UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

COMPONENT IMPORTANCE OF MULTI-COMMODITY NETWORKS

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE

By

MACKENZIE WHITMAN
Norman, Oklahoma
2016

COMPONENT IMPORTANCE OF MULTI-COMMODITY NETWORKS

A THESIS APPROVED FOR THE
SCHOOL OF INDUSTRIAL AND SYSTEMS ENGINEERING

BY

____
Dr. Kash Barker, Chair

____
Dr. Charles Nicholson

____
Dr. Suleyman Karabuk

To my family,
Who have loved me and supported me every day, all of my life.

Acknowledgements

I would first like to thank my thesis advisor, Dr. Kash Barker, of the School of Industrial and Systems Engineering here at the University of Oklahoma. I can truly, honestly say that I would never have made the choices I made that brought me here to this point without Kash's unfailing support and guidance. He brings the best out of his students and is never short of witty comebacks and snarky memes.

I would also like to thank Dr. Jonas Johansson from Lund University, Sweden for providing the Swedish Railway Network data used in this work and for the valuable feedback and insight used throughout this project.

Finally, I would like to thank all of the faculty from the school of ISE for the education I have achieved here over the last six years. During my time here at OU, I have received scholarships, employment, advisement, plenty of candy, and wonderful friendships from the dedicated and passionate professors and administrators that make up the school of ISE. Thank you.

Boomer Sooner!

Mackenzie Whitman

Table of Contents

List of Tables

List of Figures

Abstract

Preparedness planning for critical infrastructure networks requires evaluating the impact to the network when its components are disrupted. We extend the well-studied problem of component importance measures in single-commodity networks to multi-commodity networks by integrating a multi-commodity optimization model with a multi-criteria decision analysis tool to evaluate the impact of one-at-a-time component disruptions. We analyze commodity-specific impacts on network performance of a Swedish railway system application to rank critical links.

Chapter 1.0 Introduction

Critical infrastructure systems such as telecommunications, energy, water, and transportation provide essential services to society. With the advancements in technology, these services are becoming increasingly interdependent and create cascading impacts across systems when disrupted [54]. Disruptions can be caused by natural disasters, accidents, worker strikes, terrorist attacks and can cascade across infrastructures, modes, and regions [45]. The impact of disruptions can be reflected in the cost of recovery and/or the cost of delays such as Hurricane Sandy which cost over $117 million in debris cleanup or the I-35 W bridge collapse over the Mississippi River which cost an estimated $400,000 per day due to rerouting delays [41, 63]. There is an increasing interest in research and policy "to strengthen and maintain secure, functioning, and resilient critical infrastructure" [49] with multiple stakeholders and limited ability to adapt to rapidly changing risks [13]. In particular, the continuity of the transportation system is critical for the sustainment of other infrastructure systems and remains especially vulnerable to disruptions due to aging infrastructure [5, 45].

Transportation infrastructure is considered vital and fundamental to the United States' economy for the flow of goods through complex multi-modal networks of over four million miles of highway, 138,500 miles of rail, 11,000 miles of waterway, and an integrated network of airports [62]. In 2013, the U.S. moved a daily average of 55 million tons of freight valued at more than $49 million with trucks carrying the majority of the weight and value of freight [62]. With 42% of major urban highways congested and costing annually over $101 billion in wasted time and fuel, railway networks have experienced a resurgence as an energy-efficient alternative with over $75 billion invested in capital to reinforce the infrastructure since 2009 [5]. Recent interest in increasing resilience of critical infrastructure systems such as the rail network involves more than the current "patch and repair" mindset to maintenance and requires an understanding of the risks associated with disruptions and identifying vulnerabilities within a network [45]. With the renewed investment in rail networks, there is a need to increase the performance of the network for immediate gains in operational efficiency and to increase the infrastructure resilience for sustained long-term performance.

1.1 Network Vulnerability

In literature, there are many definitions for resilience and we define it here in two dimensions: vulnerability and recoverability as shown in Figure 1 adapted from Henry and Ramirez-Marquez [23]. Vulnerability describes the system's ability to mitigate impacts of a disruption and recoverability describes the system's ability to recover timely from a disrupted state [23, 24, 52]. In transportation, network vulnerability describes how disruptions reduce accessibility of network components which results in decreased system performance [7, 11]. A network is generally described as a set of nodes connected by a series of links. Network component vulnerability can be classified by either node vulnerability, the criticality of a node in system performance, or link vulnerability, reduction in system capability after selective link deletion [48]. We focus here on the vulnerability of a network defined by the magnitude of damage in system performance (i.e. change in commodity flow) when critical components, more specifically links, are disrupted [32]. Identifying critical links that have the largest impact on network performance will allow for a targeted resource allocation to the links that make the network most vulnerable to decreased system performance after a disruption.

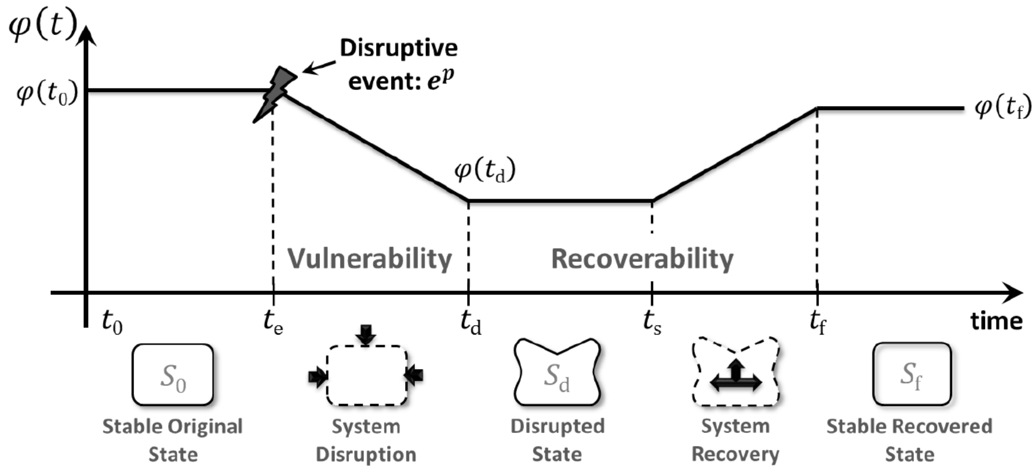

Figure 1. System Performance, $\varphi(t)$, over time adapted from Henry and Ramirez-Marquez [23].

There are three general approaches to evaluate network vulnerability: scenario-specific assessment, strategy-specific assessment, and mathematical modeling [43]. Scenario-specific assessments evaluate the impact of a losing a specific network component and its impact on network performance and is useful when applying relatively complex analytical approaches per scenario (e.g. [27, 36, 55]). Strategy-specific assessments evaluate network performance under a hypothesized sequence or strategy of disruption such as random link removal and is beneficial when assessing different network configurations to identical attack strategies for a comparison of effectiveness (e.g. [16, 31, 33]). Finally, mathematical modeling seeks to identify scenarios that have the greatest impact on network performance through simulation and establishes bounds on infrastructure vulnerability of the system (e.g. [20, 28, 59]). This work will focus on scenario-specific disruptions to identify critical links in a network by evaluating network performance when one link at a time is removed from the network for every link in the network.

When evaluating system performance, research generally classifies network component vulnerability measures as either graph theoretic measures known as structural vulnerability [31, 32] or flow-based measures known as functional vulnerability [51]. Graph theoretic measures are physical characteristics of the network such as average shortest distance, average edge betweenness, closeness centrality, etc., and is a well-studied area of research [15, 30, 60, 61]. Alternatively, evaluating network vulnerability from a functional vulnerability approach is a relatively new area of research that describes network vulnerability with respect to network flow such as the N-Q network performance/efficiency measure, flow capacity rate, edge flow centrality, etc. [44, 47].

Recent work on network vulnerability evaluates flow-based importance measures (IMs) by combining scenario-specific and strategy-specific disruption approaches to rank critical components of a network [6, 47]. When ranking critical components, IMs provide valuable information to decision makers such as identifying bottle necks or rerouting alternatives [12, 26]. Because each IM provides different information that can result in unique component rankings, there is a challenge for decision makers to use this information effectively. Hence, research in flow-based IMs has been expanded to integrate multiple-flow based measures with multi-criteria decision analysis tools such as TOPSIS or PROMEETHEE to provide a comprehensive ranking of critical components of a network based on multiple IMs [3, 4, 14]. However, to the authors' knowledge, no research has been found that integrates multi-commodity flow networks with flow based component importance measures.

1.2 Research Focus

In transportation, multiple types of goods are moved throughout the network and represent multiple stakeholders attempting to satisfy commodity-specific demand through a capacitated network. The added complexity of a multi-commodity flow might identify network components that are more important to specific commodities rather than looking at a single-commodity flow alone. We seek to answer: What links or group of links of a transportation network have the most impact on system performance from a multi-commodity flow perspective? Considering a multi-commodity flow in transportation networks is appropriate because of the regionalization of commodities based on historical movement of goods through a network and the difference in value each commodity might have to the decision maker. This research addresses (i) measuring multi-commodity vulnerability from a flow-based network performance approach, and (ii) using this multi-commodity vulnerability to rank critical links in a network that provides more holistic information than a single commodity flow approach.

This work expands on previous research on network vulnerability and flow-based link importance measures and is applied to a multi-commodity network flow optimization model. Given a set supply and demand in a deterministic capacitated network, a multi-commodity network is optimized to minimize total unmet demand. We evaluate network vulnerability by applying a one-link-at-a-time interdiction strategy and measure the drop in performance from a flow-based approach of each commodity moving through the network per scenario. This will result in system impacts per commodity of each link in the network that is then integrated with a multi-criteria decision analysis tool (TOPSIS) to consolidate multiple commodity-specific impacts into a single ranking that incorporates decision maker criteria and commodity-specific performance. This ranking of critical links would provide a different perspective of network vulnerability than analyzing total commodity movement alone.

This paper is arranged as follows. Chapter 2 gives brief definitions and notations, an overview of the multi-commodity network flow optimization framework, the network vulnerability performance measures, and an introduction to the multi-criteria decision analysis approach, TOPSIS, applied in this paper. The chapter concludes with the integrated framework of the multi-criteria decision analysis tool with the network vulnerability measures evaluated from the specific interdiction strategy applied to the multi-commodity optimization model. In Chapter 3, an illustrative example is presented of the Swedish railway system provided from publicly available data in collaboration with Lund University, Sweden and includes an overview of the data manipulation and key assumptions. The analysis is presented in Chapter 4 of the research methodology discussed in Chapter 2 applied to the network presented in Chapter 3. Chapter 5 provides concluding remarks and areas for future research.

Chapter 2.0 Research Methodology

This chapter describes the methodology used to define the multi-commodity network flow model, interdiction strategy, decision analysis method, and concludes with the integrated framework approach proposed in this paper.

2.1 Multi-Commodity Network Flow

Multi-commodity network flow models are used to solve various types of problems in transportation, supply chain, disaster relief, communication etc. with algorithmic study dating back to the 1970's. The model used in this paper is adapted from the *equal-flow problem* in which the flows through a given set of arcs are required to take equal values [9, 19, 38]. The *equal-flow* problem is a subset of the traditional multi-commodity network flow problem that seeks to minimize cost while satisfying demand [1]. The classic *minimum-cost multi-commodity flow optimization (MCMF)* framework has been modified in this paper to relax the constraint that all demand must be met and to remove the cost criteria from the objective function in order to adapt the model to an interdiction process discussed later in this chapter. With the original MCMF model, the model would be considered infeasible if demand is not fully met, so the constraint has been relaxed and transformed to the objective function that replaces cost criteria. This reflects a shift in model objective from an assumption that demand is always met with a minimal cost objective to a model where the goal is to meet demand, ignoring cost. The main goal of the multi-commodity network flow model in this paper is reduced to demand feasibility that seeks to measure how well the model responds to link closures in effectively rerouting data [12].

We begin with a directed graph denoted by $G = (V, E)$ where V is a set of n vertices or nodes and $E \subset \{(i, j): i, j \in V, i \neq j\}$ is a set of m directed links or arcs as shown in Figure 2.

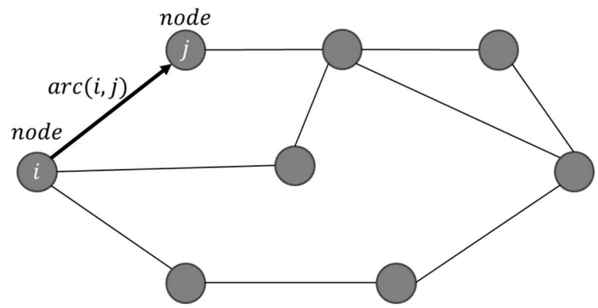

Figure 2. Example of directed graph $G = (V, E)$ with $arc(i, j)$ linking node i to node j.

Let there be K types of commodities, labeled by $\mathrm{k} = 1, \dots, K$ and for each arc (i, j), let the overall capacity per link be denoted by c_{ij} or $c(i, j)$ and the commodity-specific capacity per link be denoted by $c_{ij}^k = c_{ij}^1, \dots, c_{ij}^K$ or $c^k(i, j)$. We assume that commodity k is located at s_k different nodes within $\boldsymbol{S}^k$ indexed by $s_k = i_1^k, \dots, i_{\boldsymbol{S}_k}^k$, with amount of supply $\lambda_{i_n^k}$ at node i_n^k. The demand for commodity k is represented by d_k different nodes within $\boldsymbol{D}^k$ indexed by $d_k = j_1^k, \dots, j_{\boldsymbol{D}_k}^k$, with amount of demand $\mu_{j_n^k}$ at node j_n^k. A graphical representation of a two-commodity network with capacity, supply, and demand is shown below in Figure 3. From the figure, it can be seen that each node can be a sink for one type of commodity and a source for another, but not both a sink and source for the same commodity k.

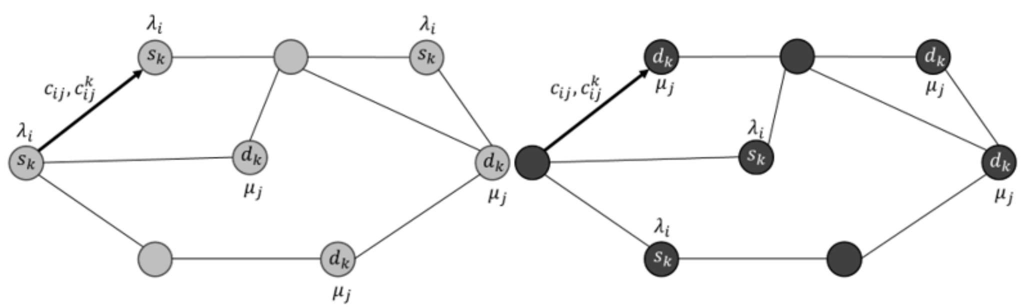

Figure 3. Two-commodity network example with each commodity shown separately (right, left) with s_k supply nodes and d_k demand nodes each with λ_i supply amount and μ_j demand amount. Each link (i, j) has link capacity, c_{ij}, and commodity-specific link capacity, c_{ij}^k.

To simplify the model, "supersource" and "supersink" nodes are introduced to separate the multiple sources and sinks for each commodity from the network [1, 38]. This reduces the multiple origins $\boldsymbol{S}^k$ and multiple destinations $\boldsymbol{D}^k$ into single origin S^k and single destination D^k nodes for each commodity k. The new "supersource" S^k and "supersink" D^k are connected only to the nodes in $\boldsymbol{S}^k$ and $\boldsymbol{D}^k$ with the following capacity parameters:

$$c(S^k, i_n^k) = c^k(S^k, i_n^k) = \lambda_{i_n^k}, \qquad c(i_n^k, S^k) = c^k(i_n^k, S^k) = 0, \qquad \forall\, i_n^k \in \boldsymbol{S}^k \quad (1)$$

$$c(j_n^k, D^k) = c^k(j_n^k, D^k) = \mu_{j_n^k}, \qquad c(D^k, j_n^k) = c^k(D^k, j_n^k) = 0, \qquad \forall\, j_n^k \in \boldsymbol{T}^k \quad (2)$$

From above, the first assignment in Eq. (1) sets the capacity parameters, c_{ij} and c_{ij}^k, for each newly added link from the "supersource" nodes S^k to each node i_n^k of $\boldsymbol{S}^k$ equal to the supply, $\lambda_{i_n^k}$. The second assignment in Eq. (1) ensures that there is no flow into node S^k and sets the capacity, c_{ij} and c_{ij}^k, for each newly added link from each node in i_n^k of $\boldsymbol{S}^k$ to zero. For Eq. (2), the reverse is applied to the "supersource" nodes D^k to ensure no there is no flow exiting demand node D^k, and the capacity entering the demand node D^k from j_n^k of $\boldsymbol{D}^k$ is equal to the demand amount, $\mu_{j_n^k}$. An example of the enlarged network $GE = (VE, EE)$ is shown in Figure 4.

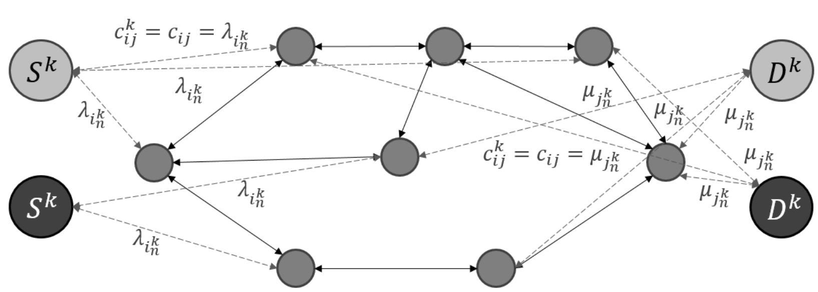

Figure 4. Enlarged two-commodity network with "supersource" nodes S^k and "supersink" nodes D^kadded per commodity k to the network and links (dashed) added for each node within $\boldsymbol{S}^k$ and $\boldsymbol{D}^k$.

2.2 Optimization Model

The following section describes the linear programming model modified from the classic *MCMF* problem discussed earlier in the section and breaks down the model components into the decision variables, objective, and constraints.

2.2.1 Decision Variables and Objective

Once the network has been enlarged, the multi-commodity optimization model can be formulated as a linear programming problem with the decision variable $x^k(i,j)$ as the flow of commodity k from node i to node j of the enlarged network GE. The objective is to minimize the sum of total unmet demand percentage per commodity k as shown below in Eq. (3) :

$$minimize: \qquad \sum_{k} \frac{\sum_{j_n^k}^{\boldsymbol{D}^k} \mu_{j_n^k} - \sum_{j_n^k}^{\boldsymbol{D}^k} x^k(j, D^k)}{\sum_{j_n^k}^{\boldsymbol{D}^k} \mu_{j_n^k}} \tag{3}$$

To calculate the unmet demand percentage per commodity k, the demand amount, $\mu_{j_n^k}$, for all demand nodes j_n^k of $\boldsymbol{D}^k$ is summed to give total demand for commodity k. This total is then subtracted from the total flow, $x^k(i,j)$, flowing from nodes j_n^k of $\boldsymbol{D}^k$ into the demand "supersink" D^k to give the unmet demand for commodity k. The percentage is then calculated from dividing the unmet demand by the total demand per commodity k. Finally, the unmet demand percentage calculated for each commodity k is then summed across k and minimized per the model objective.

2.2.2 Model Constraints

The last component of the model seeks to minimize the objective equation by modifying the decision variables and is subject to the following constraints:

$$x^k(i,j) \le c^k(i,j), \qquad \forall\ i,j,k \tag{4}$$

$$\sum_k x^k(i,j) \le\ c(i,j), \qquad \forall\ i,j \tag{5}$$

Shown above, Eqs. (4)-(5) shown above ensure that the commodity-specific link capacity and overall link capacity constraints are met. The constraint shown below in Eq. (6) is to balance the flow across the network and ensure that the flow into node i equals the flow out of node i for all nodes i in V. The index selection, $j: (j,i)$, used below would select all links (j,i) that flow into node i to represent all inflow while $j: (i,j)$ represents the outflowing links (i,j) and is applied for every k commodity.

$$\sum_{j:(j,i)}^{j} x^k(j,i) = \sum_{j:(i,j)}^{j} x^k(i,j)\,, \qquad \forall\ i \in V, k \in K \tag{6}$$

The next constraints shown below in Eqs. (7)(8) deal with all links leading to and from the "supersource" nodes S^k and "supersink" nodes D^k. For all commodities k, all flow, $x^k(i,j)$, out of S^k nodes and all flow, $x^k(i,j)$, into D^k nodes must be less than the supply amount $\lambda_{i_n^k}$ and demand amount $\mu_{j_n^k}$, respectively. Alternatively, all flow into supply nodes S^k and out of demand nodes D^k must equal zero.

$$\sum_{j:(j,i)}^{j} x^k(j,i) \le \lambda_{i_n^k}\,, \qquad \sum_{j:(i,j)}^{j} x^k(i,j) = 0, \qquad i = S^k, \forall\ k \tag{7}$$

$$\sum_{i:(i,j)}^{i} x^k(i,j) \le\ \mu_{j_n^k}, \qquad \sum_{i:(j,i)}^{i} x^k(j,i) =\ 0, \qquad j = D^k, \forall\ k \tag{8}$$

Finally, the last key constraint made in the multi-commodity network flow optimization model is that $x^k(i,j) \ge 0$, and must be integer values only. By restricting the decision variables solution space to integer values only, the computation time to solve large networks is greatly reduced. In order to solve multi-commodity network flow models, Gurobi optimization software has been used to efficiently solve integer multi-commodity flow problems with a built-in solver to detect the most efficient algorithm based on the problem structure [21]. In this instance, the dual simplex algorithm, for full details see [53], was detected to be the most efficient algorithm for this problem structure and was used to solve the given problem application which is discussed later in Chapter 3, for further details see Appendix A.2.

