DU^v}$ $X_{S_p^v-DU^v}$	Part of flow from existing and expanded generation capacity go to pseudo demand DU^v subnode. This expansion need not be done.
(7b) → (9d) (7c) → (9d) (7d) → (9d)	$X_{S_p^v-D^v}$ $X_{S_p^v-L_{vu}^b}$ $X_{S_p^v-L_{vu}^e}$ $X_{S_p^v-DU^v}$	This is decomposition of the existing generation capacity. Some of the flow from generation capacity go to satisfy local demand D^v; remaining go to the other nodes through transmission nodes L_{vu}^b or L_{vu}^e. The unused capacity goes to DUG through DU^v.
(7b) → (9e) (7c) → (9e)	$X_{E_p^v-D^v}$ $X_{E_p^v-L_{vu}^b}$ $X_{E_p^v-L_{vu}^e}$ $X_{E_p^v-DU^v}$	This is decomposition of the expanded generation capacity. Some of the flow from expanded capacity go to satisfy local demand D^v; remaining go to the other nodes through transmission nodes L_{vu}^b or L_{vu}^e. The unused capacity goes to DUG through DU^v.
(7e) → (9f) (9g) (9h)	$X_{L_{uv}^b-T^v}$ $X_{L_{uv}^e-T^v}$ $X_{T^v-L_{vk}^b}$ $X_{T^v-L_{vk}^e}$ $X_{S_p^v-L_{vk}^b}$ $X_{E_p^v-L_{vk}^b}$ $X_{S_p^v-L_{vk}^e}$ $X_{E_p^v-L_{vk}^e}$	This is about flow transmissions conservation between connected nodes. Transmission nodes T^v receives input flows from each node u through arc nodes L_{uv}^b and L_{uv}^e. Similarly, from node v, flows are sent to each connected node k using arc nodes L_{vk}^b and L_{vk}^e from subnodes T^v, S_p^v, and E_p^v.

TABLE 8: The summary of transformation from Eqs. (7b), (7c) and (7e) to Eqs. (9b), (9c), (9d), (9e), (9f), (9g) and (9h)

5.2 Flow demand and supply balance

In Figure 7, we add a global pseudo demand (**DUG**) node such that the "extra" flow can be absorbed by it and the flow conservation is maintained in the whole network. In this section, we show that how adding the **DUG** node with the value determined by Eq. (5) can keep the flow conservation in the whole network.

Lemma 1: *In the multi-commodity flow network decomposition, adding a global pseudo demand (***DUG***) node allows the flow conservation in the network of Model in Eq. (7) if it is feasible.*

Proof: Assume that Model in Eq. (7) is feasible with possible generation and expansion generation capabilities. Since

we set the values of all existing and expansion generation capacities to their maximum, it is clear that we have excess generation capacity in the network in Eq. (7). The role of node DUG is to absorb only the excess capacity. The total generation capacity in the network using the maximum of existing and possible expanded capacities is $\sum_{v\in V}(\sum_{p=1}^{|P|}(\overline{S_i^v}+\overline{E_i^v}))$. The total demand of the network is $(\sum_{v\in V} D^v)$. Thus the excess capacity in the network is given by the following expression.

$$\mathrm{EC} = \sum_{v\in V}(\sum_{p=1}^{|P|}(\overline{S_i^v}+\overline{E_i^v})) - \sum_{v\in V} D^v \tag{12}$$

Since we set the capacity of node DUG to be exactly equal to EC, and the capacity can flow from all existing and expansion generation nodes to node DUG with zero cost, DUG will be able to absorb exactly the set capacity and no more or no less. The remaining generation and expansion capacity is used to meet the demands of the other nodes exactly. The nodes' demand will be governed by the generation and transmission cost. Since none of these costs are changed, the demands are still met in exactly the same way as without node DUG. Hence the network maintain flow conservation. □

We present **Lemma 2** and **Lemma 3** to show that a feasible solution in Model in Eq. (7) is also a feasible solution in Model in Eq. (9) and vice versa.

