University of Memphis
University of Memphis Digital Commons

Electronic Theses and Dissertations

1-1-2018

Efficient Solution of Minimum Cost Flow Problems for Large-scale Transportation Networks

Yuan Zhou

Follow this and additional works at: https://digitalcommons.memphis.edu/etd

Recommended Citation
Zhou, Yuan, "Efficient Solution of Minimum Cost Flow Problems for Large-scale Transportation Networks" (2018). *Electronic Theses and Dissertations*. 1870.
https://digitalcommons.memphis.edu/etd/1870

This Dissertation is brought to you for free and open access by University of Memphis Digital Commons. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of University of Memphis Digital Commons. For more information, please contact khggerty@memphis.edu.

EFFICIENT SOLUTION OF MINIMUM COST FLOW PROBLEMS FOR LARGE-SCALE TRANSPORTATION NETWORKS

by

Yuan Zhou

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

Major: Engineering

The University of Memphis

May 2018

Copyright© Yuan Zhou
All rights reserved

Dedication

To my dear parents, my sweet husband for their unconditional love and support.

Acknowledgements

This dissertation could not be completed without the great support that I have received from so many people over the years. I wish to acknowledge my sincere gratitude to the following individuals.

At the forefront, I would like to express my deepest appreciation to my major advisor, mentor and dissertation committee chair, Dr. Stephanie S Ivey (the University of Memphis), who has been extremely helpful to me during the entire period of my study in the Department of Civil Engineering at the University of Memphis, for her excellent guidance, caring, patience and providing me with perfect atmosphere to pursue research on topics I am truly interested in. I have been privileged to be under her direction on the path of pursing my Ph.D. Degree. Without her persistent support this dissertation would not have been possible.

I am grateful to have worked with my dissertation advisory committee members: Dr. Roger W Meier (the University of Memphis), Dr. Ricardo Taborda (the University of Memphis) and Dr. Sabya Mishra (the University of Memphis). Precious comments of each committee member were very important for improving this dissertation. I also would like to thank them for the tremendous knowledge I have gained from them during several graduate courses.

I would like to express my heartfelt thanks to the faculty and staff in the Department of Civil Engineering and the Intermodal Freight Transportation Institute at the University of Memphis for the knowledge I have learned, and the help and support I have gained over the years.

I would also like to thank the Department of Civil Engineering and the Intermodal Freight Transportation Institute who supported me financially by awarding me the assistantship for pursuing my PhD degree.

Last but certainly not least, I would like to express my thanks to my sweet husband from bottom of my heart. I could never have accomplished this dissertation without his love, immense impact and strong support. I am extremely grateful to my dear parents for their continuously unconditional love and encouragement which was beyond description.

Preface

This dissertation is submitted for the degree of Doctor of Philosophy at the University of Memphis. The research described herein was conducted under the supervision of Dr. Stephanie S Ivey, Department of Civil Engineering, the University of Memphis, between August 2013 and May 2018.

To the best of my knowledge, this study is original, except where acknowledgements and references are made to previous work. To my knowledge, neither this dissertation nor any other substantially similar dissertation has been or is being submitted for any other degree at any other university.

Part of this work has been presented in the following publications:

Zhou, Y., and Ivey, S. (2016). Liner Shipping Network Design of Inland and Sea Transportation Nodes: A Two-Stage Programming Approach. Proceedings of the Transportation Research Board 95th Annual Meeting, Washington, DC, January 2016.

Zhou, Y., and Ivey, S. (2016). An Agglomerative Clustering Based Large-Scale Minimum Cost Flow Algorithm. Proceedings of 2016 INFORMS annual meeting, Nashville, TN, November 2016.

Zhou, Y., and Ivey, S. (2017). A Study of Large-Scale Minimum Cost Network Flow Problem. Proceedings of the Transportation Research Board 96th Annual Meeting, Washington, DC, January 2017.

Zhou, Y., and Ivey, S. (2017) A Decomposition Solution to Large-Scale Multi-Commodity Networks, Proceedings of 2017 METRANS International Urban Freight Conference (I-NUF), Long Beach, October 2017.

Zhou, Y., Ivey, S., Yu, H. (2018) Large-scale Minimum Cost Flow Algorithm: an Agglomerative Clustering based Approach, Transportation Research Board 97th Annual Meeting, Washington, DC, January 2018.

Abstract

Zhou, Yuan. PhD. The University of Memphis. May, 2018. Efficient Solution of Minimum Cost Flow Problems for Large-scale Transportation Networks. Major Professor: Dr. Stephanie S. Ivey.

With the rapid advance of information technology in the transportation industry, of which intermodal transportation is one of the most important subfields, the scale and dimension of problem sizes and datasets is rising significantly. This trend raises the need for study on improving the efficiency, profitability and level of competitiveness of intermodal transportation networks while exploiting the rich information of big data related to these networks. Therefore, this dissertation aims to investigate intermodal transportation network design problems, especially practical optimization problems, and to develop more realistic and effective models and solution approaches that will assist network operators and/or decision makers of the intermodal transportation system.

This dissertation focuses on developing a novel strategy for solving the Minimum Cost Flow (MCF) problem for large-scale network design problems by adopting a divide-and-conquer policy during the optimization process. The main contribution is the development of an agglomerative clustering based tiling strategy to significantly reduce the computational and peak memory consumption of the MCF model for large-scale networks. The tiling strategy is supported by the regional-division theorem and α-approximation regional-division theorem that are proposed and proved in this dissertation. The region-division theorem is a sufficient condition to exactly guarantee the consistency between the local MCF solution of each sub-network obtained by the aforementioned tiling strategy and the global MCF solution of the whole network. Furthermore, the α-approximation region-division theorem provides worst-case bounds,

so that the practical approximation MCF solution closely approximates the optimal solution in terms of its “optimal value”.

A series of experiments are performed to evaluate the utility of the proposed approach of solving the large-scale MCF problem. The results indicate that the proposed approach is beneficial to save the execution time and peak memory consumption in large-scale MCF problems under different circumstances.

Table of Contents

Chapter Page

List of Tables

List of Figures

1 Introduction

With the globalization of trade and the ensuing rapid growth in freight tonnage transported, significant demand has been placed on transportation networks. The complexities inherent in freight transport and the drawbacks associated with a unimodal approach gave rise to expanded intermodal operations. Intermodal transportation has emerged as an increasingly important segment of the global economy, and offers opportunities for developing more efficient, reliable and sustainable freight transportation systems (SteadieSeifi et al. 2014). On both the regional and national levels, policies have been implemented in recent years to stimulate intermodal transport (U.S. Congress 1991; U.S. Congress 1998; European Commission Directorate-general for Energy and Transport 2009). As a consequence, research interest in intermodal freight transportation problems has accelerated during the last few decades.

Substantial changes in the spatial organization of supply and distribution networks have been introduced due to emerging trends that influence the development of the freight transportation system. Among these are (a) geographical expansion of distribution networks, as well as increases in long-distance freight transport movements; (b) changes in the size and functionality of supply and distribution network nodes, such as the development of break-bulk, cross-docking, or transshipment systems, and the development of reverse logistics processes, light manufacturing and packaging, and so forth; (c) increases in the development of hub-and-spoke networks, while concentrating international trade in hub terminals, such as ports and airports; and (d) decreases in the size of consignments, while increasing delivery frequency and vehicle utilization (Zografos & Regan 2004).

The aforementioned development trends of the structure and scale of intermodal freight transportation networks have stimulated related research in this field. One of the important

aspects is with regard to intermodal transport network design. By its nature, the intermodal transport network design problem demonstrates an increased complexity due to the use of multiple modes of transportation and the involvement of multiple decision makers. Therefore, network operators and decision makers who are confronted with long-term decisions on the layout of intermodal terminal infrastructure networks and intermodal service networks, as well as the design of linkages between these two, will need more theoretical and technical support (Caris et al. 2013).

Along with the rapid development of intermodal transportation, the scale and complexity of the intermodal network problem increases dramatically, which brings researchers more opportunities and challenges in network programming problems. Many traditional network problem algorithms are challenged by large-scale networks and the big size of data inherent to them. The operational efficiency and consumption of computing resources of the network optimization process become critical issues when the traditional network problem algorithms encounter extreme large-scale problem, since the large amount of data may lead to long execution time or exceeding available memory. Studies on many network problems with large problem scales have been put forward, such as the large-scale minimum spanning tree problem, the large-scale shortest path problem, the large-scale set covering problem, the large-scale maximum flow problem and so on.

Therefore, the main objective of this dissertation is to investigate large-scale transportation network problems, especially practical optimization problems, mainly focusing on large-scale minimum cost flow problems. Then the effort is put on developing models and solution approaches to assist network operators and/or decision makers of the intermodal transportation system.

Contributions

Among the conventional techniques that are applied to solve network problems, the Minimum Cost Flow (MCF) model is one of the most widely used, and is a mature and practical programming technique. However, as the scale of networks becomes increasingly larger, the MCF faces a new problem in terms of computational and memory requirements.

Based on an exploration of related network problem technologies, this dissertation develops an efficient large-scale MCF problem solving approach that is based on a divide-and-conquer framework. Specifically, an agglomerative clustering based tiling strategy is developed to significantly reduce the computational and memory consumption of the large-scale MCF model. In addition, from the theoretical point of view, this dissertation also proposes and proves the regional-division theorem and the α-approximation regional-division theorem as the theoretical basis of the proposed approach.

The regional-division theorem is a sufficient condition to exactly guarantee the consistency between the local MCF solution of each sub-network obtained by the aforementioned tiling strategy and the global MCF solution of the whole network. In addition, the α-approximation regional-division theorem gives us the worst-case bounds, so that the practical approximation MCF solution closely approximates the optimal one in terms of its "optimal value". In other words, this dissertation produces a methodology that can solve the large-scale network problem without sacrificing much solution accuracy, while greatly shortening the execution time and reducing the memory usage. Experimental results demonstrate the utility of the proposed approach.

Structure of the Manuscript

The structure of the rest of this dissertation is as follows. Chapter Two presents a comprehensive literature review of the related studies. The first subsection of the literature review focuses on the research process and achievements with regard to intermodal transportation network problems, and then summarizes the issues worthy of concern related to these problems. The following subsection further reviews the study of different types of large-scale network problems, and points out the necessity of further research on the MCF problem for large-scale transportation networks.

Chapter Three discusses the problems associated with large-scale intermodal transportation networks, especially the challenge to conventional methods caused by big data derived from large-scale networks. After analyzing the problem, two theorems are proposed and proved in the following subsections. Then an agglomerative clustering based tiling strategy is proposed to deal with this problem.

Furthermore, six groups of experiments are presented in Chapter Four to measure the performance of the proposed methodology for the large-scale minimum cost flow problem from different aspects. Related technologies and methods are described in corresponding subsections as well. The experiment results are also demonstrated and analyzed.

Chapter Five provides final conclusions and a general description of ongoing research and future work directions.

In addition, Appendix A provides detailed explanations of the current MATLAB® implementation of the algorithm produced in this dissertation. It also outlines preliminary ideas about the extension of the study to large-scale multi-commodity minimum-cost flow problems.

2 Literature Review

Development of Intermodal Transportation

Intermodal transportation research was emerging as a new transportation research field in the late 1980s and early 1990s, and moved on to a more mature independent research field during the following decades (Bontekoning et al. 2004). Intermodal transportation is regarded as a competing mode that can be applied as an alternative to unimodal transportation in practice, as depicted in Figure 1.

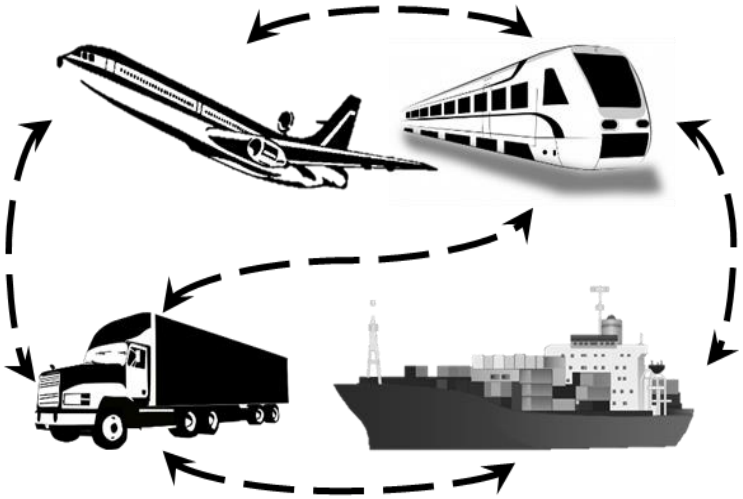


Figure 1 Demonstration of intermodal transportation.

Beginning with the Intermodal Surface Transportation Efficiency Act (ISTEA) in 1991, a more comprehensive approach to intermodal systems became a focus in the United States. ISTEA stated that it is the policy of the United States to develop a National Intermodal Transportation system that " ... shall consist of all forms of transportation in a unified, interconnected manner, including transportation systems of the future, to reduce energy consumption and air pollution while promoting economic development and supporting the Nation's preeminent position in international commerce." (U.S. Congress 1991). Several

handbooks and reference texts addressing intermodal transportation issues were published during the same time (Hayuth 1987; Cambridge Systematics Inc. et al. 1995). After 1990, more literature was developed specifically addressing intermodal transportation issues (Min 1991; European Conference of Ministers of Transport 1993; Southworth & Peterson 2000; Van Duin & Van Ham 1998; Bookbinder & Fox 1998). In the next several years, substantial research has been put forward on intermodal network problems. The issues worthy of concern related to this field are summarized as follows.

Intermodal transportation network design Study of various physical topologies of intermodal networks is necessary. In many cases, the intermodal network is assumed or designed as a hub-and-spoke topology. Research is needed to better understand advancing hub network design methods for full logistics costs (regarding coordination costs), as well as the mechanisms of collaboration in hub networks. Moreover, terminals with both modal and intermodal transfers are an important part of the intermodal transport network and have a strong influence on the competitiveness of intermodal freight alternatives. Thus, terminal network design and terminal allocation problems need further study. Depending on the real-world application, various topologies of intermodal networks also need some attention. For example, a corridor network is more appealing in a region with waterway transportation. Furthermore, a reliable intermodal network is a network that can recover from any disruption by preventing, absorbing, or mitigating its effects. Therefore the flexibility requirement of the intermodal network design is of interest. (Caris et al. 2008; SteadieSeifi et al. 2014)

Cooperation among multiple modes So far, the main attention is given to intermodal transport by rail and truck cooperation. But, in regions with an extensive inland waterway network, such as Western Europe, intermodal transport including inland navigation is also

important. Thus, future research is necessary to improve operations in various combinations of barge, rail, truck and air. Further, the modes involved in intermodal transportation have their own specific characteristics with respect to infrastructure and load units. Therefore, the design and the management of such an intermodal transportation network is restricted by the existing transportation infrastructure, location of modal transfer terminals and logistics cost structure. Complexity of assignment problems among various modes is also increased due to the large variety of load units (type and size), rail wagons and trailer chassis.

Moreover, due to the limited number and capacity of various modes, and additional regulations (e.g. working hour regulations for drivers), simultaneous planning of multiple resources (e.g. vehicles plus drivers) should be incorporated. Taking dynamicity and stochasticity of the data into account also remains a major research challenge. (Macharis & Bontekoning 2004; Ishfaq & Sox 2010; Groothedde et al. 2005; Caris et al. 2008; SteadieSeifi et al. 2014).

Collaboration among sectors of the intermodal transportation system The control of the intermodal transportation system has to be organized by a set of actors all of whom are responsible for only a part of the whole. The cooperation between actors in the intermodal transport chain is a worthwhile research field. Few studies take multiple decision makers into account. Most studies assume that the intermodal transportation system is managed centrally and only considers the requirements of the operator. Actually, the execution of the plans is largely influenced by the interactions and competitions among the carriers. Their collaboration, for example, ensures on-time delivery. It is worth considering the development of a cooperation mechanism between transport providers and terminal operators.

Furthermore, integrating different levels (such as strategic, tactical and operational levels) of planning might provide more reliability, flexibility, and more importantly sustainability, generating more efficient solutions for the industry, which will need more consideration in the collaboration among the sectors of the intermodal transportation system. (Nabais et al. 2015; Caris et al. 2008; SteadieSeifi et al. 2014)

Environmental effect of intermodal transportation Additional considerations related to intermodal and sustainable transportation can also be brought forward. Environmental effects of transportation should be considered because both traffic congestion and transportation modes have a direct effect on exhaust emission of vehicles. Intermodal transportation brings more environmental benefits than unimodal transportation. It is worthwhile to quantify those benefits into current models, for instance, to add new objectives or constraints related to the environmental impact of intermodal transportation. On the other hand, how much environmental requirements can influence the development of intermodal transportation systems can also be evaluated. (Resat & Turkay 2015; Wang et al. 2016)

Lack of the efficient solution approach to the complex intermodal transportation problems For all three planning levels (i.e., strategic, tactical and operational), many intermodal transportation problems have yet to be tackled by operations research techniques. For instance, a tactical planning problem that requires more research attention is the design of the intermodal service network. The optimal number of terminals in a service network, location decisions for hub-terminals, optimal consolidation strategy, allocation of capacity to jobs and scheduling of jobs in terminals, as well as determining truck and chassis fleet size in drayage operations, in particular, need further research.

Due to the complexity of the problems, solving them is still a challenge in itself. Research efforts are needed into the further development of solution methods and the comparison of proposed techniques. Some of the approaches have been studied in the reviewed literature, such as decomposition and relaxation techniques, Branch-and-Cut algorithms, and some metaheuristic algorithms (e.g. the family of Tabu search heuristic). It leaves a great opportunity to study other appropriate algorithms for the problems, for example, other families of metaheuristic algorithms, and comparisons among their performance.

Problem size (network scale) and related computational considerations are issues which increase the complexity of decision support in intermodal transport. Also, computational times required to obtain an optimal solution increase with the size of the instance being solved. In addition, parallel computing offers a capability to handle time and memory consuming solution algorithms, especially for large and decentralized planning problems. Some more efficient solution approaches such as column generation could be applied to solve the problem of larger-size problem instances. (Ghane-Ezabadi & Vergara 2016; Caris et al. 2008; Caris et al. 2013; SteadieSeifi et al. 2014)

Study on Large-scale Network Problem

Consequent with the development of intermodal transportation, the growing size and complexity of the intermodal network problem brings researchers more opportunities and challenges in the study of network programming problems. For many mature network programming problems the optimal solution can be easily obtained when the problem size is small. However, with the rapid increases in globalization processes, the scale and dimension of the practical network (for example, intermodal network) are becoming increasingly larger, which

could be troublesome to many traditional network problem algorithms in terms of computational and memory requirements.

The growing interest in large-scale network problems is also fueled by increasing pressure on various industries to operate more efficiently. Substantial research devoted to large-scale network problems provides the basis to assess the current state of the art. Furthermore, the experience accumulated through previous research can provide vital input into anticipated changes and trends in the study on the solution approaches of various types of large-scale network problem.

The shortest path problem. It is a famous and extensively-studied network problem that is also plagued by the issue brought by large-scale networks, with the number of vertices and edges ranging from several hundreds of thousands to billions. Many researchers have already put effort into related studies during the years.

Dijkstra (1959) introduced the original idea for solving the shortest path problem, which is to construct the tree of minimum total length between *n* nodes, and to find the path of minimum total length between two given nodes.

Meyer and Sanders (2003) designed the Δ-stepping non-negative edge weights (NSSP) algorithm which divides Dijkstra's algorithm into a number of phases, each of which can be paralleled. Their algorithm can be implemented very efficiently in sequential and parallel settings for a large class of graphs.

Crobak (2007) performed an experimental study on applying the Δ-stepping parallel algorithm to the single source shortest path problem with non-negative edge weights on largescale graphs. A remarkable parallel speedup is obtained during the implementation, when compared with competitive sequential algorithms, for low-diameter sparse graphs. It takes less

than ten seconds on 40 processors of the MTA-2 to apply Δ-stepping on a directed scale-free graph of 100 million vertices and 1 billion edges with a relative speedup of close to 30. These are the first experimental results of solving a shortest path problem on a realistic graph with billions of vertices and edges. Their study reveals the enormous potential of dealing with large-scale network problems by applying parallel computing with wise algorithm or methodology design.

Klunder & Post (2006) conducted an extensive computational study on existing shortest path algorithms, and describe combinations of them with bidirectional search and heuristic-estimate technique and a new label correcting techniques utilizing the Euclidean distance and landmarks. The test result shows that the combination of their bucket algorithm, bidirectional search, and the use of a landmark estimate is the best one-to-one shortest path algorithm of all the algorithms they tested in their study.

Sakumoto et al. (2010) conducted a comparison study on Thorup's algorithm and Dijkstra's algorithm. Thorup's algorithm for the single-source shortest path problem with computational complexity of *O(N)*, is smaller than that of Dijkstra's algorithm, which is *O(NlogN)*. The test on a large-scale simulated network of both algorithms shows that Thorup's algorithm is slightly faster but has larger memory consuption than Dijkstra's algorithm.

Gubichev et al. (2010) determined the actual shortest path (i.e., the real sequence of the nodes involved) of large graph problems and proposed a scalable sketch-base index structure to compute the shortest-paths and estimate the length of them. They present a technique leading to near-exact shortest-path approximations in real world graphs with tens of thousands to millions of nodes and edges, while providing several orders of magintude speedup over traditional path computation.

Maruhashi et al. (2012) were aware of the fact that it is getting harder to efficiently estimate the exact length of the shortest path between given pairs of nodes in a graph of real world large-scale networks due to the time complexity of the exact algorithms. They propose a novel method, EigenSP, that estimates the shortest-path length by using an adjacency matrix approximated by a few eigenvalues and eigenvectors. Comparison between their method and the landmark-based method shows that EigenSP estimates smaller distances.

Zhou et al. (2015) concentrated attention on Single-Source Shortest Path (SSSP) which is a fundamental graph algorithm for large-scale networks involving millions or even billions of vertices and links. Based on the well-known Bellman-Ford algorithm, they designed a single-FPGA based method to accelerate SSSP for massive networks. During the process, the network information is stored in external memory with their proposed optimized data layout to enable efficient utilization of external memory bandwidth. This facilitates the single-FPGA based method to achieve the maximum data parallelism to concurrently process multiple edges in each clock cycle. It is claimed that their design is capable of processing 1.6 billion links per second, while achieving a high clock rate of over 200 MHz.

Xu et al. (2016) studied the high-performance shortest-path query algorithm, which normally has two stages: preprocessing and query answering. They put effort on reducing the running time of both stages. An efficient shortest-path query algorithm was proposed, which is called BBQ, to reduce the preprocessing time over the existing algorithms on large-scale graphs, by constructing a distance oracle in a bottom-top-bottom manner. Meanwhile, by traversing the decomposed tree instead of executing separate queries, BBQ can answer batch queries in bulk, which leads to remarkable acceleration.

Aridhi et al. (2015) investigated the shortest path problem in large-scale real-road networks and proposed a MapReduce-based approach. Noticing that it is a time-consuming task for the classical shortest path algorithms to construct the distance matrix between each pair of nodes on a large-scale network due to its size, they design an efficient MapReduce-based approach, of which the objective is to provide high quality solutions in acceptable computational time but not to guarantee optimality. Their approach actually follows the popular divide-and-conquer concept. Afanasyev et al. (2016) introduced a hybrid architecture approach for solving various large-scale graph problems, i.e., the minimum spanning tree and shortest paths problems, which allow using all available resources on both multi-core CPUs and GPUs.

The maximum flow problem. It is a classical combiniatiorial optimization problem which often arises in scientific and engineering application fields. It is also of importance in computer science and operational research fields. Zhang et al. (2011) designed a new method to approximately solve the maximum flow problem in complex and large-scale networks by taking advantage of granular computing with combining the maximal clique. The experiment result indicates the proposed algorithm can obtain the approximate maximum flow in a relatively short running time.

The minimum spanning tree problem. It is one of the most popular network problems, which has been applied in various research fields. Given *n* points in a plane, a minimum spanning tree is a set of edges that connects all the points and has a minimum total length. Many researchers have worked on finding an efficient method to solve the large-scale minimum spanning tree problem for years.

Lin and Xue (2000) studied on the hexagonal minimum spanning tree problem and provided an improved linear time algorithm for computing an optimal layout of this kind of

network problem, which is an extension of their earlier achievement on a quadratic time algorithm dealing with the same problem.

Based on a general concept of spanning graphs, which is a natural definition, Zhou et al. (2002) designed a framework for minimum spanning tree construction; and provided an O(*NlogN)* sweep-line algorithm to construct a rectilinear minimum spanning tree. Later, Karloff et al. (2010) provided an algorithm for finding the minimum spanning tree of a dense graph by adopting the MapReduce model that is designed for computations on terabyte and petabyte scales.

A specific type of minimum spanning tree problem, called a *capacitated minimum spanning tree problem*, has a goal to find a minimum cost spanning tree in a network where nodes have specified demands, with additional capacity constraints on the subtrees incident to a given source nodes. This is an important sub-problem in many telecommunication network design problems.

Ahuja et al. (2003) combined their previously proposed node-based and tree-based neighborhood structure algorithms to provide an advanced composite neighborhood structure for solving the large-scale capacitated minimum spanning tree problem, and proposed a dynamic programming based exact algorithm for searching the composite neighborhood.

The set covering problem. It is one of the most studied NP-hard problems. There are many relevant practical applications of the set cover problem, such as crew scheduling in airline and mass-transit companies, with the objective being to find a set of pairings having minimum-cost that covers a given set of trips.

Marchiori and Steenbeek (2000) focused on obtaining the approximated solution of large-scale set covering problems, mainly arising from crew scheduling. They proposed an adaptive

heuristic-based evolutionary algorithm to find covers of good quality in rather short time. The main ingredient of the proposed algorithm is selecting a small core subproblem which dynamically updates during the excution.

Zhang et al. (2016) were also aware that since the existing set cover and evolutonary algorithms are unable to provide satisfactory efficiency, the growing scale of networks has made the scheduling problem more challenging. They proposed a Kuhn-Munkres parallel genetic algorithm to solve the set cover problem for large-scale wireless sensor networks. The divide-and-conquer strategy is used to reduce the dimension, and then the polynomial Kuhn-Munkres algorithm is applied to splice the feasible set cover problem solutions of each subarea to enhance the search efficiency.

The dial-a-ride problem. The growing demand on share transportation services with flexible routes in many large cities stimulated study on *the Dial-a-Ride problem*, where a number of passengers need to be picked up from and delivered to different locations. The system goal is to minimize the routing cost while respecting a set of pre-specified constraints (pickup time, ride duration and load per vehicle).

Muelas et al. (2013) proposed a variable neighborhood search algorithm to solve the problem of computing the best routes that a public transportation company could service to satisfy a number of passengers' requests. After in depth study, Muelas et al. (2015) designed a new distributed algorithm based on the partition of the requests space and the combination of the routes, for large-scale problem instance (up to 16,000 requests or 32,000 locations).

Ordonez et al. (2016) introduced a data summarization which is an essential mechanism to accelerate analytic algorithms on large data sets. Then they provided a summarization matrix to transform the algorithms to compute linear models to work in two phases: summarization and

iteration. This process can remove main memory limitations of Principal Component Analysis, linear regression, and variable selection, and enable parallel processing.

Minimum Cost Flow (MCF) problem. It is one of the most fundamental problems within network flow theory. L.V. Kantorovich (1960) first studied a group of transportation problems known as network flow problems. These problems are characterized by a need to distribute flow throughout a network in such a way that costs are minimized without violating capacity constraints, and can result in complex problem formulations.

Actually, the maximum flow problem and the shortest path problem address different components of the overall minimum cost flow problem. The maximum flow problems consider link capacity and the simplest cost structure; the shortest path problems consider link cost but not capacity. MCF combines these problem ingredients together (Ahuja et al. 1993). Therefore, many algorithms for solving MCF problem combine ingredients of both shortest path and maximum flow algorithms. For example, many of the MCF algorithms solve a sequence of shortest path problems with respect to maximum the flow.

Since Minimum Cost Flow problems are pervasive in practice, arising in almost all industries, the MCF optimization model is a very important and practical network programming technique with applications in transportation, distribution system planning, capacity planning, manufacturing, etc. Many strategic decisions can be facilitated by more efficient performance of the MCF solving methodology. Therefore, the MCF problem and related algorithms have been investigated profoundly during the last few decades. Researchers have developed a number of different algorithmic approaches that have led both to theoretical and practical improvements in the running time.

Tardos (1986) found that there is a small theoretical drawback to the previous polynomial algorithms to solve the linear programming problem, which is that the number of arithmetic steps depends on the size of the input numbers. The offered a polynomial algorithm for the minimum cost flow and multi-commodity flow problems in which the number of arithmetic steps is independent of the size of the costs and capacities. But the problem of whether any algorithm has a running time that is independent even of the size of the numbers in the constraint matrix remains open in this study.

Orlin (1989) presented a strongly polynomial time algorithm for the un-capacitated minimum cost flow problem based on a refinement of the Edmonds-Karp scaling technique. The proposed algorithm addresses the un-capacitated minimum cost flow problem as a sequence of $O(n \log n)$ shortest path problems on networks with n nodes and m arcs and runs in $O(n \log n (m + n \log n))$ time. His algorithm yields an attractive running time for solving the minimum cost flow problem in parallel.

Goldberg and Tarjan (1990) introduced a framework for solving the minimum cost flow problem that starts by finding an approximate solution then iteratively improves the current solution. Their approach measures the quality of a solution by the amount that the complementary slackness conditions are violated and extends techniques developed for the maximum flow problem to improve the quality of a solution. When the error parameter is small enough, the current solution is optimal and the algorithm terminates.

The algorithms for solving MCF problems may be categorized into several principal approaches, such as primal, dual, primal-dual and scaling algorithms (Kelly et al. 1991; Bertsekas & Tseng 1988; Ahuja et al. 1993). Some of these include polynomial algorithms, such as the interior-point algorithm (Resende & Pardalos 1996), the minimum mean cycle-canceling

algorithm (Goldberg & Tarjan 1989) and network simplex algorithms (Kelly et al. 1991). There are several mature MCF solvers already published such as Relax IV (Bertsekas & Tseng 1994) and LEMON (EGRES 2003). Although the MCF problem has been studied for a long time and can theoretically be solved by these aforementioned algorithms, the effect of the problem size was not well considered in the design of most existing MCF solving methodologies. This poses a significant challenge with the rapid advance of e-commerce, as the network scales of the aforementioned domains are becoming larger and larger. For example, there were 17.7 billion USD in orders placed on T-mall (a large Chinese business to consumer website) during the 24 hours of the "TMALL 11.11 Global shopping Festival" (Nov. 11 2016), with rapid shipment and delivery expected by consumers. Even worse these large-scale problems sometimes need to be resolved very efficiently because in some cases, demands are in real-time. For instance, in the field of biologistics, which deals with operations and logistics for temperature and time sensitive biologic materials and products, there are strict requirements of time efficiency and quality control; otherwise the manufacturer would suffer a complete write-off of the value of their products.