2.3 Network Interdiction Approach

In this section, two main components of the network interdiction approach used in this paper are defined: disruption scenarios and system performance metrics.

2.3.1 Disruption Scenario

Network interdiction is a common evaluation method for network vulnerability analysis of network-based critical infrastructure such as transportation or telecommunications. There are different approaches to network interdiction and they generally fall into three categories: scenario-specific, strategy specific, and mathematical modeling. We focus here on scenario-specific assessment of vulnerability which seeks to measure impacts of a specific set of disruption scenarios to identify the subset of disruption scenarios that result in the most damage to the network. The results of scenario-specific assessments depend greatly on the defined disruption scenarios and the selected system performance measures. Generally, disruption scenarios describe the set of network components impacted, the decreased functionality of the disrupted components, and the baseline operating conditions prior to disruption. Link disruption can be reflected as either completely obstructed, similar to a road closure, or only partially disrupted such as an accident blocking a single lane of the interstate. Once disruption scenarios are defined, impacts can be evaluated and compared between disruption scenarios. Decision makers concerned with network vulnerability are usually interested in identifying network components that result in the most damage to network performance which makes scenarios-specific disruptions ideal for identifying critical network components [43].

In this paper, we define a disruptive scenario as the removal of a particular link between nodes i and j that leads to a decreased network accessibility and performance. Once a link is disrupted, the model has two options: reroute the flow of goods through the remaining capacity of the network or hold freight until the link is restored. Based on the model objectives discussed previously, the model will always seek to reroute instead of holding until the link is restored. When the disruption scenario is applied to a directed graph $GE = (EE, VE)$ of an enlarged multi-commodity network outlined earlier, a removed link (i, j) would impact the flow for all k commodities across $x'(i, j)$ and $x'(j, i)$ which would result in the following constraint being added to the previously defined model shown below in Eq. (9):

$$x_{ij}'^{k} = x_{ji}'^{k} = 0, \qquad \forall\, k \in K \tag{9}$$

Eq. (9) ensures no flow is allowed between node i and node j of the disrupted links, (i, j). By defining each disruption scenario as a one-at-a-time link removal strategy, the most critical links can be identified from the set of all possible links in the network. This does not include the links added in the enlarged network $GE = (EE, VE)$, but the set of the original links in E.

2.3.2 System Performance

Once the disruption scenarios are defined, system performance metrics are selected to reflect network vulnerability. These component importance measures are calculated for each disruption scenario and are the basis for identifying critical links. This work builds upon recent research by selecting network performance measures rather than graph theoretic measures and expands to a multi-commodity perspective [3, 47]. Two component importance measures, change in unmet demand percentage and link usage count > 90%, are selected and defined below in Eqs. (10)(11):

$$\frac{\sum_{j_n^k}^{\boldsymbol{D^k}} \mu_{j_n^k} - \sum_{j_n^k}^{\boldsymbol{D^k}} x^k(j, D^k)}{\sum_{j_n^k}^{\boldsymbol{D^k}} \mu_{j_n^k}}, \qquad \forall\, k \tag{10}$$

$$\sum_{(i,j)}^{V} \frac{x^k(i,j)}{c^k(i,j)} \geq 90\% = 1,\ else\ 0 \tag{11}$$

Unmet demand percentage per commodity k is the first component importance measure selected as shown above in Eq. (10) and reflects the network's ability to reroute data once a link is disrupted as shown in the first equation above. This importance measure is represented as a percentage of total demand per commodity k, so that each commodity is treated equally, regardless of commodity volume in the network. The second importance measure, link usage count > 90%, is shown in Eq. (11) and indicates how likely a removed link is to create bottlenecks. For every link in the network, the link usage percentage is calculated by dividing the flow, $x^k(i,j)$, by the commodity-specific capacity, $c^k(i,j)$. If the link usage is greater than 90%, it is considered a potential bottleneck for that specific commodity and is given a count of 1, otherwise 0. The link usage count is then calculated by counting all edges in the network with link usage greater than 90% for each commodity k. Each importance measure provides different information about the network and are both used in the vulnerability analysis.

Applying the one-link-at-a-time removal strategy results in component importance measures for each link removed. In order to evaluate critical links, the component importance measurements from the interdiction process must be compared to the baseline optimization flow system performance. This is to ensure that the impact from the link removal is fully captured and does not include baseline performance results. The two baseline component importance metrics, unmet demand percentage and link usage count, u_k^b and l_k^b per commodity k are subtracted from the interdiction impacts, u'_k and l'_k , to reflect net change $\Delta u_k$and Δl_k for every link (i, j) in E disrupted as shown below in Eqs. (12)(13):

$$\Delta u_k = u_k^b - u'_k \qquad (12)$$
$$\Delta l_k = l_k^b - l'_k \qquad (13)$$

The net change component importance measure for every edge removed provide commodity-specific impacts must then be aggregated in some way to provide a single critical link ranking.

2.3.3 Decision Analysis

In order to combine commodity-specific component importance measures in a weighted fashion to rank critical network components, we make use of TOPSIS, or the Technique of Order Preference Similarity to the Ideal Solution. Often decision makers are not necessarily interested in making the best choice among several alternatives, but avoiding the worst [22]. TOPSIS addresses this idea based on the philosophy of the compromise solution, providing a ranking of alternatives according to their (shortest) distance from the best alternative for a particular criterion and the farthest distance from the worst alternative for that criterion [25]. The simplicity of TOPSIS makes it an appealing decision analysis tool with a variety of applications in supply chain logistics, engineering and manufacturing systems, energy management, water resources management, and many others [3, 57]. Recent interest in critical network component ranking has incorporated TOPSIS with flow-based network performance criteria, but has not yet been expanded to rank critical components of multi-commodity flow networks and is part of the proposed approach of this paper to provide comprehensive rankings that incorporate multi-commodity impacts [4, 14].

Two important inputs for the TOPSIS model are the performance of alternatives and criteria weights. The performance of m alternatives a with respect to k criterion b are collected in a decision matrix $\boldsymbol{Y} = (y_{ab})$ where $a = 1, \dots, m$ and $b = 1, \dots, k$. In this specific problem, the units for all performance criteria correspond to the importance measure being considered which is either net unmet demand percentage b for k commodities or net link usage count > 90% b for k commodities described earlier in Eqs. (12)-(13). Each alternative, a, corresponds to each link evaluated in graph G. The corresponding weight w_b per decision criteria k are determined by the decision maker and will be used to determine commodity importance in this specific problem described in Chapter 4. The selection of criteria weights has a significant impact on the final solution and should be determined by the decision maker with domain experience [50].

The first step of the TOPSIS method is to normalize the different performance criteria in order to compare performance criterion of different units. There are several normalization techniques, but the distributive normalization is proven to be the most consistent and is applied as shown in Eq. (14) [10].

$$r_{ab} = \frac{y_{ab}}{\sqrt{\sum_{a=1}^{m} y_{ab}^2}} for\ a = 1, \dots, m\ and\ b = 1, \dots, k. \tag{14}$$

Once the performance data is normalized, the weights are taken into account by multiplying the normalized scores r_{ab} by their corresponding weights w_b as shown below in Eq. (15).

$$v_{ab} = w_b r_{ab} \tag{15}$$

The weighted scores, v_{ab} will be used to compare each element to the ideal and anti-ideal criteria for each industry as shown in Eqs. (16)-(17). The ideal solution describes the most beneficial outcome while the anti-ideal solutions describes the most disadvantageous outcome. The ideal solution corresponds to $v_b^+ = \min_a(v_{ab})$ because criterion b is to be minimized and $v_b^- = \max_a(v_{ab})$ corresponds to the anti-ideal solution for all alternatives for each criterion.

$$A^+ = (v_1^+, \dots, v_k^+) \tag{16}$$
$$A^- = (v_1^-, \dots, v_k^-) \tag{17}$$

Once the ideal, A^+, and anti-ideal solutions, A^-, are determined for each performance criterion, the Euclidean distance from the ideal and anti-ideal solution is calculated for each element as shown in Eqs. (18)(19).

$$d_a^+ = \sqrt{\sum_b (v_b^+ - v_{ab})^2}, \;\; a = 1, \dots, m \tag{18}$$
$$d_a^- = \sqrt{\sum_b (v_b^- - v_{ab})^2}, \;\; a = 1, \dots, m \tag{19}$$

Finally, the relative closeness coefficient is calculated and is shown in Eq. (20). The closeness coefficient is always between 0 and 1 with scores closer to 1 being closer to the positive ideal solution and scores closer to 0 being closer to the negative ideal solution. The Matlab function used to calculate the proposed steps in Eqs. (14 -20) is provided in Appendix A.3

$$C_a = \frac{d_a^-}{d_a^+ - d_a^-} \tag{20}$$

2.4 Integrated Framework

The integrated framework is shown below in **Figure 5** and combines the previously discussed methods into a three step process to evaluate critical component importance of multi-commodity networks. First, the baseline optimization model is optimized and the baseline system performance is evaluated. Second, links are removed one-at-a-time for each edge in the network and the system performance is evaluated per commodity k based on the component importance measures defined as unmet demand percentage and link usage count. Once the system impacts are evaluated for every disruption scenario, they are compared against the baseline performance to calculate net change per importance measure which corresponds to performance criteria for alternative edge (i, j). These performance criteria are then used as a data input for the decision analysis. Step 3 provides a single ranking of critical components in the network that incorporates the multi-commodity impacts from Step 2 along with decision maker criteria for the weights.

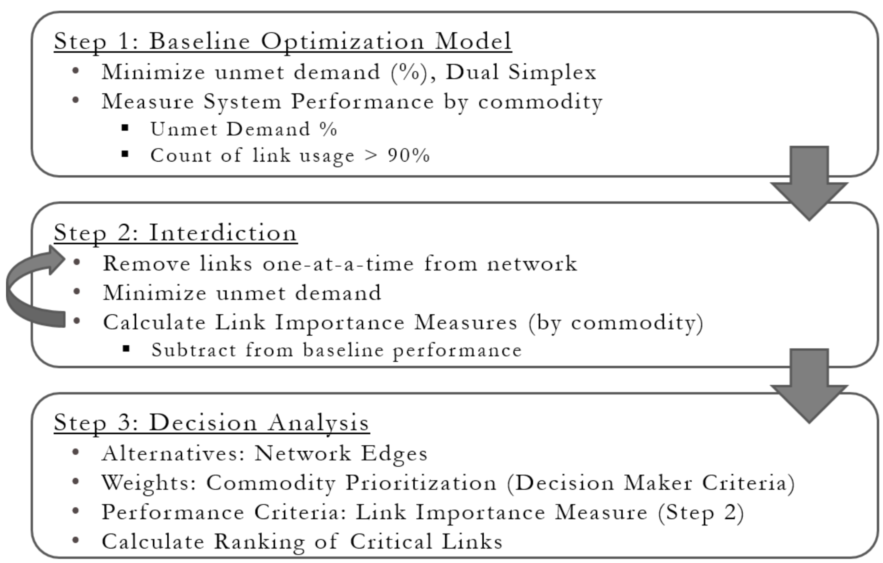

Figure 5. Proposed approach to assessing critical component importance with multi-commodity impacts on network vulnerability.

Chapter 3.0 Network Application

In this section, the proposed application, the Swedish Railway System, is introduced and briefly described. The Swedish Railway data used for this project, also referred to as SwRail, was provided in collaboration with Dr. Jonas Johansson of Lund University, Sweden. The freight transported throughout the Swedish Railway System was collected for the year 2012 from numerous publicly available data sources [29]. In order to apply the methodology discussed in Chapter 2, the data provided was modified to create the graph $G = (V, E)$ discussed in Chapter 2.1 and will be described in further detail later in this chapter.

3.1 Swedish Railway Network

The Swedish Railway Network is a system of connected train stations, tracks, and equipment that is operated by both public and private train operators for both commercial freight and passenger transport located in the third largest county in the European Union. With over 13,000 km (over 8,000 miles) of track, the Sweden's rail network is ranked 21st in the world in size and transported over 7 billion passengers and 65,000 metric tons of freight in 2015 [8, 46]. Sweden's rail network is a vital mode of transportation for Sweden's economy with over 30% of inland freight transported by rail in 2014 (most recently available data) [17]. The 20 different types of commodities transported in the network and the amount transported, in kTons, are shown on the next page in

Table 1, sorted by volume:

Table 1. Total amount of commodity type, k, transported in kTons for year 2015 (NST 07 groupings), sorted by kTons [18].

Commodity	Commodity Group Name	Transported (kTons)	% of Total	Cumulative % of Total
3	Ore	27,829	42.81%	42.81%
19	Unidentifiable goods	9,218	14.18%	57.00%
1	Agriculture, Forrest, Fishing	8,859	13.63%	70.62%
6	Wood, Cork, Pulp, Paper	6,081	9.36%	79.98%
10	Fabricated metal products	4,768	7.34%	87.32%
14	Return materials and recycling	1,557	2.40%	89.71%
7	Petroleum products	1,437	2.21%	91.92%
8	Chemicals, rubber, plastics	1,290	1.98%	93.91%
16	Equipment for transportation	1,002	1.54%	95.45%
12	Transport equipment	927	1.43%	96.87%
4	Food, Beverage, Tobacco	846	1.30%	98.18%
9	Other non-metallic mineral	339	0.52%	98.70%
18	Loader and grouped goods	264	0.41%	99.10%
15	Post and packages	233	0.36%	99.46%
2	Coal, Crude oil, Natural gas	196	0.30%	99.76%
13	Furniture, Other manufactured	77	0.12%	99.88%
11	Machinery and equipment	76	0.12%	100.00%
5	Textile, leather	1	0.00%	100.00%
17	Moving Goods, vehicles for repair	0	0.00%	100.00%
20	Goods not in group of 1-19	0	0.00%	100.00%

The top 5 commodities in the table above accounts for over 87% of the goods moved throughout the network. By far, commodity group 3, Ore, is the largest amount shipped which is expected since the Kiruna mine is the largest underground iron ore mine in the world located in the northern most part of Sweden [42]. Considering this, decision makers in Sweden might want to protect ore due to its importance to Sweden's economy over other types of commodities.

Recently, Sweden's rail traffic volume has increased by 5% from 2006 to 2010 in both goods and passengers which corresponds to a more rapid degradation of the rail infrastructure [2]. Similar to the United States, the government of Sweden is invested in maintaining current system accessibility and long term sustainability [56]. There is much research interest in the efficiency, capacity, utilization, and sustainability of the rail network and current research interests seek to develop a long-term maintenance strategy that has the largest impact with limited resources [2, 34, 35, 37, 39, 40]. An understanding of what links cause the most damage to the network performance with specific commodities in mind would lead to targeted maintenance strategy, more efficient utilization of resources, and improved resilience of the system long-term.

3.2 SwRail Data

The data provided for the Swedish Railway Network, otherwise referred to as SwRail, was obtained from publicly available data sources and was aggregated to protect any sensitive information. The key data inputs and their descriptions are shown below in Table 2.

Table 2. Important data fields from the SwRail data used to generate the network graph $G = (V, E)$.

SwRail Data Field Name	SwRail Data Field Description
SwRail.ICM	InterConnectionMatrix (ICM) with size equal to number of nodes. 1=link exists between nodes and 0=no link exists between nodes. Mirrored: link(i,j) = link(j,i) = 1.
SwRail.x, SwRail.y	Coordinates (x,y) for the nodes. Used for plotting the station locations on a map. Distance between stations not used in scope of project.
SwRail.Routes	Structure that contains the data for all unique 1,091 discrete routes with a unique origin-destination path.
Routes.NodeRoute	List of nodes on path for unique route that is direction sensitive of the stations the train passed on its scheduled origin/departure route.
Routes.kTon	The amount (in kTon) and type of commodity transported on specific route. (~estimated)
Routes.NbrTrainsPerYear	Total number of trains per year (2012) that took this unique route. Based on collected train schedule data (commercial freight trains only).

From the previous table, the structural characteristics of the network were obtained from the SwRail.ICM table which lists all of the nodes, V, in the network in an adjacency matrix that defines all of the links, E, in the network. The SwRail.ICM table defines the structure of the graph $G = (V, E)$ and contains 1,363 stations connected by 1,439 bidirectional links. To apply the methodology described in Chapter 2, there are two main components missing from the network: capacity and supply/demand parameters. In order not to disclose sensitive information, the freight movement was aggregated to the level of "cargo routes" and is not given for any specific train operator or specific train cargo. Therefore, these parameters must be estimated from the provided data by applying assumptions to the network data.

The SwRail data field used to estimate the missing network parameters, SwRail.Routes, describes the movement of cargo through the structural network graph $G = (V, E)$. The Routes data is defined by an origin and destination pair with a specific network route, Routes.NodeRoute from Table 2, that lists the stations passed throughout its journey. For example, one unique train route could be described with an origin at node 153 that passes two intermediate nodes, 152 and 151, on the path to its final destination at node 77. In addition, train schedule data was used to derive the number of trains that traveled a specific route for the year of 2012 (Routes.NbrTrainsPerYear) which could indicate the rate of freight movement with popular routes receiving higher volume than others. Referring to the previous example, 1,527 trains, approximately 4 trains per day, operated the specified path from station 153 to station 77 according to the provided SwRail.Routes data which can be assumed to carry more freight than a route that only schedules one train per week. The actual cargo amount data was collected from a separate source voluntarily provided by train operators and was aggregated to the total amount and type of freight moved in 2012 and what routes their freight was transported on. This data is modified by applying key assumptions to estimate the missing parameters needed for the analysis. The next section gives an overview of the modification to the SwRail data provided and outlines the key assumptions used to generate the SwRail network.

3.3 SwRail Modification

Due to the level of aggregation of the provided data, assumptions must be applied to the SwRail data to estimate capacity and supply/demand parameters for the network. Each of these parameters are discussed separately in this section and the Matlab code that applies all of the assumptions discussed can be found in Appendix A.1 which generates the multi-commodity network used in the analysis in Chapter 4.

3.3.1 Supply and Demand

The two components used to define the supply and demand parameters are the node locations for both supply and demand as well as the amount, measured in kTons (1000 metric tons), for each commodity. To derive these, source and sink locations must be selected among the available nodes in the network and freight must be distributed from the total amount of each type of commodity. In order to solve this network flow model, a node can't be both a sink and a source location for a commodity which might not reflect real-world operating conditions. The steps used to select source and sink locations are described below:

1. Loop through every route of unique origin-destination paths (1,091 total).
2. Sample source and sink nodes proportionate to the length of the path (sample a lot of source/sink nodes on very long paths and only a small number on paths with few stations visited). This does not assume how many stops a train makes on a schedule, just possible origin and destinations a train could pick up and deliver freight to.
    - Only sample nodes from routes where an operator could have carried that particular commodity.

3. Remove duplicate source/sink assignments. If a node has been sampled as both a source and a sink, an alternate method is used to determine assignment (see Appendix A.1).
4. Distribute kTon to each route if there are source/sink nodes located on that route. Once a node is selected as a source or sink, it is a source or sink for all routes it belongs to.
    - kTon amount proportionate to the number of trains scheduled on route.
5. Distribute freight from route to individual source/sink nodes (random).
6. Repeat for every commodity type results in unique source/sink nodes selected for each commodity type.