Lemma 2: *A feasible solution in Model in Eq. (7) is also a feasible solution in Model in Eq. (9).*

Proof: Given a feasible solution $(\overline{X})$ of Model in Eq. (7), suppose that a node a's demand is satisfied by a node b in Model of eq. (7). The corresponding flow from a to b go through some intermediate nodes and transmission arcs. Without loss of generality, suppose there is one intermediate node c and therefore two intermediate arcs, yc and cx (The proof will be similar if there are more than one intermediate nodes or none between any nodes of a and b). Suppose that this flow is generated by a group of subnodes in node y. This flow solution of yc and cx can be superimposed in Model of Eq. (9) as is without violating any constraints. This can be guaranteed by **Equivalence 1 & 2**. Actually, we can take the whole solution of Model in Eq. (7) and impose it as a solution of Model in Eq. (9). We will not violate any constraints as the generation capacities of all nodes and transmission capacities of all arcs in Model in Eq. (9) are either higher than or equal to those of Model of Eq. (7). The costs and transmission capacity upperbounds of all arcs in Model in Eq. (9) are the same as in Model of Eq. (7). Thus, the solution of Model in Eq. (7) is a feasible solution of Model in Eq. (9). □

Lemma 3: *A feasible solution of Model in Eq. (9) is also a feasible solution in Model in Eq. (7).*

Proof: Following the similar approach as in the proof of **Lemma 2**, we can map the solution of Model in Eq. (9) to Eq. (7). Given a feasible solution $(\overline{X'})$ of Model in Eq. (9), the flow solution of arc bc and ca (here the arcs between DUG and S_p^v, E_p^v are not a part of $\overline{X'}$ because DUG's demand is pseudo and only used to absorb "extra" flows and maintaining flow conservation in the whole network. Moreover, all arcs connected to DUG have 0 cost, which means that any flows on these arcs do not have impacts on the objective function values.) can be superimposed into Model in Eq. (7) because $(\overline{X'})$ does not violate any constraints that of Model in Eq. (7) for the same reason as in the proof of **Lemma 1**. □

5.3 Proof of the Equivalence of Models in Eq. (7) and Eq. (9)

We have already shown the constraints in the two models have one-to-one correspondence. We also showed that the including node DUG with a specified capacity and setting up the maximum existing and expansion generation node capacities to their maximum maintain flow conservation in the network. Any feasible solutions of Models in Eq. (7) and (9) can be mapped as solutions of of each other.

Next, we show that the two models produce solutions that have the same values of the optimal objective functions in **Lemma 4** that follows.

Lemma 4: *The model in Eq. (7) and Model in Eq. (9) have the same optimal objective function value.*

Proof:

To prove this result, we need to show that the flow distributions in Model in Eq. (7) are the same (or of the similar equivalent costs) to Model in Eq. (9). Model in Eq. (7) does not have DUG and DU^vs to absorb the "extra" flow, and allows the values of subnode E_p^v and S_p^v (in Figure 7) to be any values as long as they are not more than the upperbound values. This idea is expressed by Eq. (7b), Eq. (7c) and Eq. (7d) in Model (7). In Eq. (7b), for the flow $p = 1$, X_{v1}^{g1} is generated by v's existing flow 1 generation capacity C_v^{g1} and X_{v2}^{g1} is generated by flow generation capacity expansion ΔC_v^{g1} with the upperbound as B_v^{g1}.

The generated flow 1 from the both sources is used to satisfy v's local demand D_v or transmitted to neighbor nodes connected to v, as expressed in Eq. (7c) and Eq. (7d) in Model in Eq. (7). The same argument also hold for X_{v1}^{gp} and X_{v2}^{gp}.

The **Equivalence 1 & 2** have shown that both models have the same values of flow demand and generation and transmission costs and their expansion costs. In Model in Eq. (9), the arc flow cost between S_p^v, E_p^v and DU^v and the arc flow cost from DU^v to DUG are all 0s. It means that the "extra" flows in the transformed model have no impacts on the objective function value.