In fact, how to efficiently and effectively deal with the traditional solvable problem on a "big network" is becoming a popular research direction (Jarrah et al. 2009; Babonneau et al. 2006). Therefore, even if the MCF problem is a P problem (i.e., the problem can be solved in polynomial time) and we can obtain the optimal solution easily, when the scale of the input network exceeds computer hardware capacities, MCF will be problematic in terms of computational and memory requirements. In other words, the increasing network scale and data volume has opened new possibilities and challenges for MCF applications.

Chidananda Gowda and Ravi (1995) introduced the clustering principle of minimizing intra-cluster dissimilarity and maximizing inter-cluster dissimilarity to achieve the clustering of numerical vectors. They proposed a hierarchical symbolic clustering algorithm, which makes use of both the similarity and the dissimilarity measures. The proposed symbolic clustering algorithm works on symbolic objects of complex types instead of simple numeric ones and it makes use of both the similarity and the dissimilarity values.

Literature Review Summary

After a comprehensive review of the previous research in the intermodal transportation field, research performed to date offers insights into the complex relationships in the intermodal transportation system as well as the dilemma brought by large-scale intermodal networks. More study is needed on improving the efficiency, profitability and level of competitiveness of intermodal transportation while exploiting the rich information of big data related to the intermodal transportation network.

As summarized and discussed in detail in the previous sections, there are still many problems in this field that deserve more attention. Based on the analysis of the previous sections, this dissertation focuses on exploring an effective strategy for solving the large-scale intermodal network optimization problem. The following chapters provide the detailed study and related experiments of this topic.

3 A Novel Strategy for Solving Large-Scale Minimum Cost Flow Problems

As discussed in the literature review, increasing network scale and data volume has brought new challenges to the solution of network problem. To address this, an efficient large-scale MCF problem solving approach (abbreviated as ELS), which is based on a divide-and-conquer framework, is proposed in this dissertation. The solution uses an agglomerative clustering based tiling strategy (called the AC tiling strategy) to partition the whole large-scale network into several smaller sub-networks. Then, the MCF in each sub-network is solved independently to reduce peak memory consumption and improve efficiency. The optimal solutions of all sub-networks are spliced together as the global MCF solution to this large-scale network. Figure 2 demonstrates the basic process of this divide-and-conquer approach.

The AC tiling strategy can provide a practical solution of a large-scale MCF problem that closely approximates the optimal solution, while greatly shortening the execution time and reducing the peak memory consumption.

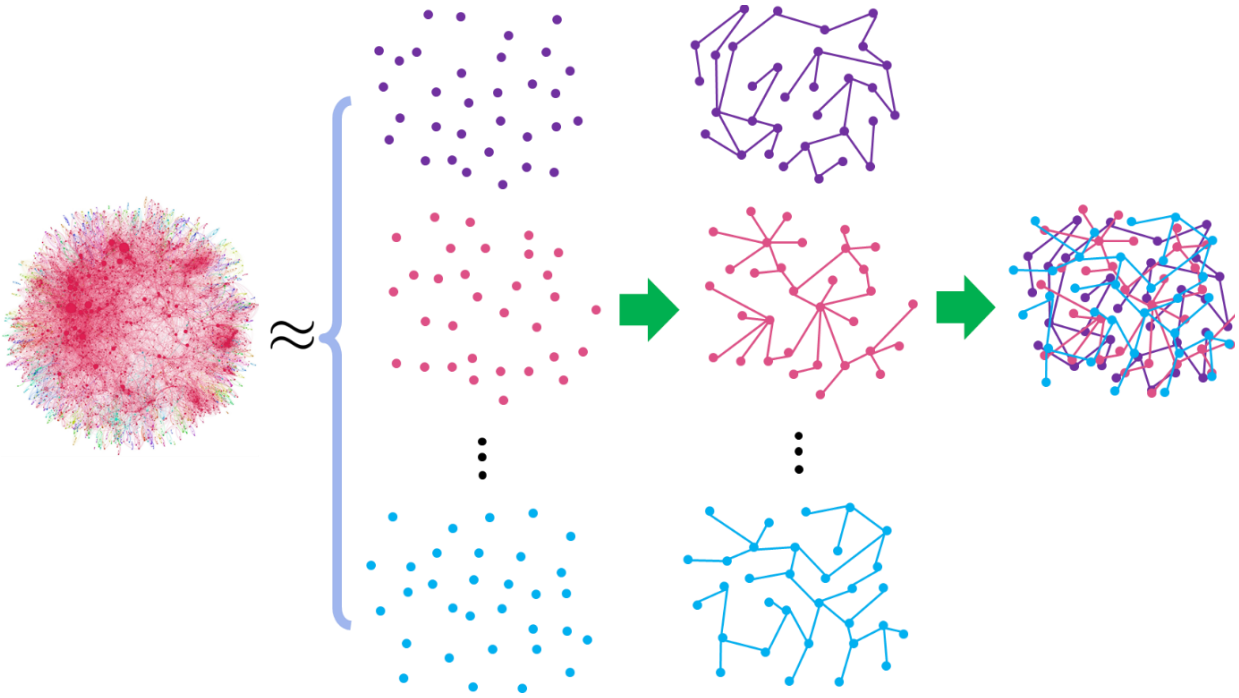


Figure 2 Demonstration of divide-and-conquer framework.

The rest of this chapter is organized as follows. The problem definition and analysis is articulated first. Then, the regional-division theorem and α-approximation regional-division theorem that underlie the AC tiling strategy are proposed and proved successively. Next, the AC tiling is designed, and finally the ELS solution approach is proposed and demonstrated.

Problem Analysis

To begin our discussion of the MCF problem, it is necessary to state that the key information of any network includes: (a) the network topology, (that is, the network node and arc structure); and (b) data such as costs, capacities and supplies/demands associated with the network's nodes and arcs (Ahuja et al. 1993). These are referred to as ***the network features*** in this study. Let $G = (N,E)$ be a directed network with a unit cost c_{ij} and a capacity u_{ij} associated with every arc $(i,j) \in E$; and a number $b(i)$ indicating the supply or demand at every node $i \in N$. Let x_{ij} represent the flow of arc $(i,j) \in E$. Then the MCF problem can be stated as follows:

$$Minmize \sum_{(i,j)\in E} c_{ij}x_{ij} \tag{3.1}$$

subject to

$$\sum_{\{j:(i,j)\in E\}} x_{ij} - \sum_{\{j:(j,i)\in E\}} x_{ji} = b(i) \quad \text{for all } i \in N, \tag{3.2}$$

$$0 \le x_{ij} \le u_{ij} \quad \text{for all } (i,j) \in E. \tag{3.3}$$

Many researchers have worked on efficient implementation and experimental analysis of the MCF problem and have put forward many mature techniques which can exactly solve it (Dantzig 1951; Bradley, Gordon H. Gerald 1977; Armstrong et al. 1980; Goldberg 1997; Lobel 1996; Portugal et al. 2008; Frangioni & Manca 2004; Ahuja et al. 1993; Goldberg & Tarjan 1989; Kelly et al. 1991; Bertsekas & Tseng 1988). However, when the size of the input network exceeds computer hardware capabilities, MCF becomes problematic in terms of computational and memory requirements. In other words, when the coefficient matrix of the MCF model (Equation (3.2)) is very large, it will severely challenge the computer hardware capabilities. In such cases, the large-scale MCF problem cannot be solved as a whole piece. Even if some large coefficient matrices can be put into memory, the solution time will be excessively long. Thus there is a need for another strategy to solve large-scale MCF problems. This study will focus on investigating the large-scale MCF problem, which is described and analyzed in subsequent sections.

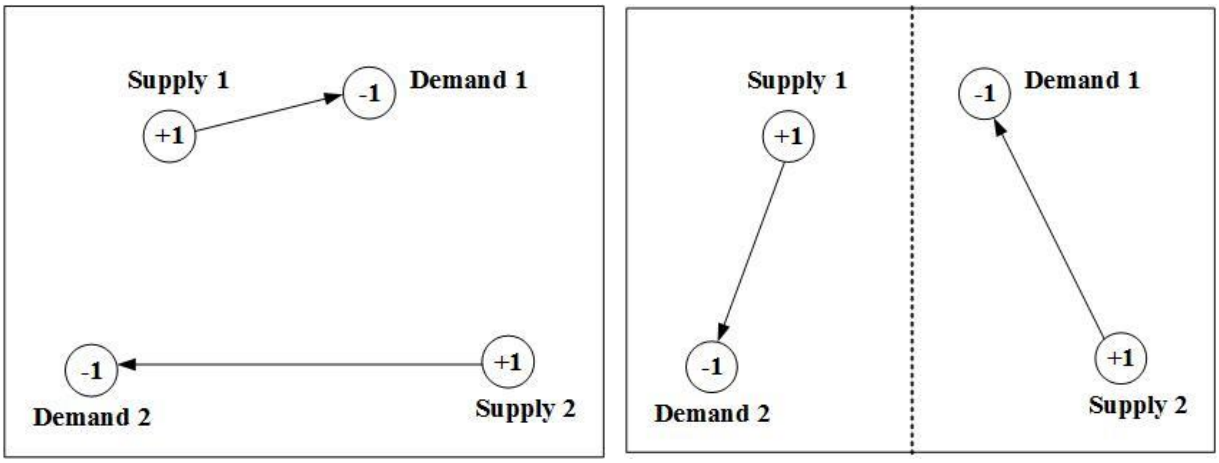
Problem Definition and Difficulty Analysis.

Problem Definition: Given a large-scale network (i.e., the number of nodes and links of this network is very large), we design a tiling strategy that can partition the whole network into several small-scale sub-networks and guarantee the consistency between the local MCF solution of each sub-network and the global MCF solution of the whole network. Then, we can efficiently

obtain the optimal solutions of each sub-network using sequential or parallel processing, and splice them together as the global MCF solution to this large-scale network.

Definition 1: **Consistency.** The local MCF solutions of each sub-network are consistent with the global MCF solution when they are equal to the MCF solution of the same region of the whole network obtained from the global processing.

The difficulty of this problem arises from several aspects. To begin with, the main purpose of the tiling strategy is to use less memory to pre-process a large-scale network for achieving a number of sub-networks with small sizes, so that the time and space complexity of the tiling strategy should not be very high. If the time and space complexity of a tiling strategy is higher than for the MCF algorithm itself, it will be unwise to use it (because it is worse than using the MCF algorithm directly). In addition, whether the MCF optimal solution of each sub-network is consistent with that of the whole network or not is critical to a tiling strategy. In other words, it is necessary to make sure that the MCF solution of each sub-network obtained by a tiling strategy is independent of each other sub-network. If the local optimum solution (i.e., the optimum solution of the sub-network) is inconsistent with the global one, the optimality of the global MCF optimum solution based on this tiling strategy will not be guaranteed. For instance, Figure 3(a) shows a global MCF solution, but if the network is segmented from the middle, the local MCF optimum solution of each piece will be as shown in Figure 3(b). It can be seen that since the MCF solution is decided by the local information, the optimality of the MCF result shown in Figure 3(b) is lost.



(a) Solution obtained as a whole piece (b) Solution obtained as two sub-networks

Figure 3 Illustration of the inconsistency of global and local MCF solutions.

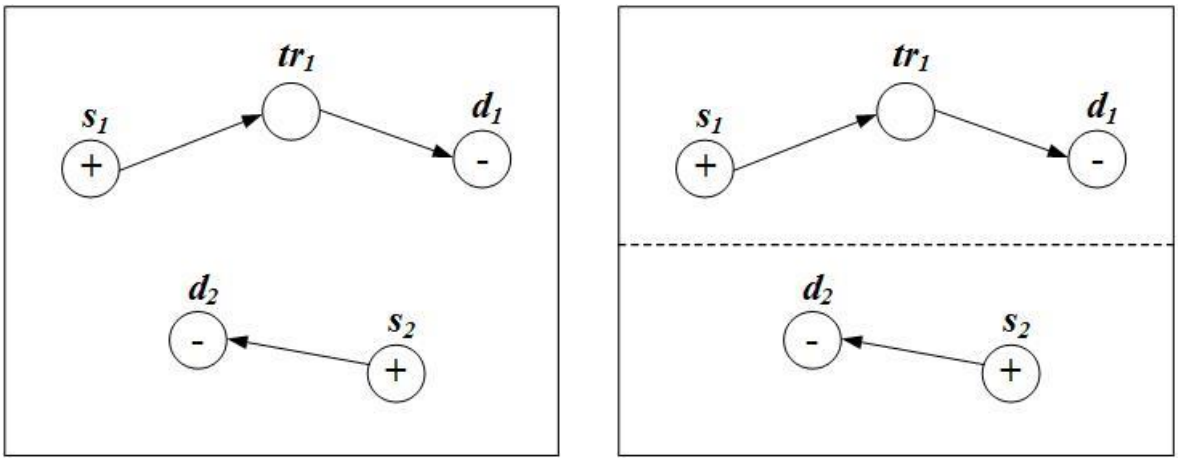
However, not all segmentations lose optimality. For instance, if our segmentation can ensure Supply 1 and Demand 1 are in a sub-network, and Demand 2 and Supply 2 are in a sub-network, the consistency between the local and global MCF results will be guaranteed. In other words, if a tiling strategy can ensure that the supply, demand and transshipment nodes that must have freight flow linkages under the global situation can be partitioned into one sub-network, inconsistency between the local and global MCF results will be avoided. In order to ensure this property in a tiling strategy, a sufficient condition (i.e., the regional-division theorem) that can exactly guarantee the consistency between the local and global MCF solutions, will be discussed in the next section.

Claim: If a tiling strategy can ensure that the supply, demand and transshipment nodes, that must have freight flow linkages under the global situation, can be partitioned into one sub-network, the local and global MCF results will be consistent with each other.

Proof: Let the supply, demand and transshipment nodes in a network denoted by s_i, d_j and tr_k respectively, where i, j and k are the ordinal indexes. The amount of supply/demand of

s_i and d_j is denoted by x_{s_i} and y_{d_j} respectively. Assume that the global optimal MCF solution exists. Under this condition, the network flow will connect some of the nodes in the network (an example is shown in Figure 4(a)). Furthermore, for simplicity, we assume that the global optimal MCF solution is unique (the case of multiple optimal solutions can be easily extended). Now, we will use reduction to absurdity to prove our claim.

We assume that there is a tiling strategy that can ensure that the supply, demand and transshipment nodes that must have freight flow linkages under the global situation, are partitioned into sub-networks together (an example is shown in Figure 4(b)) for which the MCF solution exists, and that the local optimal MCF solution of each sub-network could be obtained. It is also assumed that the local optimal MCF solution of each sub-network is not consistent with the global optimal one.



(a) Global MCF optimal solution. (b) Local MCF optimal solution.

Figure 4 Example of a network with supply, demand and transshipment nodes.

Assume that the MCF objective value of the supply, demand and transshipment nodes of the *m*th sub-network under global processing is θ_m, and the MCF objective value of the *m*th sub-network obtained by local processing is η_m. Because we already assumed that $\theta_m \neq \eta_m$, there are two possible situations: i) If θ_m is greater than η_m, it means that the global MCF optimal

solution is not the optimum because the objective value of the MCF solution of the supply-demand-transshipment nodes of the *m*th group can be replaced by η_m and a better solution will be obtained; ii) If η_m is greater than θ_m, it means that the local MCF optimal solution of the *m*th sub-network is not the optimum because the objective value can be replaced by θ_m and a better solution will be obtained. Therefore, it will be in contradiction to the notion that the objective value of the assumed global and local MCF solution is optimum. **Q.E.D.**

The Regional-Division Theorem

In this section, the regional-division theorem will be proposed and proved. To begin, some useful definitions are introduced as follows.

Definition 2: **Balanced area**: An area in which the supply of origins and demand of destinations are balanced is called a balanced area.

Definition 3: **Region cost of a balanced area**: The region cost of a balanced area is the maximum value of the minimum unit cost between any two nodes that are in this balanced area, i.e., it is the least upper bound of all minimum unit costs between pairs of nodes in this balanced area as shown in Equation (3.4).

$$Regioncost(\mathcal{N}) = sup\{c(n_i, n_j): n_i, n_j \in \mathcal{N}\} \tag{3.4}$$

where $\mathcal{N}$ denotes a balanced area, n_i, n_j denote two arbitrary nodes in the balanced area $\mathcal{N}$, $c(n_i, n_j)$ is the minimum unit cost between n_i and n_j, and $sup\{\cdot\}$ is the least-upper-bound operator.

Definition 4: **Cost between two balanced areas:** The cost between two balanced areas is the minimum value of the minimum unit cost between any two nodes which are in two different

balanced areas, i.e., the cost between two balanced areas is the greatest lower bound of all unit costs between pairs of nodes in two different balanced areas as shown in Equation (3.5).

$$Cost(\mathcal{N}_k, \mathcal{N}_l) = inf\{c(n_i, n_j) : n_i \in \mathcal{N}_k, n_j \in \mathcal{N}_l\} \tag{3.5}$$

where $\mathcal{N}_k, \mathcal{N}_l$ denote two balanced areas, n_i denotes one arbitrary node in the balanced area $\mathcal{N}_k$, n_j denotes one arbitrary node in the balanced area $\mathcal{N}_l$, and $inf\{\cdot\}$ is the greatest-lower-bound operator.

In the aforementioned definitions, two terms (i.e., region and area) are introduced that need a little more clarification. A region, in the definitions of this study, includes both nodes and links, while an area includes only nodes.

Taking Figure 5 as an example, there are two balanced areas outlined by dashed circles, of which the demand (indicated by a negative sign) and supply (indicated by a positive sign) are balanced in their own regions, and the value of the numbers indicates the amount of demand or supply. In balanced area1, assuming that the value of the minimum unit cost between node +5 and node -3 is the maximum value of the minimum unit cost between any two nodes in this area, then it can be referred to as the region cost of balanced area1, which is shown as region cost1 in Figure 5. Then region cost2 is defined in the same way in balanced area2. Assuming that the value of the minimum unit cost between node -1 in balanced area1 and node +2 in balanced area2 is the minimum value of the minimum unit cost between any two nodes in balanced area1 and balanced area2 respectively, then it is termed as the cost between these two balanced areas, as shown in Figure 5. Based on these definitions, the regional-division theorem is described as follows.

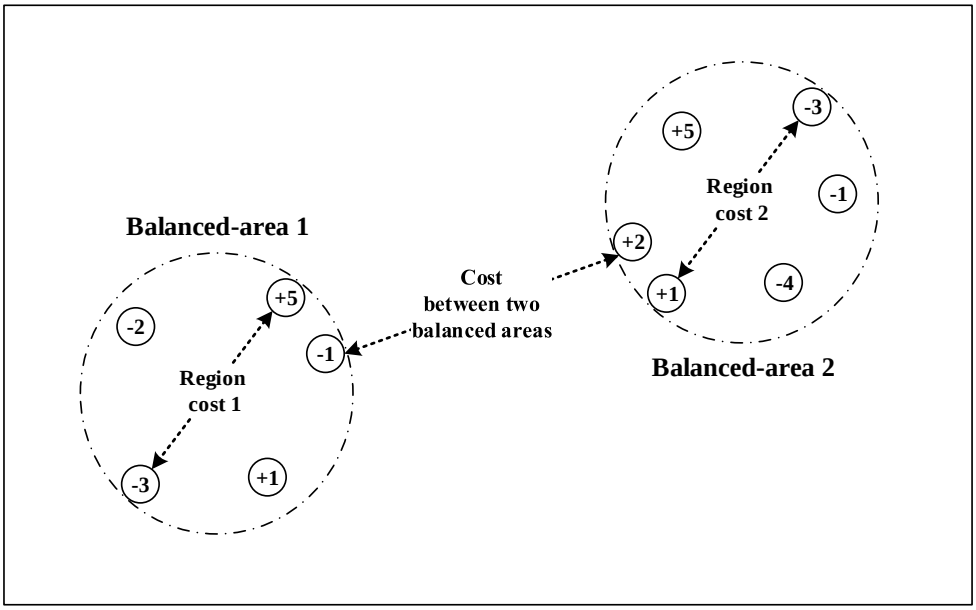


Figure 5 Illustration of balanced area, region cost of a balanced area and cost between two balanced areas.

The regional-division theorem: For one type of commodity, if the cost between any two balanced areas is greater than the region cost of either of them, the optimal MCF solution in both balanced areas will be consistent with the global MCF solution, provided the link capacity constraints can guarantee the existence of those MCF solutions.

This regional-division theorem expects to fulfill the demand within each balanced area so that it is more economical than that between the balanced areas. Taking Figure 5 as an example, if the cost between balanced area1 and balanced area2 is greater than both region cost1 and region cost2, the optimal MCF solutions in balanced area1 and balanced area2 will be consistent with the global optimal MCF solution (i.e., the optimal MCF solution of the union of balanced area1 and balanced area2), under the condition that the existence of the MCF solution within each balanced area is guaranteed by the link capacity constraints and consideration of only one type of commodity. The regional-division theorem is proved by using reduction to absurdity as follows.

Proof: Let us postulate that there are *k* balanced areas satisfying all the requirements of the proposed regional-division theorem, so the MCF solutions of each sub-network and the whole network exist. Assuming that the regional-division theorem cannot guarantee the consistency between the local and global MCF optimal results, in this case, the global MCF optimal result must include some network flows that cross the aforementioned *k* balanced areas. Under this condition, if we can find a new feasible global MCF solution whose objective value is less than that of the assumed global MCF optimal result, and there is no network flow between any two of the assumed *k* balanced areas in this new solution, the regional-division theorem will be proved because it will be in contradiction to the earlier postulate that the objective value of the assumed global MCF optimal result is the minimum.

Assume that the objective value of the global MCF optimal solution of the k balanced areas is λ, that the region cost of the ith balanced area is c_i, and that the cost between the ith and the jth balanced area is $c_{i,j}$. The minimum value of $c_{i,j}$ over all js is denoted by $c_{i,min}$. Further, assume that the total amount of flows (i.e., the absolute value of the network flows, without considering the direction of flow) obtained by the MCF model under the global condition between the ith balanced area and other balanced areas is v_i. For example, there are three balanced areas in Figure 6, i.e., $k = 3$. For balanced area1, $c_{i.min}$ is $c_{1,3}$ (if $c_{1,3} < c_{1,2}$), and v_1 is the summation of the absolute values of *flow1*, *flow2*, *flow3*, *flow4*, *flow5 and flow6*, which is equal to 10 units.

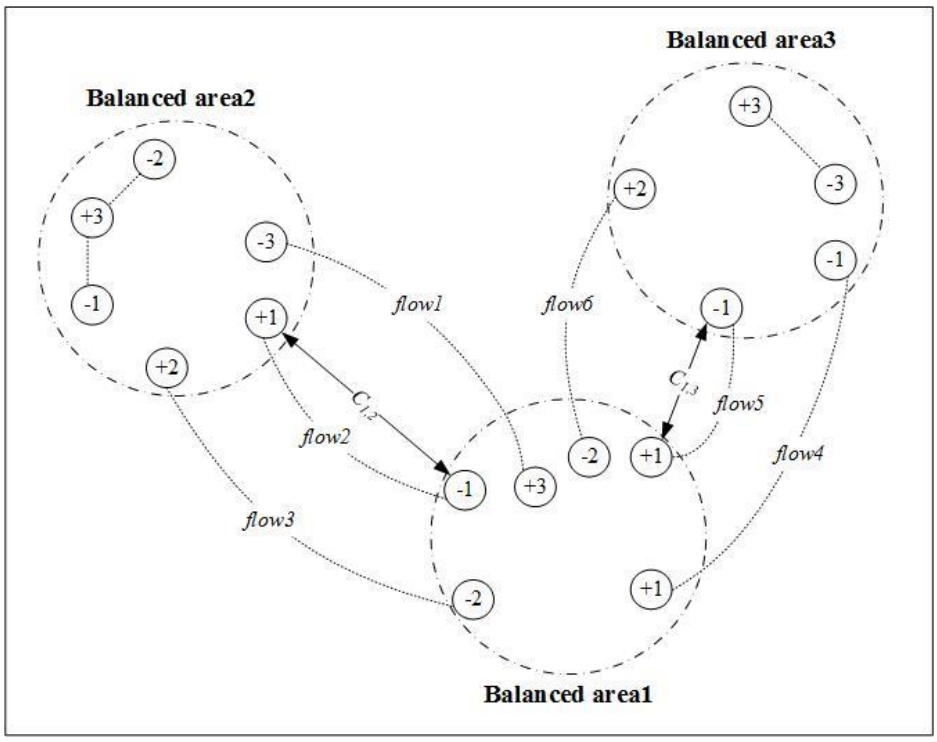


Figure 6 An example of the regional-division theorem.

Now, let's erase all the network flows crossing different balanced areas, (i.e., all the network flows between any two different balanced areas are removed) and keep the rest of the assumed network flows. In this case, the total cost of the removed network flows is greater than or equal to $\frac{1}{2} \cdot \sum_{i=1}^{k} c_{i,min} \cdot v_i$; and a number of unbalanced supplies and demands will be generated in each balanced area, since some connections are cut off. Assume that the total amount of unbalanced supply and demand in the ith balanced area is v_i'. For instance, in Figure 6, v_1' is 10 units, v_2' is 6 units and v_3' is 4 units. Because the supply and demand of each balanced area should be balanced as described in the definition, these v_i' of supply and demand can be satisfied within the balanced area by each other, and $v_i' = v_i$. Hence, these v_i' supplies and demands can be balanced by some new feasible network flows that are exactly inside the ith balanced area, and the cost of these new feasible flows must be less than or equal to $\frac{1}{2} \cdot c_i \cdot v_i'$. That is to say, there is a new feasible MCF global solution whose objective value is less than or

equal to $\lambda - \frac{1}{2} \cdot \sum_{i=1}^{k} c_{i,min} \cdot v_i + \frac{1}{2} \cdot \sum_{i=1}^{k} c_i \cdot v_i'$. Furthermore, $c_i < c_{i,min}$ which is guaranteed by the regional-division theorem. Therefore, $\frac{1}{2} \cdot \sum_{i=1}^{k} c_i \cdot v_i' < \frac{1}{2} \cdot \sum_{i=1}^{k} c_{i,min} \cdot v_i$. This means that the cost of this new feasible MCF global solution is less than that of the assumed global MCF optimal solution, and there is no network flow between any two of the assumed k balanced areas in this new solution. Therefore, it is in conflict with the assumption that the regional-division theorem cannot guarantee the consistency between the local and global optimal solutions of the MCF model. **Q.E.D.**

It follows that the regional-division theorem gives us a sufficient condition that ensures the consistency between the local and global MCF solutions. Since this theorem is proposed to divide large-scale regions into small ones by an appropriate means, we name it the regional-division theorem. It shows that the regional-division theorem takes advantage of the distribution of the cost between supply and demand nodes of a network. However, this theorem is data-dependent, and not all network features of practical networks can fulfill the requirements of the regional-division theorem. In other words, if the network features fail to meet the requirements of the regional-division theorem naturally, it is difficult to apply this theorem to divide the large region. To handle this problem, an updated theorem is proposed and proved in the following section.

The α-Approximation Regional-Division Theorem

As described in the last section, if the features of the input network do not completely satisfy the conditions of the regional-division theorem, the regional-division theorem will not be applicable for this network problem. For example, if there is a network in which the values of minimum unit cost between any two nodes are similar to each other, it is difficult to find an

appropriate division scenario for this network to satisfy the conditions of the regional-division theorem. In order to make the regional-division theorem more practical, we propose the following α-approximation regional-division theorem.

The α-approximation regional-division theorem: For one type of commodity, if the cost between any two balanced areas is greater than or equal to $\frac{1}{\alpha}$ $(\alpha \geq 1)$ times the region cost of each balanced area respectively, the summation of the MCF objective values of each area is always less than or equal to α times of the global MCF objective value, provided the link capacity constraints can guarantee the existence of those MCF solutions.

The α-approximation regional-division theorem reduces the requirement of the original regional-division theorem so that the input network can always be split by choosing a reasonable α $(\alpha \geq 1)$. The higher the α is, the lower the requirement is. It is worth mentioning that when α = 1, the α-approximation regional-division theorem can exactly ensure the consistency between local and global MCF solutions. Therefore, the closer the value of α is to one the better from an accuracy perspective. Although increasing the value of α will decrease the optimality of the final MCF solution, it is more practical and can offer us the worst-case bounds, so that the practical approximation MCF solution closely approximates the optimal one in terms of its "optimal value". The α-approximation regional-division theorem is proved as follows.

Proof: Let us postulate that there are k balanced areas that satisfy the α-approximation regional-division theorem, so the MCF solutions of each sub-network and the whole network exist. c_i is the region cost of the ith balanced area, and $c_{i,j}$ is the cost between the ith balanced area and the jth balanced area. Moreover, $c_{i,min}$ denotes the minimum value of $c_{i,j}$ over all js. In addition, assume that the optimal objective value of the global MCF solution of the k balanced areas is λ, and the summation of the MCF objective solution of each balanced area is $\hat{\lambda}$. Further

assume that the total amount of flows (i.e., the absolute value of the network flows, without considering the direction of flow), that are obtained by the MCF model under the global condition between the ith balanced area and other balanced areas, is v_i.

Now, let us erase all the network flows crossing different balanced areas, (i.e., all the network flows between any two different balanced areas are removed) and keep the rest of the assumed MCF network flows. Then assume that the cost of the rest of the MCF network flows in the ith balanced area is w_i. In this case, the cost of the removed network flows should be greater than or equal to $\frac{1}{2} \cdot \sum_{i=1}^{k} c_{i,min} \cdot v_i$, and a number of unbalanced supply-demands will be generated in each balanced area since some network flows are cut off. Further assume that the total amount of unsatisfied supply and demand in the ith balanced-set is v_i'. As described in the definition of balanced area, the supplies and demands of each balanced area are balanced, so that these v_i' supplies and demands must be balanced, and $v_i' = v_i$. Hence, these v_i' supplies and demands can be met by some new feasible network flows that are exactly within the ith balanced area, and the cost corresponding to these new feasible flows must be less than or equal to $\frac{1}{2} \cdot c_i \cdot v_i'$. Then a new feasible MCF solution of the ith balanced area is generated, of which the upper bound of the objective value is $\frac{1}{2} \cdot c_i \cdot v_i' + w_i$, (i.e., the optimal MCF solution of the ith balanced area should be less than or equal to this value). Furthermore, $c_i \leq \alpha \cdot c_{i,min}$, which is guaranteed by the α-approximation regional-division theorem. Therefore, $\hat{\lambda} \leq \frac{1}{2} \cdot \sum_{i=1}^{k} (c_i \cdot v_i') + \sum_{i=1}^{k} w_i \leq \frac{1}{2} \cdot \alpha \cdot \sum_{i=1}^{k} (c_{i,min} \cdot v_i) + \sum_{i=1}^{k} w_i$. Because $\alpha \geq 1$, it is clear that $\frac{1}{2} \cdot \alpha \cdot \sum_{i=1}^{k} (c_{i,min} \cdot v_i) + \sum_{i=1}^{k} w_i \leq \alpha \cdot \left(\frac{1}{2} \cdot \sum_{i=1}^{k} (c_{i,min} \cdot v_i) + \sum_{i=1}^{k} w_i\right) \leq \alpha \cdot \lambda$, it follows that $\hat{\lambda} \leq \alpha \cdot \lambda$. **Q.E.D.**

If a given network satisfies the conditions of the α-approximation regional-division theorem, there will be a division scenario that can ensure that the obtained balanced areas satisfy the conditions of the α-approximation regional-division theorem. Thus, we are ready to take advantage of the agglomerative clustering pattern to pursue the division scenario in the next section.