The steps briefly described above results in a sample of sources and sinks that are located on paths that operators could have shipped that commodity over in the last year. The freight amount distributed to the source and sink locations is proportionate to the trains scheduled over routes which supports the assumption that a node that is part of several routes probably receives more freight than a node with only one scheduled infrequently. The actual source/sink locations may be different in reality, but are a realistic interpretation of the provided dataset. In the next section, a brief overview is given of the assumptions used to estimate capacity parameters.

3.3.2 Capacity

We define capacity here as the maximum amount of flow allowed, measured in kTons (1,000 metric tons), across each link in the network. This is further broken down to a general link capacity shared among commodities, c_{ij}, and a commodity-specific capacity, c_{ij}^k, which restricts the amount of flow of commodity k across each link in the network. Some key assumptions are applied to the SwRail data to estimate these parameters. First, we assume the number of trains per year reflects the capacity of the network in that no additional trains can be scheduled. This constrains the number of trains that flow across each link and is derived directly from the SwRail.Routes data. However, due to the level of aggregation of the data, there is no indication of how much freight was transported per train, maximum amount of each commodity that would fit on a train, or other information that would allow a conversion of trains per link to kTons per link. When determining the maximum amount, in kTons, of a particular commodity allowed per train, the density of freight could vary by commodity which restricts the total amount that would fit on a train. For example, furniture is significantly less dense than a heavy metal like steel and would be expected to have a much smaller capacity, c_{ij}^{k}, than steel since you can transport more kTons per train of steel than of a less dense freight like furniture. Other factors that could limit the amount of a particular commodity could be consignment size in that some commodities might only be ordered in small frequent batches while others might be ordered infrequently in very large quantities.

The second assumption made to estimate commodity-specific link capacity was to calculate the average amount of kTon per train by dividing the total amount of kTon shipped in 2012 by the number of trains that commodity could have been shipped on. This average was calculated from the SwRail.Routes data and was used to estimate commodity-specific capacity, c_{ij}^{k}, per link by multiplying the average by number of trains per link. The overall link capacity, c_{ij}, was assumed to be the maximum of the calculated commodity-specific link capacities. Since the commodity-specific link capacities are calculated from an average, it is assumed that the commodity-specific link capacities might not be a true limit per train, but some other constraint like consignment sizes (the size of the order determined from the buyer). Therefore, the overall link capacity reflects the assumption that commodities are competing for a shared resource that can't be satisfied for all commodities.

Applying the previously mentioned assumptions results in a capacitated network based on existing freight movement in the network. Inherently, this creates a network that when fully optimized, might not have enough slack in the calculated capacity to allow for rerouting. To account for this, adjustment factors were applied to both parameters to ensure both the feasibility of meeting the demand and to allow enough slack for rerouting of disrupted freight.

The goal of calculating the capacity parameters for the modified SwRail network was to derive them from the provided historical data. Some key assumptions were applied to estimate parameters that are able to fully satisfy baseline total demand for all commodities with a limited amount of excess capacity for rerouting. The final modified SwRail network graph is presented in the next section and is analyzed in Chapter 4 with the methodology described in Chapter 2.

3.4 SwRail Network Graph

The final SwRail network graph is generated from modifying the provided data on the Swedish Railway Network, using assumptions discussed in the previous sections. The final network graph $G = (V, E)$ is summarized on the next page in Table 3. The first two columns label and describe the 20 different commodity types transported in the network and columns 3 and 4 present the total amount in kTons (column 3) and percentage of total freight (column 4) moved in 2012 derived directly from the SwRail data. The last five columns describe the parameters that were estimated and include the number of sinks and sources, the average sink and source size, and average overall link capacity. This data is presented to demonstrate the large number of sources and sinks selected per commodity k with the exception of commodity 2. In addition, the size of the sink or source is proportionate to the demand of that commodity and the average link capacity is significantly larger than the average size of any sink or source.

Table 3. Network summary of modified SwRail network by commodity group, k.

Commodity	Commodity Group Name	Demand (kTon)	Demand % of Total	Number of Sources	Number of Sinks	Avg. Source Size (kTon)	Avg. Sink Size (kTon)	Average Capacity (kTon)
1	Agriculture, Forrest, Fishing	8463	13.35%	228	284	37.12	29.80	955.00
2	Coal, Crude oil, Natural gas	280	0.44%	27	19	10.34	14.69	209.94
3	Ore	29427	46.41%	210	262	140.12	112.31	3156.89
4	Food, Beverage, Tobacco	250	0.39%	281	366	0.89	0.68	27.88
5	Textile, leather	1	0.00%	240	262	0.00	0.00	0.00
6	Wood, Cork, Pulp, Paper	4701	7.41%	245	276	19.19	17.03	614.94
7	Petroleum products	1410	2.22%	198	217	7.12	6.50	221.39
8	Chemicals, rubber, plastics	1257	1.98%	186	187	6.76	6.72	174.44
9	Other non-metallic mineral	449	0.71%	270	258	1.66	1.74	50.04
10	Fabricated metal products	4017	6.33%	216	193	18.60	20.81	559.82
11	Machinery and equipment	95	0.15%	263	251	0.36	0.38	10.95
12	Transport equipment	894	1.41%	240	269	3.72	3.32	100.41
13	Furniture, Other manufactured	59	0.09%	248	239	0.24	0.25	8.33
14	Return materials and recycling	1125	1.77%	256	380	4.39	2.96	118.62
15	Post and packages	0	0.00%	0	0	0.00	0.00	0.00
16	Equipment for transportation	1187	1.87%	238	260	4.99	4.56	135.24
17	Moving Goods, vehicles for repair	0	0.00%	0	0	0.00	0.00	0.00
18	Loader and grouped goods	39	0.06%	287	241	0.14	0.16	5.29
19	Unidentifiable goods	9738	15.36%	293	267	33.23	36.47	1024.14
20	Goods not in group of 1-19	18	0.03%	227	195	0.08	0.09	2.81

A visual of the distribution of sources and sinks are shown below in Figure 6 for commodity 1 and 2. The sources are shown in blue and the sinks are shown in red with the circle size proportionate to the amount of supply and demand for that commodity overlaid on a map of the actual network. As you can see from the figure, commodity 2 source and sink locations are restricted to the northern region of Sweden and there are some portions of the network that do not have a sink or source located on them for that commodity. All commodities' source and sink locations, except 15 and 17 for which there is no demand, are shown in Figure 7 and Figure 8 to give a visualization of the size and complexity of the generated SwRail network that is analyzed in Chapter 4.

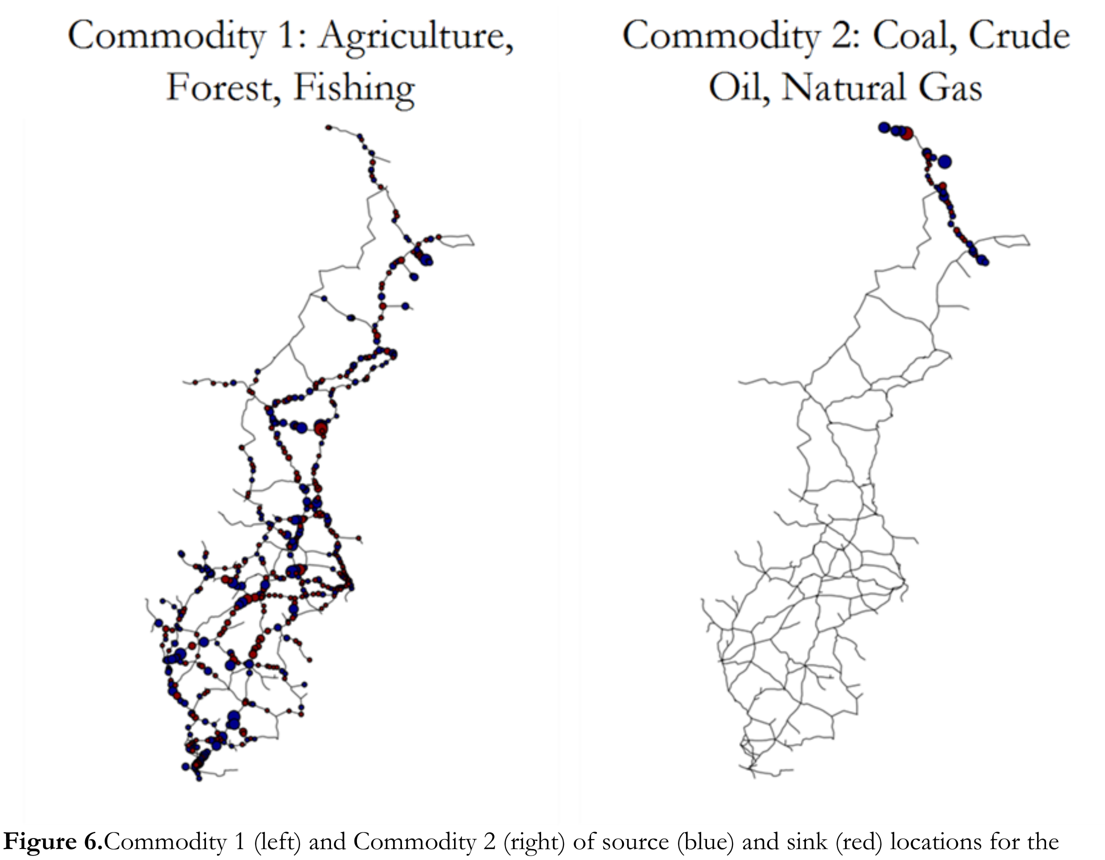

Figure 6. Commodity 1 (left) and Commodity 2 (right) of source (blue) and sink (red) locations for the modified SwRail network

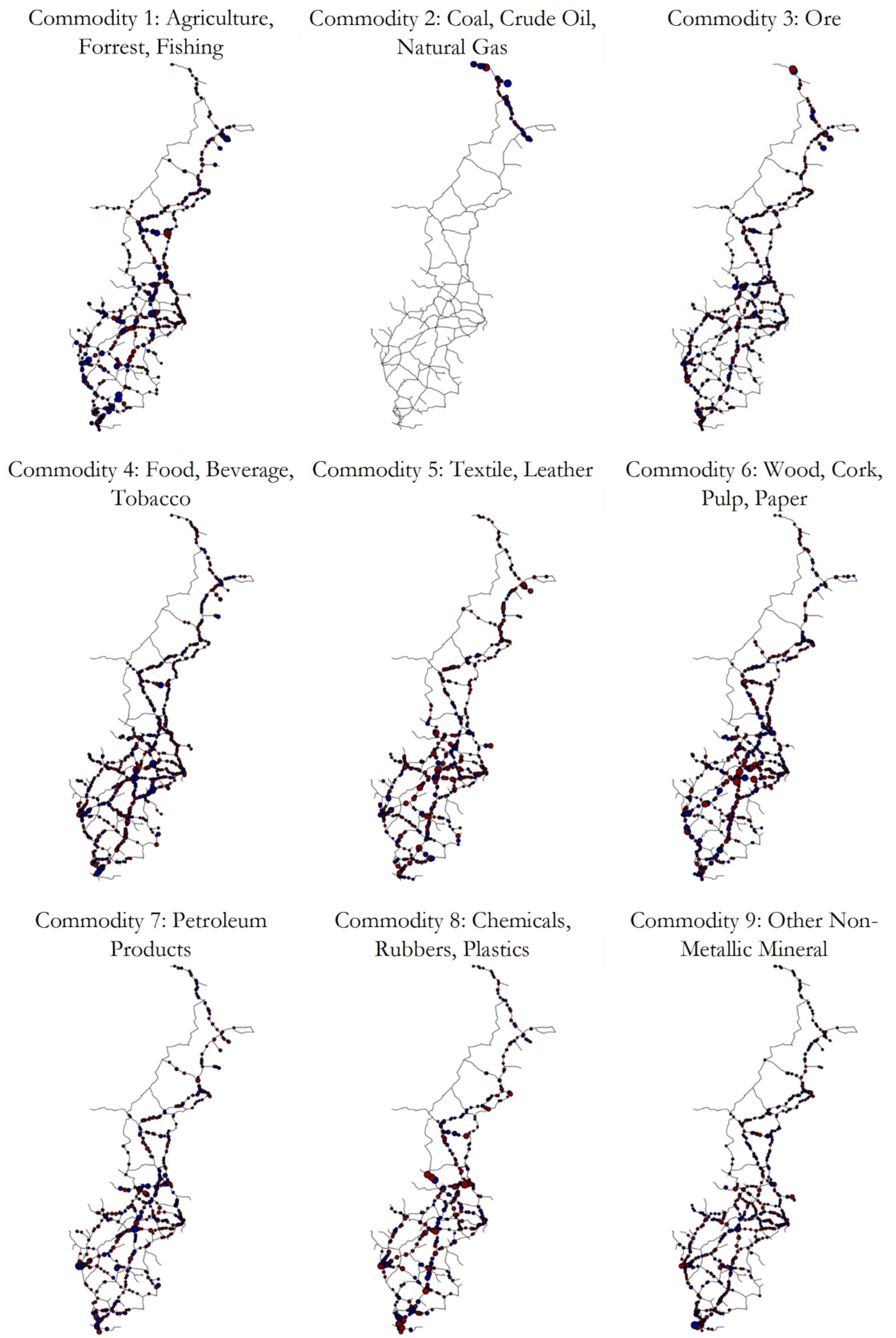

Figure 7. Commodities 1-9 Sink/Source locations of the modified SwRail network.

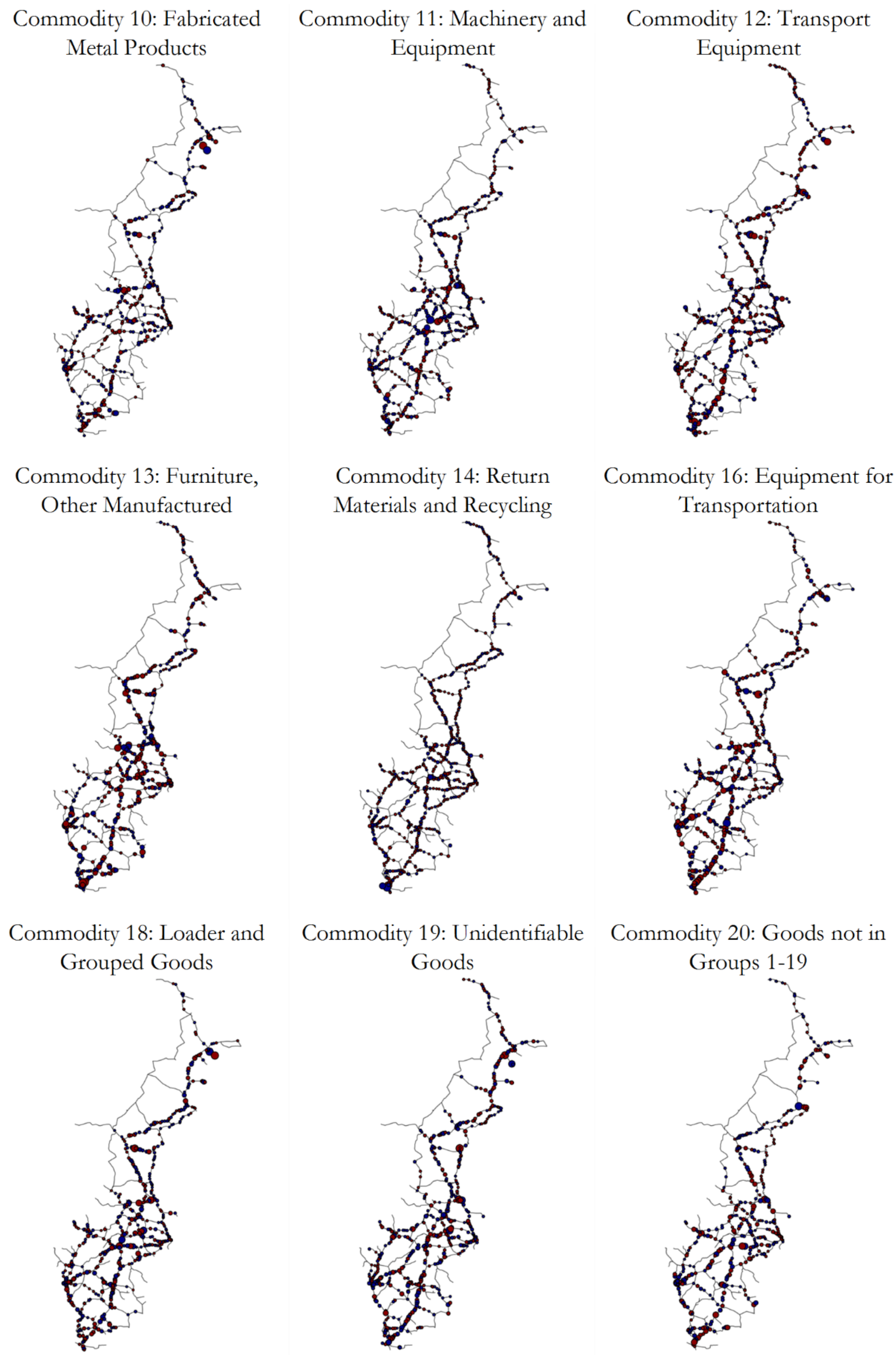

Figure 8. Commodities 10-20 (except 15 and 17 since no demand present) of the modified SwRail Network.

Chapter 4.0 Analysis

In this section, the methodology framework described in Chapter 2 is applied to the generated Swedish Railway Network in Chapter 3 and is organized in three steps referenced previously in Figure 5: baseline optimization, interdiction strategy, and finally, decision analysis.

4.1 SwRail Baseline Optimization

The network generated in Chapter 3 was optimized from the modified *MCMF* model discussed in Chapter 2 and the objective was to minimize the summation of total unmet demand percentage by commodity. The performance of the baseline optimization model was defined by the system performance measures selected in the methodology, unmet demand % and link usage.

The results of the demand feasibility model are shown in Table 4 in regards to unmet demand percentage. From the table, the first two columns list the commodity groups while column 3 and 4 outlines the total demand in the network. The results of the optimization model are shown in the last four columns in different perspectives. The total flow in kTon that was able to satisfy demand is shown in column 5, which leaves demand unsatisfied in column 6. The last two columns give perspective of how much demand is met per commodity by percentage, and the relative unmet demand across commodities. For example, commodity 1, Agriculture, Forrest, and Fishing, resulted in less than 1% of its total commodity demand unmet, but accounted for 35.9% of the total unmet demand. Overall, over 99% of the total demand was met and at least 98% of total demand was met for any commodity.

Table 4. Unmet demand percentage results of baseline optimization, by commodity group.

Commodity	Commodity Group Name	Demand (kTon)	Demand %	Flow (kTon)	Unmet Demand (kTon)	Unmet Demand % of Commodity	Commodity % of Total
1	Agriculture, Forrest, Fishing	8,463	13.3%	8,393	69.5	0.822%	35.9%
2	Coal, Crude oil, Natural gas	279	0.4%	279	0.0	0.000%	0.0%
3	Ore	29,426	46.4%	29,426	0.0	0.000%	0.0%
4	Food, Beverage, Tobacco	249	0.4%	248	1.2	0.490%	0.6%
5	Textile, leather	0	0.0%	0	0.0	1.255%	0.0%
6	Wood, Cork, Pulp, Paper	4,701	7.4%	4,692	8.9	0.190%	4.6%
7	Petroleum products	1,409	2.2%	1,409	0.0	0.000%	0.0%
8	Chemicals, rubber, plastics	1,257	2.0%	1,257	0.0	0.000%	0.0%
9	Other non-metallic mineral	448	0.7%	446	1.8	0.407%	0.9%
10	Fabricated metal products	4,017	6.3%	3,948	68.7	1.710%	35.5%
11	Machinery and equipment	94	0.1%	94	0.0	0.000%	0.0%
12	Transport equipment	894	1.4%	894	0.0	0.000%	0.0%
13	Furniture, Other manufactured	59	0.1%	58	0.5	0.804%	0.2%
14	Return materials and recycling	1,124	1.8%	1,124	0.4	0.032%	0.2%
15	Post and packages	0	0.0%	0	0.0	0.000%	0.0%
16	Equipment for transportation	1,187	1.9%	1,182	4.9	0.415%	2.5%
16	Equipment for transportation	0	0.0%	0	0.0	0.000%	0.0%
18	Loader and grouped goods	39	0.1%	39	0.2	0.495%	0.1%
19	Unidentifiable goods	9,738	15.4%	9,701	37.3	0.383%	19.3%
20	Goods not in group of 1-19	18	0.0%	18	0.1	0.297%	0.0%
T	Total	63,401	100.0%	63,208	193	0.305%	100.0%

The second performance measure, link usage > 90%, is presented in Table 5 and is includes two summary tables of link usage from two different perspectives. In the top table, the counts of links are grouped by link usage % (column 1) and commodity groups (row 1). For example, commodity 1 had 40 links with link usage % greater than 90% which is 1.4% of the 2,877 total links in the network. An alternate perspective is shown in the bottle table and presents the link usage count as a percentage of links with flow greater than zero. If you ignore edges without flow of commodity 1, then there are only 959 edges with flow greater than zero and 4.2% of those edges have link usage greater than 90%. This accounts for the regionalization of commodities as some might utilize more links than others.