The above analysis hold for each $v \in V$ in Model in Eq. (9) and Model in Eq. (7). The transformation procedures only change the flow distributions among subnodes (of Model in Eq. (9)) but do not change the total flow generation needed by the total demand in both models. In both models, the flows go to cheapest unsaturated arcs at first in order to minimize the objective function values. **Lemma 2 & 3** have shown that one model's feasible solution can be mapped on the other model. Therefore, we obtain that

$$Z_p(\widetilde{X^{p*}}, \widetilde{X^{gp*}}, \widetilde{\Delta C^{gp*}}, \widetilde{\Delta T^*}) = Z(\widetilde{X^*}) \tag{13}$$

Here, $(\widetilde{X^{p*}}, \widetilde{X^{gp*}}, \widetilde{\Delta C^{gp*}}, \widetilde{\Delta T^*})$ are optimal solutions of Model in Eq. (9). $\widetilde{X^*}$ are optimal solutions of Model in Eq. (9). □

From the above discussion, we conclude that the transformation of formulations of **CEMNF** into **LNF** makes it possible to solve **CEMNF** problems by the (ϵ-algorithm method, which is known to be convergent and yields optimal solutions.

5.4 Analysis on the complexity of the transformation

In this methodology, (refer to Figure 4, 5, 6 and 7), we split each node into some subnodes (including existing generation capacity subnodes, expanded generation capacity subnodes, demand subnode, pseudo demand subnode, global pseudo demand node, transmission subnode, arc-transmission subnodes) and also need to split each arc into two arcs and add one new node for each new arc to represent transmission capacity expansions. These steps increase the time and space complexities of the model.

Number of subnodes. For each node ($v \in V$), with $|P|$ types of existing and expansion generations, we will have $|P|$

existing (S) $|P|$ expansion (E) subnodes. We add one subnode D^v to represent the flow demand at each node $v \in V$ and one subnode T^v to receive flows from the other nodes . We also need to add a subnode DU^v, which represents the pseudo flow demand of each node $v \in V$. Thus each node is represented by $2 * |P| + 3$ subnodes.

We also add one global pseudo demand node (DUG) node.

For each arc $uv \in E$, two arc-transmission nodes (L^b_{uv}, L^e_{uv}) are added. Thus total arc nodes are $2 * |E|$.

This is the total number of nodes in the network are $|V| * (2 * |P| + 3) + 2 * |E| + 1$.

Number of links. At each node $v \in V$, each S and each E subnodes have two links, one to D^v and one to DU^v, and thus a total of $2 * 2 * |P| * |V|$ such links.

Each S and each E subnodes at node $v \in V$ have one link to L^b_{vu} and one link to L^e_{vu} subnodes for all outgoing links. This results into a total of $2 * |E| * 2 * |P|$ links.

Each T subnode at node $v \in V$ has one link to L^b_{vu} and one link to L^e_{vu} subnodes for all outgoing links. This results into a total of $2 * |E|$ links. Also, for each link uv, each pair of node L^b_{uv} and L^e_{uv} subnodes has a link to node T^v, resulting into $2 * |E|$ links.

In addition, each DU^v subnode for each $v \in V$ has a link to node DUG.

Thus the total number of links in the new network are $2 * 2 * |P| * |V| + 2 * |E| * 2 * |P| + 2 * |E| + 2 * |E| + |V| = 4 * |P| * (|V| + |E|) + 4 * |E| + |V|$ links in the network.

Thus the overall problem is a linear scaled version of the original problem.