The AC Tiling Strategy

In this section, a tiling strategy (i.e., the AC tiling strategy) that is based on the α-approximation regional-division theorem is designed. As discussed in the previous sections, if the features of a network satisfy the conditions of the α-approximation regional-division theorem, this indicates that a clustering phenomenon exists in the node distribution of this network, which is not only referring to the geographic distribution, but also the economic distribution involving the unit cost between nodes. Therefore, an agglomerative clustering analysis approach could be used to design this AC tiling strategy. The corresponding approach framework is introduced in the following section.

First, each $node_i$ is considered as a clustering object whose clustering radius is denoted as γ_i. When the minimum unit flow cost between $node_i$ and $node_j$ is smaller than $\frac{1}{\alpha}\ max(\gamma_i, \gamma_j)$, these two nodes will be clustered together in one area. If all the nodes fulfill the clustering termination conditions, the clustering process will stop. These conditions are i) the supply and demand of each cluster should be balanced; ii) the clustering radius of each node in a cluster should be greater or equal to the region cost of this cluster; iii) the MCF solution of each cluster exists. Otherwise, the clustering radii of the node that do not meet the termination condition will be increased; and then the nodes will be clustered again. The related pseudo-code is shown in Algorithm 1.

Algorithm 1 The pseudo-code of the AC tiling strategy framework

Input: all the nodes and the initialized clustering radii of all the nodes (γ_i denotes the cluster radius of the *i*th node.).
Output: the clustering result of all the nodes.

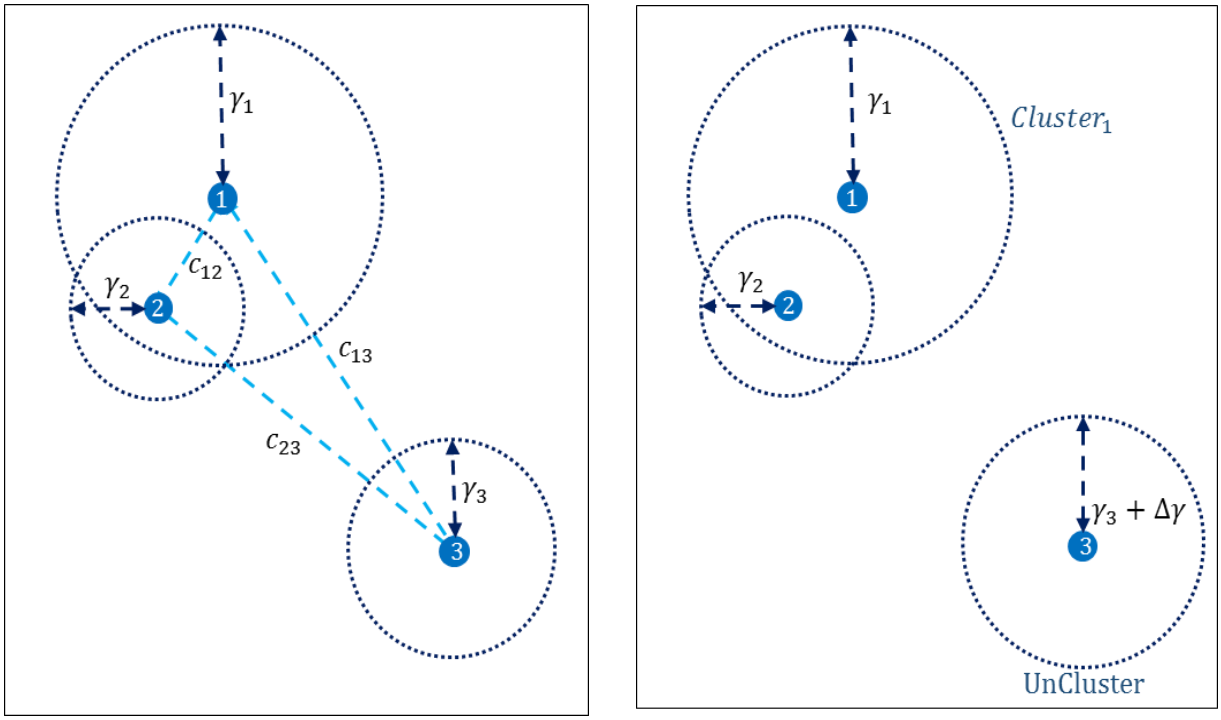
•*UnClustered* = all nodes
•*n* = 1
while *UnClustered* ≠ Φ
∀*node* ∈ *UnClustered*
•$Cluster_n = Cluster_n \cup \{node\}$
•*UnClustered* = *UnClustered* –{*node*}
for *i* = 1:K % K is the number of nodes in $Cluster_n$
for *j* = 1:L % L is the number of nodes not belong to $Cluster_n$
if minimum unit-cost between $node_i$ and $node_j \leq \frac{1}{\alpha} Max(\gamma_i, \gamma_j)$
% $node_i \in Cluster_n$, $node_j \notin Cluster_n$
•$Cluster_n = Cluster_n \cup \{node_j\}$
if $node_j \in$ *UnClustered*
•*UnClustered* = *UnClustered* –{ $node_j$ }
end if
end if
end for
end for
•*n* = *n*+1
end while
for *m* = 1: N % N is the total number of the nodes
if $node_m$ does not satisfy the clustering termination condition
•$UnClustered = UnClustered \cup \{node_m\}$
• $\gamma_m = \gamma_m + \Delta\gamma$
end if
end for
if *UnClustered* ≠ Φ
•Go to step 3
else
•Terminate the clustering process
end if

In Algorithm 1, γ_i and $\Delta\gamma$ are parameters defined by users. To begin with, we know that each obtained node cluster is supply-demand balanced, i.e., a balanced area. In addition, the termination condition of the AC tiling strategy can guarantee that the clustering radius of each

node in a cluster is greater than or equal to the region cost of this balanced area. Furthermore, the AC tiling strategy can also ensure that the obtained balanced areas must satisfy the α-approximation regional-division theorem. Since if two clusters are not merged by the AC tiling strategy, it indicates that these two clusters are supply-demand balanced, the MCF solution of each cluster exists, and the least value of the minimum unit cost between two nodes which are in these two different areas will be greater than 1/α times their clustering radii, respectively. Because the clustering radius of each node is the region cost of its own cluster, the cost between these two clusters will be greater than 1/α times the region cost of each cluster respectively, i.e., the requirements of the α-approximation regional-division theorem are satisfied. Therefore, the output of Algorithm 1 (i.e., the node cluster) satisfies the α-approximation regional-division theorem. It can be seen that a smaller α will offer higher optimality.

However, this clustering method is data-dependent, and not all network features can fulfill the requirement for a small α. If the data itself is not sparse enough, an over-small α will cause all the nodes to be clustered in one cluster, i.e., the AC tiling strategy will be useless for the large-scale MCF problem.

Figure 7 uses a simple examvple demonstrating a part of the basic process of Algorithm 1. Figure 7 (a) includes three nodes having their own clustering radii labeled (γ_i), and the turquoise dashed lines indicate the unit-cost between each pair of nodes (c_{ij}). According to Algorithm 1, node 1 and node 2 are clustered together after comparing the unit-cost between them (c_{12}) with $max(\gamma_1, \gamma_2)$. Node 3 is not clustered with the other two nodes based on the same condition, as shown in Figure 7 (b). Then, the clustering radius of node 3 is increased to $\gamma_3 + \Delta\gamma$ for the next round in the clustering process.



(a) Three unclustered nodes (b) Cluster result of first round clustering

Figure 7 An example of Algorithm 1 application.

A detailed explanation of the current implementation of Algorithm 1 is provided in the Appendices section of this dissertation, including the main data structures, code prototypes with their attributes and functionalities (a workflow is produced to demonstrate the detailed steps of this algorithm, shown in Figure 43), potential parallelization implementation and code maintenance and version control plan.

In Algorithm 1, calculating the region cost of a cluster and the minimum unit cost between two nodes is a time-consuming procedure. The reason for this is that finding the minimum unit cost between two nodes requires an expensive algorithm, e.g., the Dijkstra algorithm. However, the minimum unit cost between two nodes could be calculated off-line, and we can also approximately use the straight-line distance between two nodes instead of their minimum flow cost in practice (because, in reality, the minimum flow cost between two nodes is usually proportional to their distance).

In addition, to determine whether the MCF solution of a balanced area exists, we could adopt the arc capacity sensitivity analysis (Ahuja, Magnanti and Orlin, 1993). Since the demand and supply is balanced in the balanced area, the only constraint on the existence of the MCF solution is the capacity of the link. In practice, the connection and link capacity of a balanced area are usually capable of guaranteeing the existence of an MCF feasible solution. Thus, we can ignore this termination condition to speed up our algorithm. On the other hand, even if we have to consider this termination condition in some circumstances, since the problem size of each sub-network (i.e., each cluster) is usually far less than that of the global network, the computation of this part is acceptable.

Therefore, based on the aforementioned discussion, the time complexity of Algorithm 1 is approximately $\mathrm{O}(N^2)$, where N is the number of nodes in the global network. Admittedly, there are many mature algorithms dealing with MCF problem, such as minimum-circle canceling algorithm (Rader 2010), whose complexity is $O(N^4)$. Since this proposed tiling strategy has much lower complexity, to cluster the huge size network before applying (any of) the MCF algorithms will save execution time and peak memory. So, it is a profitable tiling strategy for the large size MCF problem.

Based on the proposed AC tiling strategy, the ELS is operated as follows. First, all nodes in the original large-scale network are clustered into groups (i.e., sub-networks) using the AC tiling strategy. Second, the MCF program is applied to each sub-network either in parallel or in sequential computing. Then, the results of local MCF problems are combined together as the final global solution.

4 Numerical Experiments

In this chapter, the performance of the efficient large-scale MCF problem solving approach (ELS) is tested from different aspects through a series of numerical experiments.

Experiment Introduction

Theoretically, the tiling MCF strategy proposed in this study offers a new approach that can be combined with any existing of MCF problem solver, such as Relax IV (Bertsekas & Tseng 1994), which is the code for solving MCF problems, and LEMON (Kovács 2015; EGRES 2003) , which is a C++ template library providing efficient implementations of common data structures and algorithms with a focus on combinatorial optimization tasks connected mainly with graphs and networks. To test the effectiveness of ELS, it is necessary to compare the execution time (and/or peak memory consumption) using the same solver to solve the MCF problem as a whole as is used to solve each sub-network in the ELS approach. The numerical experiments in this chapter are developed on the grounds of this basic train of thought.

It is also necessary to test the ELS on networks of various scales and characteristics. A common method to obtain test problem instances is to generate them with standard random generators, such as NETGEN, GRIDGEN, GOTO, and GRIDGRAPH (Kovács 2015). For example, LEMON and RELAX IV were both tested using these randomly generated problem instances. The common generators, such as GRIDGEN, GOTO and GRIDGRAPH, usually produce grid networks as the MCF problem instances for testing, as shown in Figure 8. Nie (2010) used grid networks for testing and analyzing his bush-based methods for a traffic assignment problem.

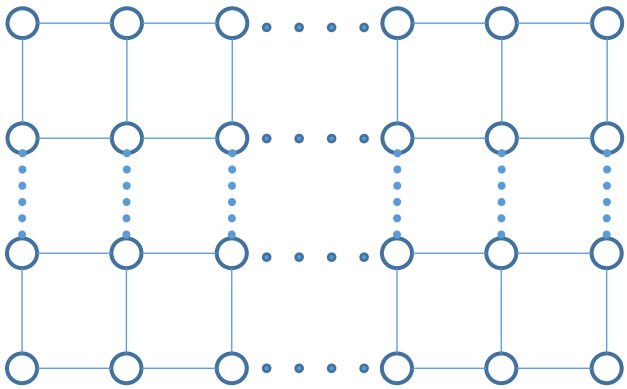


Figure 8 Grid network example

Therefore, the proposed MCF tiling strategy is tested on randomly generated grid networks with different scales and characteristics by a series of experiments in this chapter. Each node in the grid network (as shown in Figure 8) will be randomly assigned *n* units supply or demand. Note that *n* can be a different integer for each node, and that when *n* equals zero it indicates that the node is a transshipment node. Since each network is generated directly as a connected grid network, the connectivity of the network is guaranteed in the experiments.

Several groups of experiments for the performance measurement of ELS are presented in this chapter. The first group of experiments tests the effectiveness of ELS on grid networks with binary supply and demand when the scale and characteristics of the networks are changing. The second group of experiments aims to measure the performance of ELS on grid networks with multiple supplies and demands when the scale and characteristics of the networks are changing. The third group of experiments is conducted to compare the ELS performance between serial and parallel computing when the scale and characteristics of the networks are changing. The fourth group of experiments introduces the greedy-based tiling strategy as a base case to compare with the ELS. The fifth and sixth groups of experiments are both tested on realistic networks. The fifth experiment is performed on real networks with binary supply and demand simulation, and

the sixth experiment includes the performance measure for a real network with multiple supplies and demands.

Experiment One: Effectiveness Testing with Binary Supply and Demand

The aim of the first experiment is to measure the performance of ELS with binary supply and demand (Binary-SD) assigned to the nodes. The statistical execution time and the statistical peak memory consumption are compared between the global solution method and ELS using the interior-point algorithm which is one of the most famous linear programming algorithms used to solve MCF problems (Goldberg & Tarjan 1989; Ahuja et al. 1993; Rader 2010; Resende & Pardalos 1996). It is worthwhile mentioning that, in all of the experiments, the MCF problem is solved by the interior-point algorithm which is supplied by the MATLAB software. In other words, all of the MCF solutions (global and local in ELS) are obtained by means of the 'canned' program in MATLAB. Therefore, it is appropriate to consider the global MCF solutions as credible benchmarks for the application of the ELS strategy.

Simulation Setting.

As explained earlier, the test problem instances are generated as grid networks with increasing scales from 200 × 200 nodes to 1000 × 1000 nodes with an increment of 200 rows and 200 columns, and from 1200 × 1200 nodes to 3000 × 3000 nodes with an increment of 400 rows and 400 columns. The unit cost of each direct link (i.e., c_{ij} in Equation (3.1)) in each network is simulated as un-weighted cost in this experiment. For each network scale, the demand and supply is randomly assigned on each node with one or zero unit commodity (i.e., +1 indicates one unit commodity of supply, -1 indicates one unit of demand, and zero indicates that the node is a transshipment node), while the total demand and total supply is balanced.

We use the simulation method proposed in (Yu et al. 2011) which is named as ODLS to generate the supply and demand nodes for this experiment. This method can generate the grid networks with user assigned scale and randomly generate an integer number for each node, it fulfilled the requirements of the generated problems instances for the experiment, as discussed in the experiment introduction section. In this simulation software, the role of supply and demand nodes is similar to the noise of the image (i.e., each noise node can have positive or negative value which indicates supply or demand node in this experiment).

Within each round of the simulation the location of the supply and demand nodes is changed. In addition, the number of the supply and demand nodes is inversely proportional to the quality of the image, i.e., the better the quality of the image, the less the number of supply and demand nodes. Under this setting, we cannot directly control the number of the supply and demand nodes, but we can indirectly control them through the expected image-quality parameter of ODLS. The parameter is scaled from zero to one, with zero indicating all of the nodes have supply or demand, and one indicating none of the nodes has supply or demand. In this experiment, 0.7 is selected as the value of the image-quality parameter. Furthermore, the amount of supply and demand of each node is controlled by a probability distribution (i.e., hypergeometric distribution). During this process, the upper and lower bounds of the amount of each supply and demand node can be controlled, but not the percentage of the supply and demand nodes.

When the network scale is smaller than or equal to 1000×1000 nodes, 20 instances of supply and demand nodes setting are generated for each scale; and when that is greater than 1000×1000 node, one instance is generated for each scale. Table 1 shows the statistical information of the number of supply and demand nodes with different scales of networks

generated for this experiment. Figure 9 shows the supply and demand nodes generation information with the scales of networks as well. Figure 10 shows two examples of the supply and demand nodes generated for networks of 200×200 nodes and 1000×1000 nodes. In Figure 10 blue pixels indicate demand nodes, red pixels indicate supply nodes and green pixels indicate transshipment nodes.

Table 1 Number of supply and demand nodes for different scales of networks (Binary SD)

Network Scale	Mean value	Min-value	Max-value
200×200	4917	4780	5004
400×400	19773	19446	20066
600×600	44607	44096	45098
800×800	79331	78652	79868
1000×1000	124027	123374	124863
1200×1200	452233	-	-
1800×1800	748827	-	-
2200×2200	1118863	-	-
2600×2600	1563355	-	-
3000×3000	2082696	-	-

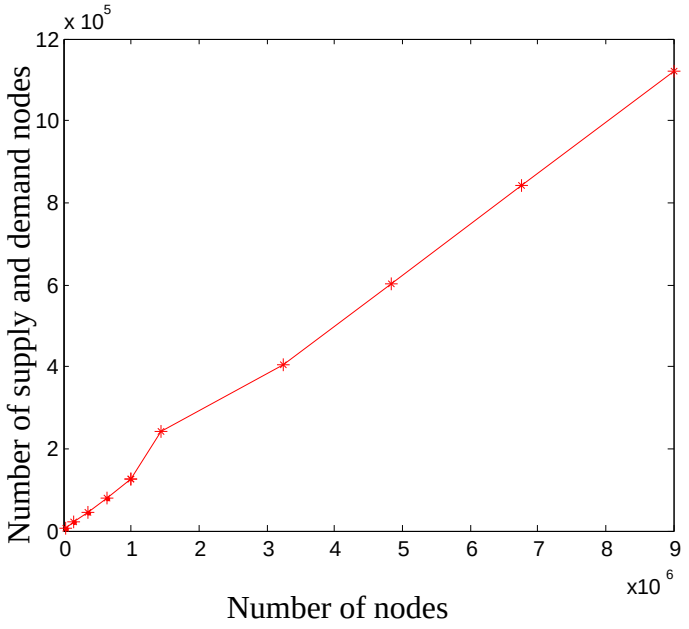
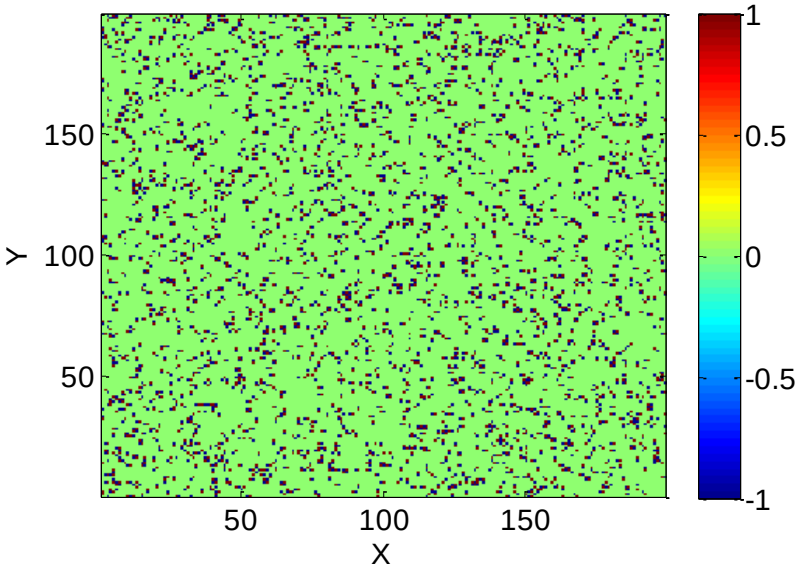
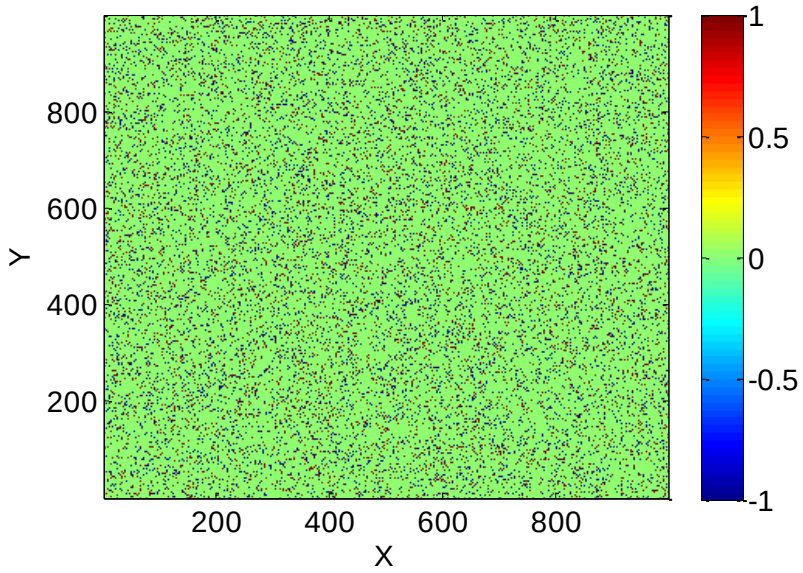


Figure 9 Number of supply and demand nodes for different scales of networks (Binary-SD)



(a) Supply and demand nodes with network scale of 200×200 nodes



(b) Supply and demand nodes with network scale of 1000×1000 nodes

Figure 10 Examples of supply and demand nodes generation (Binary-SD)

Experiment One Process and Results.

In this experiment, for each scale of simulated grid network, the binary supply or demand of each node is randomly generated 20 times by applying ODLS. Each time, the AC tiling strategy (i.e., Alg. 1) is applied to obtain the clustering result of all nodes (i.e., the sub-network

information) with different assigned-α values, where the minimum unit cost between two nodes is approximately substituted by the distance between them (the rationale is described in Chapter Three). Then, the actual-α value is calculated to compare with the assigned-α value. To be specific, according to the clustering result, the interior-point algorithm is applied to obtain the local optimal objective values of the MCF problem of each sub-network. Thus, the actual-α can be calculated by dividing the combined optimal objective value, which is the summation of the local optimal objective values of all the sub-networks, by the global optimal objective value of the entire network which is also obtained by the interior-point algorithm. Several values were tested for assigned-α from $\alpha = 5$ to $\alpha = 25$. The related testing results (i.e., actual-α, number of clusters, peak memory consumption and execution time, and objective value) are shown in Table 2. It shows that, in this experiment, when assigned-α equals 5, the original network is clustered into one cluster, which does not offer benefits over the global solution process. On the other hand, when the assigned-α is greater than 20, the clustering results do not change significantly compared to assigned-$\alpha = 20$. Therefore, in the following part of this experiment, testing is only conducted on assigned-$\alpha = 15$ and assigned-$\alpha = 20$.

Table 2 Assigned-α selection test results (Binary-SD)

assigned-α	Network Scale	Execution time (min)	Peak memory consumption (MB)	Number of clusters	Objective value
5	200×200	0.46	115.34	1	3223.01
	400×400	0.53	85.49	1	12775.06
	600×600	1.41	200.62	1	28444.16
	800×800	3.73	405.82	1	50664.32
	1000×1000	9.92	722.20	1	79141.56
15	200×200	0.26	0.52	1724	3223.01
	400×400	1.01	0.16	6763	12775.06
	600×600	2.55	2.89	15157	28444.16
	800×800	4.86	5.13	26937	50669.04
	1000×1000	8.39	8.01	41953	79144.28
20	200×200	0.25	0.15	1724	3223.01
	400×400	1.17	0.14	6763	12775.06
	600×600	2.57	2.89	15158	28444.16
	800×800	4.33	5.13	26938	50669.05
	1000×1000	7.89	8.00	41955	79144.28
25	200×200	0.27	0.16	1724	3223.01
	400×400	1.05	1.28	6763	12775.06
	600×600	2.46	2.89	15158	28444.16
	800×800	4.52	5.13	26938	50669.05
	1000×1000	7.48	8.01	41955	79144.28

The values are similar for performance of the ELS with increasing assigned-α as shown in Table 2, but this is likely due to the simple geometry of the generated grid structure. The differences in performance and efficiency among different assigned-αs are expected to become more apparent in more complex test cases.

For each assigned-α, while running the test 20 times on each scale of network, the value of each actual-α is documented. In this experiment, it is assumed that the value of actual-α has an approximate normal distribution. Therefore, according to the results of the 20 replicates, statistical results of actual-α are produced in Table 3, in which the mean actual-α with the minimum and maximum value of the actual-αs are listed corresponding to each scale of network.

The 95-percent confidence interval for the mean actual-α of each scale of network is provided in Table 3 as well. This is obtained by applying the method introduced by Ledolter and Hogg (2009). Figure 11 displays the statistical results of actual-α with increasing network scales. From Table 3 and Figure 11 we can see that, although the assigned-α are 15 or 20, the actual-α is not far from one, even when the network scale as large as 3000×3000; and the actual-α value tends to be stable when the network scale is larger than 1200×1200 in this experiment. Figure 12 shows the number of clusters obtained by different assigned-αs while network scale is increasing, which is approximately linear.

Table 3 Statistical results of actual-α (Binary-SD)

assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
15	200×200	1.0001492	1.0000010	1.0006370	[1.0000345,1.0002640]
	400×400	1.0000632	1.0000016	1.0002756	[1.0000257,1.0001007]
	600×600	1.0000393	1.0000019	1.0002105	[1.0000144,1.0000724]
	800×800	1.0000384	1.0000022	1.0001207	[1.0000222,1.0000625]
	1000×1000	1.0000382	1.0000024	1.0001025	[1.0000272,1.0000571]
	1200×1200	1.0001813	-	-	-
	1800×1800	1.0001550	-	-	-
	2200×2200	1.0001688	-	-	-
	2600×2600	1.0001629	-	-	-
	3000×3000	1.0001666	-	-	-
20	200×200	1.0001492	1.0000010	1.0006370	[1.0000345,1.0002640]
	400×400	1.0000632	1.0000016	1.0002756	[1.0000257,1.0001007]
	600×600	1.0000428	1.0000019	1.0002105	[1.0000164,1.0000692]
	800×800	1.0000463	1.0000022	1.0001207	[1.0000272,1.0000654]
	1000×1000	1.0000465	1.0000024	1.0001038	[1.0000315,1.0000614]
	1200×1200	1.0001813	-	-	-
	1800×1800	1.0001550	-	-	-
	2200×2200	1.0001688	-	-	-
	2600×2600	1.0001641	-	-	-
	3000×3000	1.0001684	-	-	-

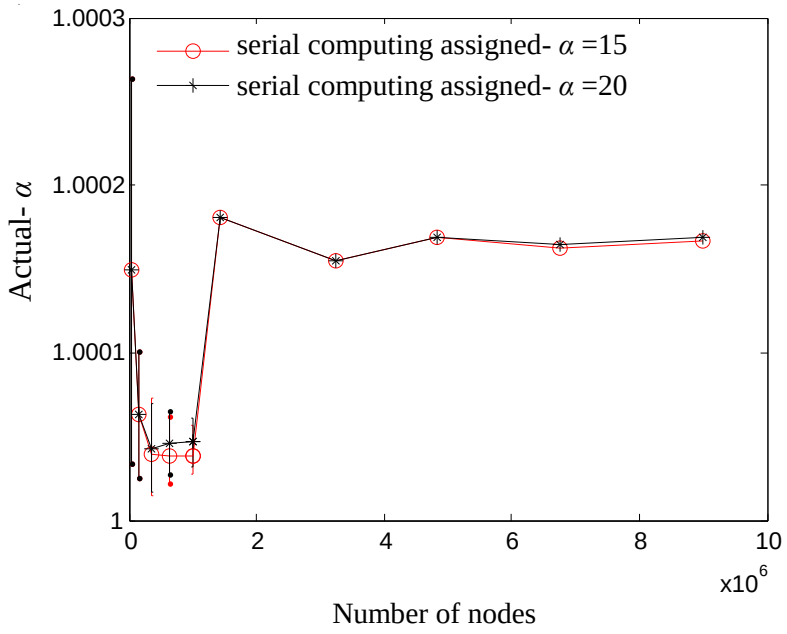


Figure 11 The statistical result of actual-α (Binary-SD)

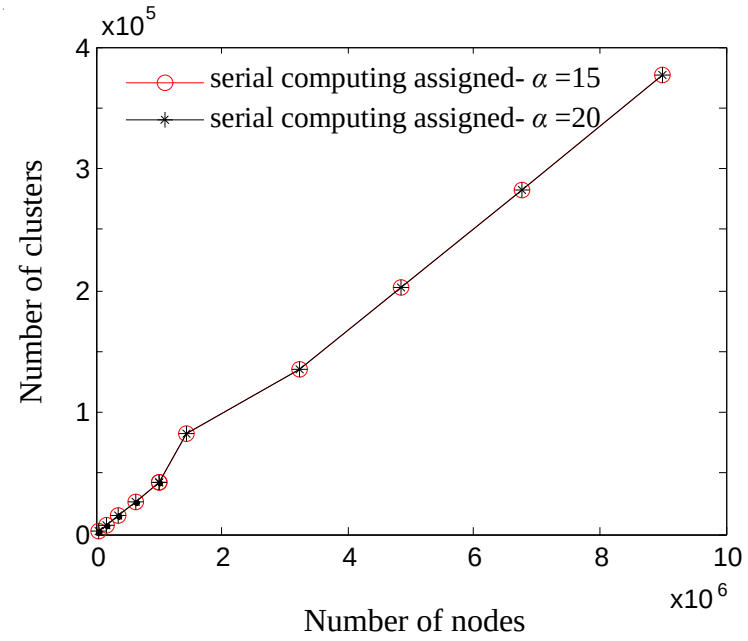


Figure 12 The number of clusters obtained by different assigned-αs (Binary-SD)

During this experiment, for each assigned-α, the execution time and peak memory consumption of the global MCF algorithm and that for when the ELS is applied (corresponding to each scale of the testing network) is recorded. The ELS execution time is recorded as the

summation of the solution times for all clusters and the time of the clustering process. According to the results of the 20 replicates, results of the ELS execution time and peak memory consumption is produced in Table 4 and Table 5, as well as the results for the execution time and peak memory consumption of the global MCF algorithm (which solves the MCF problem of the entire network as one piece).