Table 5. Link Usage results from baseline optimization shown in count from total links (top) and percentage of links with flow greater than 0.

Link Usage % Total	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20
0.0-0.1	2719	2859	2738	2687	2716	2730	2710	2757	2712	2691	2728	2674	2775	2743	2877	2714	2877	2743	2654	2776
0.1-0.2	65	2	77	83	83	76	85	78	78	76	70	74	37	44		73		74	99	56
0.2-0.3	16	2	35	18	27	18	25	19	20	29	18	37	13	13		25		12	24	20
0.3-0.4	7		9	13	10	8	14	11	12	7	6	18	7	19		16		7	17	5
0.4-0.5	4	1	7	22	9	4	1	2	7	16	12	18	3	4		3		1	7	2
0.5-0.6	2	1	6	7	2	1		5	5	7	3	5	5	5				13	1	
0.6-0.7	4		2	3	2	2	4	1	3	3	3	3	7	6		2		3	5	4
0.7-0.8	10		1	7	2	1	2		3	3	3	2	4	3		2		2	7	1
0.8-0.9	10	1		2	2	4	3		2	8	6	6	1	3		5		4	13	1
0.9-1.0	40	11	2	35	24	33	33	4	35	37	28	40	25	37		37		18	50	12
% > 90%	1.4%	0.4%	0.1%	1.2%	0.8%	1.1%	1.1%	0.1%	1.2%	1.3%	1.0%	1.4%	0.9%	1.3%	0.0%	1.3%	0.0%	0.6%	1.7%	0.4%
Total	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877	2877

Link Usage % Flow > 0	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20
0.0-0.1	84%	73%	83%	80%	81%	84%	81%	85%	82%	79%	83%	78%	88%	85%		82%		85%	76%	88%
0.1-0.2	7%	3%	9%	9%	10%	9%	10%	10%	9%	9%	8%	8%	4%	5%		8%		8%	10%	7%
0.2-0.3	2%	3%	4%	2%	3%	2%	3%	2%	2%	3%	2%	4%	2%	1%		3%		1%	3%	2%
0.3-0.4	1%	0%	1%	1%	1%	1%	2%	1%	1%	1%	1%	2%	1%	2%		2%		1%	2%	1%
0.4-0.5	0%	2%	1%	2%	1%	0%	0%	0%	1%	2%	1%	2%	0%	0%		0%		0%	1%	0%
0.5-0.6	0%	2%	1%	1%	0%	0%	0%	1%	1%	1%	0%	1%	1%	1%		0%		1%	0%	0%
0.6-0.7	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	1%		0%		0%	1%	0%
0.7-0.8	1%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%		0%		0%	1%	0%
0.8-0.9	1%	2%	0%	0%	0%	0%	0%	0%	0%	1%	1%	1%	0%	0%		1%		0%	1%	0%
0.9-1.0	4.2%	16.7%	0.2%	3.8%	2.8%	3.7%	3.7%	0.5%	3.9%	4.3%	3.2%	4.3%	2.9%	4.0%		4.2%		2.1%	5.3%	1.4%
Total	959	66	842	933	866	892	882	811	905	867	886	937	849	914		890		874	948	851

To visualize the baseline optimization results, the flow of commodities through the network is plotted on the Swedish Railway Network map as shown for commodity 1 in Figure 9 with a side by side comparison to the sink/source locations presented in the previous chapter. Bright red lines represent a bidirectional flow, while a slightly darker red line corresponds to a flow that only flows in one direction. The thickness of the line represents the size of the relative flow and a grey line received no flow at all for that commodity. From the figure it can be observed that most of the flow for commodity 1 is concentrated in the southern region near the concentration of large sinks and sources, but most of the flow in the northern parts of the country appear to only flow in one direction. When compared to commodity 2, as shown on the next page in Figure 10, all of the flow is concentrated in the northern region.

All 20 commodity groups' baseline network flow (except for 15 and 17) is plotted in subplots shown in

Figure **11** and Figure 12. From the graphs, there are different regions of the map that receive more flow for that particular commodity and some of the western corridors appear to not be used at all. With the exception of commodity 2 which is concentrated in the northern region, flow moves through the entire network and most two-way flow occurs in the central and southern region and varies by commodity. Now that the baseline optimization model system parameters have been established, the interdiction strategy is applied (step 2 from Figure 5) to measure system performance when one link at a time is removed from the network.

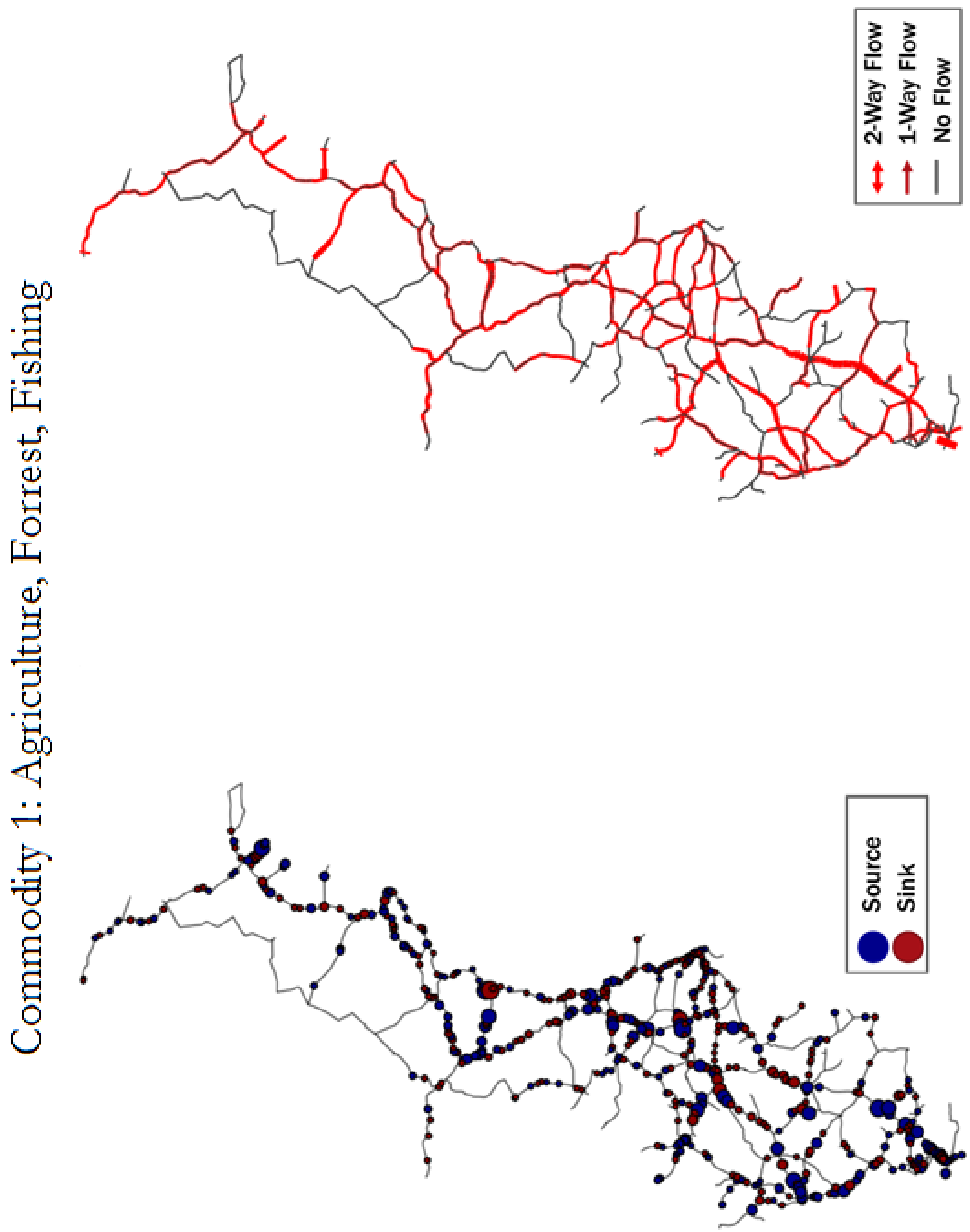

Figure 9. Side by side comparison of sink/source location (left) to baseline network flow (right) of commodity 1.

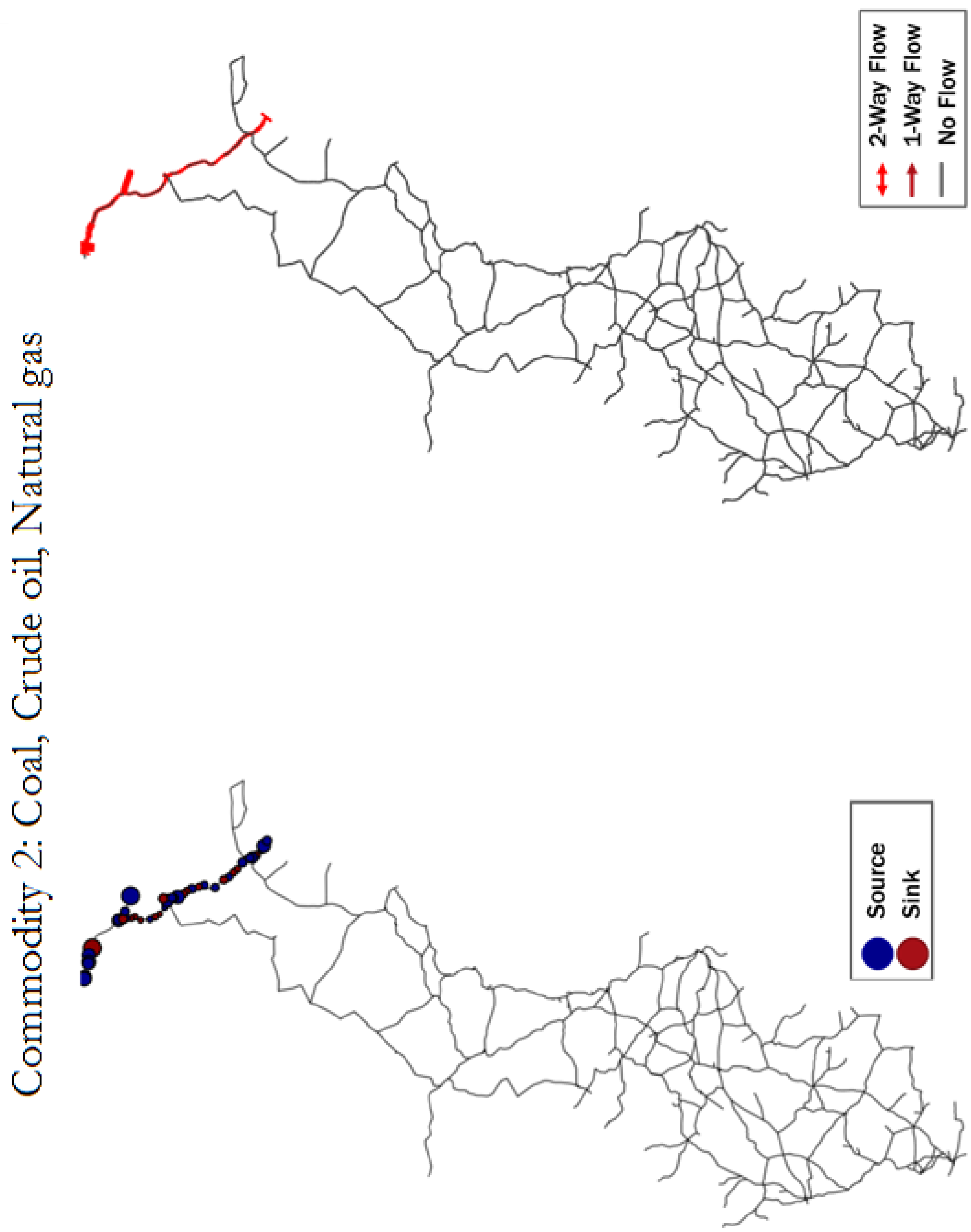

Figure 10. Side by side comparison of sink/source location (left) to baseline network flow (right) of commodity 2

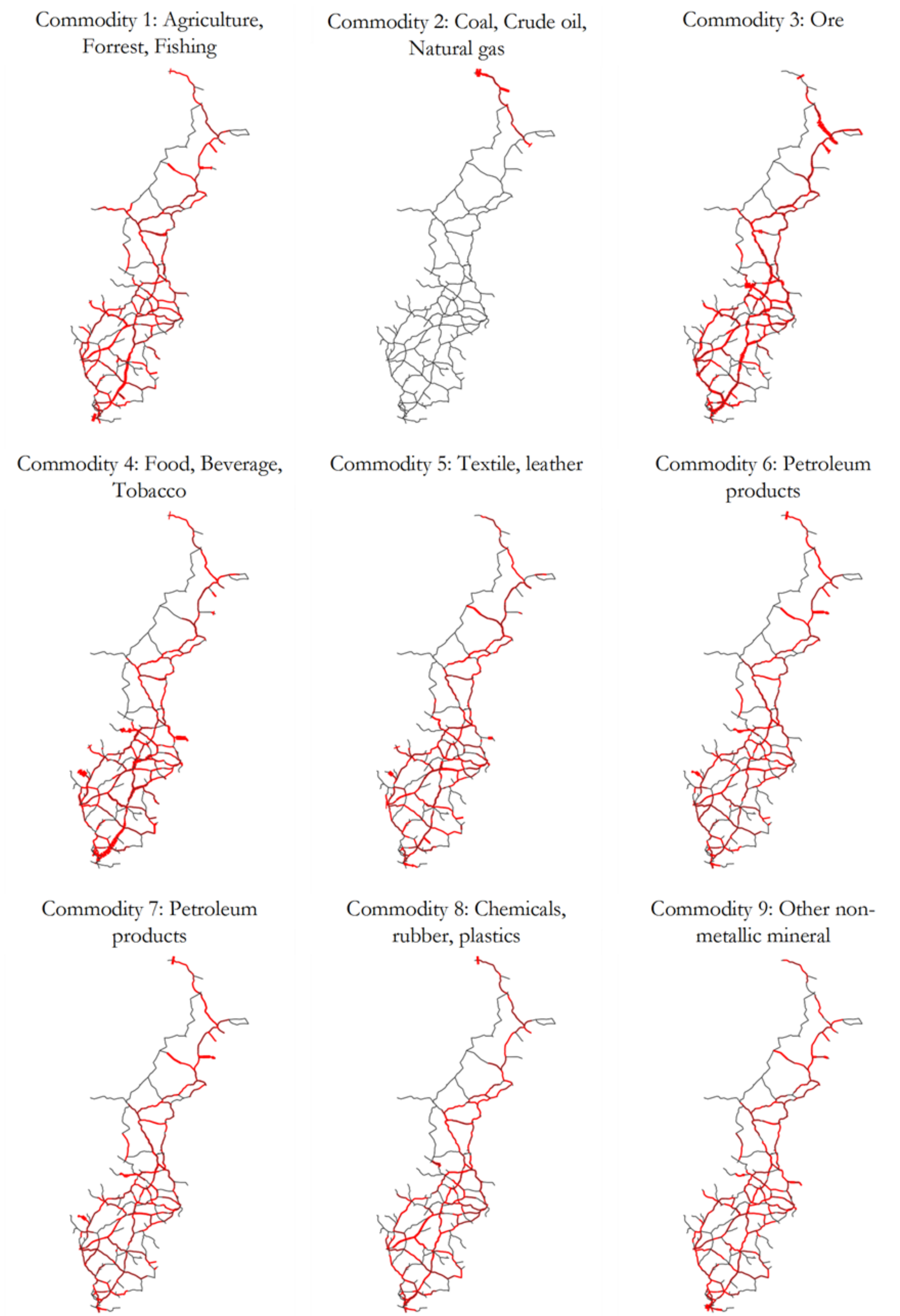

Figure 11. Commodity groups 1-9 of baseline network flow shown in red.

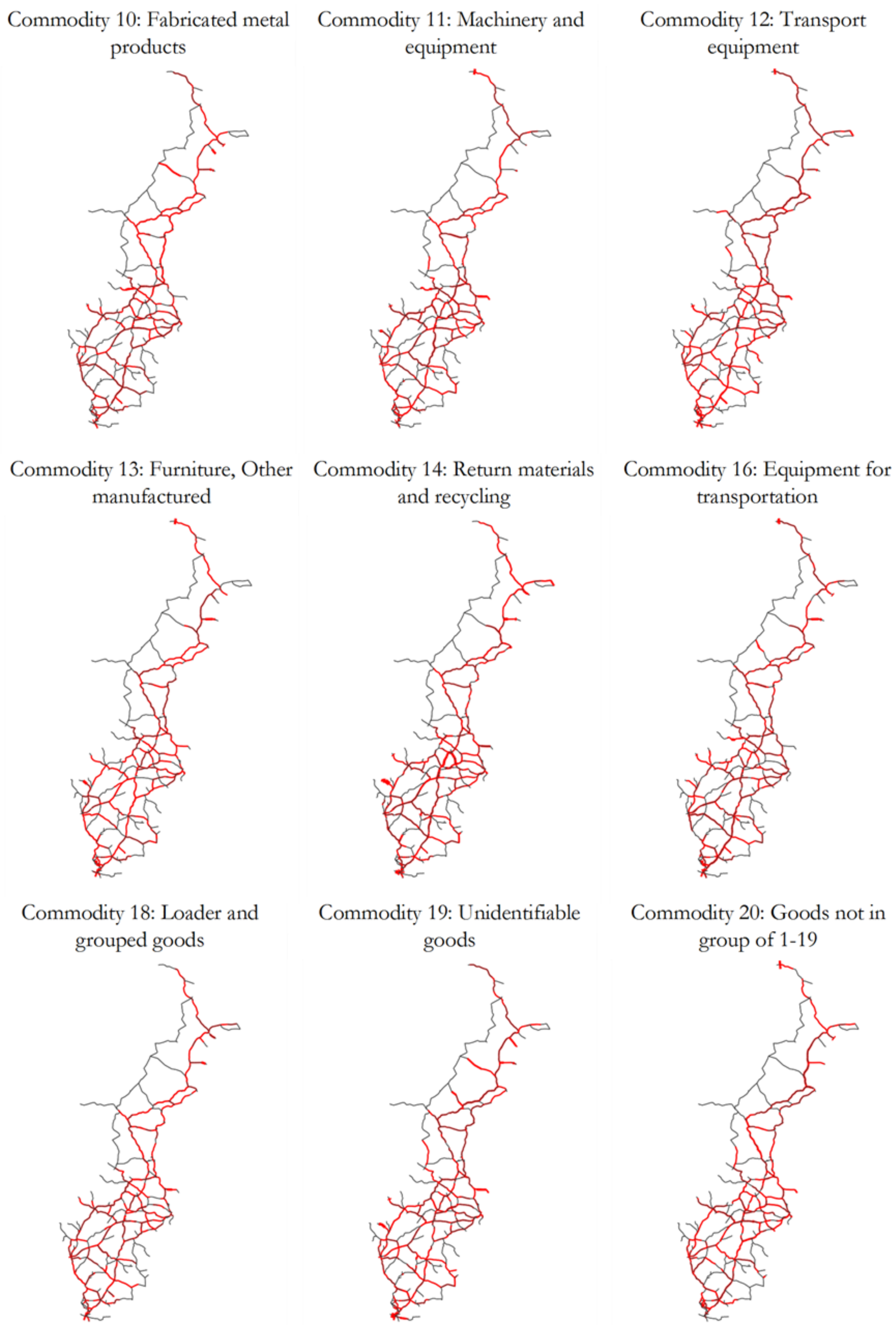

Figure 12. Commodity groups 10-20 (except 15, 17) of baseline network flow.

4.2 SwRail Interdiction

The results of the one-at-a-time link removal strategy are presented here according to the system performance metrics discussed in Chapter 2. The interdiction results summary for unmet demand percentage by commodity is shown below in Table 6 in columns 4 – 7. Column 4 and 5 represents the maximum and minimum amount of demand that is disrupted out of all edges removed. Recall in from Chapter 2, that the system performance measures are the net change from the baseline performance metrics. The last two columns track what percentage of links results in a change from the base. For example, commodity 1 had a maximum demand disruption of 2.23% and only 11.47% of links removed resulted in a change from the baseline optimization performance for unmet demand percentage for commodity 1. In general, less than 20% of the links removed impacted the unmet demand percentage by commodity which indicates successful rerouting of freight in most cases.