6 Numerical Results

We implement the transformation algorithm **Decomposition-Transform** presented in Section 4.1, which decomposes a **CEMNF** network ($G(V, E)$ as shown in Fig. 4) into a **LNF** network ($G'(V', E')$ as shown in Fig. 7). We generate **CEMNF** network models with random nodes and arcs. The values of flow generation and transmission costs are random numbers. In order to avoid trivial solutions of linear programming models, we keep all numerical values at the similar magnitude range, e.g. $\{10^k | k = 1, 2, 3\}$. Both networks are formulated as linear programming models, which are solved by *lp_solve5.5*, which is an open-source linear/integer programming model solver [18]. We compare the optimal objective function values of the two models and show the results in Table 9.

The results show that the transformation algorithm **DecompositionTransform** transforms **CEMNF** problems into **LNF** problems correctly and the optimal objective function values are the same. It also shows that for a given node number, more arcs in **CEMNF** generally implies more nodes and arcs in **LNF**, but it depends on how many commodities and expansions are available.

case	CEMNF [$\|V\|, \|E\|$]	CEMNF Z^*	LNF [$\|V\|, \|E\|$]	LNF Z^*
1	[4, 4]	14455.7	[33, 84]	14455.7
2	[5, 6]	19006.5	[43, 117]	19006.5
3	[6, 9]	24135.8	[55, 162]	24135.8
4	[7, 10]	20595.5	[63, 183]	20595.5
5	[7, 11]	16432.8	[65, 195]	16432.8
6	[7, 12]	21053	[67, 207]	21053
7	[7, 14]	20353.2	[71, 231]	20353.2
8	[7, 15]	14068.7	[73, 243]	14068.7
9	[8, 11]	25143.1	[71, 204]	25143.1
10	[8, 12]	27139.3	[73, 216]	27139.3
11	[8, 14]	21986.8	[77, 240]	21986.8
12	[8, 16]	20450.7	[81, 264]	20450.7
13	[8, 17]	20653.1	[83, 276]	20653.1
14	[8, 19]	23346	[87, 300]	23346
15	[8, 20]	24691.5	[89, 312]	24691.5
16	[9, 12]	27788.4	[79, 225]	27788.4
17	[9, 14]	19279.3	[83, 249]	19279.3
18	[9, 18]	22989	[91, 297]	22989
19	[9, 19]	23020.6	[93, 309]	23020.6
20	[9, 22]	24839.3	[99, 345]	24839.3
21	[10, 17]	30926.8	[95, 294]	30926.8
22	[20, 50]	61821.3	[221, 780]	61821.3
23	[30, 111]	87425.7	[403, 1602]	87425.7
24	[40, 193]	127327	[627, 2676]	127327
25	[50, 298]	161139	[897, 4026]	161139
26	[60, 444]	191531	[1249, 5868]	191531
27	[70, 592]	199378	[1605, 7734]	199378
28	[80, 791]	228148	[2063, 10212]	228148
29	[90, 1005]	239985	[2551, 12870]	239985
30	[100, 1217]	293334	[3035, 15504]	293334

Z^*: optimal objective function value ; $|V|$: node number; $|E|$: arc number

TABLE 9: The comparison of numerical results of **CEMNF** and **LNF** networks

7 Experimental Results of Parallel Computing

In this section, we present the experimental results of solving the **LNF** models in Eq. (8) transformed from **CEMNF** models in Eq. (7) by the parallel computing algorithm (ϵ-relaxation) in [14]. We choose 25 **CEMNF** models with node size from 4 to 100 and use **Simplex** algorithm [19] to solve **CEMNF** models and use the ϵ-relaxation algorithm to its transformed **LNF** models. The speedup is computed as dividing the run-time of solving **CEMNF** models by the run-time of transforming the **CEMNF** models and solving their corresponding **LNF** models.

The results are shown in Table 10. The time unit is second. Because ϵ-relaxation [16] is an approximation algorithm, in which an ϵ error is tolerated in solving a linear programming model with the format of **LNF** in order to solve it by parallel computing pattern, we set ϵ value as 0.01 in order to assure that the solutions are accurate enough. Let Z^*_L represent the optimal objective value of **LNF** solved by ϵ-relaxation and let Z^*_C represent the optimal objective value of **CEMNF** solved by Simplex. The accuracy is computed as $1 - |Z^*_C - Z^*_L|/Z^*_C$.