Table 4 Execution time of serial computing (Binary-SD) (min)

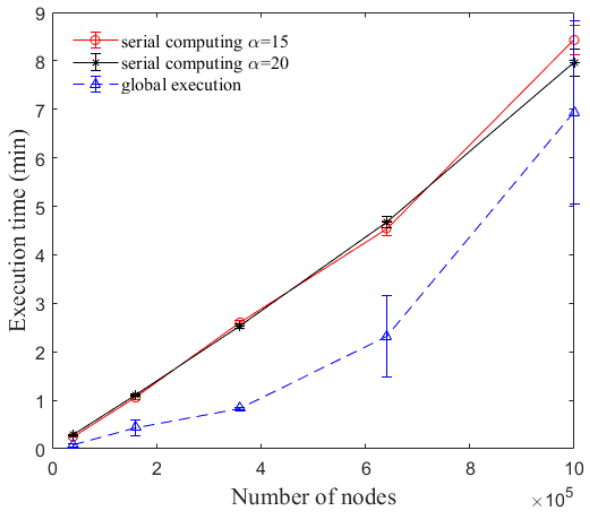
assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
15	200×200	0.25	0.24	0.26	[0.25,0.25]
	400×400	1.06	0.97	1.18	[1.02,1.09]
	600×600	2.61	2.49	2.74	[2.57,2.65]
	800×800	4.55	4.29	5.38	[4.39,4.70]
	1000×1000	8.43	7.41	9.63	[8.12,8.74]
	1200×1200	22.51	-	-	-
	1800×1800	52.09	-	-	-
	2200×2200	104.06	-	-	-
	2600×2600	184.25	-	-	-
	3000×3000	333.56	-	-	-
20	200×200	0.28	0.24	0.30	[0.27,0.29]
	400×400	1.11	0.97	1.17	[1.08,1.13]
	600×600	2.53	2.29	2.64	[2.48,2.57]
	800×800	4.68	4.27	4.98	[4.57,4.80]
	1000×1000	7.97	7.01	8.82	[7.69,8.24]
	1200×1200	23.13	-	-	-
	1800×1800	51.73	-	-	-
	2200×2200	103.33	-	-	-
	2600×2600	183.77	-	-	-
	3000×3000	332.85	-	-	-
Global	200×200	0.07	0.05	0.33	[0.04,0.11]
	400×400	0.44	0.31	1.78	[0.28,0.59]
	600×600	0.83	0.81	1.05	[0.81,0.86]
	800×800	2.32	1.88	10.01	[1.47,3.17]
	1000×1000	6.94	4.34	15.60	[5.05,8.82]
	1200×1200	20.16	-	-	-
	1800×1800	136.54	-	-	-
	2200×2200	337.67	-	-	-
	2600×2600	495.49	-	-	-
	3000×3000	743.20	-	-	-

Table 5 Peak memory consumption of serial computing (Binary-SD) (MB)

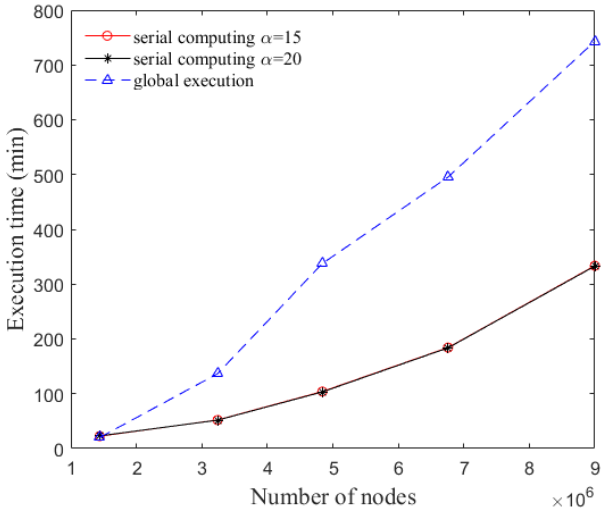
assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
15	200×200	0.33	0.13	1.18	[0.17,0.48]
	400×400	0.37	0.14	1.28	[0.22,0.52]
	600×600	2.88	2.88	2.89	[2.88,2.89]
	800×800	5.13	5.13	5.13	[5.13,5.13]
	1000×1000	8.01	8.01	8.01	[8.01,8.01]
	1200×1200	15.70	-	-	-
	1800×1800	27.25	-	-	-
	2200×2200	40.34	-	-	-
	2600×2600	54.17	-	-	-
	3000×3000	72.90	-	-	-
20	200×200	0.16	0.10	0.26	[0.14,0.18]
	400×400	0.20	0.13	0.53	[0.14,0.27]
	600×600	2.89	2.88	2.89	[2.89,2.89]
	800×800	5.13	5.13	5.13	[5.13,5.13]
	1000×1000	8.01	8.00	8.01	[8.00,8.01]
	1200×1200	15.96	-	-	-
	1800×1800	25.96	-	-	-
	2200×2200	39.31	-	-	-
	2600×2600	54.17	-	-	-
	3000×3000	72.90	-	-	-
Global	200×200	36.71	15.90	329.20	[2.86,70.56]
	400×400	171.47	85.48	545.95	[89.07,253.86]
	600×600	200.62	200.62	200.62	[200.62,200.62]
	800×800	492.94	405.82	2146.32	[310.81,675.08]
	1000×1000	984.79	721.67	2032.82	[733.19,1236.40]
	1200×1200	1477.58	-	-	-
	1800×1800	7197.88	-	-	-
	2200×2200	9325.27	-	-	-
	2600×2600	15676.92	-	-	-
	3000×3000	20889.42	-	-	-

Figure 13 shows the execution time and peak memory consumption comparison between ELS with different assigned-αs and global MCF algorithm. Figures 13 (a) and (b) are the execution time comparison, while Figures 13 (c) and (d) are the peak memory comparison. Figure 13 (a) indicates that when the network scale is smaller than 1000×1000 nodes in this experiment, ELS uses longer execution time to obtain the MCF optimal solution of the entire network than global MCF algorithm. However, when the network scale is larger than 1200×1200

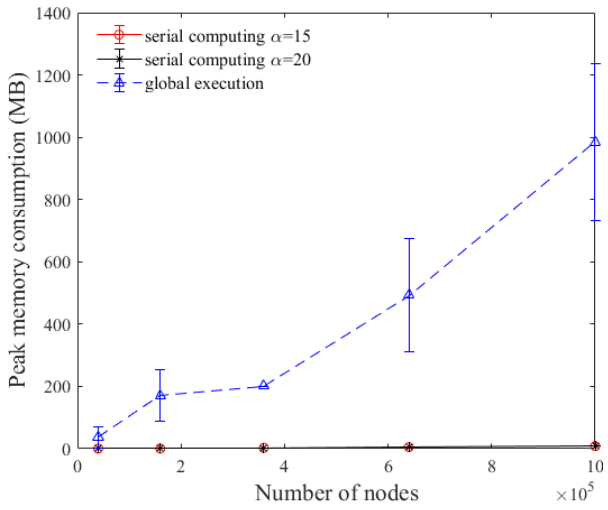
nodes, in Figure 13 (b), it shows that ELS starts to save execution time. As to the peak memory consumption, regardless of whether the network scale is smaller than 1000×1000 nodes (shown in Figure 13 (c)) or larger than 1000×1000 nodes (shown in Figure 13 (d)), ELS always performs better than the global MCF algorithm.



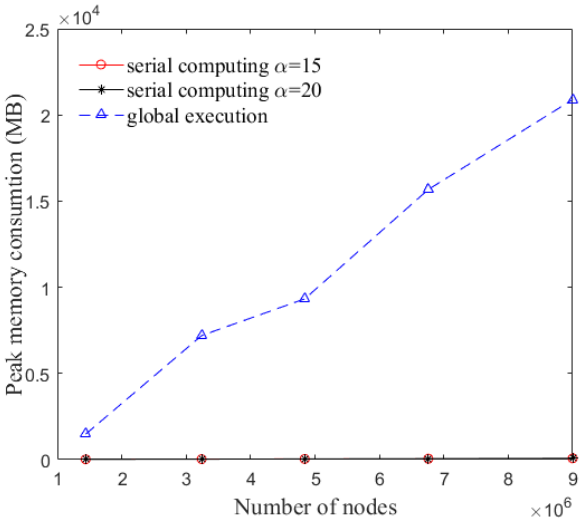
(a) Execution time comparison (less than 1000×1000 nodes)



(b) Execution time comparison (greater than 1000×1000 nodes)



(c) Peak memory consumption comparison (less than 1000×1000 nodes)



(d) Peak memory consumption comparison (greater than 1000×1000 nodes)

Figure 13 Performance comparison between ELS and global MCF (Binary-SD)

The flow cost of the entire network obtained from the objective function value of ELS is recorded in Table 6, as well as the flow cost directly obtained from the objective function of the global MCF algorithm. Since this experiment is conducted on simulated networks, the flow cost can be generalized cost with corresponding cost unit (i.e., unit of currency or unit of time). The comparison between the objective value obtained by ELS and the global MCF algorithm is shown in Figure 14. From the results shown in Table 6 and Figure 14, we can tell that when assigned-α is equal to 15 or 20, the objective values obtained by ELS are very close to the global objective value, which indicates that the ELS does not sacrifice much accuracy in this experiment.

Table 6 The flow cost of the entire network (Binary-SD)

Assigned-α	Network scale	ELS objective value		
		Mean value	Min-value	Max-value
15	200×200	3157	3022	3239
	400×400	12639	12348	12853
	600×600	28501	28102	28857
	800×800	50647	50107	51077
	1000×1000	79135	78719	79572
	1200×1200	983328	-	-
	1800×1800	1626493	-	-
	2200×2200	2427373	-	-
	2600×2600	3394014	-	-
	3000×3000	4518181	-	-
20	200×200	3157	3022	3239
	400×400	12639	12348	12853
	600×600	28501	28102	28857
	800×800	50647	50107	51077
	1000×1000	79135	78724	79572
	1200×1200	983328	-	-
	1800×1800	1626493	-	-
	2200×2200	2427373	-	-
	2600×2600	3394018	-	-
	3000×3000	4518188	-	-
		Global objective value		
	Network scale	Mean value	Min-value	Max-value
	200×200	3156	3022	3238
	400×400	12638	12348	12851
	600×600	28500	28102	28851
	800×800	50645	50107	51075
	1000×1000	79132	78719	79566
	1200×1200	983150	-	-
	1800×1800	1626241	-	-
	2200×2200	2426964	-	-
	2600×2600	3393461	-	-
	3000×3000	4517428	-	-

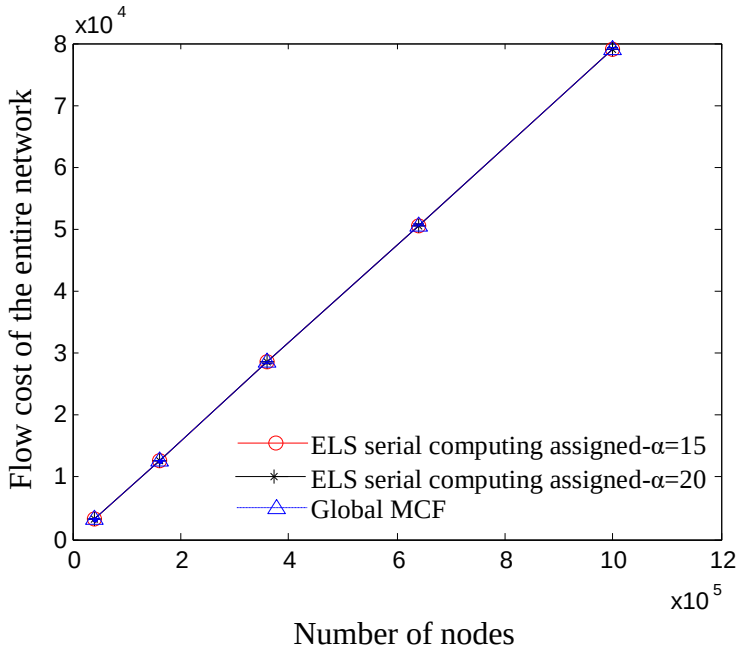


Figure 14 The flow cost of the entire network comparison (Binary-SD)

Since the package of the interior-point algorithm in MATLAB is applied to obtain the global MCF solution, the global objective value as well as the corresponding execution time and peak memory consumption are treated as benchmarks in this experiment. This experiment is conducted on a grid network with binary SD setting rather than a realistic dataset; the performance between different assigned-αs could be more apparent in a noisier test case.

Experiment Two: Effectiveness Testing with Multiple Supplies and Demands

The aim of the second experiment is to measure the performance of ELS with multiple supplies and demands (Multi-SD) assigned to the nodes of the network. The execution time and the peak memory consumption are compared between the global solution method and ELS.

Simulation Setting.

The general idea of the second experiment is similar to the first one but the ***n*** units of demand and supply are randomly assigned to nodes, while the total demand and total supply of the entire network is balanced. The test problem instances are generated as grid networks with

increasing scales (i.e., the network scale is from 200 × 200 to 1000 × 1000 with an increment of 200 rows and 200 columns), during this experiment. The unit cost of each direct link (i.e., c_{ij} in Equation (3.1)) in each network is simulated as an un-weighted cost in this experiment as well. For each scale of the network, the demand and supply is randomly assigned on each node with *n* units commodity (i.e., +*n* indicates *n* units of supply, -*n* indicates *n* units of demand, and zero indicates the node is a transshipment node), while the total demand and total supply is balanced.

The ODLS software is applied to generate the supply and demand nodes of different scales of networks in this experiment as well. Table 7 shows the number of supply and demand nodes for the different scales of networks generated for this experiment in the multiple supplies and demands setting. Figure 15 demonstrates the supply and demand of nodes generation information with the scales of networks. Figure 16 shows two examples of supply and demand of nodes corresponding to difference network scales, which are 200×200 nodes and 1000×1000 nodes displayed in (a) and (b) respectively. In Figure 16, as displayed by the color bar, pixels with warm color (positive integers) indicate supply nodes, pixels with cold color (negative integers) indicate demand nodes and the green pixels indicate transshipment nodes. The maximum absolute value of demand or supply amount can be chosen as a certain integer number as the boundary value in ODLS.

Table 7 Number of supply and demand nodes for different scales of networks (Multi-SD)

Network Scale	Mean value	Min-value	Max-value
200×200	4793	1380	9305
400×400	19417	5817	37285
600×600	43695	13218	84020
800×800	77615	23639	149105
1000×1000	121115	37100	232530

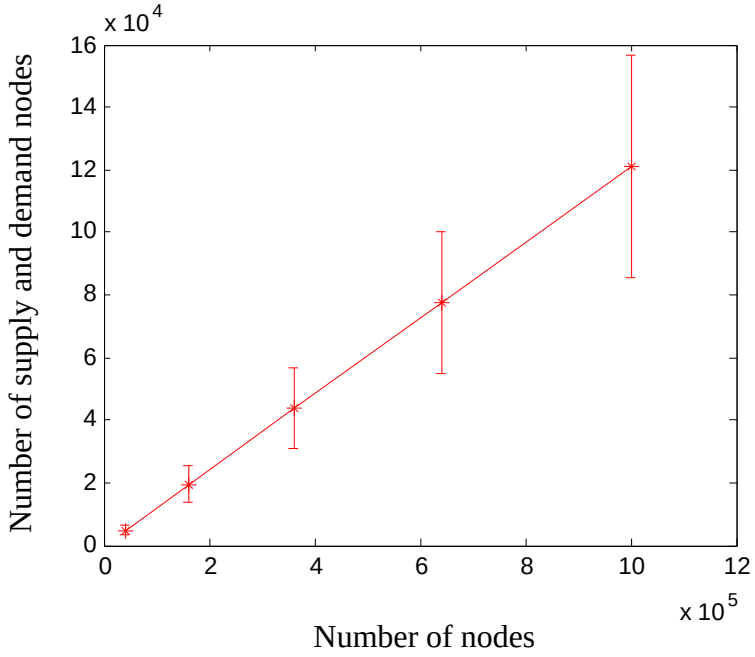
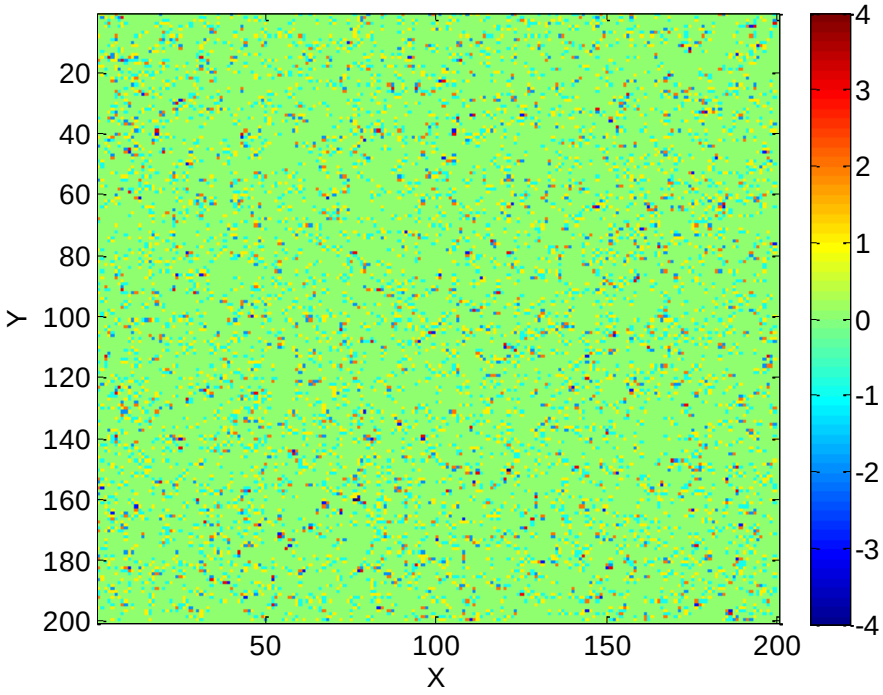
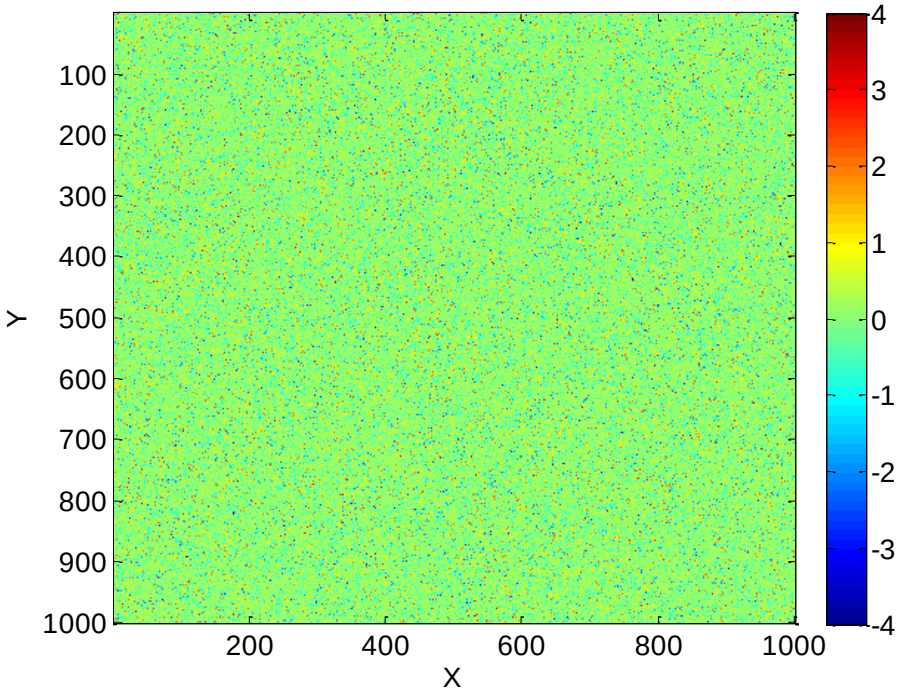


Figure 15 Number of supply and demand nodes for difference scales of networks (Multi-SD)



(a) supply and demand of nodes with network scale 200×200



(b) supply and demand of nodes with network scale 1000×1000

Figure 16 Examples of supply and demand of nodes (Multi-SD)

Experiment Two Process and Results.

In this experiment, for each scale of simulated grid network, multiple supplies or demands for each node are randomly generated 20 times. The range of allowance value is set as 0~4. Each time, the AC tiling strategy is applied to obtain the clustering result of all the nodes (i.e., sub-network information) with different assigned-α values from $\alpha = 20$ to $\alpha = 45$, where the minimum unit cost between two nodes is approximately substituted by the distance between the pair of nodes. Then, the actual-α value is calculated to compare with the assigned-α value, through the same process as described in the first experiment. The results are summarized in Table 8. When assigned-α equals 20, the original network is clustered into one cluster, which provides no benefit over the global solution approach. On the other hand, when the assigned-α is greater than 40, the clustering results change little compared to assigned-$\alpha = 40$. Therefore, in the

subsequent part of this experiment, testing was only conducted on assigned-α=35 and assigned-α=40.

Table 8 Assigned-α selection test results (Multi-SD)

assigned-α	Network Scale	Execution time (min)	Peak memory consumption (MB)	Number of clusters	Objective value
20	200×200	0.14	17.13	1	6753
	400×400	0.88	85.48	1	27582
	600×600	3.00	200.61	1	61720
	800×800	8.83	405.82	1	109461
	1000×1000	26.10	722.21	1	170705
35	200×200	0.36	0.15	1962	6756
	400×400	1.57	0.16	8030	27591
	600×600	3.88	4.33	17954	61732
	800×800	7.98	7.70	31712	109483
	1000×1000	21.41	12.03	49243	170705
40	200×200	0.36	0.19	1962	6756
	400×400	1.55	0.68	8030	27591
	600×600	3.81	4.33	17954	61732
	800×800	8.17	7.70	31712	109483
	1000×1000	16.14	12.02	49243	170747
45	200×200	0.36	0.15	1962	6756
	400×400	1.50	0.78	8030	27591
	600×600	4.03	4.33	17954	61732
	800×800	8.48	7.69	31712	109483
	1000×1000	16.95	12.02	49270	170749

In Table 8, the values are similar for performance of the ELS with increasing assigned-α as in experiment one. This is likely due to the simple geometry of the generated grid structure. The differences in performance and efficiency are expected to become more apparent in realistic test cases.

For each assigned-α, after running the test 20 times on each scale of network, the value of each actual-α is documented. In this experiment, it is assumed that the value of actual-α has an approximate normal distribution. Therefore, according to the results of the 20 replicates, statistical results of actual-α are produced in Table 9. From Table 9 and Figure 17 we can see

that, although the assigned-α are 35 or 40, the actual-α is not much bigger than one. Figure 18 shows the number of clusters obtained with the different assigned-αs.

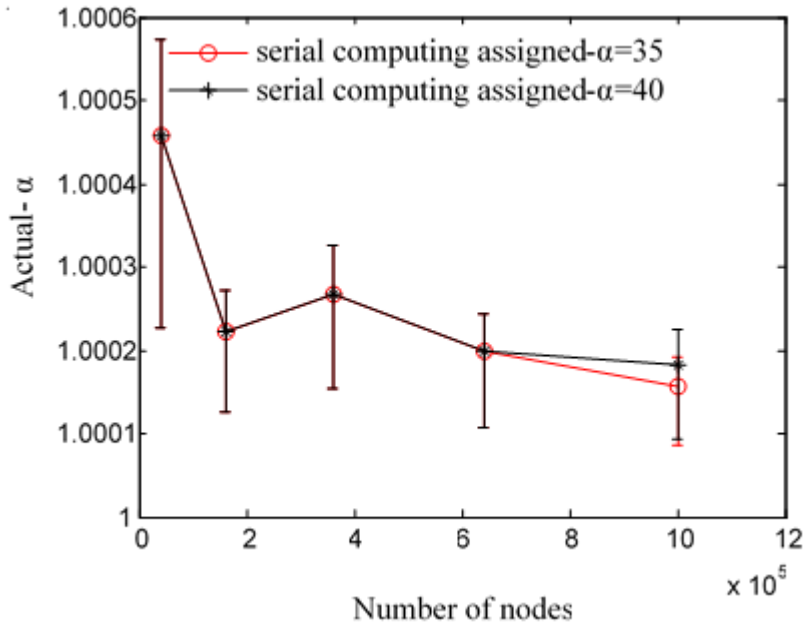


Figure 17 The statistical result of actual-α (Multi-SD)

Table 9 Statistical results of actual-α (Multi-SD)

assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
35	200×200	1.000458	1.000000	1.000779	[1.000227,1.000574]
	400×400	1.000223	1.000000	1.000321	[1.000127,1.000271]
	600×600	1.000268	1.000000	1.000196	[1.000154,1.000325]
	800×800	1.000198	1.000000	1.000269	[1.000108,1.000244]
	1000×1000	1.000156	1.000000	1.000265	[1.000085,1.000191]
40	200×200	1.000458	1.000000	1.000779	[1.000227,1.000574]
	400×400	1.000223	1.000000	1.000321	[1.000127,1.000271]
	600×600	1.000268	1.000000	1.000196	[1.000154,1.000325]
	800×800	1.000198	1.000000	1.000269	[1.000108,1.000244]
	1000×1000	1.000182	1.000000	1.000265	[1.000094,1.000226]

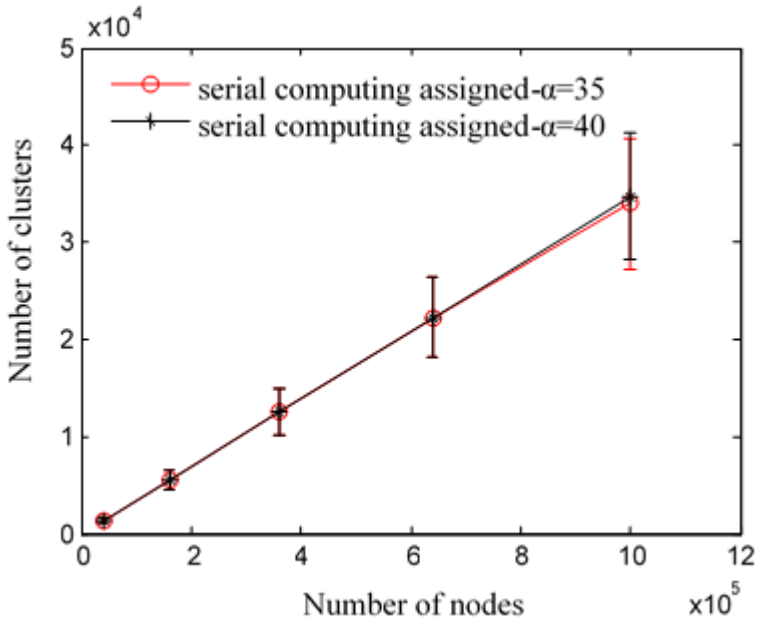


Figure 18 The statistical result of number of clusters (Multi-SD)

During the process of this experiment, for each assigned-α, the execution time and peak memory consumption of the global MCF algorithm and of the ELS, at each scale of the testing network are recorded. These results are provided in Tables 10 and 11.

Table 10 Execution time of serial computing (Multi-SD) (min)

assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
35	200×200	0.25	0.11	0.37	[0.20,0.29]
	400×400	1.02	0.44	1.58	[0.81,1.22]
	600×600	2.45	1.00	3.92	[1.93,2.98]
	800×800	5.06	2.02	8.48	[3.91,6.21]
	1000×1000	9.16	3.59	16.01	[6.93,11.39]
40	200×200	0.25	0.11	0.40	[0.20,0.30]
	400×400	1.01	0.44	1.55	[0.80,1.21]
	600×600	2.43	0.99	3.81	[1.91,2.94]
	800×800	5.02	1.96	8.19	[3.89,6.15]
	1000×1000	9.47	3.48	16.14	[7.25,1.70]
Global	200×200	0.12	0.05	0.49	[0.06,0.17]
	400×400	0.54	0.29	1.31	[0.37,0.71]
	600×600	1.54	0.76	7.11	[0.81,2.28]
	800×800	2.96	1.84	7.98	[2.07,3.85]
	1000×1000	6.69	4.14	15.66	[5.04,8.34]

Table 11 Peak memory consumption of serial computing (Multi-SD) (MB)

assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
35	200×200	0.16	0.14	0.18	[0.15,0.16]
	400×400	0.20	0.15	0.52	[0.14,0.25]
	600×600	4.33	4.33	4.33	[4.33,4.33]
	800×800	7.70	7.69	7.70	[7.70,7.70]
	1000×1000	12.02	12.01	12.02	[12.02,12.02]
40	200×200	0.15	0.15	1.96	[0.20,0.72]
	400×400	0.14	0.14	1.46	[0.39,0.76]
	600×600	4.33	4.33	4.33	[4.33,4.33]
	800×800	7.69	7.69	7.70	[7.70,7.70]
	1000×1000	12.01	12.01	12.02	[12.02,12.02]
Global	200×200	32.10	15.90	282.64	[3.96,60.24]
	400×400	171.46	85.30	504.65	[103.71,239.20]
	600×600	298.24	200.62	688.87	[204.49,392.00]
	800×800	532.79	405.81	1252.55	[387.65,677.93]
	1000×1000	1123.45	722.02	3505.01	[761.02,1485.88]

Figure 19 shows the execution time and peak memory consumption comparison between ELS with different assigned-αs and the global MCF algorithm. Figure 19 (a) indicates that ELS uses longer execution time to obtain the MCF optimal solution of the entire network than the global MCF algorithm. However, as to the peak memory consumption shown in Figure 19 (b), it indicates that ELS performs better than the global MCF algorithm.