Table 6. Net unmet demand percentage results from interdiction strategy.

Commodity	Commodity Group Name	Baseline Met Demand	Max Unmet Demand % Disrupted	Min Unmet Demand % Disrupted	% of Links Change	% of Links No Change
1	Agriculture, Forrest, Fishing	99.18%	2.23%	0.00%	11.47%	88.53%
2	Coal, Crude oil, Natural gas	100.00%	31.88%	0.00%	4.10%	95.90%
3	Ore	100.00%	5.82%	0.00%	13.91%	86.09%
4	Food, Beverage, Tobacco	99.51%	3.06%	0.00%	9.25%	90.75%
5	Textile, leather	98.74%	1.96%	0.00%	9.53%	90.47%
6	Wood, Cork, Pulp, Paper	99.81%	2.82%	0.00%	9.74%	90.26%
7	Petroleum products	100.00%	3.07%	0.00%	14.81%	85.19%
8	Chemicals, rubber, plastics	100.00%	5.73%	0.00%	12.38%	87.62%
9	Other non-metallic mineral	99.59%	8.26%	0.00%	9.87%	90.13%
10	Fabricated metal products	98.29%	6.26%	0.00%	10.08%	89.92%
11	Machinery and equipment	100.00%	2.97%	0.00%	14.60%	85.40%
12	Transport equipment	100.00%	3.47%	0.00%	18.98%	81.02%
13	Furniture, Other manufactured	99.20%	2.37%	0.00%	8.62%	91.38%
14	Return materials and recycling	99.97%	16.89%	0.00%	14.05%	85.95%
15	Post and packages	0.00%	0.00%	0.00%	0.00%	0.00%
16	Equipment for transportation	99.59%	1.33%	0.00%	11.54%	88.46%
17	Moving Goods, vehicles for repair	0.00%	0.00%	0.00%	0.00%	0.00%
18	Loader and grouped goods	99.50%	3.59%	0.00%	9.32%	90.68%
19	Unidentifiable goods	99.62%	2.81%	0.00%	13.21%	86.79%
20	Goods not in group of 1-19	99.70%	3.67%	0.00%	10.71%	89.29%
T	Total	99.69%	2.79%	0.00%	28.72%	71.28%

On the following page, in Table 7, interdiction results are presented for the second component importance metric, link usage count > 90%. Columns 1-4 carry over previous commodity information and baseline performance metrics for reference and columns 5 and 6 represent the maximum and minimum met change observed when compared to the baseline. The last three columns give a sense of what percentage of increased the link usage count, decreased it, or observed the same level. For example, commodity 1 saw a maximum increase of 29 additional links with usage greater than 90% and at one point decreased the link usage by 19 links. Most of the links removed from the graph saw an net change in the link usage % with most links removed from the network causing an increase in the count of links with usage greater than 90%. This supports the previous results from Table 6 that data was successfully rerouted when an edge was disrupted.

To visualize the distribution of interdiction results, subplots in Figure 13 and Figure 14 are generated for all commodities except 15 and 17. In Figure 13, only unmet demand results greater than 0 are plotted in the histogram for visibility. For all commodities, as shown in Table 6, most of the links disrupted, did not change the unmet demand percentage per commodity. In addition, the x-axis is not standardized for all plots for visibility purposes. As shown, commodity 2 and 14 had the most extreme demand perturbation and only a small number of edges caused the perturbation. Moving on to Figure 14, all links are included in the histogram and the distribution varies by commodity. Some commodities are centered with a high frequency of 0 links changed, while others are either skewed or completed shifted off-center. For example, commodity 8 and 20 both have a high number of edges that show no net change, but have a shifted distribution to the right. Alternatively, Commodity 2 had the highest percentage of improved link usage capacity for disrupted edges and commodity 8 almost always increased.

Table 7. Interdiction results for net change in link usage count > 90% by commodity group.

Commodity	Commodity Group Name	Baseline Link Usage Count > 90%	Baseline Count Flow > 0	Max Link Usage Count Change	Min Link Usage Count Change	% of Links Change +	% of Links Change -	% of Links No Change
1	Agriculture, Forrest, Fishing	40	959	29	-16	57.93%	17.32%	24.76%
2	Coal, Crude oil, Natural gas	11	66	3	-11	0.42%	76.08%	23.50%
3	Ore	2	842	20	-2	47.22%	13.00%	39.78%
4	Food, Beverage, Tobacco	35	933	22	-17	30.32%	43.53%	26.15%
5	Textile, leather	24	866	37	-12	61.96%	12.59%	25.45%
6	Wood, Cork, Pulp, Paper	33	892	16	-21	31.57%	41.59%	26.84%
7	Petroleum products	33	882	17	-22	30.60%	42.35%	27.05%
8	Chemicals, rubber, plastics	4	811	27	-1	78.16%	0.07%	21.77%
9	Other non-metallic mineral	35	905	29	-12	61.61%	13.63%	24.76%
10	Fabricated metal products	37	867	23	-23	36.16%	38.04%	25.80%
11	Machinery and equipment	28	886	23	-15	37.55%	35.40%	27.05%
12	Transport equipment	40	937	26	-19	46.73%	28.30%	24.97%
13	Furniture, Other manufactured	25	849	22	-12	53.34%	19.54%	27.12%
14	Return materials and recycling	37	914	20	-23	28.86%	45.97%	25.17%
15	Post and packages	0	0	0	0	0.00%	0.00%	0.00%
16	Equipment for transportation	37	890	25	-19	51.95%	22.88%	25.17%
17	Moving Goods, vehicles for repair	0	0	0	0	0.00%	0.00%	0.00%
18	Loader and grouped goods	18	874	25	-11	50.49%	21.21%	28.30%
19	Unidentifiable goods	50	948	23	-23	34.01%	39.64%	26.36%
20	Goods not in group of 1-19	12	851	33	-5	77.75%	0.42%	21.84%
T	Total	1	2217	14	-1	48.75%	7.23%	44.02%

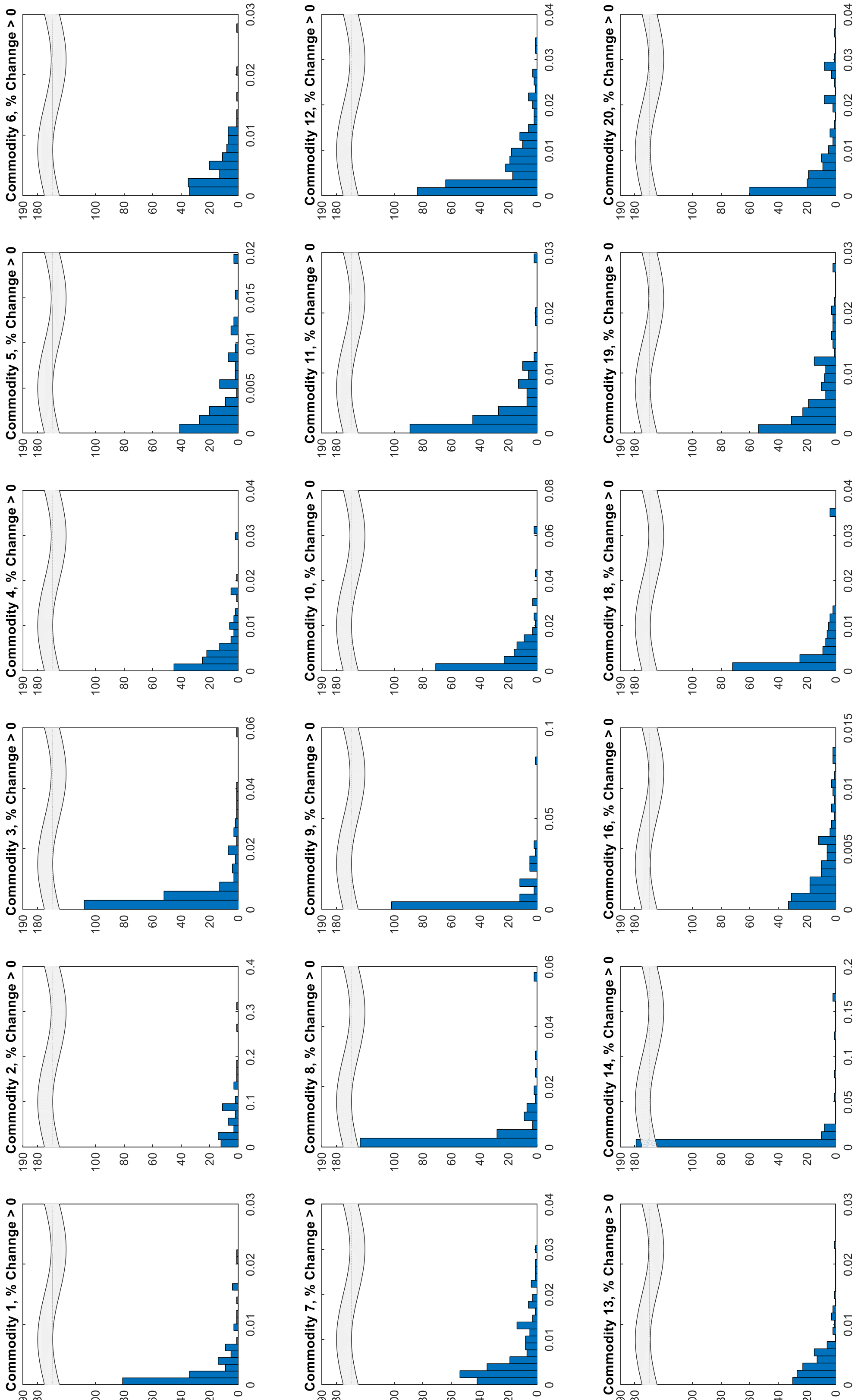

Figure 13. Histogram of all 20 commodity groups (except 15 and 17) of unmet demand % change > 0.

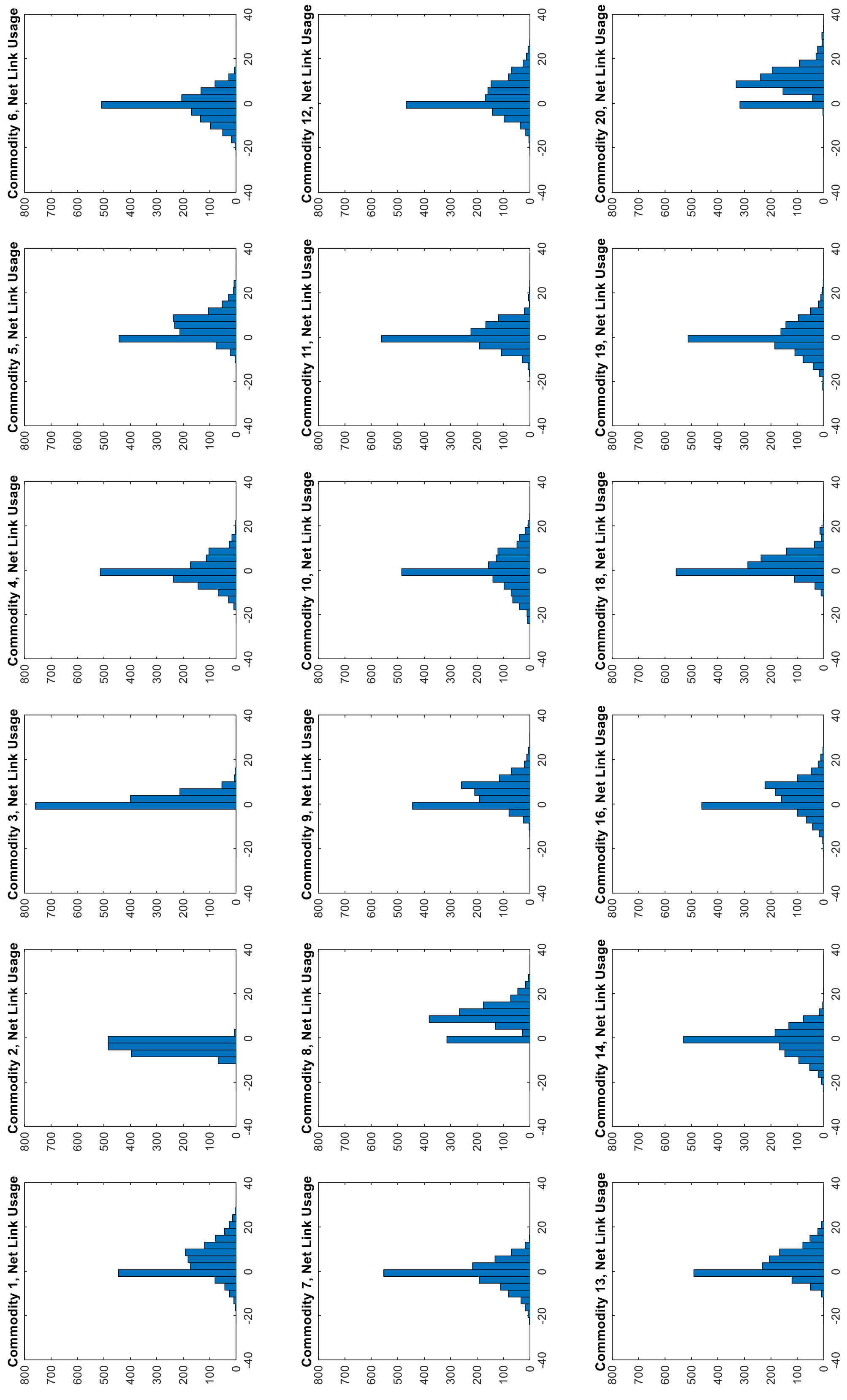

Figure 14. Histogram of 20 commodity groups (except 15 and 17) of net change in link usage capacity count.

The last figure shown here, Figure 15, compares the system performance metrics with each other to investigate if there is a correlation. Each subplot shown per commodity plots link usage counts versus non-zero values of unmet demand percentage to determine if an increase in unmet demand is associated with an increase or decrease with link usage. From the figure, you can see a large concentration of points near where unmet demand percentage is close to zero for all commodities. For each commodity subplot, the Pearson correlation coefficient and associated p-value are shown on the graph, with the least-squares line plotted in grey if the Pearson coefficient results in an insignificant p-value > 0.5 and highlighted in red if significant, p-value<0.5. The Pearson coefficient value is between -1 and 1 to indicate the level of linear correlation between the two variables, link usage count and unmet demand percentage, with values equal to 0 indicates no correlation. Interestingly enough, four commodities have significant Pearson correlation coefficients that can be both positive and negative even though the relatively low Pearson coefficient values does not indicate a linear relationship. A possible explanation for the negative correlation between unmet demand percentage and link usage count could be that there are no rerouting alternatives due to the lack of redundancy in the network, not lack of excess capacity, hence, the unmet demand percentage increases and the link usage decreases. This would also explain the positive correlation results in that unmet demand percentage and link usage count both increase because of lack of excess capacity. The interdiction strategy of one-link-at-a-time removal process had different impacts to certain commodities, but all commodities were affected. The two system performance measures were impacted very different from the interdiction process and indicate rerouting of freight occurred often when a link was disrupted. The next section will present the decision analysis findings and provide final rankings of critical links.

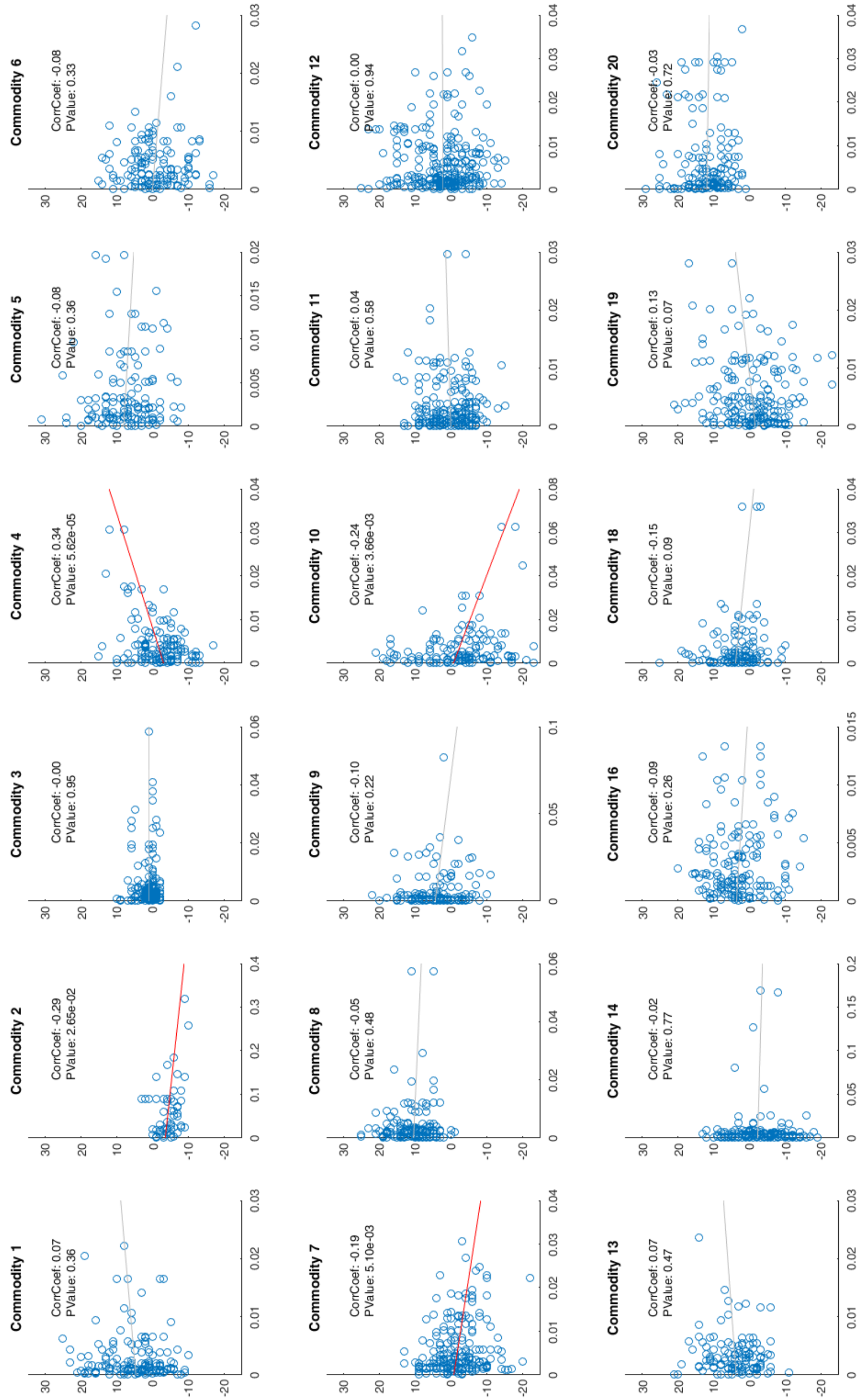

Figure 15. Link Usage Count > 90% vs. Unmet Demand %, non-zero values of unmet percentage. Pearson's coefficient and p-value are shown as well as a least-square lines plotted for each commodity.

4.3 SwRail Decision Analysis

Step 3 of the proposed methodology from Chapter 2 is to apply a multi-criteria decision analysis tool, TOPSIS, to aggregate the commodity-specific component importance measures discussed in the previous sections. Commodity weightings are required from the decision maker in order to apply the TOPSIS methodology. These weightings should prioritize what commodities the decision maker is invested in protecting. Early in Chapter 3, the total amount of freight moving through the network by commodity group was shown in

Table 1 and it was suggested that decision makers might want to prioritize the largest commodity group shipped, commodity 3: Ore. However, if we consider the value of the commodities to Sweden's economy instead of just volume, we would have commodity weightings similar to the following obtained from Sweden GDP data shown in Table 8 [58]. Column 3 was derived from Sweden's economic value for the first 14 commodity groups that are shipped in the Swedish Railway Network. From the table, the bottom six commodities darkened in grey had no equivalent economic value from the collected data and were given a weight of 0 for that commodity group and are not considered in the analysis. Two of the six excluded commodity groups 15 and 17, had no associated demand in the applied network example and commodity group 19 is classified as Unidentifiable goods. Even though Unidentifiable Goods makes up over 15% of the given demand for the applied network, it makes sense to exclude it because of the difficulty of prioritizing unknown freight as a decision maker. The remaining three commodity groups that are being excluded from the analysis (groups 16, 18, and 20) make up only 2% of the total demand of the network.