We implement the ϵ-relaxation algorithm by multithreaded computing pattern and solve each model with 6 threads on an Intel(R) Xeon(R) (6 cores) multi-core processor. The run time is the average value of solving 10 models with the same network topologies.

All models in Table 10 have accuracy 99% according to the above accuracy formula. The results show that our transformation method can help solving **CEMNF** problems more efficiently than **Simplex** algorithm [19] by parallel computing.

8 Conclusions

In this paper, we have presented a general purpose problem that allows expansion of capacity of nodes and transmission links network expansion flow problems, which arises in many real-world problems involving multi-commodity flow scheduling and generation planning. Compared to classical linear network flow (**LNF**) problems, in which each node has a constant flow demand or supply and each arc has a constant flow transmission capacity, (**CEMNF**) problems have some distinct properties. In (**CEMNF**) problems, the total flow demand is greater than the total currently existing flow supply in the whole network, which means that some node need to do expand their flow generation capacity to meet the total demand. Similarly, the transmission capacities of arcs connecting the nodes are not constants either, which can be expanded to allow more flow to be generated or transmitted

case	CEMNF [$\|V\|, \|E\|$]	CEMNF time	LNF [$\|V\|, \|E\|$]	LNF time	Speedup
1	$[4, 4]$	0.006	$[33, 84]$	0.0011	5.61
2	$[8, 11]$	0.0125	$[71, 204]$	0.0022	5.66
3	$[12, 22]$	0.0246	$[117, 372]$	0.0044	5.63
4	$[16, 34]$	0.0416	$[165, 552]$	0.0074	5.62
5	$[20, 50]$	0.0633	$[221, 780]$	0.0112	5.67
6	$[24, 70]$	0.0952	$[285, 1056]$	0.017	5.61
7	$[28, 99]$	0.1801	$[367, 1440]$	0.032	5.63
8	$[32, 122]$	0.251	$[437, 1752]$	0.045	5.59
9	$[36, 169]$	0.3687	$[555, 2352]$	0.066	5.62
10	$[40, 193]$	0.4474	$[627, 2776]$	0.079	5.65
11	$[44, 249]$	0.5925	$[763, 3252]$	0.106	5.58
12	$[48, 278]$	0.7481	$[845, 3768]$	0.133	5.62
13	$[52, 344]$	1.2056	$[1001, 4596]$	0.214	5.64
14	$[56, 375]$	1.4709	$[1087, 5504]$	0.261	5.63
15	$[60, 444]$	1.7191	$[1249, 5868]$	0.303	5.68
16	$[64, 497]$	2.3751	$[1379, 6539]$	0.422	5.63
17	$[68, 573]$	3.0401	$[1555, 7488]$	0.542	5.61
18	$[72, 615]$	2.7965	$[1663, 8028]$	0.495	5.65
19	$[76, 711]$	4.4323	$[1879, 9216]$	0.789	5.62
20	$[80, 791]$	4.9213	$[2063, 10212]$	0.87	5.66
21	$[84, 871]$	5.518	$[2211, 11087]$	0.97	5.69
22	$[88, 926]$	6.6459	$[2381, 11904]$	1.172	5.67
23	$[92, 1085]$	8.792	$[2511, 12728]$	1.548	5.68
24	$[96, 1136]$	10.151	$[2737, 14156]$	1.793	5.66
25	$[100, 1217]$	12.2032	$[3035, 15504]$	2.152	5.67

TABLE 10: Experimental Results of Parallel Computing

among nodes. The goal, like in classical (**LNF**) problems, is to minimize the flow transmission cost, but we have additional constraints that we also need to minimize the generation cost and generation and transmission capacity expansion cost in the whole network. From the perspective of mathematical programming optimization techniques, both of two problems can be formulated as linear programming models, but their formulations are different.