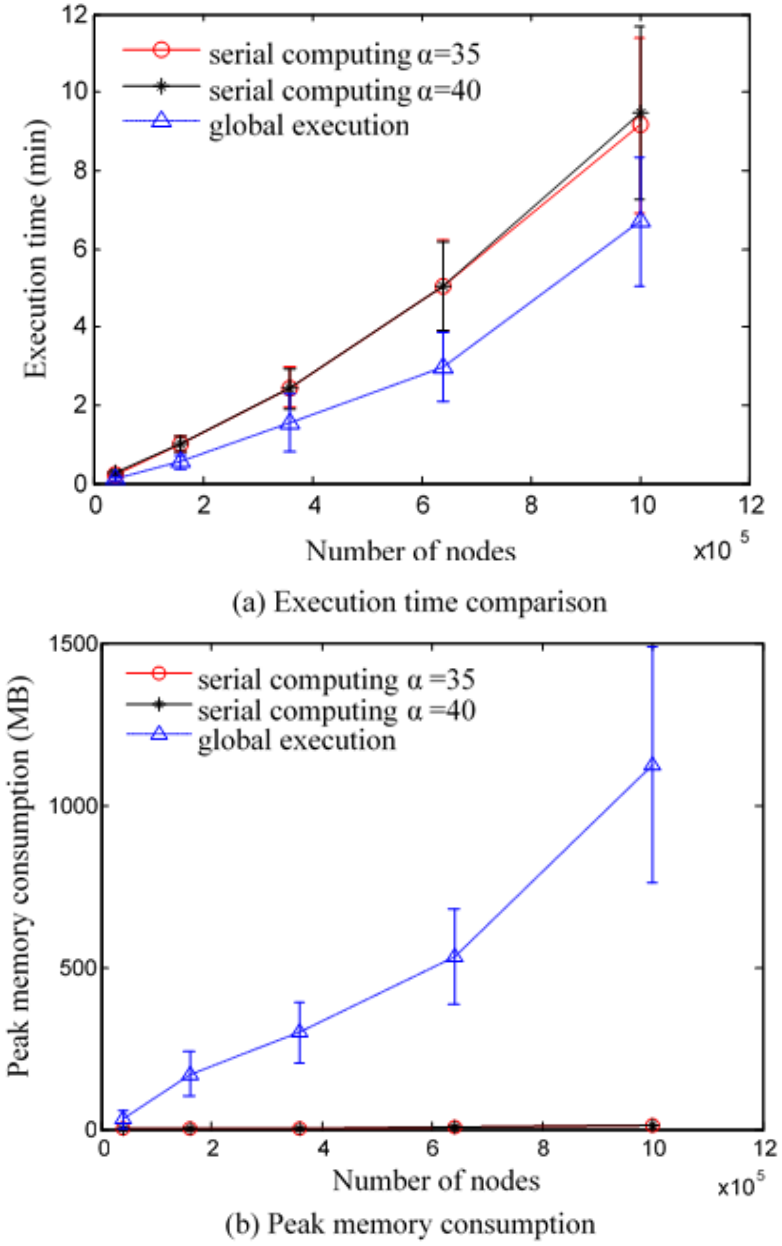


Figure 19 Performance comparison between ELS and global MCF (Multi-SD)

The flow cost of the entire network obtained from the objective function value of ELS is recorded in Table 12, as well as the flow cost directly obtained from the objective function of the global MCF algorithm (i.e., the interior-point algorithm). Since this experiment is conducted on simulated networks, the flow cost can be generalized cost with corresponding cost unit (i.e., unit of currency or unit of time). The comparison between the objective value obtained by ELS and global MCF algorithm is shown in Figure 20. From the results shown in Table 12 and Figure 20, we can tell that when assigned-α equals to 35 or 40, the objective values obtained by ELS are very close to the global objective value, which indicates that the ELS does not sacrifice much accuracy in this experiment.

Table 12 The flow cost of the entire network (Multi-SD)

		ELS objective value		
Assigned-α	Network scale	Mean value	Min-value	Max-value
35	200×200	3239	831	6914
	400×400	13071	3469	27591
	600×600	29392	7928	61981
	800×800	52179	14133	110330
	1000×1000	81471	22222	172190
40	200×200	3239	831	69144
	400×400	13071	3469	27591
	600×600	29392	7928	61981
	800×800	52179	14133	110330
	1000×1000	81473	22222	172190
		Global objective value		
	Network scale	Mean value	Min-value	Max-value
	200×200	3237	831	6909
	400×400	13069	3469	27582
	600×600	29384	7928	61969
	800×800	52168	14133	110300
	1000×1000	81458	22222	172140

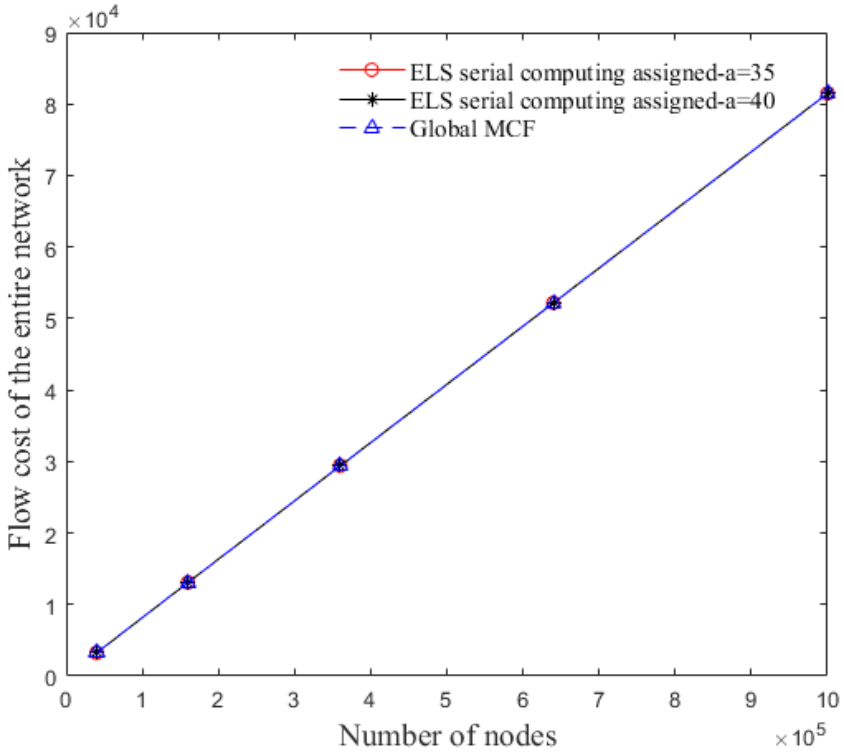


Figure 20 The flow cost of the entire network comparison (Multi-SD)

Experiment Three: Performance Examination of Parallel Computing Application

The first and second experiments are both performed in a serial computing environment. It is worth mentioning that one of the merits brought by the AC tiling strategy is offering the opportunity to utilize parallel computing in the large-scale problem solving process. Therefore, the third experiment is conducted to compare the performance between serial and parallel computing for the large-scale MCF problem.

Experiment Setting.

This parallel computing performance experiment is performed by utilizing the ‘parfor’ loop. A ‘parfor’-loop in MATLAB executes a series of statements in the loop body in parallel. The MATLAB client issues the ‘parfor’ command and coordinates with MATLAB workers to

execute the loop iterations in parallel on the workers in a parallel pool. The client sends the necessary data on which ‘parfor’ operates to workers, where most of the computation is executed. The results are sent back to the client and assembled. A ‘parfor’-loop can provide significantly better performance than its analogous for-loop, because several MATLAB workers can compute simultaneously on the same loop. Each execution of the body of a ‘parfor’-loop is an iteration. MATLAB workers evaluate iterations in no particular order and independently of each other. Because each iteration is independent, there is no guarantee that the iterations are synchronized in any way, nor is there any need for this. If the number of workers is equal to the number of loop iterations, each worker performs one iteration of the loop. If there are more iterations than workers, some workers perform more than one loop iteration. In this case, a worker might receive multiple iterations at once to reduce communication time. It can be seen that ‘parfor’ only can be applied on the data structure that can be totally parallelized. Therefore in this experiment, the ‘parfor’ is applied on the MCF problem solving step after the original network has been clustered by AC tiling strategy. The parallel computing process of this experiment proceeds on a workstation with 12 cores, 3.4- GHz CPU frequency, 64 bit operating system, 128G memory.

Experiment Three Process and Result.

The first part of this experiment is conducted under the binary supply and demand setting. Table 13 presents the parallel computing execution time by applying different values of assigned-α (i.e., assigned-α = 15 and assigned-α = 20), with increasing sizes of network, which have the same input information as that of the previous serial computing experiment. Table 14 presents the peak memory consumption of parallel computing with different assigned-α, corresponding to increasing network scales.

Table 13 Execution time of parallel computing (Binary-SD) (min)

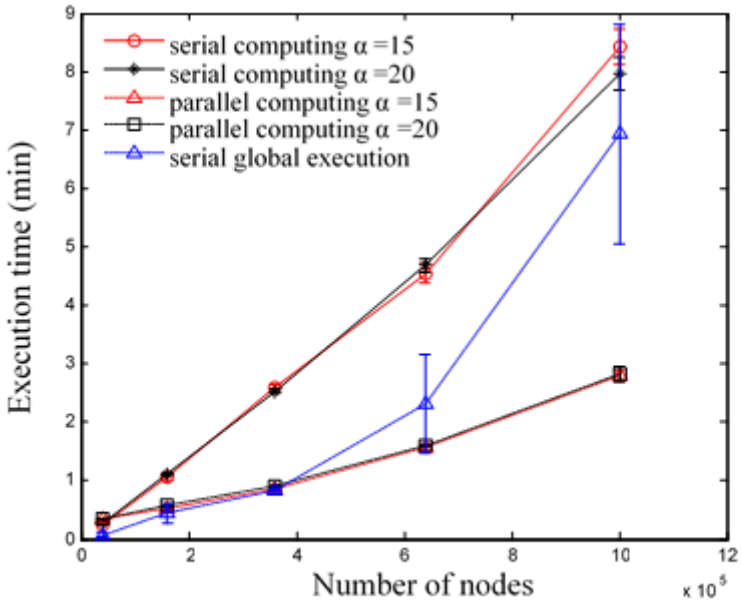
assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
15	200×200	0.34	0.33	0.35	[0.34,0.34]
	400×400	0.52	0.51	0.53	[0.52,0.52]
	600×600	0.87	0.86	0.89	[0.86,0.87]
	800×800	1.56	1.52	1.78	[1.53,1.59]
	1000×1000	2.79	2.62	3.16	[2.7,2.89]
20	200×200	0.34	0.34	0.39	[0.34,0.35]
	400×400	0.57	0.52	0.61	[0.56,0.59]
	600×600	0.90	0.89	0.91	[0.90,0.90]
	800×800	1.60	1.56	1.79	[1.58,1.62]
	1000×1000	2.83	2.67	3.58	[2.71,2.96]

Table 14 Peak memory consumption of parallel computing (Binary-SD) (MB)

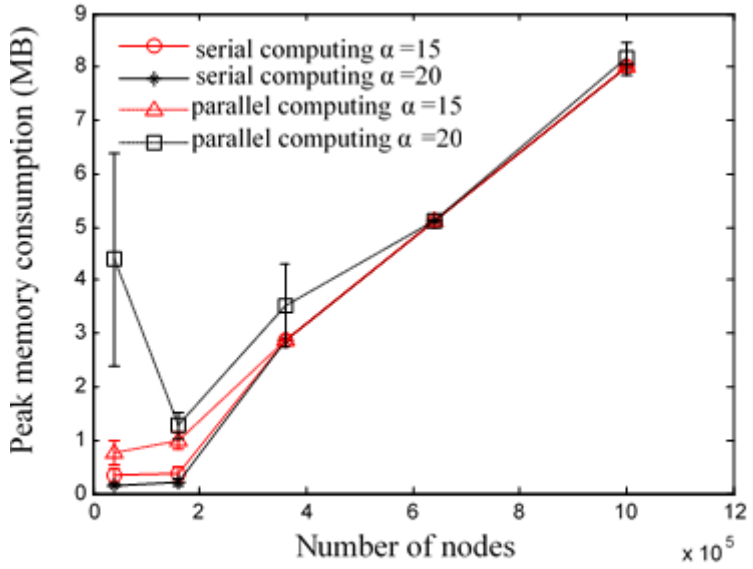
assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
15	200×200	0.77	0.21	1.97	[5.34,9.97]
	400×400	1.00	0.21	1.29	[8.34,11.70]
	600×600	2.89	2.88	2.89	[28.85,28.88]
	800×800	5.13	5.12	5.13	[51.27,51.29]
	1000×1000	8.00	7.95	8.02	[79.97,80.13]
20	200×200	4.39	2.74	144.14	[2.39,6.38]
	400×400	1.27	3.73	26.26	[1.03,1.50]
	600×600	3.52	28.84	103.38	[2.73,4.30]
	800×800	5.12	50.50	51.28	[5.11,5.13]
	1000×1000	8.16	80.08	109.24	[7.85,8.46]

Figure 21 displays the performance comparison between serial and parallel computing. Figure 21 (a) compares the execution time between serial computing and parallel computing with different assigned-α values while the network scale is increasing, as well as the global MCF execution process as a base case. It shows that when the network scale is small, (i.e., 200×200 nodes) parallel computing has longer execution time than serial computing. The possible reason for this phenomena is the data communication between ‘workers’ and data initialization take much time for the parallel computing compared to serial computing. As the network scale gets larger, parallel computing shows its advantage on saving execution time. Comparing to the global MCF process, parallel computing shows its advantage on saving execution time when the network scale is greater than 600×600 nodes.

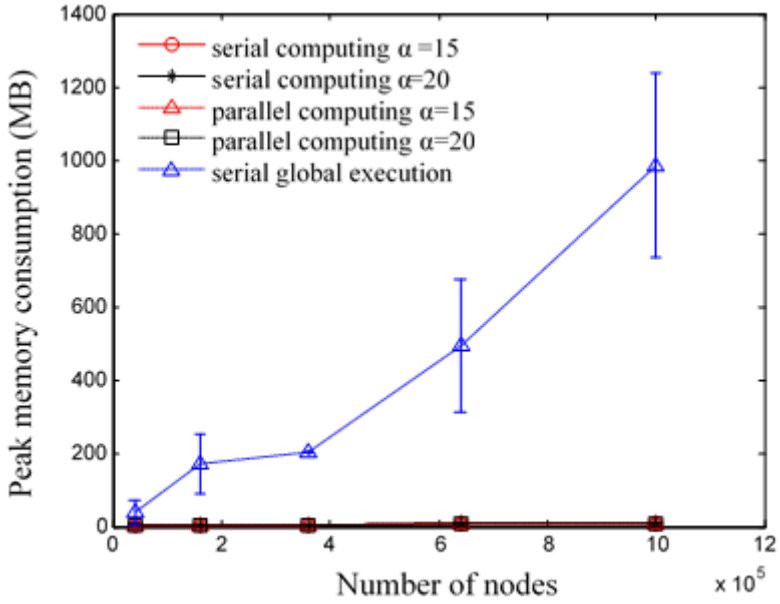
Figure 21 (b) compares the peak memory consumption between serial computing and parallel computing with different assigned-α while the network scale is increasing. It shows that the parallel computing is not stable when the network scale is small (i.e., 200×200 nodes in this experiment), which may be due to the computation resource initialization. It is also noticed that in this experiment the serial computing has less or similar peak memory consumption compared to the parallel computing. Figure 21 (c) added the peak memory consumption of serial global execution to compare with both serial and parallel computing. It shows that both serial and parallel computing has less peak memory consumption than serial global MCF process.



(a) Execution time comparison



(b) Peak memory consumption comparison



(c) Peak memory consumption comparison (including serial global execution)

Figure 21 Comparison between serial and parallel computing (Binary-SD)

To demonstrate the performance difference between serial and parallel computing, Figure 22 shows the performance ratios of them, when assigned-α = 15. The blue line is the ratio of the difference between the parallel and serial computing peak memory consumption to the serial computing peak memory consumption, while the red line is the ratio of the difference between

the serial and parallel computing execution time to the serial computing execution time. When the network scale is equal to or greater than 600×600 nodes the performance ratio between parallel computing and serial computing becomes stable.

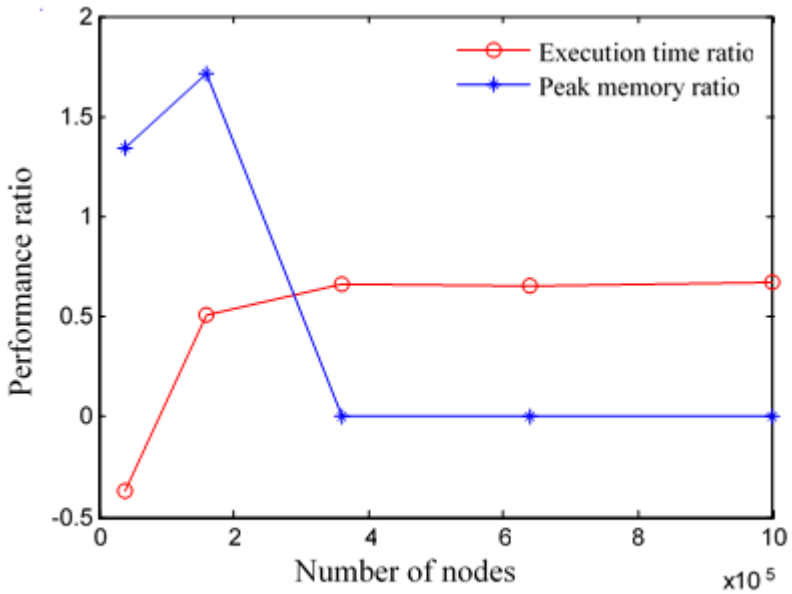


Figure 22 The performance ratio between serial computing and parallel computing (Binary-SD)

Figure 23 shows the performance ratio between parallel computing and serial global MCF process (refer to Function X.1 in Appendices). The trend indicates that when the network scale is equal to or greater than 600×600 nodes, parallel computing uses much less execution time compared to global MCF process.

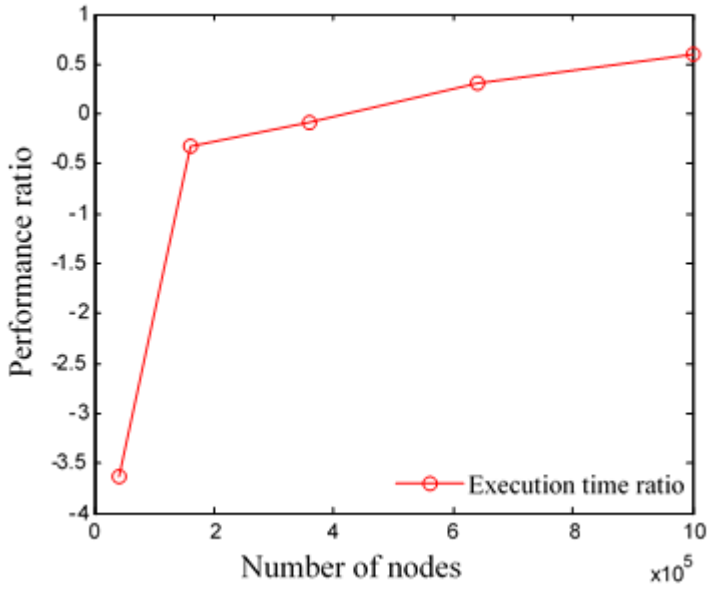


Figure 23 The performance ratio between parallel computing and serial global execution (Binary-SD)

The second part of this experiment is conducted under the multiple supplies and demands setting. Table 15 presents the parallel computing execution time by applying different values of assigned-α (i.e., assigned-α = 35 and assigned-α = 40), with increasing scales of network, which have the same input information as for the previous serial computing experiment. Table 16 presents the peak memory consumption of parallel computing with different assigned-α, corresponding to increasing network scales.

Table 15 Execution time of parallel computing (Multi-SD) (min)

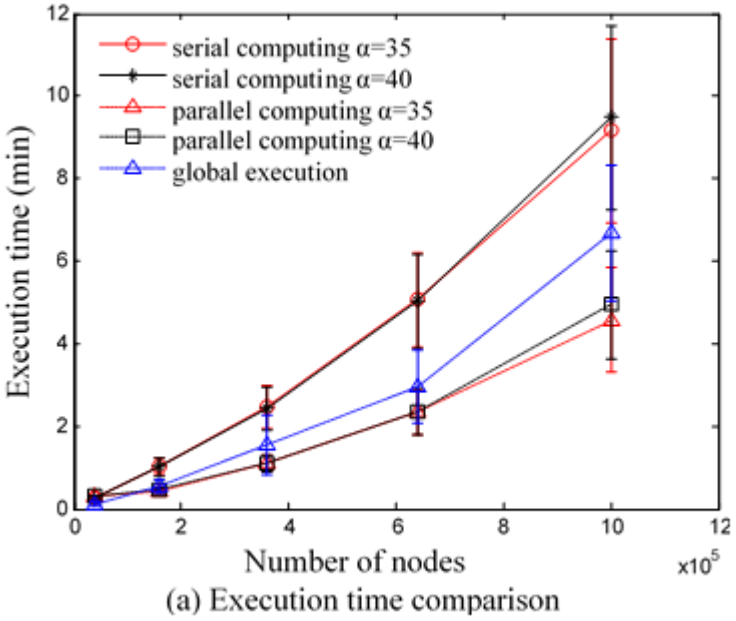
assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
35	200×200	0.29	0.26	0.32	[0.28,0.30]
	400×400	0.42	0.32	0.55	[0.38,0.46]
	600×600	1.10	0.63	1.72	[0.89,1.30]
	800×800	2.33	1.04	4.07	[1.78,2.88]
	1000×1000	4.56	1.75	8.88	[3.29,5.82]
40	200×200	0.30	0.27	0.33	[0.29,0.31]
	400×400	0.45	0.32	0.63	[0.40,0.50]
	600×600	1.11	0.65	1.73	[0.92,1.31]
	800×800	2.35	1.06	4.08	[1.80,2.89]
	1000×1000	4.94	1.86	9.05	[3.65,6.24]

Table 16 Peak memory consumption of parallel computing (Multi-SD) (MB)

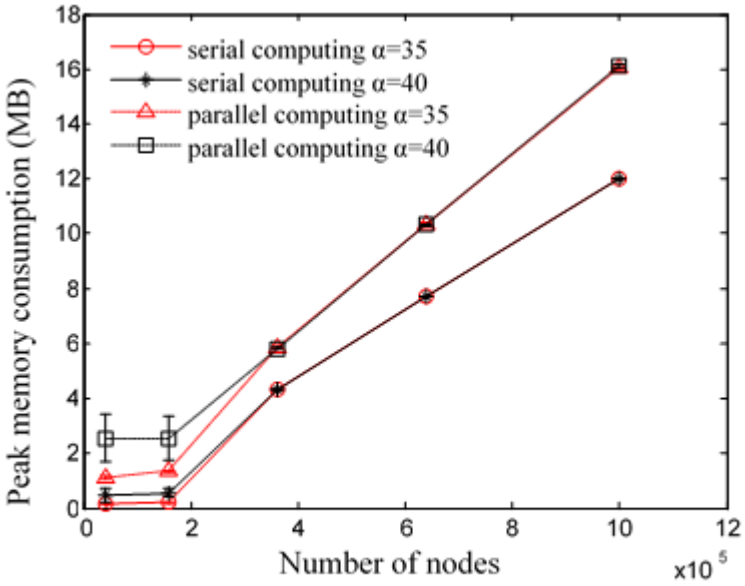
assigned-α	Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
35	200×200	1.11	0.76	1.54	[1.03,1.19]
	400×400	1.37	1.29	1.52	[1.34,1.39]
	600×600	5.87	5.72	6.41	[5.80,5.95]
	800×800	10.34	10.21	10.57	[10.29,10.39]
	1000×1000	16.05	15.96	16.14	[16.01,16.08]
40	200×200	2.54	0.93	8.41	[1.67,3.41]
	400×400	2.56	1.29	7.00	[1.77,3.34]
	600×600	5.81	5.71	5.93	[5.78,5.85]
	800×800	10.32	10.22	10.57	[10.27,10.37]
	1000×1000	16.12	15.98	16.19	[16.07,16.16]

Figure 24 displays the performance comparison between serial and parallel computing. Figure 24(a) compares the execution time between serial computing and parallel computing with different assigned-α while the network scale is increasing. The execution time of global MCF program is also displayed in Figure 24 (a) as a base case. Parallel computing shows its advantage on saving execution time in this experiment result. Figure 24 (b) compares the peak memory consumption between serial computing and parallel computing with different assigned-αs while the network scale is increasing. It shows that serial computing has less peak memory

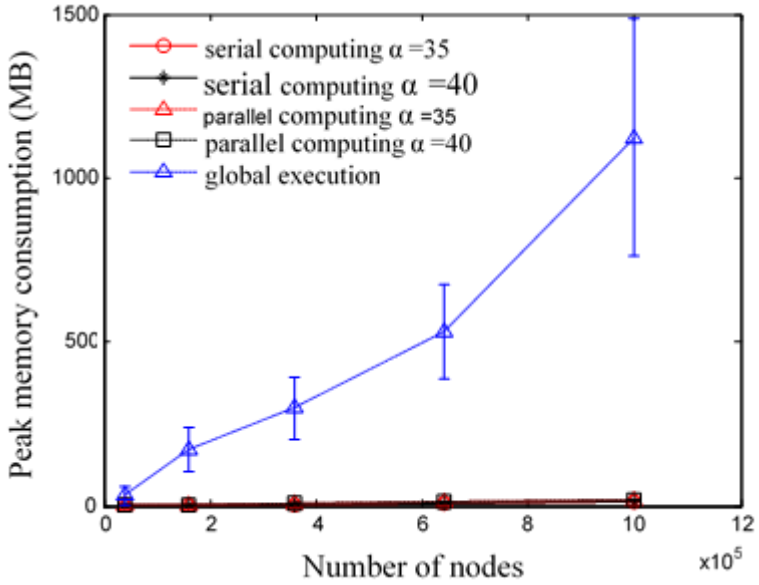
consumption in this experiment. Figure 24 (c) added the peak memory consumption of serial global execution to compare with both serial and parallel computing. It shows that both serial and parallel computing have less peak memory consumption than global MCF process.



(a) Execution time comparison



(b) Peak memory consumption comparison



(c) Peak memory consumption comparison (including serial global execution)

Figure 24 Comparison between serial and parallel computing (Multi-SD)

Figure 25 shows the performance ratios of the difference between serial and parallel computing relative to serial computing (assigned-α = 35). The blue line is the ratio of the difference between the parallel and serial computing peak memory consumption to the serial computing peak memory consumption, while the red line is the ratio of the difference between the serial and parallel computing execution time to the serial computing execution time. It shows that when the network scale is equal to or greater than 600×600 nodes, the performance ratio between parallel computing and serial computing becomes stable.

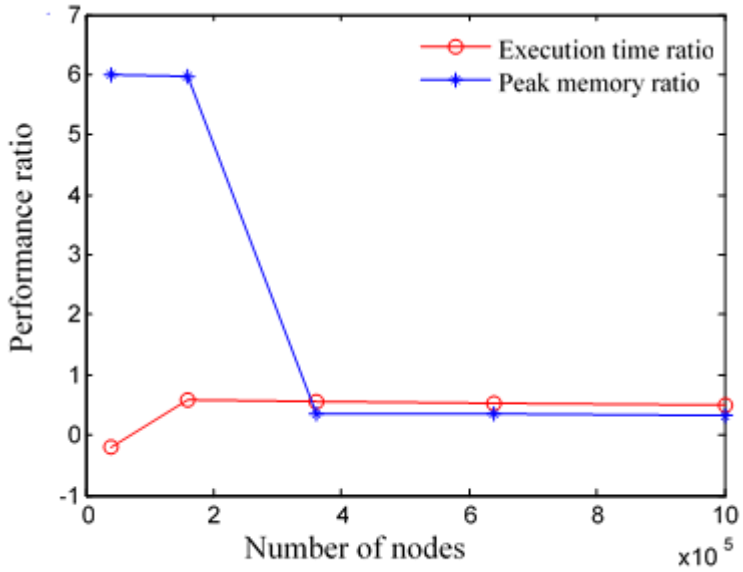


Figure 25 The performance ratio between serial computing and parallel computing (Multi-SD)

Figure 26 shows the performance ratio between parallel computing and serial global MCF process (refer to Function X.1 in Appendices). The trend indicates that when the network scale is equal to or greater than 400×400 nodes, the parallel computing uses much less execution time compared to the global MCF process.

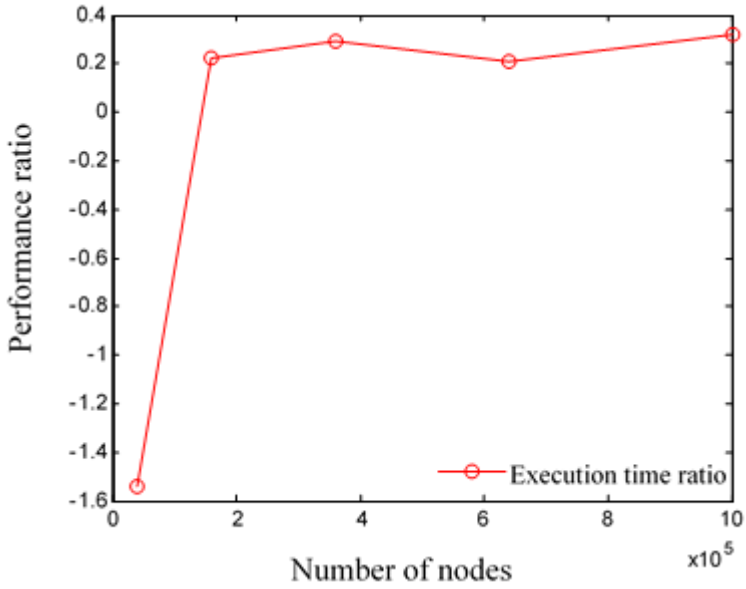


Figure 26 The performance ratio between parallel computing and serial global execution (Multi-SD)

Experiment Four: Comparison with Greedy-based Tiling Strategy

Since there is no current tiling strategy applied in solving large-scale MCF problems, this experiment is performed to compare ELS with a greedy-based tiling strategy. If there is no corresponding published existing algorithm or solution method, it is a common and basic criterion that a new algorithm or solution method is compared with the greedy-based algorithm or solution method design. The reason is that greedy-based algorithms or solution methods are straightforward algorithms or solution methods that are typically convenient to implement, and also are commonly used heuristics (Willianmson & Shmoys 2010). Carballo (2000) provides a greedy-based tiling strategy applied in the image processing field. In this experiment, the two-step greedy-based tiling strategy (GTS), which is a refined method for transportation applications, is utilized to perform the comparison. The detailed description of GTS follows.

The GTS is performed to compare with the proposed tiling strategy in this experiment. Specifically, in the first step, GTS evenly divides the input grid network into fixed-size and rectangular tiles (the size of each tile is chosen by the user). Then the MCF solution of each sub-network can be calculated within each tile. Note that the supply and demand in each sub-network may not be balanced since the original network is 'crudely' divided using proximity alone. We can achieve the MCF solution of each tile in step one by introducing a 'ground point' temporarily, i.e., allowing the supply and demand of each sub-network get satisfied as much as possible, while the unsatisfied part is temporarily fulfilled by the 'ground point'. The node with unsatisfied supply or demand in step one is called a residual supply or demand node. Figure 27 demonstrates the functionality of the ground point, using binary supply/demand as an example, which is also applicable to multiple supply and demand situations.

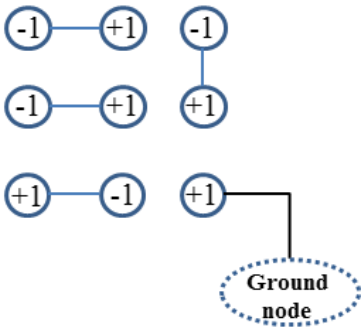


Figure 27 Demonstration of the ground point functionality

After the first step, we may have a number of residual supply or demand nodes from some of the sub-networks. In the second step, the residual supply or demand nodes are balanced by a simple star topology network, in which the central node of the original network will be connected with each of the residual supply or demand nodes by the shortest path. Then, the MCF solution of the residual nodes network is obtained. Figure 28 shows an example of the step two process. The MCF solutions of each sub-network in the first step and of the residual nodes network are combined together to achieve the MCF solution of the entire network.