Table 8. Commodity weights for TOPSIS analysis derived from commodity value to GDP economy.

Group	Commodity Group	Total Use (SEK 100K)	SIOT Weights %	Equal Weights
1	Agriculture, Forrest, Fishing	130,329	4.8%	7.1%
2	Coal, Crude oil, Natural gas	91,241	3.3%	7.1%
3	Ore	91,241	3.3%	7.1%
4	Food, Beverage, Tobacco	222,790	8.2%	7.1%
5	Textile, leather	58,167	2.1%	7.1%
6	Wood, Cork, Pulp, Paper	237,093	8.7%	7.1%
7	Petroleum products	190,674	7.0%	7.1%
8	Chemicals, rubber, plastics	316,821	11.6%	7.1%
9	Other non-metallic mineral	47,472	1.7%	7.1%
10	Fabricated metal products	180,394	6.6%	7.1%
11	Machinery and equipment	718,996	26.3%	7.1%
12	Transport equipment	308,902	11.3%	7.1%
13	Furniture, Other manufactured	81,663	3.0%	7.1%
14	Return materials and recycling	57,265	2.1%	7.1%
15	Post and packages	N/A	0.0%	0.0%
16	Equipment for transportation	N/A	0.0%	0.0%
17	Moving Goods, vehicles for repair	N/A	0.0%	0.0%
18	Loader and grouped goods	N/A	0.0%	0.0%
19	Unidentifiable goods	N/A	0.0%	0.0%
20	Goods not in group of 1-19	N/A	0.0%	0.0%

There are three different weight combinations that are examined in this section and two different importance measures incorporated into the analysis: unmet demand percentage and count of link usage greater than 90%. Columns 2 and 3 from the table above make up the first two sets of rankings, SIOT value and equal weight value, that are used with each importance measure to provide a ranking of critical components. The last analysis weights only examine the top five commodities (highlighted in light grey) and are used to give a ranking that incorporates both importance measures each given equal importance. This results in five TOPSIS rankings that are compared and discussed below.

4.3.1 Rankings Without TOPSIS

In this section we summarize the ranking of critical components of the Swedish Railway Network when analyzing total flow of goods, regardless of commodity type. This is done by ranking the total unmet demand performance results for total goods moved through the network and by ranking link usage > 90% for overall link capacity rather than commodity-specific capacity. The rankings are shown below in Table 9, columns 2 and 4. Interestingly, columns 3 and 5 list the rank the respective edge holds in the other ranking and you can see that neither ranking shares edges in the top 20 or even in the top 100. Possible reasons for this could be the potential correlation that unmet demand percentage might have with total link usage count in that a high unmet demand percentage might correspond to improve link usage performance since less flow moves through the network.

Table 9. Rankings of critical links without TOPSIS.

Rank	Total Unmet Demand %	Rank of Edge in Link Usage	Total Link Usage Count	Rank of Edge in Unmet Demand
1	(18,17)	359	(130,105)	91
2	(518,458)	1335	(890,889)	414
3	(578,518)	702	(151,77)	26
4	(57,56)	359	(1158,1134)	414
5	(153,152)	151	(896,895)	414
6	(56,55)	359	(1138,1137)	414
7	(152,151)	1335	(1136,1135)	414
8	(68,67)	702	(897,896)	414
9	(58,57)	702	(885,884)	414
10	(74,67)	702	(1139,1138)	414
11	(54,53)	702	(1144,1143)	414
12	(55,54)	359	(1135,1134)	414
13	(71,70)	359	(1137,1136)	414
14	(70,69)	359	(1065,1064)	414
15	(75,50)	359	(1033,1032)	414
16	(51,50)	359	(1025,1024)	414

17	(53,52)	46	(1026,1025)	414
18	(52,51)	359	(889,888)	414
19	(1289,1288)	151	(873,872)	414
20	(69,68)	702	(357,356)	414

In order to visualize the spread of the rankings provided in Table 9, the spread of both rankings are shown in Table 10 and Table 11 below:

Table 10. Spread of total net unmet demand % network performance results for all edges in network.

Total Unmet Demand %	Count of Rank
0-0.001	1235
0.001-0.002	63
0.002-0.003	55
0.003-0.004	26
0.004-0.005	14
0.005-0.006	11
0.006-0.007	5
0.007-0.008	7
0.008-0.009	2
0.009-0.01	1
0.01-0.011	2
0.011-0.012	4
0.012-0.013	1
0.013-0.014	3
0.014-0.015	2
0.015-0.016	1
0.016-0.017	1
0.017-0.018	1
0.019-0.02	3
0.027-0.028	1

The spread of unmet demand performance is consistent with the interdiction results discussed in the previous section in that only a small percentage of edges had a significant effect on the network performance. In contrast, the spread of the link usage results is much more even spread with a large percentage of links impacting the link usage count performance. Also different from the unmet demand performance data was the small percentage of edges that caused an improvement in the link usage performance in that removed edges actually resulted in less bottlenecks in the network.

Table 11. Spread of net link usage > 90% count for overall link capacity, for all edges in network.

Link Usage Count, All Commodities	Count of Rank
-1	104
0	633
1	343
2	208
3	74
4	10
5	5
6	5
7	11
8	4
9	13
10	9
11	10
12	4
13	3
14	2

4.3.2 TOPSIS Results

In this section we present the final component rankings from TOPSIS based on different combinations of imported commodity weights and selected performance criteria. The first importance measure selected, unmet demand percentage, is analyzed with two different weights, SIOT (importance to Sweden economy) and equal weights. The results are shown in **Table 12** with columns 2 and 3 listing the top 20 ranked edges and columns 3 and 4 listing the respective ranking each edge takes in the other ranking. For example edge (1288,1287) is ranked first according to SIOT weights and is second with equal weights. There are several common edges between the two rankings, but no real agreement except for edge (1288,1287) which is considered very important by both ranking structures.

Table 12. TOPSIS results for unmet demand percentage.

TOPSIS	Unmet Demand Percentage			
Rank	TOPSIS SIOT Weights	TOPSIS Equal Weights	Rank of SIOT in Equal Weights	Rank of Equal Weights in SIOT
1	(1288,1287)	(1250,1249)	2	54
2	(1287,1286)	(1288,1287)	12	1
3	(1115,946)	(71,70)	37	50
4	(1286,1285)	(1362,1273)	20	145
5	(880,870)	(1274,1250)	6	83
6	(525,524)	(1249,1247)	9	64
7	(526,525)	(880,870)	10	5
8	(81,80)	(1247,1246)	73	20
9	(82,81)	(1289,1288)	102	23
10	(79,78)	(525,524)	57	6
11	(80,79)	(152,151)	59	44
12	(1270,1269)	(578,518)	47	33
13	(150,149)	(70,69)	42	135
14	(83,82)	(526,525)	89	7
15	(950,949)	(69,68)	82	207
16	(84,83)	(876,875)	110	119
17	(585,579)	(1287,1286)	24	2
18	(951,950)	(153,152)	106	43
19	(952,951)	(72,71)	108	65
20	(1247,1246)	(21,20)	8	67

The second performance metric, count of link usage > 90%, was also analyzed with two different sets of weights. The results are shown below in **Table 13** and are presented in the same structure as the previous rankings. As with unmet demand percentage, there is a general disagreement between the rankings even though common edges exit between the top 20. There are no common edges between the rankings using unmet demand percentage and the rankings used by link usage regardless of the weights chosen.

Table 13. TOPSIS results for Link Usage > 90%.

TOPSIS	Link Usage > 90%			
Rank	TOPSIS SIOT Weights	TOPSIS Equal Weights	Rank of SIOT in Equal Weights	Rank of Equal Weights in SIOT
1	(300,299)	(889,888)	8	16
2	(999,871)	(1139,1138)	3	18
3	(124,123)	(999,871)	24	2
4	(897,896)	(389,388)	14	38
5	(898,872)	(888,887)	15	128
6	(499,498)	(1008,1007)	75	63
7	(509,508)	(756,743)	156	8
8	(756,743)	(300,299)	7	1
9	(873,872)	(298,297)	46	94
10	(335,334)	(942,941)	194	158
11	(86,85)	(1143,1142)	314	52
12	(832,831)	(390,389)	100	257
13	(956,955)	(295,294)	91	116
14	(385,384)	(897,896)	330	4
15	(1014,884)	(898,872)	80	5
16	(889,888)	(1144,1143)	1	108
17	(330,329)	(766,765)	40	174
18	(1139,1138)	(890,889)	2	569
19	(159,158)	(913,912)	414	100
20	(496,458)	(1141,1140)	59	35

The last TOPSIS ranking examined is a combination of both importance measures. By taking the top 5 industry weights from Table 8 and applying them to both component importance measures, you have a weighting structure that assigns have of the weights on the unmet demand performance and half of the priority on link usage and only focusing on the top five commodities used in Sweden. The results of the discussed TOPSIS weighting scheme is shown below in Table 14. Very interestingly, the ranking from the combined importance measures almost exactly matches the top 8 ranking for the TOPSIS results for unmet demand percentage using SIOT weightings as shown in column 3. Column 5, the rankings for Link Usage with SIOT weightings, also had an impact since the very last link (82,81) is ranked 9th in unmet demand percentage, but 1275th in link usage which drove the rank down. Another example is edge (300, 299) which was not considered important from an unmet demand percentage perspective, but was ranked in the top 10 from a link usage perspective.

Table 14. TOPSIS results for combined importance measures and the respective rank of other TOPSIS results.

TOPSIS	Combined Importance Measures				
Rank	Combined SIOT Top 5	Rank in SIOT Unmet Demand	Rank in Eq Unmet Demand	Rank in SIOT Link Usage	Rank in Eq Link Usage
1	(1288,1287)	1	2	536	535
2	(1287,1286)	2	12	1221	733
3	(1115,946)	3	37	136	325
4	(1286,1285)	4	20	254	557
5	(525,524)	6	9	332	136
6	(81,80)	8	73	72	773
7	(880,870)	5	6	1076	541
8	(526,525)	7	10	1213	688
9	(83,82)	14	89	50	220
10	(84,83)	16	110	164	481
11	(86,85)	41	148	11	314
12	(90,89)	24	142	248	1168
13	(950,949)	15	82	298	574
14	(79,78)	10	57	452	1166
15	(1243,1242)	21	41	236	343
16	(956,955)	48	163	13	91
17	(952,951)	19	108	256	107
18	(300,299)	403	403	1	8
19	(80,79)	11	59	1115	1434
20	(82,81)	9	102	1275	714

The spread of the rankings for each of the five TOPSIS rankings is provided in Appendix A.4. How sparsely grouped the data varies mostly with the importance metric shown similar to the spread of the groupings in Table 10 and Table 11 discussed in the previous section in that the unmet demand percentage criteria have only a small amount of links that cause and impact compared to the link usage rankings which are more evenly spread. The next section will conclude the analysis with a visualization of the locations of the top ranked components on the Swedish Railway Network.

4.3.3 TOPSIS Visualization

In order to visualize where the critical components are located on the Swedish Railway Network according to the different TOPSIS rankings, the TOPSIS rankings are plotted on a map of the Swedish Railway Network for each of the rankings discussed previously in this chapter. The links are colored according to the closeness to the highest ranked link and distance from the lowest ranked link with lighter colored links having higher ranking than darker red links. The plotted networks are overlaid on a dark grey background to improve visibility of light colored links. The first two networks, shown in Figure 16 and Figure 17, are a visualization of the overall network performance without TOPSIS described earlier in section 4.3.1. Examining the unmet demand percentage performance graph, a large number of links are deemed important along one particular corridor which could correspond to commodity 2, Coal, Crude oil, Natural gas, that is located exclusively there. By reviewing the impacts from the interdiction process, commodity 2 demand had the largest maximum demand disrupted from column 4 in Table 6 which easily explains this ranking's emphasis on the northern most region of the network. When considering overall link usage capacity, a different perspective emerges where entire paths connecting long corridors are emphasized here as rerouting alternatives. Additionally, dead end branches of the network appear to be especially vulnerable to bottlenecks when cut off from the rest of the network.

The next two graphs presented map the TOPSIS results for the unmet demand performance criteria with two different weights, SIOT weights and Equal weights. Unlike the previous graphs, the edges are colored according to their distance to the ideal and anti-ideal solution from Eq. (20) with ideal scores closer to 1.

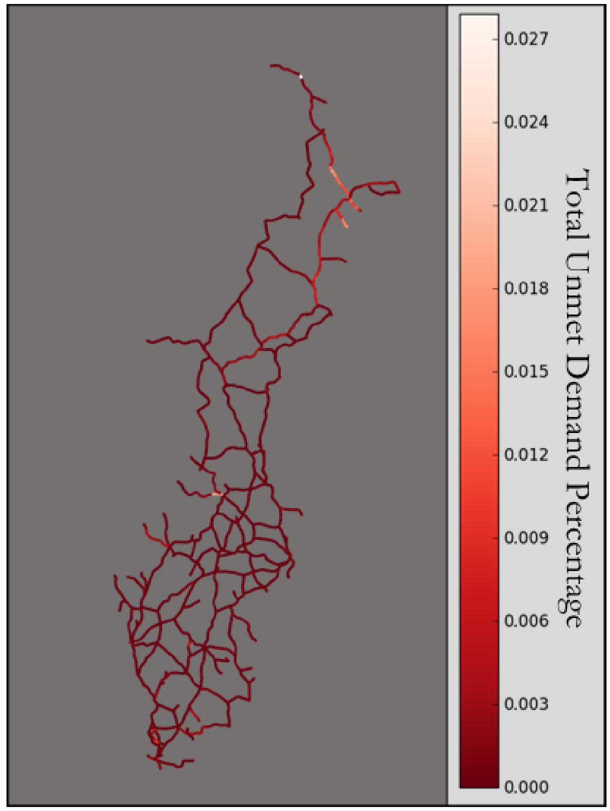

Figure 16. Total unmet demand percentage performance for SwRail Network.

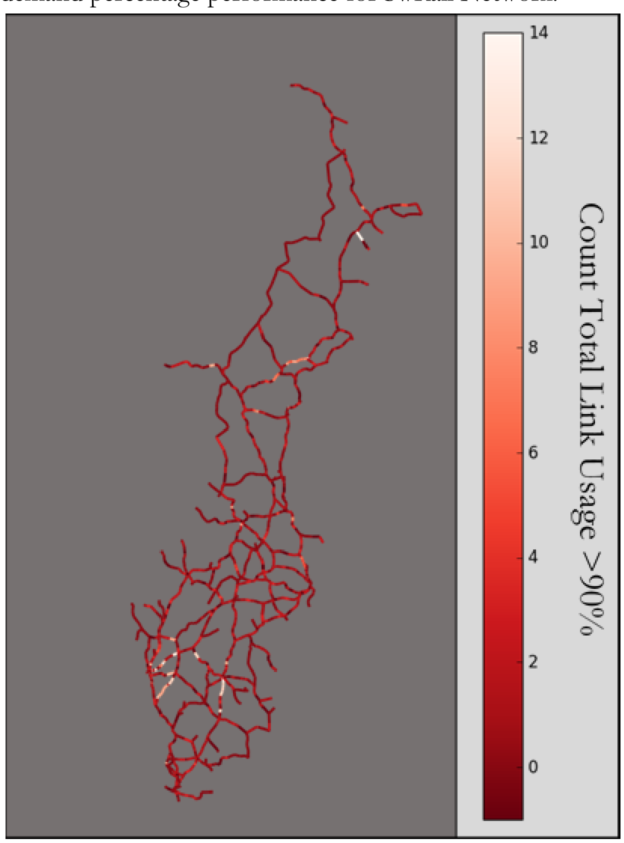

Figure 17. Total overall link usage performance (left) for SwRail network.

The graphs are stacked as shown on the next page in Figure 18 and Figure 19, with critical links located in similar sections of the graph even with different weighting structure of the commodity weights. One key difference between the TOPSIS results with SIOT weights is the indifference of most of the northern part of the network which deviates from the rankings when considering only overall total unmet demand. This is surely due to the SIOT weights emphasizing other commodities over commodity 2 which was the main focus of the overall unmet demand performance results. Also of note, the range of TOPSIS scores for the unmet demand percentage performance with equal weights have a very small range of possible values with very little space between the highest ranked links to the lower ranked links. This suggests that the equal weights diluted the performance metrics.

The next importance metric, link usage, is shown on the following page in Figure 20 and Figure 21 with both having much lighter colored edges due to the wider spread of and higher TOPSIS scores achieved. When examining both graphs, it is interesting to note that entire branches are highlighted up the mid-point where the link suddenly drops. This is possibly due to rerouting decisions since flow across a branch is completely disrupted when the mid-point of the path is removed. Finally, Figure 22, maps the TOPSIS scores for the combined importance metrics with the top 5 SIOT commodities which looks very similar to TOPSIS results for the unmet demand percentage which is expected.

Overall, the performance metric selected and commodity weights had an impact on the TOPSIS scores and location critical links in the SwRail network.

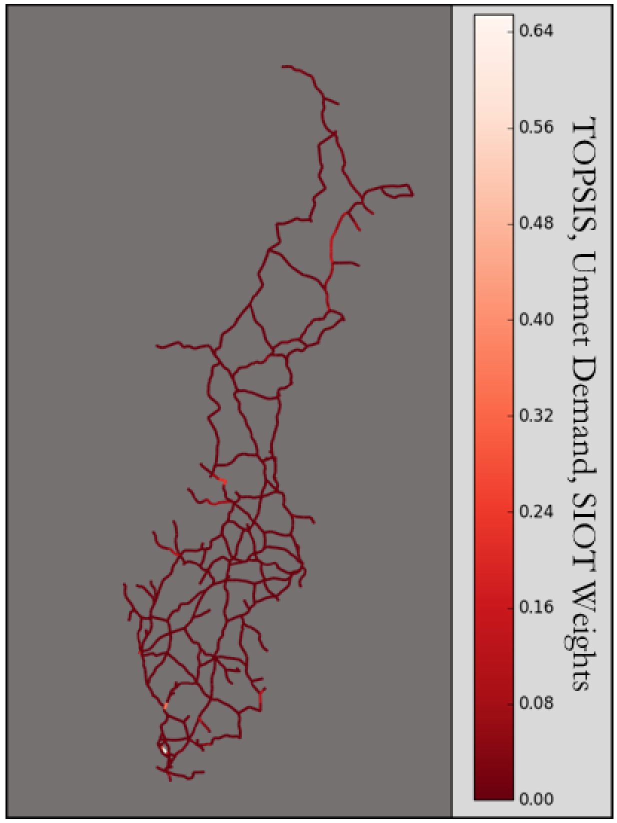

Figure 18. TOPSIS scores mapped on SwRail network for unmet demand percentage and SIOT weights.

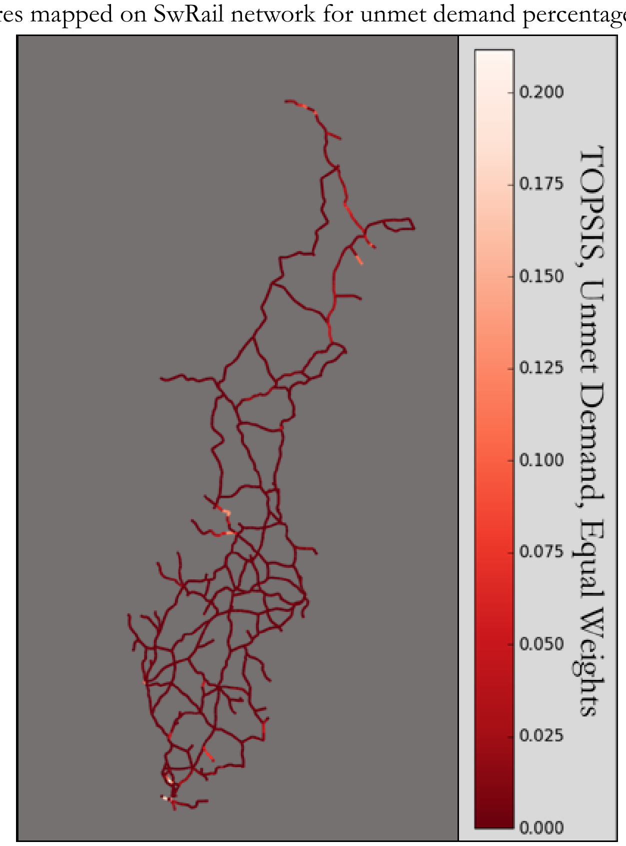

Figure 19. TOPSIS scores mapped on SwRail network for unmet demand percentage and equal weights.