The above difference makes it impossible to solve (**CEMNF**) problems in parallel computing environment and exploit parallelism available on a multi-processor cluster systems. The (**LNF**) problems have shown to offer efficient solution methods on such systems with convergence and optimality [14]. We developed a methodology to transform (**CEMNF**) problems into (**LNF**) problems such that they can be solved using a parallel high-performance computing platforms. In this paper, we have shown that the transformation methodology has polynomial time and space complexities and the transformed formulation yields the optimal solutions. The numerical results show that the efficiency of solving transformed (**CEMNF**) problems can be improved by parallel computing pattern.

REFERENCES

[1] A. Ouorou, P. Mahey and J.-Ph. Vial, "A Survey of Algorithms for Convex Multicommodity Flow Problems", Management Science, Vol. 46, No. 1 (Jan., 2000), pp. 126-147, Published by: INFORMS

[2] Mustafa Pinar, Stavros A. Zenios, "A data-level parallel linear-quadratic penalty algorithm for multicommodity network flows", ACM Trans. Math. Softw. 20(4): 531-552 (1994)

[3] Mustafa Pinar, Stavros A. Zenios, "Parallel Decomposition of Multicommodity Network Flows Using a Linear-Quadratic Penalty Algorithm", INFORMS Journal on Computing 4(3): 235-249 (1992)

[4] Jones, K. L., I. J. Lustig, J. M. Farvolden and W. B. Powell, "Multicommodity network flows: The impact of formulation on decomposition", Math. Programming 62 95-117, 1993.

[5] Cheney, W., A. Goldstein. "Newton's method for convex programming and Chebishev approximation". Num. Math. 1 (5) 253-268, 1959.

[6] Spingarn, J. E., "Partial inverse of a monotone operator", Appl. Math. Optim. 10 247-265, 1983.

[7] Bertsekas, D. P. 1982, "Projected Newton methods for optimization problems with simple constraints", SIAM J. Control Optimn. 20, 221-246.

[8] LeBlanc, L. 1973, "Mathematical programming algorithms for large scale network equilibrium and network design problems", Ph.D. Dissertation, IE/MS Dept., Northwestern University, Evanston IL.

[9] Richard D. McBride, E. Carrizosa, E. Conde and M. Munoz-Marque, "Advances in Solving the Multicommodity-Flow Problem Richard", Volume 28 , Issue 2, Pages: 32-41, February 1998.

[10] Netlib is a collection of mathematical software, papers, and databases, http://www.netlib.org/

[11] Stern, T. E. , "A class of decentralized routing algorithms using relaxation", IEEE Trans. Conlmmunications COM-25 1092-1102, 1977.

[12] Nagamochi, H., "Studies on multicommodity flows in directed networks", thesis, Eng. Dr. thesis, Kyoto University, Japan, 1988.

[13] Hossein, P. A., D. P. Bertsekas, P. Tseng, "Relaxation methods for network flow problems with convex arc costs", SIAM. J. Control Optimit5. (25) 1219-1243, 1987.

[14] D. P. Bertsekas, J. N. Tsitsiklis, "Parallel and Distributed Computation", Prentice-Hall, Englewood Cliffs, NJ, 1989.

[15] Armand A. Zakarian, "Nonlinear Jacobi And epsilon-Relaxation Methods For Parallel Network Optimization", Doctoral Dissertation, Univ. of Wisconsin, Madison, 1995.

[16] D.P. Bertsekas and P. Tseng, "Relaxation method for minimum cost ordinary and generalized network flow problems", Operations Research, vol. 36, pp. 93114, 1988.

[17] L.C. Polymenakos, "ϵ-Relaxation and auction algorithms for the convex cost network flow problem", Ph.D. Thesis, Electrical Engineering and Computer Science Department, M.I.T., Cambridge, MA, 1995.

[18] http://lpsolve.sourceforge.net/5.5/

[19] George B. Dantzig and Mukund N. Thapa, "Linear programming 1: Introduction", Springer-Verlag, 1997.