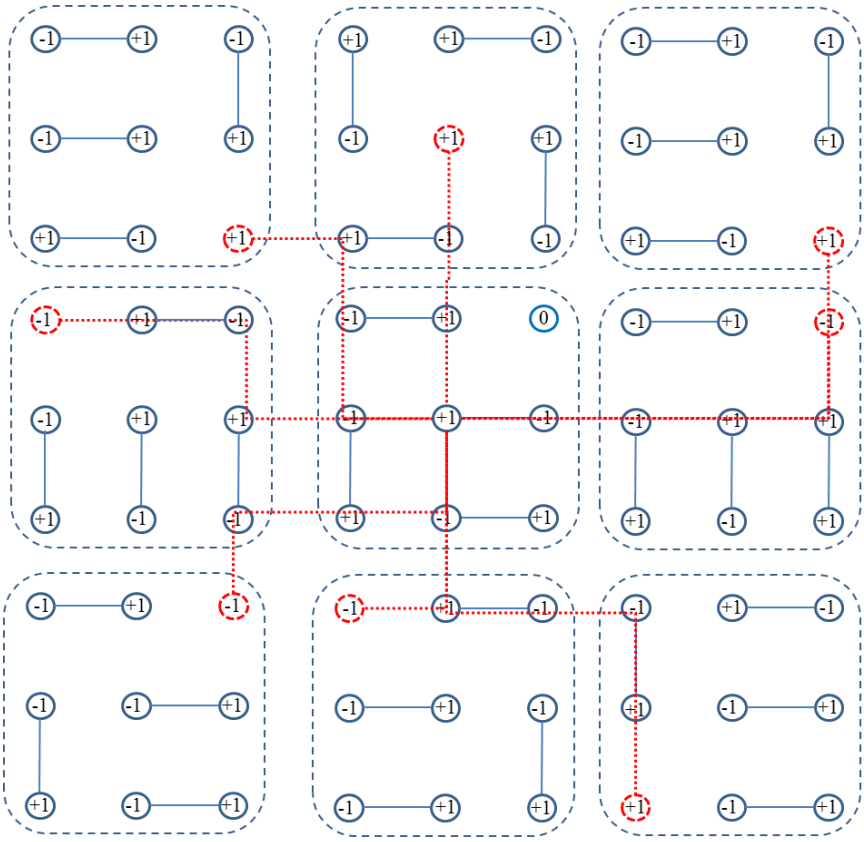


Figure 28 Demonstration of balancing residual supply/demand nodes

Simulation Setting.

Since this experiment proceeds to compare the proposed MCF tiling strategy with GTS, the same scales of networks as in the first experiment are used as the test problem, as well as the demand and supply assignment setting for each scale of the grid network. The unit cost of each direct link (i.e., c_{ij} in Equation (3.1)) in each network is simulated as an un-weighted cost in this experiment.

Experiment Four Process and Results.

In this experiment, the binary supply and demand situation is representatively performed on GTS for comparison. With each scale of simulated grid network, the same 20 settings of binary demand and supply of each node as the first experiment are used as the input of the GTS. Consequently, the input information of this experiment is the same as the first one.

The execution time and the peak memory consumption (details are shown in Table 17 and Table 18) of applying GTS for solving the MCF problem of networks with different scales are all obtained during the experiment process, which are used as a base case for comparison.

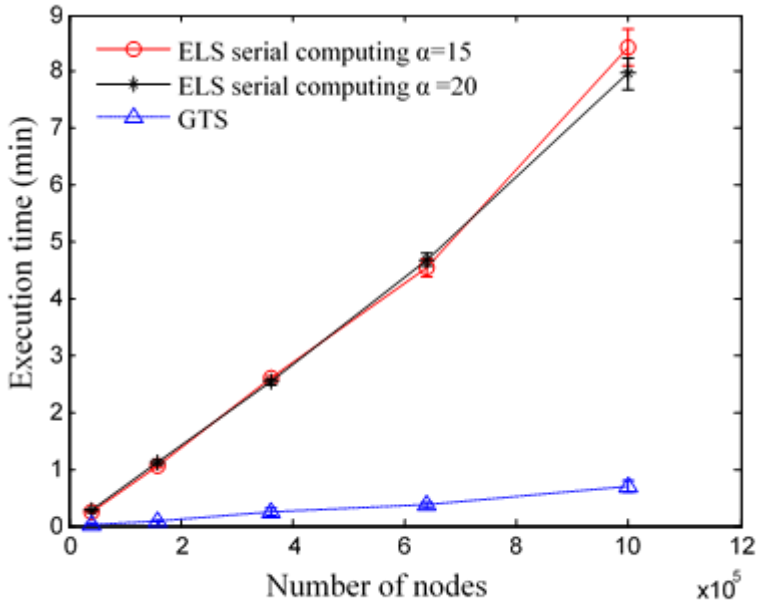
Table 17 GTS execution time (min) (Binary-SD)

Network Scale	Mean value	Min-value	Max-value	Confidence Interval
200×200	0.03	0.029	0.032	[0.030,0.031]
400×400	0.09	0.078	0.094	[0.084,0.087]
600×600	0.23	0.183	0.292	[0.219,0.242]
800×800	0.37	0.314	0.418	[0.360,0.387]
1000×1000	0.70	0.599	0.804	[0.671,0.722]

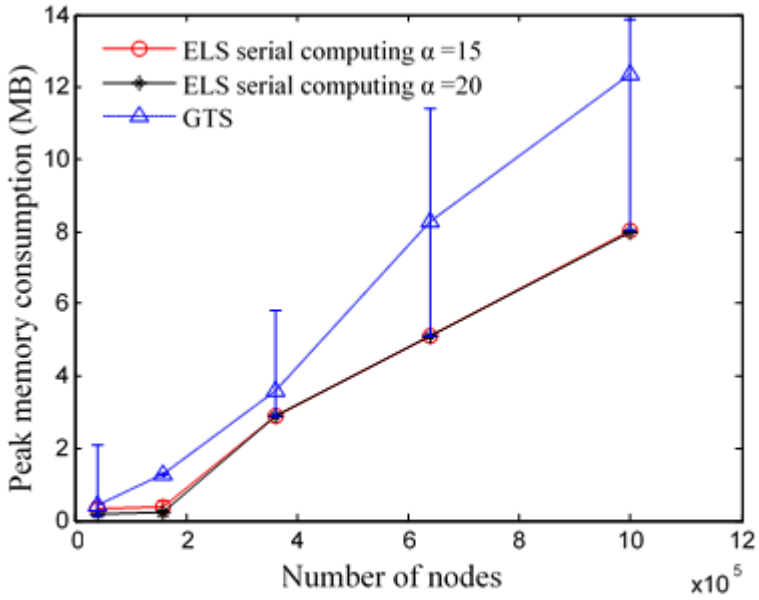
Table 18 GTS Peak Memory consumption (MB) (Binary-SD)

Network Scale	Mean value	Min-value	Max-value	Confidence Interval
200×200	0.44	0.13	2.10	[0.22,0.67]
400×400	1.29	1.29	1.29	[1.29,1.29]
600×600	3.60	2.89	5.82	[3.33,3.87]
800×800	8.26	5.14	11.39	[7.22,9.30]
1000×1000	12.39	8.02	13.85	[11.34,13.43]

Figure 29 shows the performance comparison between ELS and GTS. Figure 29 (a) demonstrates the execution time of ELS with different assigned-αs and the one of GTS. It shows that GTS uses much less execution time than ELS because GTS conducts a very straightforward tiling strategy to cluster the original network. Figure 29 (b) displays the peak memory consumption of ELS with different assigned-αs and the one of GTS. It indicates that GTS has higher peak memory consumption than ELS.



(a) Execution time comparison



(b) Peak memory consumption comparison

Figure 29 Performance comparison between ELS and GTS (Binary-SD)

The statistical actual-α obtained by GTS is shown in Table 19. The comparison of the actual-α obtained by GTS and ELS with different assigned-αs is shown in Figure 30. It is clearly shown that ELS has much smaller actual-α values as compared to GTS.

Table 19 The GTS statistical result of actual-α (Binary-SD)

Network Scale	Mean value	Min-value	Max-value	95% Confidence Interval
200×200	6.0899	5.8825	6.3440	[6.0331,6.1467]
400×400	6.0499	5.9379	6.1522	[6.0215,6.0783]
600×600	6.0331	5.9504	6.1233	[6.0110,6.0552]
800×800	6.0240	5.9669	6.0889	[6.0063,6.0417]
1000×1000	5.9977	5.9417	6.0596	[5.9822,6.0133]

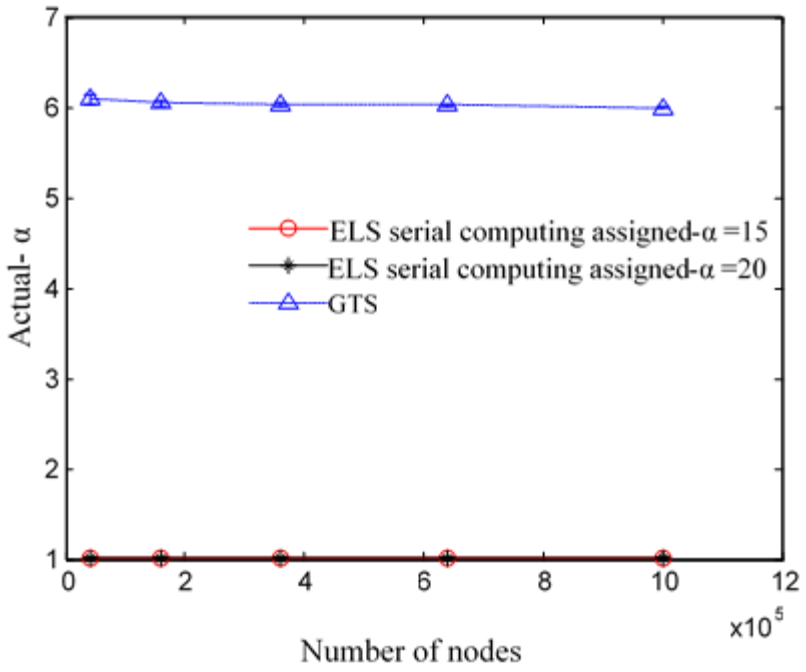


Figure 30 The statistical result of actual-α (GTS) (Binary-SD)

The objective function value obtained by GTS and ELS with different assigned-αs, as well as the one obtained by global MCF algorithm is displayed in Table 20 and Figure 31. Since this experiment is conducted on simulated networks, the flow cost can be generalized cost with corresponding cost unit (i.e., unit of currency or unit of time). It shows that ELS obtains results much closer to the global MCF as compared to the GTS.

Table 20 The flow cost of the entire network (Binary-SD)

Assigned-α	Network scale	ELS objective value		
		Mean value	Min-value	Max-value
15	200×200	3157	3022	3239
	400×400	12639	12348	12853
	600×600	28501	28102	28857
	800×800	50647	50107	51077
	1000×1000	79135	78719	79572
20	200×200	3157	3022	3239
	400×400	12639	12348	12853
	600×600	28501	28102	28857
	800×800	50647	50107	51077
	1000×1000	79135	78724	79572
	Network scale	GTS objective value		
		Mean value	Min-value	Max-value
	200×200	19218	18711	19764
	400×400	76454	75357	77298
	600×600	171940	170570	174630
	800×800	305080	302990	307930
	1000×1000	474600	471590	478560

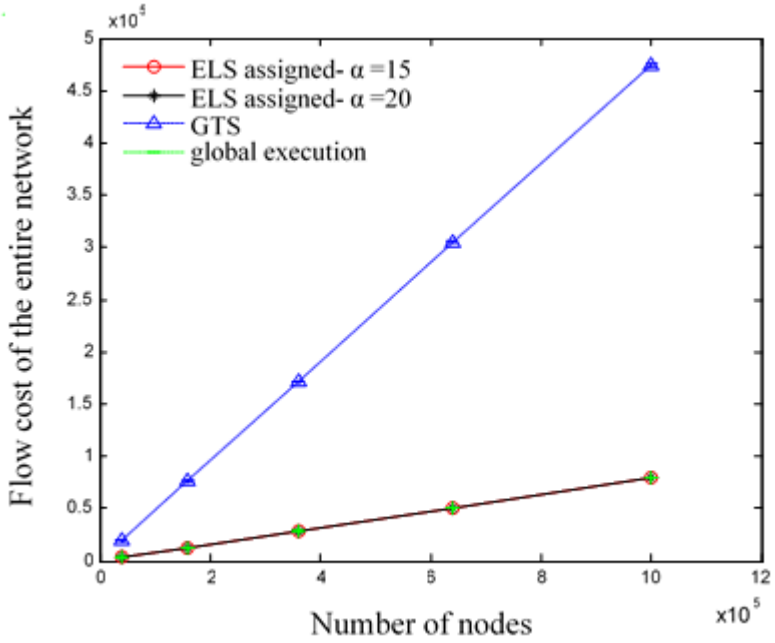


Figure 31 The objective function value comparison between GTS and ELS

Experiment Five: Performance Tests on a Real Network with Binary Demand and Supply

The previous experiments test the performance of ELS based on randomly generated grid networks. The fifth experiment aims to test the performance of ELS on real networks with binary supply and demand assigned. The first test analyzes the effect of the assigned-α value on the clustering results of the AC tilling strategy and measures the accuracy of the ELS. The second test analyzes the performance of the ELS when applied to a big scale practical network with simulated binary supply and demand.

Effectiveness Testing on Small Real Network.

This test is performed based on the network of Friedrichshain which is a district within Berlin, Germany. The data is obtained from a network repository of Ben Stabler's GitHub for transportation research (Stabler 2016). This small network is used to test the effectiveness of ELS with different values of assigned-α.

***Simulation Setting*.** The input data for this test are the coordinates of each node and the given unit cost of links of the network. The network we tested in this experiment includes 224 nodes and 523 links according to the database (Stabler 2016). The capacity of each link is assumed to be infinite. Assuming that the supply and demand of all the nodes within the network are balanced, we randomly assign each node to be a supply or demand node. For the sake of simplicity, each demand node is allocated one unit commodity and each supply node is allocated one unit commodity as well; and the number of demand nodes equals the number of supply nodes. In this case, the summation of all the demand and supply is zero, and there are no transshipment nodes in this experiment, so the network is balanced. Figure 32 shows an example of this simulation setting, in which the red pentacles represent the demand nodes and the black

asterisks represent the supply nodes. In Figure 32, the coordinates are normalized to range of [-100, 100].

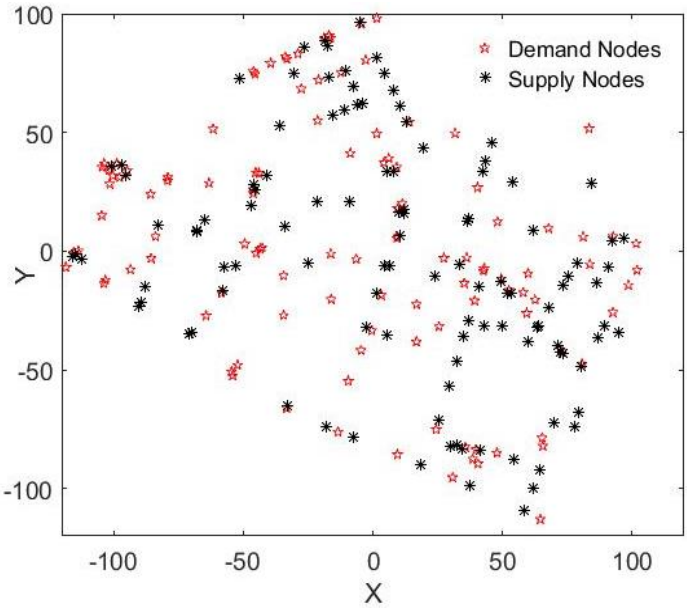
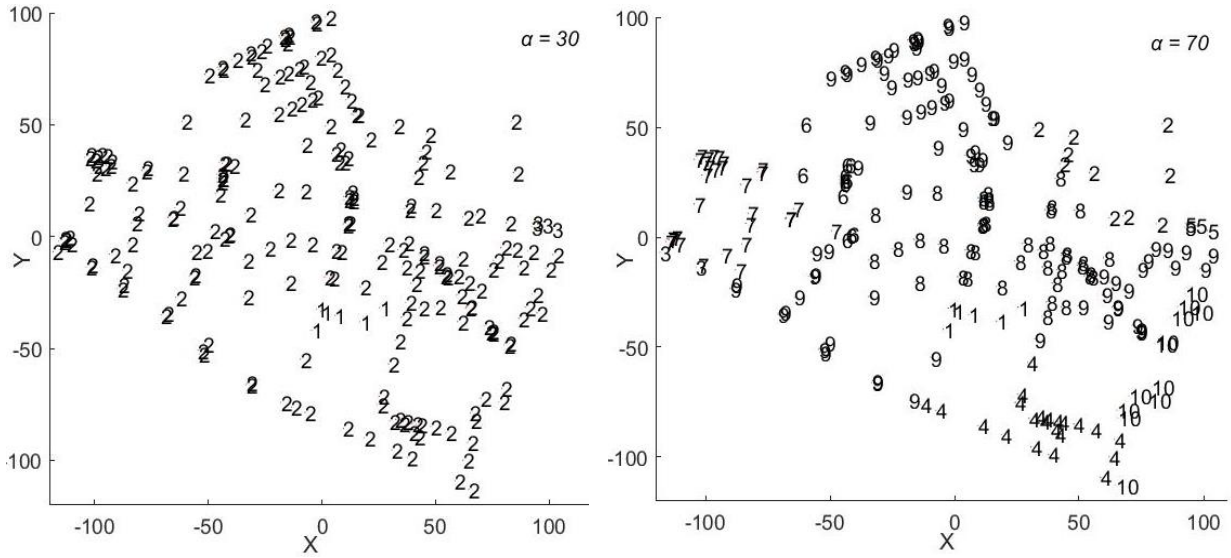


Figure 32 Demonstration of supply and demand nodes setting of Friedrichshain network.

Result. In this test, we randomly generated the supply or demand of each node on the Friedrichshain network 50 times. Each time, the AC tiling strategy (i.e., Alg. 1) is applied to obtain the clustering result of all the nodes (i.e., sub-network information) with different assigned-α values (assigned-α is from 30 to 70 with an increment of 10), where the minimum unit cost between two nodes is approximately substituted by the distance between them,. (The rationale of using the distance between two nodes instead of the minimum unit cost is explained in the AC tiling strategy section of Chapter Three.) Then, the actual-α value is calculated to compare with the assigned-α value.

Figure 33 amplifies two of the clustering results of the original network shown in Figure 31 with different assigned-αs (i.e., assigned-α = 30 in Figure 33 (a) and assigned-α = 70 in Figure 33 (b)). In each sub-figure of Figure 33, the numbers indicate the cluster index of each

node; that is to say, the nodes with same number belong to the same sub-network. From Figure 33, it can be seen that the number of sub-networks increases as the value of assigned-α grows.



(a) (b)

Figure 33 Clustering results of Friedrichshain network with different assigned-αs.

Table 21 shows the mean actual-α values of the 50 tests corresponding to different assigned-αs with corresponding standard deviations. It suggests that the value of actual-α is growing with the assigned-α. Interestingly, it also can be seen that even though the assigned-α value is large, the actual-α is very close to one. As mentioned in Chapter Three, when the actual-α value is closer to one, this indicates greater guarantee of the consistency between the local and global optimal objective values. Hence, it is safe to state that the accuracy of the ELS is reliable in practice, which means the combined local optimal objective value of the network is close to the global optimal objective value.

Table 21 Comparison of assigned-α and actual-α.

Assigned-α	30	40	50	60	70
Mean of Actual-α	1.2501	1.5622	1.7734	1.9335	1.9468
Standard deviation	0.2266	0.3258	0.3158	0.3137	0.3093

Figure 34 shows the comparison between assigned-α and mean actual-α. The mean actual-α increases with the assigned-α, but the tendency indicates that the actual-α is non-sensitive to the value of assigned-α.

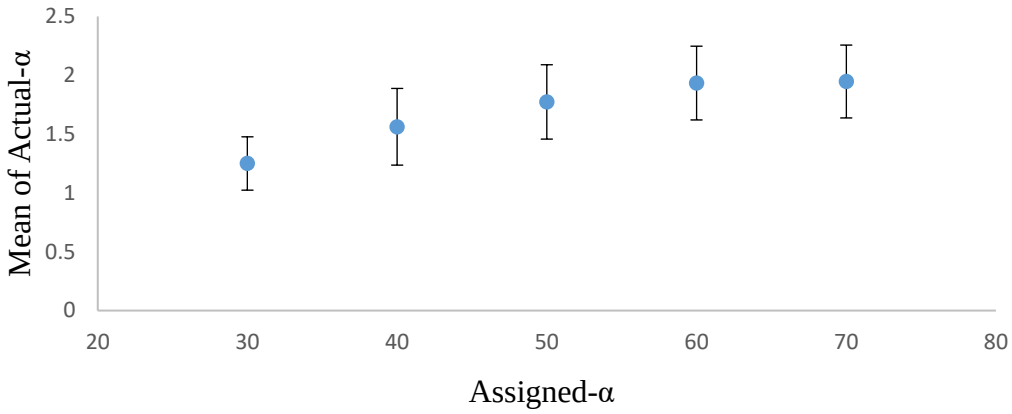


Figure 34 The comparison between assigned-α and mean actual-α

Effectiveness Testing on Large Real Network.

To test the effectiveness of the proposed model on a large-scale network, this test is performed based on the network of Mitte, Prenzlauer Berg and Friedrichshain (MPBF), which are three portions of Berlin, Germany. The data source is the same as for the previous test, but involves a larger region of Berlin City.

Simulation Setting. As with the previous test, the input data are the coordinates of each node and the given cost of links of the network. The network includes 975 nodes and 2184 links (Stabler 2016). The capacity of each link is assumed to be infinite. Since this is a much larger network than the one in the previous test, a modified scenario is applied for this simulation setting. Twenty percent of the nodes are randomly selected from all the nodes of the entire network as supply or demand nodes, while the remaining nodes are treated as transshipment nodes which have zero supply or demand. For the sake of generality and simplicity, each selected node is randomly assigned to be a supply or demand node; each demand node is

allocated one unit commodity and each supply node is allocated one unit commodity. It is still required that the supply and demand should be balanced within the entire network, so the number of supply nodes is equal to the number of demand nodes.

Figure 35 demonstrates the node distribution of the network in this experiment, in which the red pentacles represent the demand nodes, the black asterisks represent the supply nodes and the green dots represent the transshipment nodes. In Figure 35, the coordinates are normalized to a range of [-500,500].

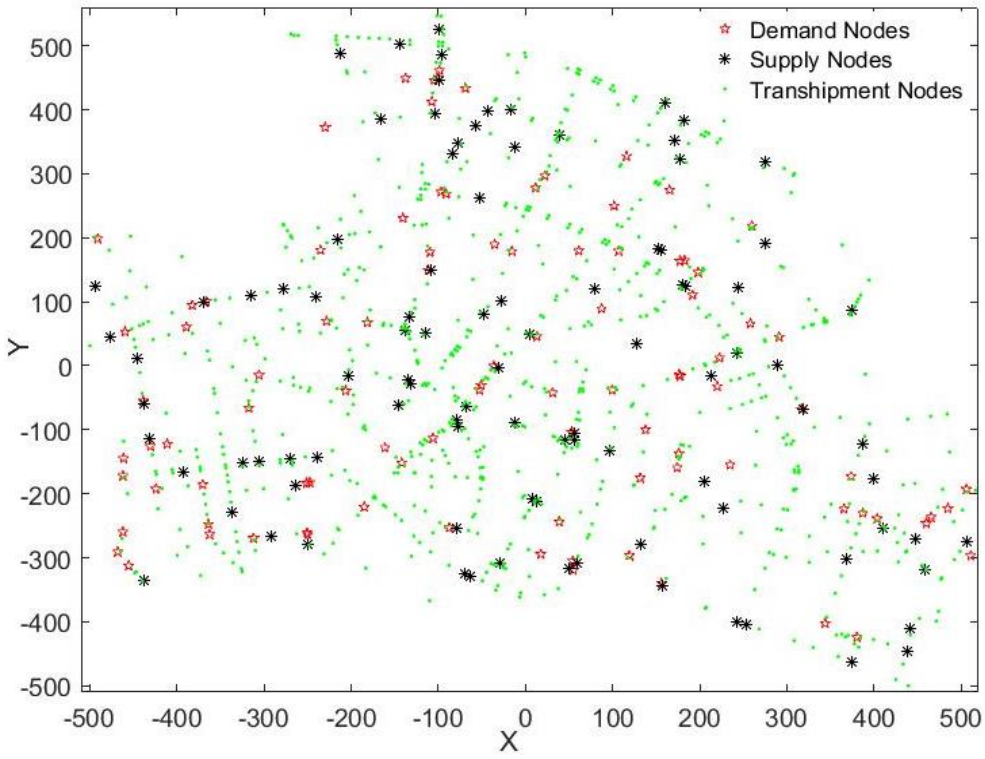


Figure 35 Demonstration of supply and demand nodes setting of Mitte, Prenzlauer Berg and Friedrichshain network.

Result. According to the simulation setting process, the supply and demand nodes were selected from the network of Mitte, Prenzlauer Berg and Friedrichshain. The AC tiling strategy (i.e., Alg. 1) was applied to obtain the clustering result of all the nodes (i.e., sub-network information) with different assigned-α values (α is once again assigned from 30 to 70 with an increment of 10). During the process the minimum unit cost between two nodes is approximately

substituted by the distance between them. Based on the clustering result, the MCF program is applied to each sub-network to obtain the local optimal solution. Then the value of actual-α is calculated to compare with the assigned-αs.

Table 22 and Figure 36 shows the actual-α values corresponding to different assigned-α values. It indicates that the actual-α value is increasing as the assigned-α value grows. But even if the assigned-α value is big, the actual-α is still not far from one, indicating that the combined local optimal objective value is a good approximation of the global optimal objective value. Furthermore, Table 23 shows the performance comparison among different assigned-αs and global execution. It shows the benefits over the global MCF solving process when the assigned-α increases.

Table 22 Comparison of assigned-α and actual-α (MPBF).

Assigned-α	30	40	50	60	70
Actual-α	1.0459	1.1414	1.6221	2.1448	2.7168

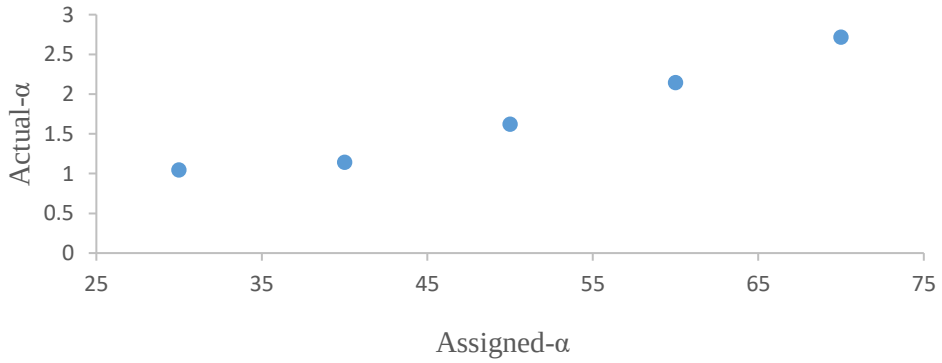


Figure 36 The comparison between assigned-α and mean actual-α (MPBF).

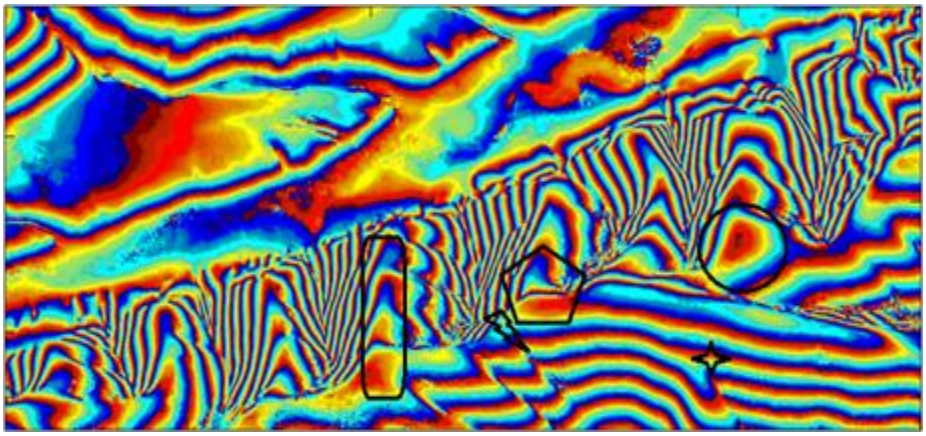
Table 23 Performance comparison among different assigned-αs (MPBF).

Execution time (min)				Peak Memory consumption (MB)			
Global MCF	ELS			Global MCF	ELS		
	α =30	α =50	α =70		α =30	α =50	α =70
8.80	8.46	3.24	2.33	36.31	34.32	17.45	8.57

Experiment Six: Performance Test on a Real Network with Multiple Demand and Supply

Background.

MCF is widely applied in the image processing domain (Ghiglia & Pritt 1998; Yu & Lan 2016). Therefore, one such MCF application, i.e., multi-baseline (MB) synthetic aperture radar interferometry (InSAR) image processing, is used to validate the ELS in this experiment. MB InSAR is used to produce a digital elevation model (DEM) from synthetic aperture radar data (For more detailed information please refer to Appendix D, which offers a brief primer on MB InSAR and explanations on related concepts). For example, Figure 37(a) is the interferogram, and Figure 37(b) is the related Google-earth image. In MB InSAR image processing, there is one processing step called phase unwrapping (PU). In the PU step, researchers first generate residues from the input InSAR image, or so-called interferogram. Residues can be positive or negative. In this experiment, the positive residues are considered as supply nodes, and the negative residues as demand nodes. The purpose of the PU step is to use the MCF model to balance the supply and demand, then obtain the final InSAR product, which is the digital elevation model (DEM) (Ghiglia & Pritt 1998; Yu & Lan 2016).



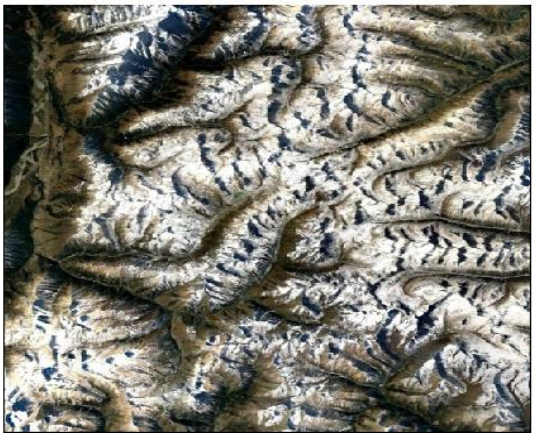
(a) Interferogram

(b) Google-earth image corresponding to (a)

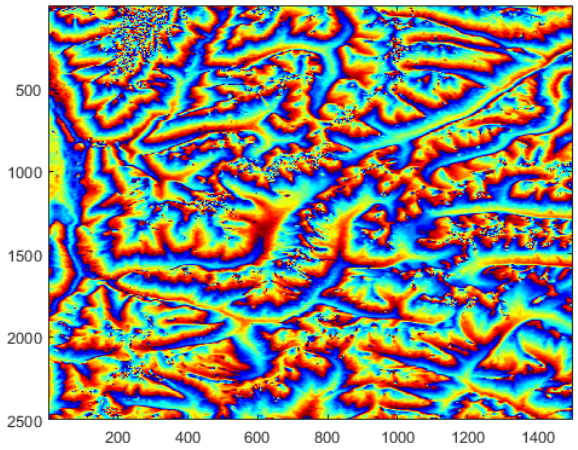
Figure 37 MB InSAR Image processing example

Data Set Introduction.