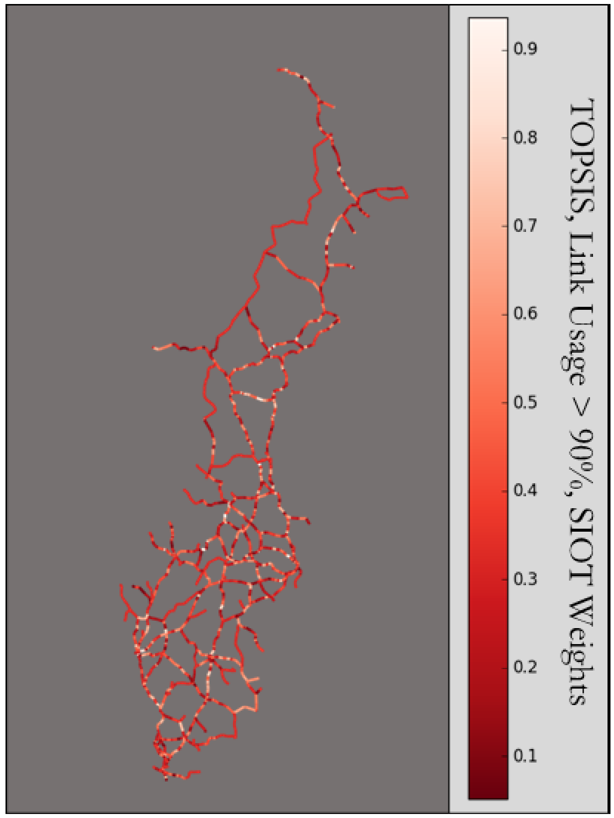

Figure 20. TOPSIS scores mapped on SwRail network for link usage > 90% with SIOT weights.

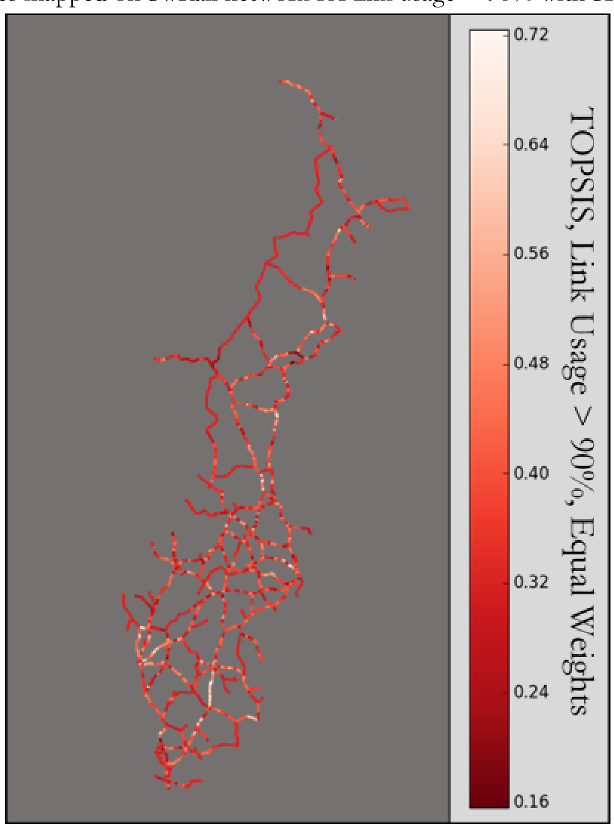

Figure 21. TOPSIS scores mapped on SwRail network for link usage > 90% with equal weights.

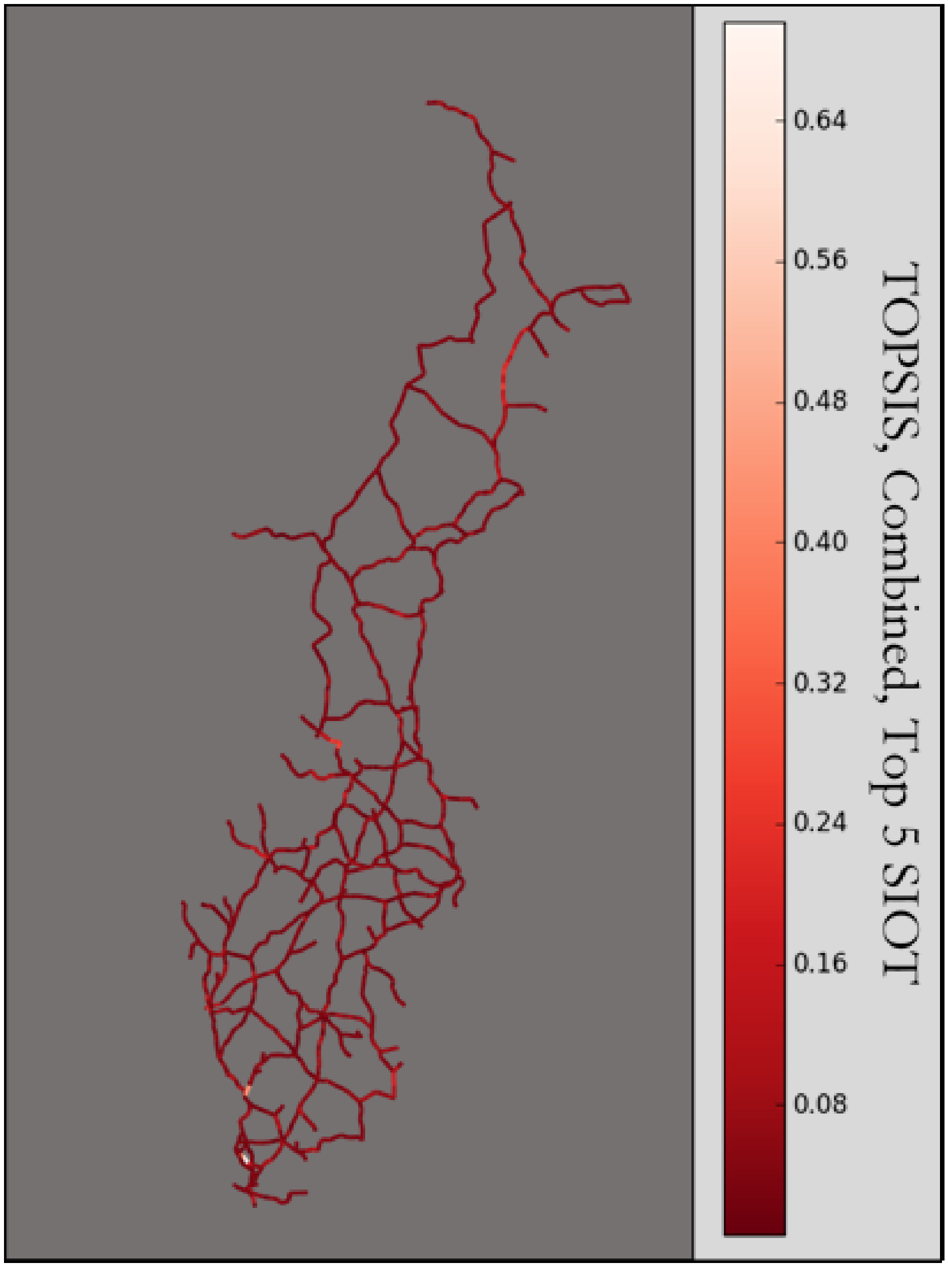

Figure 22. TOPSIS results mapped on SwRail network for combined importance measures with the top 5 SIOT commodities.

5.0 Conclusion

The goal of this thesis was to analyze network vulnerability of multi-commodity networks from a flow-based approach to identify critical links in the network. This would allow decision makers with limited resources to allocate resources to critical links in the network that would cause the most damage to the network performance criteria considered most important. This work builds on the well-studied flow-based network vulnerability analysis, but had little previous research on multi-commodity network vulnerability. To measure network vulnerability, a proposed three stage approach built a baseline modified *MCMF* optimization model to measure demand feasibility before applying an interdiction strategy to measure system performance when links were disrupted then finally applying a decision analysis tool, TOPSIS, to rank critical links from a multi-commodity perspective. As described in the previous chapter, applying different component importance measure and commodity weights resulted in different rankings than by analyzing total network performance alone.

There are several areas for future work and include investigating the impact different optimization models impact ranking strategies. In addition, identifying critical paths over critical links could provide additional information to the decision maker and is another research avenue to explore.

References

[1] Aggarwal A, Oblak M, Vemuganti RR. A heuristic solution procedure for multicommodity integer flows. Computers & operations research. 1995;22:1075-87.

[2] Al-Douri YK, Tretten P, Karim R. Improvement of railway performance: a study of Swedish railway infrastructure. Journal of Modern Transportation. 2016;24:22-37. DOI: 10.1007/s40534-015-0092-0.

[3] Almoghathawi Y, Barker K, Nicholson CD, Rocco CM. A Multi-Criteria Decision Analysis Approach for Importance Ranking of Network Components. Reliability Engineering & System Safety. 2016.

[4] Almoghathawi Y, Oommen A, Morshedlou N. A Multi-Criteria Decision Making Tool to Prioritize Network Component for Recovery based on Importance Measures. Proceedings of the 2015 International Conference on Operations Excellence and Service Engineering. Orlando, FL2015. p. 372-82.

[5] ASCE. 2013 Report Card for America's Infrastructure. American Society of Civil Engineers; 2013. p. 74. URL: http://www.infrastructurereportcard.org/wp-content/uploads/2013ReportCardforAmericasInfrastructure.pdf.

[6] Barker K, Nicholson CD, Ramirez-Marquez JE. Vulnerability importance measures toward resilience-based network design. 2015.

[7] Berdica K. An introduction to road vulnerability: what has been done, is done and should be done. Transport Policy. 2002;9:117-27.

[8] Broberg A. Railway transport 2016 quarter 1. Sweden: Trafikanalys, Sveriges officiella statistik, Trafikverke; 2016. p. 23.

[9] Calvete HI. Network simplex algorithm for the general equal flow problem. European Journal of Operational Research. 2003;150:585-600.

[10] Chakraborty S, Yeh C-H. A simulation comparison of normalization procedures for TOPSIS. Computers & Industrial Engineering, 2009 CIE 2009 International Conference on: IEEE; 2009. p. 1815-20.

[11] Chen A, Yang C, Kongsomsaksakul S, Lee M. Network-based accessibility measures for vulnerability analysis of degradable transportation networks. Networks and Spatial Economics. 2007;7:241-56.

[12] Darayi M, Barker K, Santos JR. Component Importance Measures for Multi-Industry Vulnerability of a Freight Transportation Network. Networks and Spatial Economics. 2016.

[13] DHS. The 2014 Quadrennial Homeland Security Review. In: Security USDoH, editor.2014. p. 104. URL: https://www.dhs.gov/sites/default/files/publications/2014-qhsr-final-508.pdf.

[14] Du Y, Gao C, Hu Y, Mahadevan S, Deng Y. A new method of identifying influential nodes in complex networks based on TOPSIS. Physica A: Statistical Mechanics and its Applications. 2014;399:57-69.

[15] Dunn S, Wilkinson SM. Identifying critical components in infrastructure networks using network topology. Journal of Infrastructure Systems. 2012;19:157-65.

[16] Erath A, Birdsall J, Axhausen K, Hajdin R. Vulnerability assessment methodology for Swiss road network. Transportation Research Record: Journal of the Transportation Research Board. 2009:118-26.

[17] Eurostat. Freight transport statistics - modal split. April 2016 ed: Eurostat Statistics Explained; 2014. URL: http://ec.europa.eu/eurostat/statistics-explained/index.php/Freight_transport_statistics_-_modal_split.

[18] Eurostat. Railway transport - Goods transported, by group of goods - from 2008 onwards based on NST 2007 (1 000 t, million tkm) [rail_go_grpgood]. 2015. URL: http://ec.europa.eu/eurostat/statistics-explained/index.php/Road_freight_transport_statistics#Main_tables.

[19] Evans JR. The simplex method for integral multicommodity networks. Naval Research Logistics Quarterly. 1978;25:31-7.

[20] Ghalebsaz-Jeddi B. Analysis and Sensitivity of Stochastic Capacitated Multi-Commodity Flows: University of Cincinnati; 2004.

[21] Gurobi Optimization Inc. Gurobi Optimizer Reference Manual. 2015. URL: http://www.gurobi.com/documentation/6.5/refman.pdf.

[22] Haimes YY. Risk modeling, assessment, and management: John Wiley & Sons; 2015.

[23] Henry D, Ramirez-Marquez JE. Generic metrics and quantitative approaches for system resilience as a function of time. Reliability Engineering & System Safety. 2012;99:114-22.

[24] Hosseini S, Barker K, Ramirez-Marquez JE. A review of definitions and measures of system resilience. Reliability Engineering & System Safety. 2016;145:47-61.

[25] Ishizaka A, Nemery P. Multi-criteria decision analysis: methods and software: John Wiley & Sons; 2013.

[26] Jenelius E. Redundancy importance: Links as rerouting alternatives during road network disruptions. Procedia Engineering. 2010;3:129-37.

[27] Jenelius E, Mattsson L-G. Road network vulnerability analysis of area-covering disruptions: A grid-based approach with case study. Transportation research part A: policy and practice. 2012;46:746-60.

[28] Jenelius E, Westin J, Holmgren ÅJ. Critical infrastructure protection under imperfect attacker perception. International Journal of Critical Infrastructure Protection. 2010;3:16-26.

[29] Johansson J. SwRailwayData_20141012. Lund University, Sweden 2012

[30] Johansson J, Hassel H, Cedergren A. Vulnerability analysis of interdependent critical infrastructures: case study of the Swedish railway system. International journal of critical infrastructures. 2011;7:289-316.

[31] Johansson J, Hassel H, Zio E. Reliability and vulnerability analyses of critical infrastructures: comparing two approaches in the context of power systems. Reliability Engineering & System Safety. 2013;120:27-38.

[32] Jönsson H, Johansson J, Johansson H. Identifying critical components in technical infrastructure networks. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability. 2008;222:235-43.

[33] Knoop VL, Snelder M, van Zuylen HJ, Hoogendoorn SP. Link-level vulnerability indicators for real-world networks. Transportation Research Part A: Policy and Practice. 2012;46:843-54.

[34] KTH. The prospects of Freight by Rail in the future transportation market. In: Group KR, editor. Stockholm, Sweden: Royal Institute of Technology; 1999. p. 24. URL: https://www.kth.se/polopoly_fs/1.87065!/Menu/general/column-content/attachment/00_81_summary.pdf.

[35] KTH. Efficient train systems for freight transport. In: Technology CfR, editor. Stockholm, Sweden: Royal Institute of Technology; 2005. p. 14.

[36] Lambert JH, Sarda P. Terrorism scenario identification by superposition of infrastructure networks. Journal of Infrastructure Systems. 2005;11:211-20.

[37] Landex A, Kaas AH, Schittenhelm B, Schneider-Tilli J. Evaluation of railway capacity. Trafikdage/Trafficdays. 2006.

[38] Li H, Tian S, Pan Y, Zhang X, Yu X. Minimum-Cost Optimization in Multicommodity Logistic Chain Network. In: Li H, Olver PJ, Sommer G, editors. Computer Algebra and Geometric Algebra with Applications: 6th International Workshop, IWMM 2004, Shanghai, China, May 19-21, 2004 and International Workshop, GIAE 2004, Xian, China, May 24-28, 2004, Revised Selected Papers. Berlin, Heidelberg: Springer Berlin Heidelberg; 2005. p. 97-104.

[39] Lindfeldt A. A study of the performance and utilization of the Swedish railway network. CETRA. Opatija, Croatia 2010. p. 8.

[40] Lindfeldt A. Railway Capacity Analysis. Stockholm, Sweden: KTH Royal Institute of Technology; 2015.

[41] Lipton E. Cost of Storm-Debris Removal in City Is at Least Twice the U.S. Average. The New York Times2013. URL: http://www.nytimes.com/2013/04/25/nyregion/debris-removal-from-hurricane-sandy-is-more-costly-than-average.html.

[42] LKAB. Home to the world's largest underground iron ore mine. lkab.com; 2016. URL: http://www.lkab.com/en/About-us/Overview/Operations-Areas/Kiruna/.

[43] Murray AT, Matisziw TC, Grubesic TH. A methodological overview of network vulnerability analysis. Growth and Change. 2008;39:573-92.

[44] Nagurney A, Qiang Q. A network efficiency measure with application to critical infrastructure networks. Journal of Global Optimization. 2008;40:261-75.

[45] NAIC. Transportation Sector Resilience. In: Council NIA, editor.2015. p. 248. URL: https://www.dhs.gov/sites/default/files/publications/niac-transportation-resilience-final-report-07-10-15-508.pdf.

[46] NationMaster. Transport > Railways: Sweden and United States compared. 2013. URL: http://www.nationmaster.com/country-info/compare/Sweden/United-States/Transport/Railways.

[47] Nicholson CD, Barker K, Ramirez-Marquez JE. Flow-based vulnerability measures for network component importance: Experimentation with preparedness planning. Reliability Engineering & System Safety. 2016;145:62-73.

[48] O'Kelly ME. Network hub structure and resilience. Networks and Spatial Economics. 2015;15:235-51.

[49] Obama B. Critical Infrastructure Security and Resilience. 2013. URL: https://www.whitehouse.gov/the-press-office/2013/02/12/presidential-policy-directive-critical-infrastructure-security-and-resil.

[50] Olson DL. Comparison of weights in TOPSIS models. Mathematical and Computer Modelling. 2004;40:721-7.

[51] Ouyang M, Hong L, Mao Z-J, Yu M-H, Qi F. A methodological approach to analyze vulnerability of interdependent infrastructures. Simulation Modelling Practice and Theory. 2009;17:817-28.

[52] Pant R, Barker K, Ramirez-Marquez JE, Rocco CM. Stochastic measures of resilience and their application to container terminals. Computers & Industrial Engineering. 2014;70:183-94.

[53] Raack C. Capacitated Network Design-Multi-Commodity Flow Formulations, Cutting Planes, and Demand Uncertainty. 2012.

[54] Rinaldi SM, Peerenboom JP, Kelly TK. Identifying, understanding, and analyzing critical infrastructure interdependencies. Control Systems, IEEE. 2001;21:11-25. DOI: 10.1109/37.969131.

[55] Rodríguez-Núñez E, García-Palomares JC. Measuring the vulnerability of public transport networks. Journal of transport geography. 2014;35:50-63.

[56] SE Gov. Objectives for transport and infrastructure. Government Offices of Sweden; 2016. URL: http://www.government.se/government-policy/transport-and-infrastructure/goals-for-transport-and-infrastructure/.

[57] Simonovic SP, Verma R. A new methodology for water resources multicriteria decision making under uncertainty. Physics and Chemistry of the Earth, Parts A/B/C. 2008;33:322-9.

[58] SIOT. Symmetric input-output tables 2008-2013 SNI2007, ESA2010. In: Sweden S, editor.2012. URL: http://www.scb.se/Statistik/NR/NR0103/Symmetric-input-output-tables-2008-2013-SNI2007-ESA2010.xlsx.

[59] Sullivan J, Novak D, Aultman-Hall L, Scott DM. Identifying critical road segments and measuring system-wide robustness in transportation networks with isolating links: a link-based capacity-reduction approach. Transportation Research Part A: Policy and Practice. 2010;44:323-36.

[60] Tizghadam A, Leon-Garcia A. On Robust Traffic Engineering in Transport Networks. Global Telecommunications Conference, 2008 IEEE GLOBECOM 2008 IEEE2008. p. 1-6. DOI: 10.1109/GLOCOM.2008.ECP.456 URL: http://ieeexplore.ieee.org/ielx5/4697774/4697775/04698231.pdf?tp=&arnumber=4698231&isnumber=4697775.

[61] Ukkusuri SV, Yushimito WF. A methodology to assess the criticality of highway transportation networks. Journal of Transportation Security. 2009;2:29-46. DOI: 10.1007/s12198-009-0025-4.

[62] USDOT. Freight Facts and Figures. In: Statistics USDoT-BoT, editor.2015. p. 111. URL: http://www.rita.dot.gov/bts/sites/rita.dot.gov.bts/files/FF%26F_complete.pdf.

[63] Zhu S, Levinson DM. Disruptions to transportation networks: a review. Network reliability in practice: Springer; 2012. p. 5-20.

Appendix A.1 SwRail Network Generation

This section includes all code generated in MATLAB® R2015b used to modify the SwRail data provided to generate source/sink supply/demand nodes as well as capacity limits for the edges in the network.