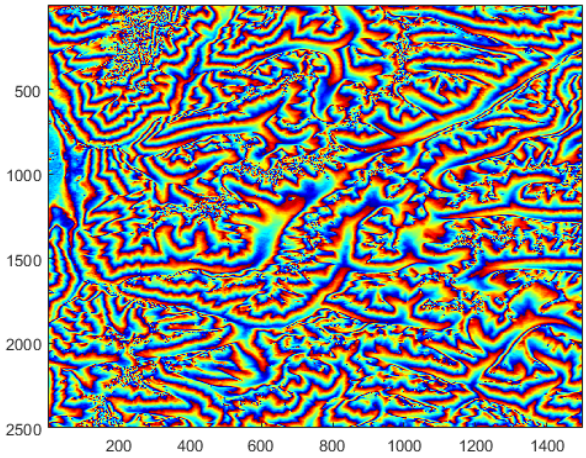
In the following section, real MB InSAR data sets are used to test the performance of the ELS. Figure 38(a) shows the Google Earth image of the real MB InSAR data set used in this experiment, which is the Himalayan mountain area, and whose center latitude and longitude are around 30.85° and 94.47°. Figure 38(b) and Figure 38(c) are the interferograms with different baseline lengths. Figure 38(d) is the residue distribution that is obtained from Figure 38(b) and Figure 38(c). There are 152,970 residues (i.e., supply and demand nodes) with the demand/supply range from -5 to +5.



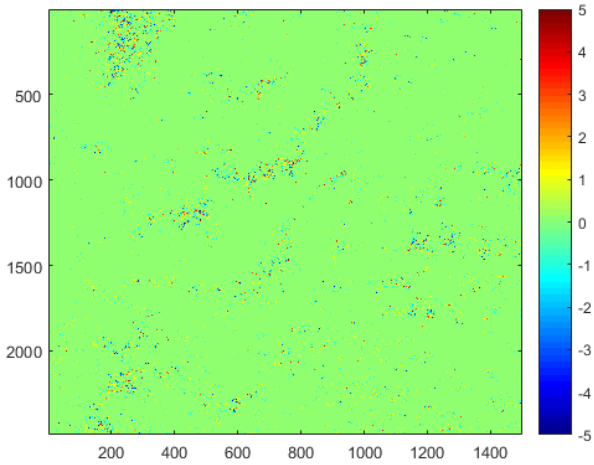
(a) Google Earth image of Himalayan Mountain area



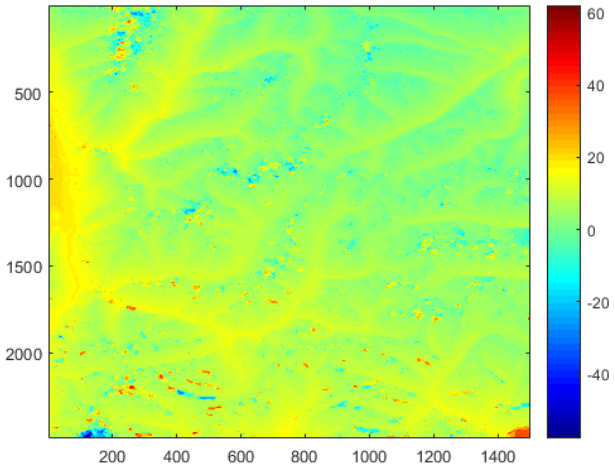
(b) Interfergram 1 of (a)



(c) Interfergram 2 of (a)



(d) Residue distribution of (a)



(e) Global MCF processing result

Figure 38 Himalayan Mountain area image data setting and benchmark result

Figure 38(e) is the global MCF based PU processing result obtained by the method available in the GAMMA remote sensing software, which is the mature InSAR signal processing software (GAMMA 1995). Figure 38(e) can be considered as the ground-truth (i.e., benchmark) in this experiment. The physical meaning of Figure 38(e) is the DEM with a scaling factor and its MCF objective value is 252,807 (GAMMA 1995).

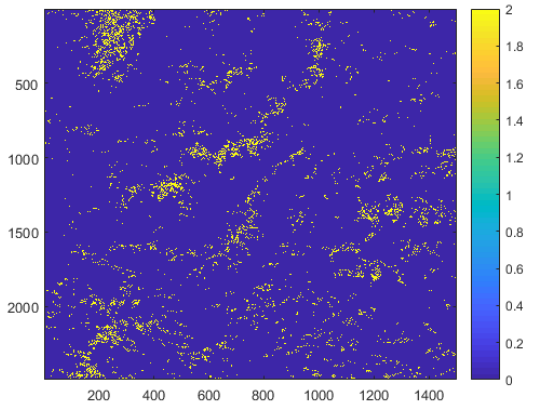
Parameter Selection Strategy.

Parameter selection is one of the common problems in the algorithm design domain. For example, the parameter *k* of the *k*-mean algorithm and the step size of the interior-point algorithm are both user-defined parameters (Han & Kamber 2006; Williamson & Shmoys 2011; Koutroumbas & Konstantinos 2008; Webb 2003). Similarly, assigned-α is a user-defined parameter of the ELS.

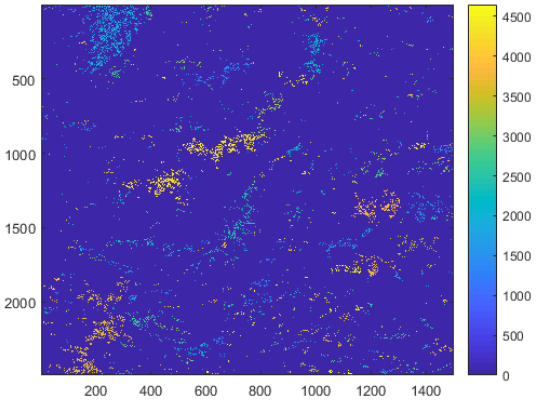
In the ELS, we need to assign a value to α. In the following, an empirical parameter selection strategy is designed. First, a very large value of α, such as 2000, is applied to run the clustering algorithm. A clustering result will be obtained with a number of clusters (i.e., balanced-areas) of relatively small size. Based on this clustering result, the average region cost of the balanced-areas and the average cost between the balanced-areas can be calculated. Then, we conversely apply the α-approximation regional-division theorem to obtain one value of assigned-α, i.e., dividing the average cost between balanced-areas by the average region cost of the balanced-areas. Taking Figure 38(d) as an example, when the value 2000 is applied, the obtained value of α is 148.6. Under this condition, we suggest the value of α be around 150. To verify this parameter selection strategy, in the following experiment 50, 150, 250, 350 and 450 are used as values of assigned-α to test the ELS.

Experiment Results.

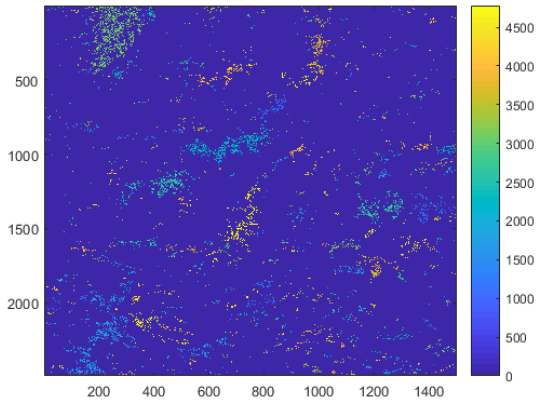
As described in the previous section, the data sets of Figure 38(d) are used as the input of this experiment. According to the parameter selection strategy, different values are assigned to α to test the ELS. Figures 39(a)-(e) are the clustering results with α-values of 50, 150, 250, 350 and 450, in which nodes with same color are clustered in the same balanced-area. The number of the clusters corresponding to the assigned-αs are shown in Table 24.



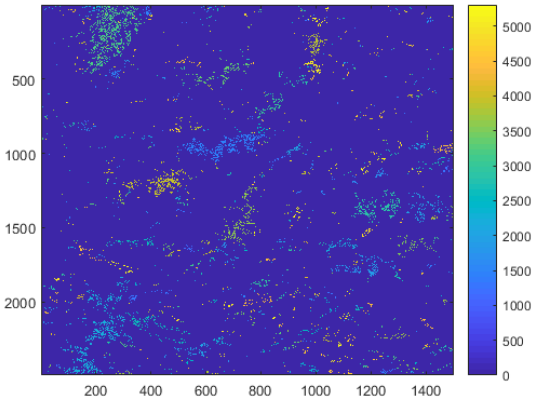
(a) assigned-α = 50



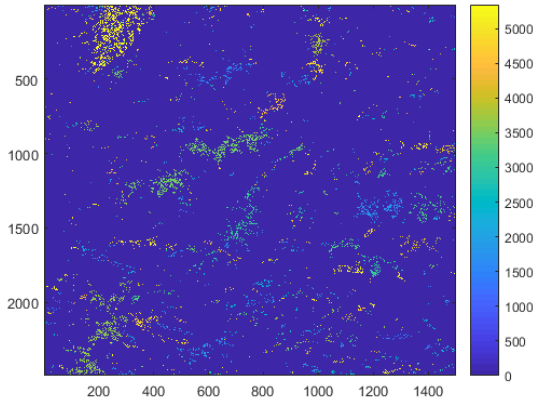
(b) assigned-α = 150



(c) assigned-α = 250



(d) assigned-α = 350



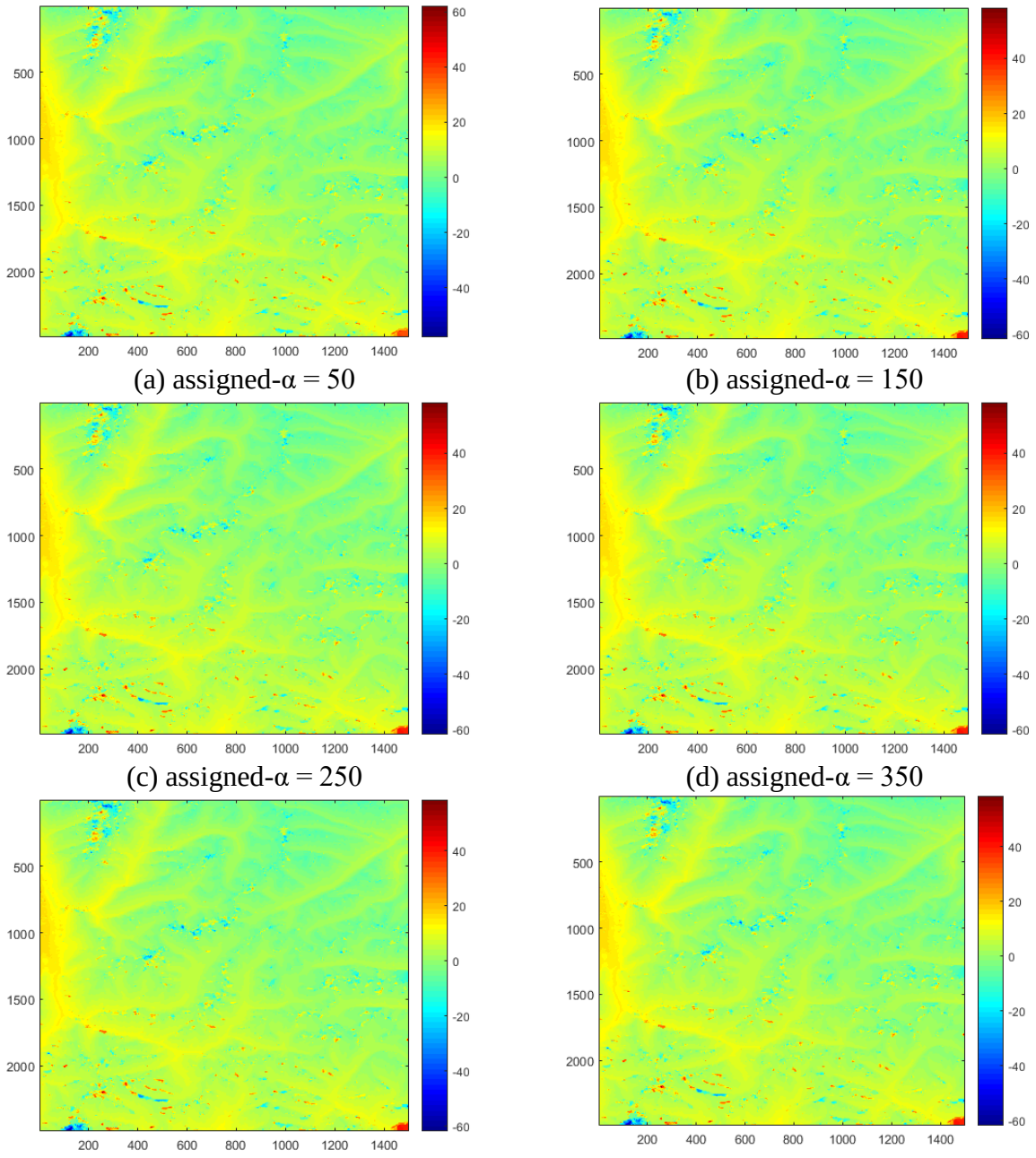
(e) assigned-α = 450

Figure 39 Cluster results corresponding to different assigned-αs

Table 24 The number of clusters corresponding to different assigned-αs

assigned-α	50	150	250	350	450
Number of clusters	1	3576	3676	4082	4111

The results show that when assigned-α equals 50, the nodes are clustered in one balanced-area. This clustering result will not provide any benefit on saving peak memory consumption and execution time. The number of clusters increases as the value of assigned-α increases. It is worthwhile mentioning that in Figures 39(a)-(e) the cluster number is not continuous due to the numbering process during the clustering process, therefore the maximum number of the color bar is greater than the number of clusters corresponding to different assigned-αs. The ELS results corresponding to different values of assigned-α are shown in Figures 40 (a)-(e), whose objective values are shown in Table 25 with the corresponding actual-αs calculated as well (i.e., the actual-α is calculated by dividing the ELS MCF solution by the Global MCF solution).



(a) assigned-α = 50

(b) assigned-α = 150

(c) assigned-α = 250

(d) assigned-α = 350

(e) assigned-α = 450

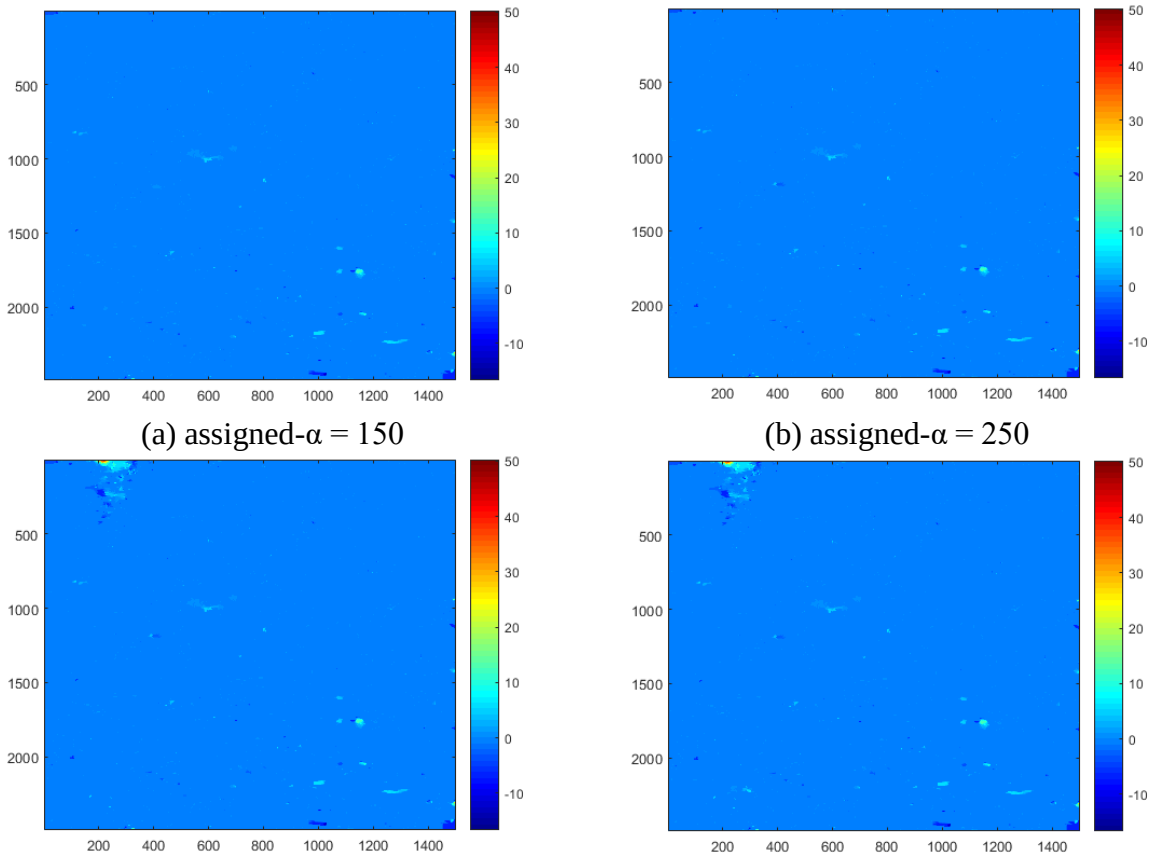
(f) Global MCF solution

Figure 40 MCF processing result obtained by ELS solution figures with different assigned-αs

Table 25 MCF objective values with different assigned-αs

assigned-α	50	150	250	350	450	Global MCF solution
ELS MCF solution	252807	253447	253463	253788	253790	252807
actual-α	1	1.0025	1.0026	1.0039	1.0039	-

Figures 41 (a)-(d) are the differences between Figure 40 (f) and Figures 40 (b)-(e), which are the differences between the global MCF solution (i.e., the DEM with a scaling factor) and the solutions obtained by ELS with different assigned-αs. When assigned-α equals 50, all of the nodes are clustered together in this experiment, and there is no difference between the global and ELS MCF solutions. Thus, Figure 41 does not include the comparison between them. The peak memory consumption and execution time corresponding to different assigned-αs are tabulated in Table 26, as well as the root-mean-square-errors of Figures 41(a)-(d) (comparing to the global MCF solution).



(a) assigned-α = 150

(b) assigned-α = 250

(c) assigned-α = 350

(d) assigned-α = 450

Figure 41 Differences between the global MCF solution and the solutions obtained by ELS with different assigned-αs

Table 26 Performance comparison among different assigned-αs

assigned-α	Peak memory consumption（MB）	Total execution time (min)	root-mean-square-error
150	122392	15.3	0.4262
250	122396	15.2	0.4302
350	122396	16.1	0.7119
450	122400	15.3	0.7137
Global	3117292	158.40	

From the results of this experiment, we can tell that when the value of α is assigned as 150 or 250, the difference between global MCF solution (Figure 40 (f)) and ELS solution (Figure 40 (b)) are very small; the root-mean-square-error is 0.4262 (assigned-α =150) or 0.4302 (assigned-α = 250). From Figure 41 and Table 26, it can be seen that when the assigned-α is increasing, the differences between the global and ELS solutions become larger (Table 26), but the peak memory consumption and execution time does not change very much. Based on the analysis of the experiment results, 150 is a reasonable value for assigned-α for this data set, and any assigned-α value between 150 and 250 could be reasonable as well.

In Figure 42, the histograms demonstrate the distribution of the big clusters (i.e., big cluster refers to clusters that include more than 2000 nodes in this experiment) of the clustering result corresponding to different assigned-αs respectively.

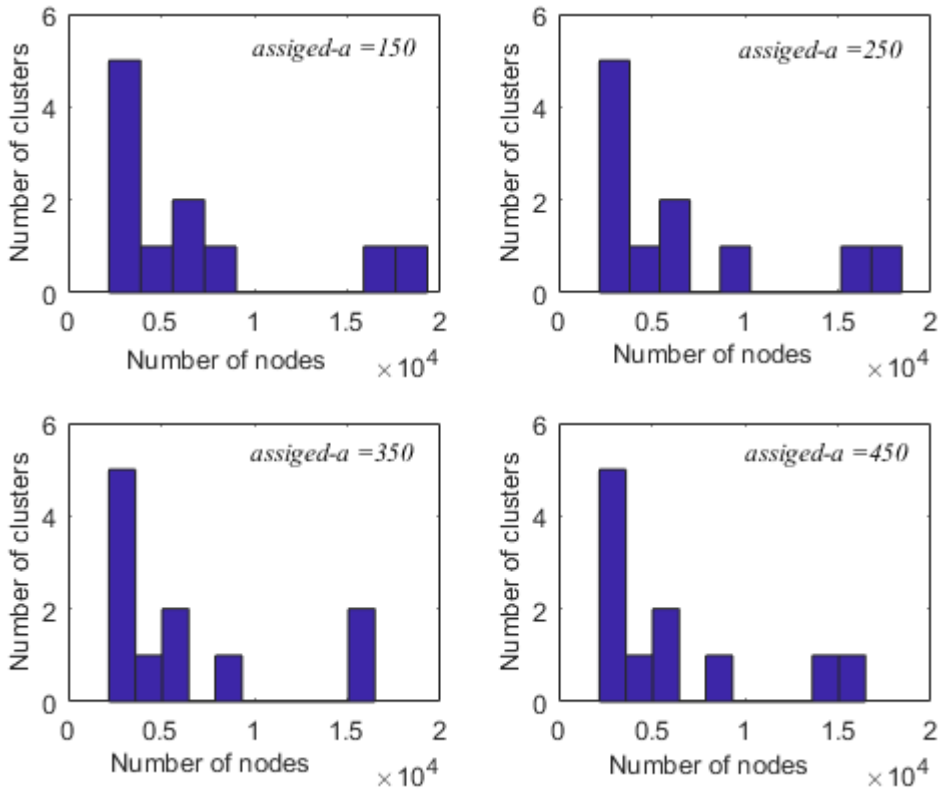


Figure 42 Demostration of the distribution of the big clusters of the clustering results

The peak memory consumption and execution time of ELS is closely related to the number and size of the largest clusters of the clustering result. From Figure 42, we notice that the numbers and the sizes of the largest clusters obtained by AC tiling strategy with different assigned-αs in this experiment are close to each other, which is decided by the features of this input network (refer to Chapter 3). Therefore, the peak memory consumption and execution time savings of the different assigned-αs in this experiment (shown in Table 26) are also relatively close to each other.

Furthermore, based on the results shown in Tables 25 and 26, compared to the global MCF execution process, ELS shows its advantage in saving peak memory consumption and execution time (i.e., ELS just uses around 3.9% peak memory consumption and 9.7% execution

time of global MCF execution process), while keeping acceptable accuracy of the objective value.

Therefore, based on the results obtained by this experiment, we can conclude that i) ELS is beneficial for realistic large-scale MB InSAR (which is a large-scale MCF problem) in saving peak memory consumption and execution time; ii) a reasonable approach to parameter selection is proposed (which is using a very large value of α to obtain an initial clustering result by AC tiling strategy, then conversely applying the α-approximation regional-division theorem to obtain one value of assigned-α) and tested in this experiment.

Conclusions

In this chapter, six groups of experiments are carried out to measure the performance of ELS from various aspects.

The first and second group of experiments tests the effectiveness of ELS on grid networks with binary and multiple supplies and demands when the scale and characteristics of the networks are changing. The actual-αs obtained in these experiments indicate that the performance of ELS is not sensitive to the assigned-α. The results indicate that ELS always uses less peak memory consumption as compared to global MCF execution (i.e., solving the same MCF problem as a whole piece), however, the advantage of ELS is only revealed when the network scale is larger than 1000×1000 nodes.

The results of the third group of experiments show that the application of parallel computing facilitated by ELS can further shorten the execution time of solving the large-scale MCF problem. Notice that, when the network scale is not large, serial computing with ELS uses more execution time than the global MCF process, while parallel computing with ELS uses less execution time than the global MCF process.

The result of the comparison tests with GTS in the fourth experiment reveals that although ELS consumes longer execution time than GTS, ELS offers more accurate solution of the large-scale MCF problem. Finally, the results of the fifth and sixth experiments demonstrate the utility of the ELS on realistic networks under different circumstances. The parameter selection strategy of assigned-α is also demonstrated and tested in the sixth experiment.

5 Conclusion and Future Research

Summary and Conclusions

Intermodal transportation systems have experienced enormous evolution throughout the years since the late 1980s. As the cooperation between various transportation modes increases with rapidly growing demand of regional and/or global actors, the intermodal transportation network must be examined so as to meet contemporary and future demand.

The consideration of large-scale problems is attracting more interest in various fields. One of the important reasons is that as technologies continue to mature and advance, the availability of massive amounts of data and increased computational capacities open the door for new avenues of research and exploration. Therefore, this dissertation has focused on large-scale network programming to explore some new opportunities in this field. Specifically, this dissertation concentrated on the large-scale MCF problem by adopting a divide-and-conquer policy during the optimization process.

The regional-division theorem and the α-approximation regional-division theorem were first proposed and proved. The regional-division theorem offers a sufficient condition to guarantee the consistency between the local and global MCF solutions, while the α-approximation regional-division theorem provides the worst-case bounds for the practical approximation MCF solution. Based on these two theorems, an agglomerative clustering based tiling strategy was proposed for decomposing the input network into sub-networks and further solving the MCF in each sub-network independently. The AC tiling strategy aims to reduce peak memory consumption and improve efficiency.

A series of experiments was carried out and the results showed that the ELS, which is designed according to the proposed AC tiling strategy, is reliable in practice and significantly

saved execution time and/or peak memory consumption in large-scale networks without losing much accuracy under different circumstances. However, so far the ELS is proposed and designed only for single commodity situations, which limits the application range of ELS. And the α-approximation regional-division theorem is designed for deterministic MCF problems. Therefore it has limitation when dealing with the dynamic MCF problem (i.e., OD information is changing over time).

Avenues for future work

The number of experiments involving a performance comparison with an alternative approach is limited in the experiment section for ELS due to insufficient suitable data sets. In terms of future research, the collection of suitable data sets for numerical experimentation will be beneficial to make the performance of ELS more convincing. Due to the limitation of computational resources, even though the performance of the ELS was already tested from various aspects, it is still not a very comprehensive test. Experiments on a gigantic network could show more potential of ELS, when the computational resources are available in the future.

The proposed theorems and methodology in Chapter Three are concentrated on a single commodity situation, which may limit the range of application. It is worthwhile to extend the theorems and algorithm to a multi-commodity situation. (In the appendices section of this dissertation, a sketch of the extension study on large-scale multi-commodity minimum cost flow problems is provided.)

The performance of comprehensive intermodal transportation network design models will be impacted by the formulation and related parameters of the relevant cost functions. When applying the ELS in this dissertation, it is assumed that the cost of the links in the network is comparable to the Euclidean distance, which is just a simplified formulation of cost. Therefore,

study on developing a corresponding approach to formulate more practical cost functions will be of interest in future work. Further consideration about how this approximation affects the approximation theorem is also worthy of study.

To make the proposed theorems more applicable from other perspectives, it will be a good extension to study modified versions of them applicable to other types of network problems confronted by the large-scale situation.

An in-depth study of the relation between the tendency of mean actual-α and the network features will be beneficial to the ELS application.

References

Afanasyev, I. et al., 2016. Techniques for Solving Large-Scale Graph Problems on Heterogeneous Platforms. In *Supercomputing*. pp. 318–332. Available from: http://link.springer.com/10.1007/978-3-319-55669-7.

Ahuja, R.K., Magnanti, T.L. & Orlin, J.B., 1993. Network flows: theory, algorithms and applications. *Network*, 1, p.864. Available from: http://www.amazon.com/Network-Flows-Theory-Algorithms-Applications/dp/013617549X%5Cnhttp://www.computing-wisdom.com/secure/NetworkFlows.pdf.

Ahuja, R.K., Orlin, J.B. & Sharma, D., 2003. A composite very large-scale neighborhood structure for the capacitated minimum spanning tree problem. *Operations Research Letters*, 31(3), pp.185–194.

Aridhi, S. et al., 2015. A MapReduce-based approach for shortest path problem in large-scale networks. *Engineering Applications of Artificial Intelligence*, 41, pp.151–165. Available from: http://dx.doi.org/10.1016/j.engappai.2015.02.008.

Armstrong, R.D., Klingman, D. & Whitman, D., 1980. Implementation and analysis of a variant of the dual method for the capacitated transshipment problem. *European Journal of Operational Research*, 4(6), pp.403–420. Available from: http://linkinghub.elsevier.com/retrieve/pii/0377221780901939 [Accessed February 16, 2017].

Babonneau, F., du Merle, O. & Vial, J.P., 2006. Solving Large-Scale Linear Multicommodity Flow Problems with an Active Set Strategy and Proximal-ACCPM. *Operations Research*, 54(1), pp.184–197. Available from: http://dx.doi.org/10.1287/opre.1050.0262.

Bertsekas, D.P. & Tseng, P., 1994. RELAX-IV : A Faster Version of the RELAX Code for Solving Minimum Cost Flow Problems. *Massachusetts Institute of Technology*, (November), pp.1–18. Available from: http://www.mit.edu/~dimitrib/RELAX4_doc.pdf.

Bertsekas, D.P. & Tseng, P., 1988. Relaxation Methods for Minimum Cost Ordinary and Generalized Network Flow Problems. *Operations Research*, pp.93–114. Available from: http://pubsonline.informs.org/doi/abs/10.1287/opre.36.1.93.

Bontekoning, Y.M., Macharis, C. & Trip, J.J., 2004. Is a new applied transportation research field emerging? - A review of intermodal rail-truck freight transport literature. *Transportation Research Part A: Policy and Practice*, 38(1), pp.1–34. Available from: http://linkinghub.elsevier.com/retrieve/pii/S0965856403000740 [Accessed March 9, 2017].

Bookbinder, J.H. & Fox, N.S., 1998. Intermodal routing of Canada–Mexico shipments under NAFTA. *Transportation Research Part E: Logistics and Transportation Review*, 34(4), pp.289–303.

Bradley, Gordon H. Gerald, B.G., 1977. Design and Implementation of Large Scale Primal Transshipment Algorithms Author (s): Gordon H . Bradley , Gerald G . Brown and Glenn W . Graves Published by : INFORMS Stable URL : http://www.jstor.org/stable/2630725 REFERENCES Linked references are avai. *Management Science*, 24(1), pp.1–34. Available from: http://www.jstor.org/stable/2630725?seq=1#page_scan_tab_contents.

Cambridge Systematics Inc. et al., 1995. *Intermodal Freight Transportation, Volume 1: Overview of Impediments, Data Sources for Intermodal Transportation planning, and Annotated Bibliography*,

Carballo, G., 2000. *Statistically-based multiresolution network flow phase unwrapping for SAR interferometry*. Royal Institute of Technolog, Stockholm, Sweden.

Caris, A., Macharis, C. & Janssens, G.K., 2013. Decision support in intermodal transport: A new research agenda. *Computers in Industry*, 64(2), pp.105–112. Available from: http://dx.doi.org/10.1016/j.compind.2012.12.001.

Caris, A., Macharis, C. & Janssens, G.K., 2008. Planning Problems in Intermodal Freight Transport: Accomplishments and Prospects. *Transportation Planning and Technology*, 31(3), pp.277–302.

Chidananda Gowda, K. & Ravi, T. V., 1995. Agglomerative clustering of symbolic objects using the concepts of both similarity and dissimilarity. *Pattern Recognition*, 28(8), pp.1277–1282.

Crobak, K.M.A.B.W.B.R., 2007. An Experimental Study of A Parallel Shortest Path Algorithm for Solving Large-Scale Graph Instances. *Proceedings of the 9th Workshop on Algorithm Engineering and Experiments (ALENEX'07)*, pp.23–35.

Dantzig, G.B., 1951. *Application of the Simplex Method to a Transportation Problem*, New York: John Wiley & Sons, Inc. Available from: http://web.eecs.umich.edu/~pettie/matching/Dantzig-using-simplex-for-transportation-Cowles-Monograph.pdf.

Dijkstra, E.W., 1959. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 271(1), pp.269–271.

EGRES, 2003. LEMON Library. *the Department of Operations Research, Eötvös Loránd University, Budapest*. Available from: http://lemon.cs.elte.hu/trac/lemon [Accessed January 1, 2016].

European Commission Directorate-general for Energy and Transport, 2009. *A sustainable future for transport ： Towards an integrated, Technology-Led and User-Friendly System*, Available from: ec.europa.eu/transport/media/publications/index_en.htm.

European Conference of Ministers of Transport, 1993. Terminology on Combined Transport. In Pairs, pp. 16–18.

Frangioni, A. & Manca, A., 2004. A Computational Study of Cost Reoptimization for Min-Cost Flow Problems. *Operations Research*, pp.61–70. Available from: http://dx.doi.org/10.1287/ijoc.1040.0081.

GAMMA, 1995. Interferometric SAR Processing: GAMMA Remote Sensing. Gmligen, Switzerland. Available from: https://www.gamma-rs.ch/ [Accessed January 1, 2017].

Ghane-Ezabadi, M. & Vergara, H.A., 2016. Decomposition approach for integrated intermodal logistics network design. *Transportation Research Part E: Logistics and Transportation Review*, 89, pp.53–69. Available from: http://dx.doi.org/10.1016/j.tre.2016.02.009.

Ghiglia, D.C. & Pritt, M.D., 1998. *Two-Dimensional Phase Unwrapping: Theory, Algorithms, and Software* 1st ed., Wiley-Interscience.

Goldberg, A. V, 1997. An Efficient Implementation of a Scaling Minimum-Cost Flow Algorithm. *Journal of Algorithms*, 22(1), pp.1–29. Available from: http://linkinghub.elsevier.com/retrieve/pii/S019667748570805X.

Goldberg, A. V. & Tarjan, R.E., 1989. Finding minimum-cost circulations by canceling negative cycles. *Journal of the ACM*, 36(4), pp.873–886. Available from: http://dl.acm.org/citation.cfm?id=76359.76368.

Goldberg, A. V. & Tarjan, R.E., 1990. Solving minimum cost flow problems by successive approximation. *Mathematics of Operations Research*, 15, pp.430–466.

Groothedde, B., Ruijgrok, C. & Tavasszy, L., 2005. Towards collaborative, intermodal hub networks. A case study in the fast moving consumer goods market. *Transportation Research Part E: Logistics and Transportation Review*, 41(6 SPEC. ISS.), pp.567–583.

Gubichev, A. et al., 2010. Fast and accurate estimation of shortest paths in large graphs. *Proceedings of the 19th ACM international conference on Information and knowledge management*, pp.499–508.

Han, J. & Kamber, M., 2006. *Data Mining: Concepts and Techniques*,

Hayuth, Y., 1987. *INTERMODALITY, CONCEPT AND PRACTICE: STRUCTURAL CHANGES IN THE OCEAN FREIGHT TRANSPORT INDUSTRY*, Lodon: Lloyd's of London Press.

Ishfaq, R. & Sox, C.R., 2010. Intermodal logistics: The interplay of financial, operational and service issues. *Transportation Research Part E: Logistics and Transportation Review*, 46(6), pp.926–949. Available from: http://dx.doi.org/10.1016/j.tre.2010.02.003.

Jarrah, a. I., Johnson, E. & Neubert, L.C., 2009. Large-Scale, Less-than-Truckload Service Network Design. *Operations Research*, 57(3), pp.609–625.

Kantorovich, L.V., 1960. Mathematical Methods in the Organization and Planning of Production (translated in). *Management Science*, 6, pp.336–422.

Karloff, H., Suri, S. & Vassilvitskii, S., 2010. A Model of Computation for MapReduce. *ACM-SIAM Symposium on Discrete Algorithms (SODA10)*, p.11. Available from: http://theory.stanford.edu/~sergei/papers/soda10-mrc.pdf.

Kelly, D.J. et al., 1991. The Minimum Cost Flow Problem and The Network Simplex Solution Method By Master of Management Science Supervised by. *Transportation*, (September).

Klunder, G.A. & Post, H.N., 2006. The shortest path problem on large-scale real-road networks. *Networks*, 48(4).

Koutroumbas, S. & Konstantinos, T., 2008. *Pattern Recognition* 4th ed., Academic Press.

Kovács, P., 2015. Minimum-cost flow algorithms: an experimental evaluation. *Optimization Methods and Software*, 30(1), pp.94–127. Available from: http://www.tandfonline.com/doi/abs/10.1080/10556788.2014.895828.

Ledolter, J. & Hogg, R. V., 2009. *Applied Statistics for Engineers and Physical Scientists* 3rd ed., Pearson.

Lin, G.-H. & Xue, G., 2000. Optimal layout of hexagonal minimum spanning trees in linear time. *Proceedings - IEEE International Symposium on Circuits and Systems*, 4, p.I-196-I-199. Available from: http://www.scopus.com/inward/record.url?eid=2-s2.0-0033681645&partnerID=40&md5=62481da74ce7ac19e930d0d29751d0bd%5Cnhttp://www.scopus.com/inward/record.url?eid=2-s2.0-0033699236&partnerID=40&md5=1734573a9d50183077ef536dab7a1025.

Lobel, A., 1996. Solving Large-Scale Real-World Minimum-Cost Flow Problems by a Network Simplex Method.

Macharis, C. & Bontekoning, Y.M., 2004. Opportunities for OR in intermodal freight transport research: A review. *European Journal of Operational Research*, 153(2), pp.400–416.

Marchiori, E. & Steenbeek, A., 2000. An evolutionary algorithm for large scale set covering problems with application to airline crew scheduling. *Real-World Applications of Evolutionary Computing*, pp.370–384.

Maruhashi, K. et al., 2012. EigenSP: A more accurate shortest path distance estimation on large-scale networks. *Proceedings - 12th IEEE International Conference on Data Mining Workshops, ICDMW 2012*, pp.234–241.

Meyer, U. & Sanders, P., 2003. Δ-stepping: A parallelizable shortest path algorithm. *Journal of Algorithms*, 49(1), pp.114–152.

Min, H., 1991. International intermodal choices via chance-constrained goal programming. *Transportation Research Part A: General*, 25(6), pp.351–362. Available from: http://linkinghub.elsevier.com/retrieve/pii/019126079190013G [Accessed March 30, 2017].

Muelas, S., LaTorre, A. & Peña, J.-M., 2015. A distributed VNS algorithm for optimizing dial-a-ride problems in large-scale scenarios. *Transportation Research Part C: Emerging Technologies*, 54, pp.110–130. Available from: http://dx.doi.org/10.1016/j.trc.2015.02.024.

Muelas, S., Latorre, A. & Peña, J.M., 2013. A variable neighborhood search algorithm for the optimization of a dial-a-ride problem in a large city. *Expert Systems with Applications*, 40(14), pp.5516–5531. Available from: http://dx.doi.org/10.1016/j.eswa.2013.04.015.

Nabais, J.L. et al., 2015. Achieving transport modal split targets at intermodal freight hubs using a model predictive approach. *Transportation Research Part C: Emerging Technologies*, 60, pp.278–297.

Nie, Y. (Marco), 2010. A class of bush-based algorithms for the traffic assignment problem. *Transportation Research Part B: Methodological*, 44(1), pp.73–89. Available from: http://dx.doi.org/10.1016/j.trb.2009.06.005.

Ordonez, C., Zhang, Y. & Cabrera, W., 2016. The Gamma Matrix to Summarize Dense and Sparse Data Sets for Big Data Analytics. *IEEE Transactions on Knowledge and Data Engineering*, 28(7), pp.1905–1918.

Orlin, J.B., 1989. *A Faster Strongly Polynomial Minimum Cost Flow Algorithm*,

Portugal, L.F. et al., 2008. Fortran subroutines for network flow optimization using an interior point algorithm. *Pesquisa Operacional*, 28(2), pp.243–261.

Rader, D.J., 2010. *Deterministic Operations Research: models and methods in linear optimization*, New Jersey: John Wiley & Sons. Inc.

Resat, H.G. & Turkay, M., 2015. Design and operation of intermodal transportation network in the Marmara region of Turkey. *Transportation Research Part E: Logistics and Transportation Review*, 83, pp.16–33. Available from: http://dx.doi.org/10.1016/j.tre.2015.08.006.

Resende, M.G.C. & Pardalos, P.M., 1996. Interior point algorithms for network flow problems. In J. E. Beasley, ed. *Advances in Linear and Integer Programming*. Oxford, UK,: Oxford University Press, pp. 147–187.

Sakumoto, Y., Ohsaki, H. & Imase, M., 2010. On the Effectiveness of Thorup's Shortest Path Algorithm for Large-Scale Network Simulation. In *10th Annual International Symposium on Applications and the Internet IEEE*. pp. 339–342.

Southworth, F. & Peterson, B.E., 2000. Intermodal and international freight network modeling. *Transportation Research Part C: Emerging Technologies*, 8(1–6), pp.147–166.

Stabler, B., 2016. TransportationNetworks. Available from: https://github.com/bstabler/TransportationNetworks [Accessed June 1, 2016].

SteadieSeifi, M. et al., 2014. Multimodal freight transportation planning: A literature review. *European Journal of Operational Research*, 233(1), pp.1–15.

Tardos, E., 1986. A Strongly Polynomial Algorithm to Solve Combinatorial Linear Programs. *Operations Research*, 34(2), pp.250–256.

U.S. Congress, 1991. *Intermodal Surface Transportation Efficiency Act of 1991*, https://www.congress.gov/bill/102nd-congress/house-bill/02950.

U.S. Congress, 1998. *Transportation Equity Act for the 21st Century*, US: https://www.fhwa.dot.gov/tea21/index.htm.

Van Duin, R. & Van Ham, H., 1998. A three-stage modeling approach for the design and organization of intermodal transportation services. *Proceedings of the IEEE International Conference on Systems, Man and Cyber- netics, Part 4, October 11–14, 1998*, 4, pp.4051–4056. Available from: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=726723.

Wang, X., Meng, Q. & Miao, L., 2016. Delimiting port hinterlands based on intermodal network flows: Model and algorithm. *Transportation Research Part E: Logistics and Transportation Review*, 88, pp.32–51. Available from: http://dx.doi.org/10.1016/j.tre.2016.02.004.

Webb, A.R., 2003. *Statistical Pattern Recognition, 2nd Edition* 2nd ed., Wiley.

Williamson, D.P. & Shmoys, D.B., 2011. *The Design of Approximation Algorithms*, Available from: http://books.google.com/books?hl=en&lr=&id=Cc_Fdqf3bBgC&pgis=1.

Willianmson, D.P. & Shmoys, D.B., 2010. *The Design of Approximation Algorithms*, New York: Cambridge University Press.

Xu, Q. et al., 2016. Fast shortest-path queries on large-scale graphs. *Proceedings - International Conference on Network Protocols, ICNP*, 2016–Decem, pp.1–10.

Yu, H. & Lan, Y., 2016. Robust two-dimensional phase unwrapping for multibaseline SAR interferograms: A two-stage programming approach. *IEEE Transactions on Geoscience and Remote Sensing*, 54(9), pp.5217–5225.

Zhang, X.-Y. et al., 2016. Kuhn–Munkres Parallel Genetic Algorithm for the Set Cover Problem and Its Application to Large-Scale Wireless Sensor Networks. *IEEE Transactions on Evolutionary Computation*, 20(5), pp.695–710. Available from: http://ieeexplore.ieee.org/document/7362161/.

Zhang, Y.P. et al., 2011. Research on the maximum flow in large-scale network. *Proceedings - 2011 7th International Conference on Computational Intelligence and Security, CIS 2011*, pp.482–486.

Zhou, H., Shenoy, N. & Nicholls, W., 2002. Efficient minimum spanning tree construction without Delaunay triangulation. *Information Processing Letters*, 81(5), pp.271–276.

Zhou, S., Chelmis, C. & Prasanna, V.K., 2015. Accelerating Large-Scale Single-Source Shortest Path on FPGA. *Proceedings - 2015 IEEE 29th International Parallel and Distributed Processing Symposium Workshops, IPDPSW 2015*, (1018801), pp.129–136.

Zografos, K.G. & Regan, A.C., 2004. Current challenges for intermodal freight transport and logistics in Europe and the United States. *Transportation Research Record: Journal of the Transportation Research Board*, 1873(1873), pp.70–78.

Appendices

Appendix A, Overview of the Algorithm 1 implementation

Algorithm 1 proposed in Chapter Three in this dissertation is implemented in a MATLAB® environment. A detailed explanation of the current implementation is provided in this section.

The main data structures in Algorithm 1.

The main data objects needed in this algorithm are array and cell array. Array is applied to store the nodes with their related information and the cell array is applied to the cluster manipulation.

Code prototypes and their attributes and functionalities.

The problem which is studied in this dissertation is closely related to graph theory. In this field, the main information is usually stored in a matrix (i.e., adjacency matrix). Therefore, the information about the graph (i.e., coordinate of nodes, links, and cost of links) is stored in 2-D arrays (i.e., matrix) in my code. It is convenient to search, retrieve and update the information through the array index. In addition, with regard to the information of clusters and the cluster elements (index of the nodes), since the number of the elements within each cluster is different and cannot be known in advance, we cannot store this information in the array. Then the cell array is adopted to deal with this kind of information.

The main functions in the current implementation.

The main function of the current implementation of Algorithm 1 is the clustering function. The input data of this algorithm are the coordinates of the nodes which are stored in array ‘XY’ (double) and the cost information of all links which is stored in arrays ‘aa’ (single), ‘bb’(single) and ‘cc’(double) indicating the starting node, ending node and the cost of each link,

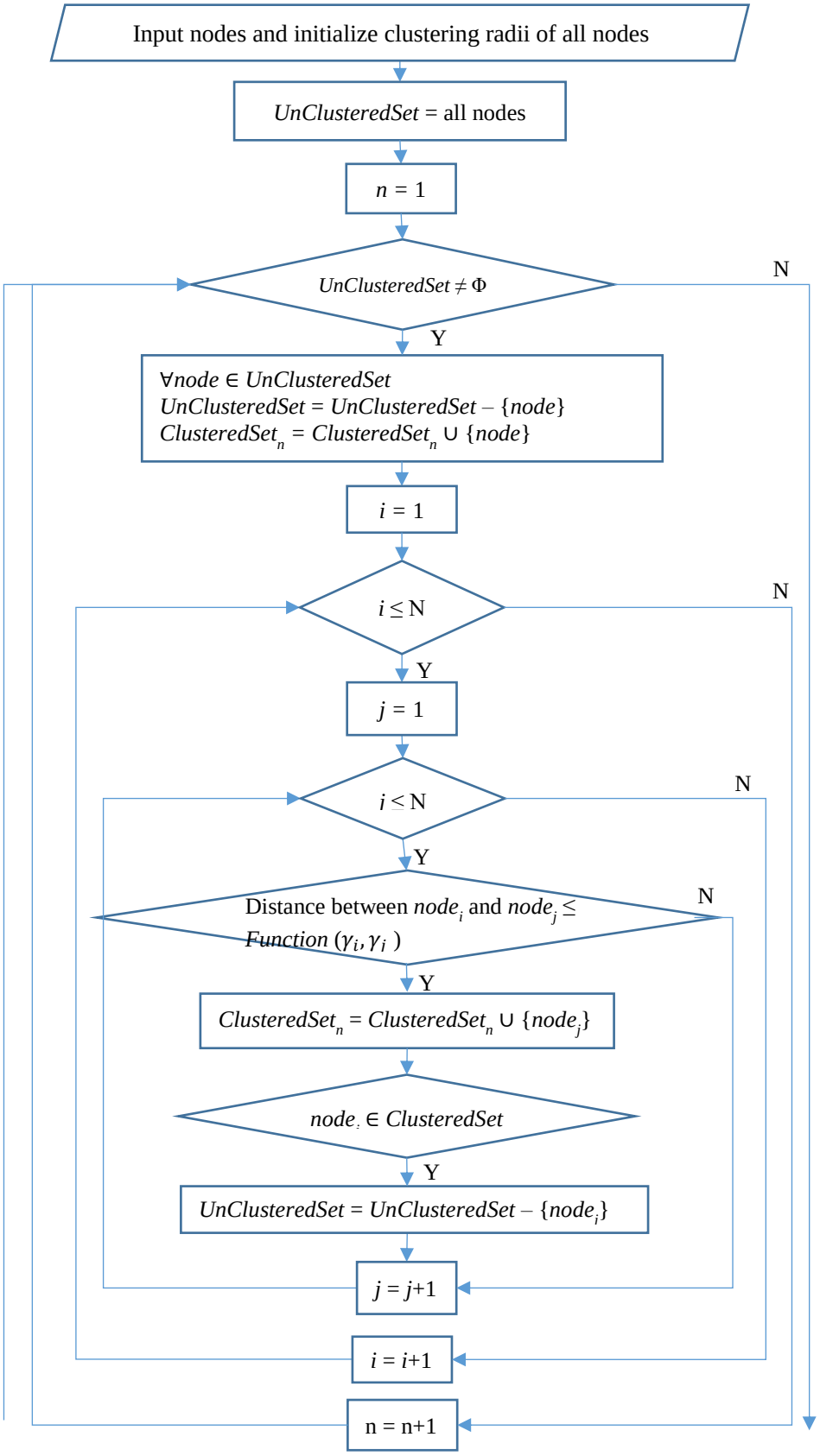
respectively. Another form of the input of link cost information is the costmatrix which stores the unit cost information of all links.

For simulation, this algorithm generates the demand or supply for each node, then the input node information is stored in the array ‘InputNodes’(double), including the coordinates and demand or supply of each node. If the actual demand and supply data is provided, we can directly input the data into the ‘InputNodes’ array. The output of the clustering function is the cell array called ‘Cluster’ (cell) where each cell keeps the indices of the nodes included in the same cluster.

Within the main function, since it is intended to test the influence of the value of different assigned-α on Algorithm 1, a ‘for loop’ is embedded for increasing the value of assigned-α while applying the cluster process. Actually, for the final application version of this algorithm, this part could be simplified, since we could just choose one proper value as the assigned-α and apply it in the algorithm.

To test the accuracy of this algorithm, a function for calculating the global minimum-cost-flow (MCF) optimal solution of the entire network is also embedded in the main function. This part could be omitted in the application version of this code, since there is no need to calculate it in the final application version. And a similar function also is applied on each sub-network after the clustering stage is completed. The input data of this MCF optimal solution function is the information of the nodes (‘InputNodes’) and the information of the clusters (‘Cluster’) which is the output of the clustering function. And the output of this function should be the MCF optimal solution of the input network. The parameter handling in those functions is by reference.

The pseudo-code of Algorithm 1 is provided in section 4.5. To clearly describe this algorithm, a workflow is produced to demonstrate the detailed steps of this algorithm, shown in Figure 43.


Input nodes and initialize clustering radii of all nodes
UnClusteredSet = all nodes
n = 1
UnClusteredSet ≠ Φ
N
Y
∀node ∈ UnClusteredSet
UnClusteredSet = UnClusteredSet – {node}
ClusteredSet_n = ClusteredSet_n ∪ {node}
i = 1
i ≤ N
N
Y
j = 1
i ≤ N
N
Y
Distance between node_i and node_j ≤ Function (γ_i, γ_i)
N
Y
ClusteredSet_n = ClusteredSet_n ∪ {node_j}
node_i ∈ ClusteredSet
Y
UnClusteredSet = UnClusteredSet – {node_i}
j = j+1
i = i+1
n = n+1

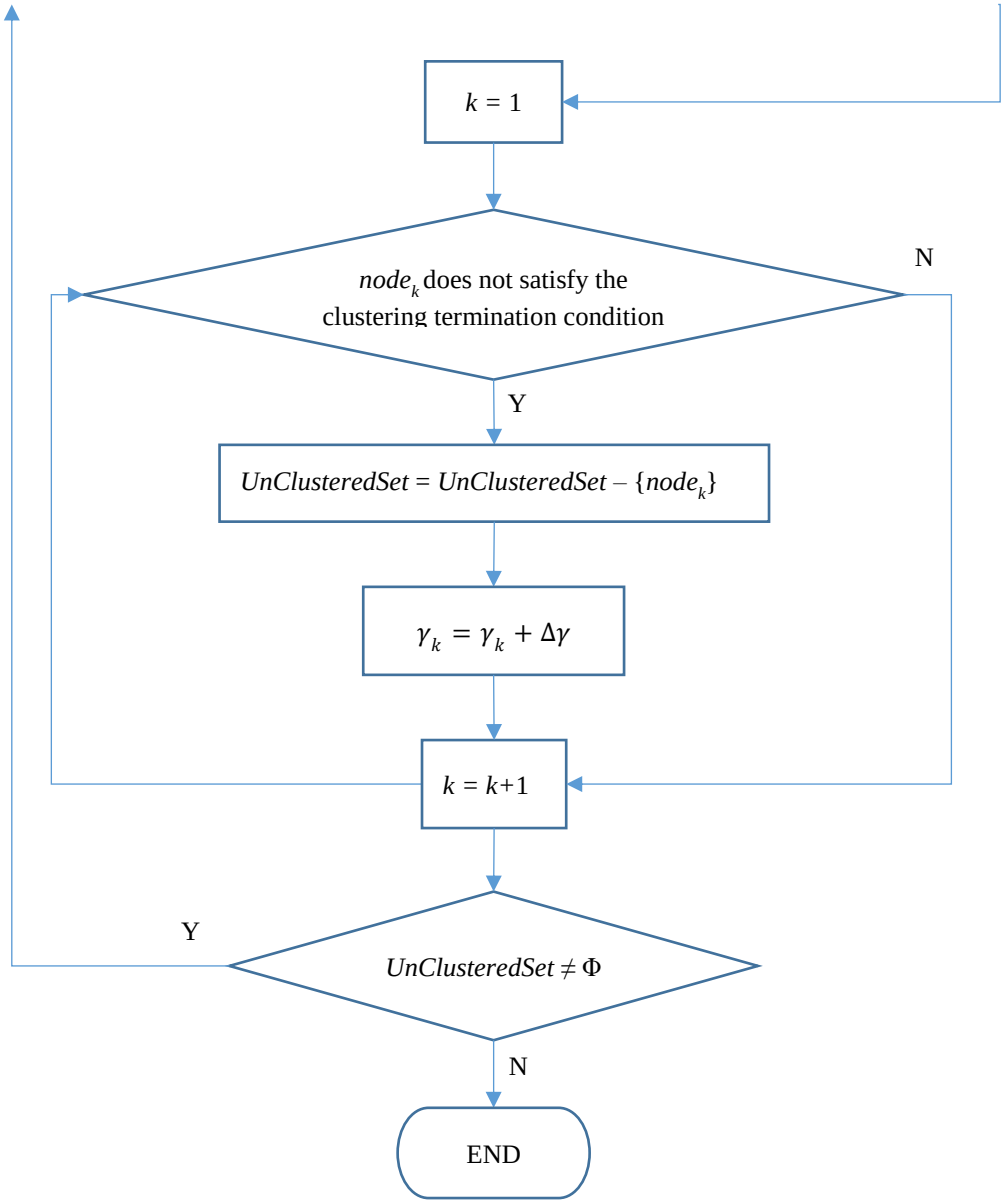


Figure 43 The workflow of the Algorithm 1.

Parallelization implementation.

The purpose of Algorithm 1 is to partition the information of large-scale networks, thereby reducing the consumption of computational resources (i.e., execution time and peak

memory consumption). Based on the α-approximation regional-division theory proposed and proved in this dissertation, which was applied in the partition procedure of this algorithm, the optimal solutions of the sub-networks are independent of each other. Therefore, parallel computing can be applied to calculate the MCF optimal solutions of the sub-networks.

For the network partition stage (clustering stage), in reality, the processing speed achieved by parallel computing will not be significantly higher. Since the time and space complexity of the algorithm of the clustering stage is approximately linear, the efficiency of this algorithm is high enough without applying parallel computing.

Code maintenance and version control plan.

For the sake of the code maintenance and version control, the detailed description of the program for each version of the code will be record in chronological order. And clear code comments also need to be documented for each version of the code.

Appendix B, Large-Scale Multi-Commodity Minimum-Cost Flows Problem

The proposed theorems and methodology in Chapter Three are concentrated on a single commodity situation, which may limit the range of application. It is worthwhile to extend the theorems and algorithm to a multi-commodity situation. The following is the preliminary ideas about the extension of the theorems. Further detailed proof is still needed.

Definition 1: **Comprehensive balanced area**: An area in which, for each type of commodity, the supply of origins and demand of destinations are balanced is called a comprehensive balanced area.

Definition 2: **Region cost vector of a comprehensive balanced area**: The *k*th element of the region cost vector of a comprehensive balanced area corresponds to the region cost of a comprehensive balanced area of the *k*th type of commodity. For the *k*th type of commodity, the region cost of a comprehensive balanced area is the maximum value of the minimum unit cost between any two nodes that are in this comprehensive balanced area.

Definition 3: **Cost vector between two comprehensive balanced areas**: The *k*th element of the cost vector between two comprehensive balanced areas corresponds to the cost between two comprehensive balanced areas of the *k*th type of commodity. For the *k*th type of commodity, the cost between two comprehensive balanced areas is the minimum value of the minimum unit cost between any two nodes that are in two different comprehensive balanced areas.

Multi-commodity regional-division theorem.

For all elements of a cost vector between any two comprehensive balanced areas, in which the link capacity constraints can guarantee the existence of the MCF solution of each comprehensive balanced area, if the *k*th element of the cost vector between any two

comprehensive balanced areas is greater than the *k*th element of the region cost vector of each of them respectively, the optimal multi-commodity MCF solution in any comprehensive balanced area will be consistent with the global multi-commodity MCF solution.

***α*-approximation multi-commodity regional-division theorem.**

For elements of a cost vector between any two comprehensive balanced areas, in which the link capacity constraints can guarantee the existence of the MCF solution of each comprehensive balanced area, if the *k*th element of the cost vector between any two comprehensive balanced areas is greater than or equal to $1/\alpha$ ($\alpha \geq 1$) times the *k*th element of the region cost vector of each comprehensive balanced area, respectively, the summation of the multi-commodity MCF objective value of each area is always less than or equal to a factor α of the global multi-commodity MCF objective value.

Appendix C, Referred Functions

Parallel performance ratio function

$$parallel\ performance\ ratio = \frac{global\ execution\ time - parallel\ execution\ time}{global\ execution\ time} \quad \text{(X.1)}$$

The link generalized cost function utilized in the six experiments of the numerical experiment chapter is introduced from Stabler (2016):

$$t_l = t_{ff} \times \left(1 + B \times \left(\frac{f}{C}\right)^p\right) \quad \text{(X.2)}$$

$$c_{link} = t_l + toll_{factor} \times toll + d_{factor} \times d \quad \text{(X.3)}$$

where t_l indicates link travel time, t_{ff} indicates free flow time, f is the link flow, C is the link capacity, c_{link} is link generalized cost, d is the distance, $toll$ is the number of tolls, and $toll_{factor}$, d_{factor}, B and p are related parameters.

Appendix D, Brief Primer on MB InSAR Phase-Unwrapping and Related Concepts

The frameworks of single-baseline (SB) phase unwrapping (PU) and multi-baseline (MB) PU can be demonstrated through the following: i) estimate the gradient of the absolute phase; ii) compute the residues; iii) distribute branch-cuts; iv) obtain the absolute phase through using an integration process. However, the biggest difference between SB PU and MB PU is on step 1 (SB PU is an ill-posed problem but MB PU is well-posed), in which SB PU uses a phase continuity assumption to estimate the absolute phase gradient, but MB PU uses the Chinese remainder theorem (CRT). Due to this difference, the residue polarities on SB and MB PUs are different. In SB PU, the residue polarity can only be ± 1, but can be $\pm n, n \in Integer$ in MB PU. The residue is the minimum loop-integration value of the phase gradients of any 2×2 neighboring pixels. Theoretically, the 2-D estimated gradient field should be irrotational so that the integration is independent of the integration path. However, because of the existence of the phase noise, the 2-D estimated gradient field is not irrotational, i.e., the integration path will affect the PU result. The meaningfulness of the ***residue*** is that, if and only if, all the positive and negative residues are connected by lines (called branch-cuts) and the integration path does not pass through any branch-cut, the integration result is independent of the integration path (like the residue in a Complex Function textbook). If we consider the positive residue as a supply-node and the negative residue as a demand-node, the flow paths obtained by the MCF will be branch-cuts. The residue polarities are the supply and demand (residue polarity does not have a physical unit) in this MCF network. From this view point, the MB PU problem is more similar to a transportation problem, because MB residue polarity is $\pm n$, but that of SB residue is just ± 1. In addition, the unit cost between two neighboring points (they could be supply, demand or transit

point) is set at one unit (again, there is no special physical meaning on this unit). In other words, the MCF network in the PU problem is an unweighted network. After we distribute branch-cuts obtained by the MCF solution, we can easily obtain the absolute phase by a flood-filling process (i.e., 2-D integration process).

In Experiment 6, the residue is computed by GAMMA remote sensing software (GAMMA 1995), so the input of ELS is produced by a 'canned' software. Then, the branch-cut information is obtained by ELS. Finally, the absolute phase is computed by GAMMA again with the branch-cut information provided by the ELS result. For more detailed information, the reader should refer to (Ghiglia & Pritt 1998).