A.1.1 Create Source/Sinks

```
%Master's Thesis - Mackenzie Whitman
%Goal: Distribute source sink nodes based on orig/destination information
%from SwRail Route data. Source/Sink per commodity can be any node on path
%of orig/dest per route.

clear;
load('SwRailwayData_20141012.mat');
Routes = SwRail.Routes;

%create structure with correct field names, node, station, pos_0, pos_1
Nodes = struct();
Nodes.station = SwRail.Stn_Name_Short;
Nodes.node = (1:size(Nodes.station)).';
Nodes.pos_0 = SwRail.y;
Nodes.pos_1 = SwRail.x;
Nodes.nodeCount = length(Nodes.node);
Nodes.commodityCount = size(SwRail.CargoTypeShort,2) - 1;

%structures to hold the source/sink samples from route paths
Nodes.source = struct();
Nodes.sink = struct();

%loop through each route (1091 total), pull sample from nodes on path
for i = 1:size(Routes,2)
    if size(Routes(i).NodeRoute,2) == 1
        continue
    end
    for k = 1:Nodes.commodityCount
        %create source/sink sample based on nodes on path
        node_sample = Routes(i).NodeRoute;
        fieldName = sprintf('c_%d',k);
        Nodes.source(i).(fieldName) = [];
        Nodes.sink(i).(fieldName) = [];

        %Check if commodity k possible from route history, otherwise skip k
        if Routes(i).kTon(k) > 0
            %50% chance commodity k is selected to be sink/souce for route
            selected = rand;
            if selected > 0.5; selected = 1; else selected = 0; end;
```

```
            if selected == 1
                %if k selected, determine sample size for source/sink
                %sample size: random uniform distribution(1 : 0.5(pathNodes))
                %Use floor to ensure size source + size sink > node_sample
                lowerR = 1;
                upperR = int64(floor(0.5*(size(node_sample,2))));

                %if upperR <= 1, set sample size = 1;
                if upperR <= 1
                    source_size = 1;
                    sink_size = 1;
                else
                    source_size = randi([lowerR upperR]);
                    sink_size = randi([lowerR upperR]);
                end

                %sample without replacement source/sink nodes
                Nodes.source(i).(fieldName) = datasample(node_sample, ...
                    source_size, 'Replace', false);
                %remove sampled node from options for source, :( complicated
                tf = ismember(node_sample, Nodes.source(i).(fieldName));
                loc = find(~tf);
                node_sample = node_sample(loc);
                Nodes.sink(i).(fieldName) = datasample(node_sample, ...
                    sink_size, 'Replace', false);
            end
        end
    end
end
```

A.1.2 Remove duplicate source/sink node assignments

```
%node can only be a source or sink per commodity
%loop through all source/sink assignments, find duplicates
%if duplicate, select source/sink with largest number of trains per year

%keep track of nmbr trains per year per commodity, per node, sink/source
Nodes.trainSink = zeros(Nodes.nodeCount,Nodes.commodityCount);
Nodes.trainSource = zeros(Nodes.nodeCount,Nodes.commodityCount);

%loop through all paths
for i = 1:size(Routes,2)
    %loop through all commodities
    for k = 1:Nodes.commodityCount
        nmbTrains = Routes(i).NbrTrainsPerYear;
        fieldName = sprintf('c_%d',k);
        sourceNodes = Nodes.source(i).(fieldName);
        sinkNodes = Nodes.sink(i).(fieldName);
```

```
        %check if sources on path i selected for commodity k
        if ~isempty(sourceNodes)
            for j = 1:size(sourceNodes,2)
                Nodes.trainSource(sourceNodes(j),k) = Nodes.trainSource(j,k) +
nmbTrains;
            end
        end
        %check if sinks on path i selected for commodity k
        if ~isempty(sinkNodes)
            for j = 1:size(sinkNodes,2)
                Nodes.trainSink(sinkNodes(j),k) = Nodes.trainSink(j,k) +
nmbTrains;
            end
        end
    end
end

%Max(number of trains) for sink or source. Remove node from sink/source so,
%that node is only a sink or source, but not both
removeSource = [];
removeSink = [];

%Modification 06/14/2016 - Limit % of source/sink nodes to 25% (~340 nodes)
%count number of nodes that are either source/sink
numberSinks = sum(Nodes.trainSink > 0, 1);
numberSources = sum(Nodes.trainSource > 0, 1);
sourceList = [];
sinkList = [];

%loop through all nodes
for i = 1:Nodes.nodeCount
    for k = 1:Nodes.commodityCount
        source = Nodes.trainSource(i,k);
        sink = Nodes.trainSink(i,k);
        if source > 0
            sourceList(end + 1, :) = [i, k];
        end
        if sink > 0
           sinkList(end + 1, :) = [i, k];
        end
        %if node is both source/sink, select one with max number of trains
        if and(source > 0, sink > 0)
            if source < sink
                %remove node from sink for all paths
                removeSource(end + 1,:) = [i,k];
                numberSources(k) = numberSources(k) - 1;
                iRemove = ~ismember(sourceList, [i,k], 'rows');
                sourceList = sourceList(iRemove,:);
            elseif source > sink
                %remove node from source for all paths
                removeSink(end + 1,:) = [i,k];
                numberSinks(k) = numberSinks(k) - 1;
                iRemove = ~ismember(sinkList, [i,k], 'rows');
                sinkList = sinkList(iRemove,:);
```

```
            else
                %remove both
                removeSource(end + 1,:) = [i,k];
                removeSink(end + 1,:) = [i,k];

                numberSources(k) = numberSources(k) - 1;
                numberSinks(k) = numberSinks(k) - 1;
                iRemove = ~ismember(sourceList, [i,k], 'rows');
                sourceList = sourceList(iRemove,:);

                iRemove = ~ismember(sinkList, [i,k], 'rows');
                sinkList = sinkList(iRemove,:);
            end
        end
    end
end

%define random number of sinks/sources allowed (no more than 30%)
sourceRange = zeros(Nodes.commodityCount, 1);
sinkRange = zeros(Nodes.commodityCount, 1);

for k = 1:Nodes.commodityCount
    if numberSources(k) > floor(0.3*Nodes.nodeCount)
        lowerR = floor(0.15*Nodes.nodeCount);
        upperR = floor(0.25*Nodes.nodeCount);
        sourceRange(k) = randi([lowerR upperR]);
    else
        sourceRange(k) = numberSources(k);
    end

    if numberSinks(k) > floor(0.3*Nodes.nodeCount)
        lowerR = floor(0.15*Nodes.nodeCount);
        upperR = floor(0.25*Nodes.nodeCount);
        sinkRange(k) = randi([lowerR upperR]);
    else
        sinkRange(k) = numberSinks(k);
    end

end

%if number of source/sinks > 30%, randomly select more for removal
for k = 1:Nodes.commodityCount
    while numberSources(k) > sourceRange(k)
        %until size is > 30%, sample one at a time from source/sinkList and
        %add to remove Source/Sink list
        kSourceList = sourceList(sourceList(:,2) == k, 1);
        rNode = datasample(kSourceList, 1);
        removeSource(end + 1, :) = [rNode k];
        iRemove = ~ismember(sourceList, [rNode,k], 'rows');
        sourceList = sourceList(iRemove, :);
        numberSources(k) = numberSources(k) - 1;
    end
```

```
    while numberSinks(k) > sinkRange(k)
        %until size is > 30%, sample one at a time from source/sinkList and
        %add to remove Source/Sink list
        kSinkList = sinkList(sinkList(:,2) == k, 1);
        rNode = datasample(kSinkList, 1);
        removeSink(end + 1, :) = [rNode k];
        iRemove = ~ismember(sinkList, [rNode,k], 'rows');
        sinkList = sinkList(iRemove, :);
        numberSinks(k) = numberSinks(k) - 1;
    end
end

%if node in list, remove from all instances of source nodes per k
for i = 1:size(Nodes.source,2)
    for k = 1:Nodes.commodityCount
        fieldName = sprintf('c_%d',k);
        %check if field empty
        if isempty(Nodes.source(i).(fieldName))
            continue
        end
        source_k = Nodes.source(i).(fieldName).';
        source_k(:,end + 1) = k;
        tf = ismember(source_k, removeSource,'rows');
        loc = find(~tf).';
        Nodes.source(i).(fieldName) = Nodes.source(i).(fieldName)(loc);
    end
end

%repeat for sink nodes per k
for i = 1:size(Nodes.sink,2)
    for k = 1:Nodes.commodityCount
        fieldName = sprintf('c_%d',k);
        %check if field empty
        if isempty(Nodes.sink(i).(fieldName))
            continue
        end
        sink_k = Nodes.sink(i).(fieldName).';
        sink_k(:,end + 1) = k;
        tf = ismember(sink_k, removeSink,'rows');
        loc = find(~tf).';
        Nodes.sink(i).(fieldName) = Nodes.sink(i).(fieldName)(loc);
    end
end
```

A.1.3 Calculate kTon/train per commodity

```
%Need to know total kTon per commodity in network, and total number of
%trains that a commodity could be on. Used to calculate kTon/train per
%commodity to distribute kTon to source/sink nodes.

%calculate total kTon in network by commodity k
```

```matlab
%sum by all k commodities all kTon for every path.
Nodes.kTonTotal = zeros(Nodes.commodityCount,1);

for i = 1:size(Routes,2)
       %loop through every path and add commodity to kTonTotal
       route_kTon = Routes(i).kTon(1:Nodes.commodityCount);
       Nodes.kTonTotal = Nodes.kTonTotal + route_kTon;
end

%determine if node is source/sink node by commodity k
%double checking previous work on ensuring node is not both
%creating list of source nodes and sink nodes by commodity
nodeSinkStatus = zeros(Nodes.nodeCount, Nodes.commodityCount);
nodeSourceStatus = zeros(Nodes.nodeCount, Nodes.commodityCount);

for i = 1:size(Routes,2)
    for k = 1:Nodes.commodityCount
        fieldName = sprintf('c_%d',k);
        sourceNodes = Nodes.source(i).(fieldName);
        sinkNodes = Nodes.sink(i).(fieldName);

        %check if sources on path i selected for commodity k
        if ~isempty(sourceNodes)
            nodeSourceStatus(sourceNodes,k) = 1;
        end
        %check if sinks on path i selected for commodity k
        if ~isempty(sinkNodes)
            nodeSinkStatus(sinkNodes,k) = 1;
        end
    end
end

%check sum, to ensure no dual source/sink assignments
checkNodeStatus = nodeSinkStatus + nodeSourceStatus;
badNodes = all(checkNodeStatus > 1); %it works, I am a genius!
Nodes.trainsTotal = zeros(Nodes.commodityCount,1);

%Calc number of trains passing through source/sink by commodity per path
for i = 1:size(Routes,2)
    for k = 1:Nodes.commodityCount
        fieldName = sprintf('c_%d',k);
        sourceNodes = Nodes.source(i).(fieldName);
        sinkNodes = Nodes.sink(i).(fieldName);

        if and(isempty(sourceNodes), isempty(sinkNodes))
            continue

        %if both not empty, add number of trains per year to total count
        elseif and(~isempty(sourceNodes), ~isempty(sinkNodes))
            Nodes.trainsTotal(k) = Nodes.trainsTotal(k) +
Routes(i).NbrTrainsPerYear;
            continue
        end
```

```
        % modification 06/15/2014 if empty, remove
        if xor(isempty(sourceNodes), isempty(sinkNodes))
            %if missing source, select from nodes on path not sinks
            if isempty(sourceNodes)
                Nodes.sink(i).(fieldName) = [];
            elseif isempty(sinkNodes)
                Nodes.source(i).(fieldName) = [];
            end
        end
    end
end
```

A.1.4 Distribute kTon per path per commodity

```
%Convert kTon to kg (1E+6) No Rounding yet!
Nodes.kgTotal = Nodes.kTonTotal * 1E+6;

%calculate kg/train per commodity, still no rounding yet.
Nodes.kgPerTrain = Nodes.kgTotal./Nodes.trainsTotal;

%calculate kg Total after distributing to see rounding errors
Nodes.kgTotalRounded = zeros(Nodes.commodityCount,1);

%if source/sink on path, assign kTon: kg/train * number of trains(per path)
for i = 1:size(Routes,2)
    Nodes.source(i).kg = zeros(Nodes.commodityCount,1);
    Nodes.sink(i).kg = zeros(Nodes.commodityCount,1);
    for k = 1:Nodes.commodityCount
        fieldName = sprintf('c_%d',k);
        sourceNodes = Nodes.source(i).(fieldName);
        sinkNodes = Nodes.sink(i).(fieldName);

        %if they are both empty, no commodity is distributed
        if and(isempty(sourceNodes), isempty(sinkNodes))
            continue
        %otherwise, distribute by number of trains per year (Routes data)
        %Round up to the nearest kg (error ~ 2 lbs per rounding).
        %keep track of distributed kg to see rounding erros at the end.
        else
            Nodes.source(i).kg(k) = ceil(Nodes.kgPerTrain(k) *
Routes(i).NbrTrainsPerYear);
            Nodes.sink(i).kg(k) = ceil(Nodes.kgPerTrain(k) *
Routes(i).NbrTrainsPerYear);
            Nodes.kgTotalRounded(k) = Nodes.kgTotalRounded(k) +
Nodes.source(i).kg(k);
        end
    end
end
```

A.1.5 Distribute kg to source/sink nodes

```
%Now, distribute kg to source/sink nodes on path. End up
%with source/sink node total for network, instead of per path.
%store inflow for sink/source per commodity k, Sink < 0, Source > 0
Nodes.inflow = zeros(Nodes.nodeCount, Nodes.commodityCount);

%for every path, every commodity k, distribute sink/source kg
for i = 1:size(Routes,2)
    for k = 1:Nodes.commodityCount
        fieldName = sprintf('c_%d',k);
        sourceNodes = Nodes.source(i).(fieldName);
        sinkNodes = Nodes.sink(i).(fieldName);
        %if sink/source for commodity k, distribute, else, continue
        if and(isempty(sourceNodes), isempty(sinkNodes))
            continue
        %two for loops (sink, source) to randomly assign kg (uniform)
        %sum(supply) === sum(demand), integer values, no switching
        else
            %source first, kgDistribute is the same for both sink/source
            kgDistribute = Nodes.source(i).kg(k);
            sumSource = 0;

            lsran = int64(kgDistribute/size(sourceNodes,2) * 0.50);
            usran = int64(kgDistribute/size(sourceNodes,2) * 1.25);

            %loop through each node in source, assign kg (random uniform)
            for j = 1:size(sourceNodes,2)
                %if not last element, random sampling
                if j < size(sourceNodes,2)
                    kgIntS = randi([lsran, usran]);
                    sumSource = sumSource + kgIntS;

                    %check if amount exceeds kgDistributed, adjust if need
                    if sumSource > kgDistribute
                        OldsumSource = sumSource - kgIntS;
                        kgIntS = kgIntS - (sumSource - kgDistribute);
                        sumSource = OldsumSource + kgIntS;
                    end
                    Nodes.inflow(sourceNodes(j),k) =
Nodes.inflow(sourceNodes(j),k) + ...
                        kgIntS;
                %if last element, balance so sumSource === kgDistribute
                else
                    if sumSource < kgDistribute
                        kgIntS = kgDistribute - sumSource;
                        Nodes.inflow(sourceNodes(j),k) =
Nodes.inflow(sourceNodes(j),k) + ...
                            kgIntS;
                    end
                end
```

```
            end

            %repeat previous for sink nodes, except change to negatvie
            kgDistribute = -Nodes.sink(i).kg(k);
            sumSink = 0;

            usran = int64(kgDistribute/size(sinkNodes,2) * 0.50);
            lsran = int64(kgDistribute/size(sinkNodes,2) * 1.25);

            %loop through each node in source, assign kg (random uniform)
            for j = 1:size(sinkNodes,2)
                %if not last element, random sampling
                if j < size(sinkNodes,2)
                    kgIntS = randi([lsran, usran]);
                    sumSink = sumSink + kgIntS;

                    %check if amount exceeds kgDistributed, adjust if need
                    %check the signs, because all values should be negative
                    if sumSink < kgDistribute
                        OldsumSink = sumSink - kgIntS;
                        kgIntS = kgIntS - (sumSink - kgDistribute);
                        sumSink = OldsumSink + kgIntS;
                    end
                    Nodes.inflow(sinkNodes(j),k) = Nodes.inflow(sinkNodes(j),k) +
kgIntS;
                %if last element, balance so sumSource === kgDistribute
                else
                    if sumSink > kgDistribute
                        kgIntS = kgDistribute - sumSink;
                        Nodes.inflow(sinkNodes(j),k) =
Nodes.inflow(sinkNodes(j),k) + kgIntS;
                    end
                end
            end
        end
    end
end
```

A.1.6 Calculate trains per link in network

```
%use Nodes.kgPerTrain (not rounded, before distribution) to calculate capacity per
train (kg)
Edges = struct();
kgPerTrain = Nodes.kgPerTrain;
Edges.trainCapacity = kgPerTrain;

%train capacity for total is max of train capacity of commodities
Edges.trainCapacity(end + 1) = max(kgPerTrain)*1.25;

%create Edges data structureUse
ICM = SwRail.ICM;
Edges.arc_ij = [];
```

```
Edges.commodityCount = Nodes.commodityCount;

for i = 1:size(ICM,1)
   for j = 1:size(ICM,2)
       %if edge exists, add to edges
       if ICM(i,j) == 1
           Edges.arc_ij(end + 1, :) = [i, j];
       end
   end
end

%calculate number of trains per link, by Routes data (if train on path)
%will be bidirectional, meaning arc(i,j) and arc(j,i) will have individual
%capacities.
Edges.trainsPerArc = struct();

%will need trains per arc for every commodity
for k = 1:Edges.commodityCount + 1
    fieldName = sprintf('c_%d',k);
    Edges.trainsPerArc.(fieldName) = zeros(Nodes.nodeCount, Nodes.nodeCount);
end

%for each route, pull the nodes on route, record arc data, number of trains
%if commodity found on path
for i = 1:size(Routes,2)
    NodeRoute = Routes(i).NodeRoute;
    kTonPerRoute = Routes(i).kTon;

    %commodity found T/F boolean (for field 21 (total link capacity)
    foundCommodity = false;

    %loop through every every node on path per route up to end - 1
    for j = 1:size(NodeRoute,2)

        %record number of trains per commodity (if commodity on route)
        for k = 1 : Edges.commodityCount
            fieldName = sprintf('c_%d',k);

            %if you are not on the last element, record arc information
            if j + 1 < size(NodeRoute,2)
                Edges.trainsPerArc.(fieldName)(NodeRoute(j), NodeRoute(j + 1)) =
...
                    Edges.trainsPerArc.(fieldName)(NodeRoute(j), NodeRoute(j + 1))
+ ...
                    Routes(i).NbrTrainsPerYear;

                Edges.trainsPerArc.(fieldName)(NodeRoute(j + 1), NodeRoute(j)) =
...
                    Edges.trainsPerArc.(fieldName)(NodeRoute(j + 1), NodeRoute(j))
+ ...
                    Routes(i).NbrTrainsPerYear;
                foundCommodity = true;
            end
        end
```

```
        %fill for total commodity data
        if j + 1 < size(NodeRoute,2)
            if foundCommodity == true
                fieldName = sprintf('c_%d',21);
                Edges.trainsPerArc.(fieldName)(NodeRoute(j), NodeRoute(j + 1)) =
...
                    Edges.trainsPerArc.(fieldName)(NodeRoute(j), NodeRoute(j + 1))
+ ...
                    Routes(i).NbrTrainsPerYear;

                Edges.trainsPerArc.(fieldName)(NodeRoute(j + 1), NodeRoute(j)) =
...
                    Edges.trainsPerArc.(fieldName)(NodeRoute(j + 1), NodeRoute(j))
+ ...
                    Routes(i).NbrTrainsPerYear;
            end
        end
    end
end
```

A.1.7 Calculate capacity per link in network

```
Edges.arcCapacity = struct();

for i = 1:size(Edges.arc_ij,1)
    arc_ij = Edges.arc_ij(i,:);
    Edges.arcCapacity(i).arc_ij = Edges.arc_ij(i,:);
    Edges.arcCapacity(i).trainsPerArc = zeros(Edges.commodityCount + 1, 1);
    Edges.arcCapacity(i).kg = zeros(Edges.commodityCount + 1, 1);
end

%loop through trains per arc, covert format
for i = 1:size(Edges.arc_ij, 1)
    for k = 1:Edges.commodityCount + 1
        fieldName = sprintf('c_%d',k);
        trainsPerArc = Edges.trainsPerArc.(fieldName);
        tArc = trainsPerArc(Edges.arc_ij(i,1), Edges.arc_ij(i,2));

        %record trains per arc per commodity information
        Edges.arcCapacity(i).trainsPerArc(k) = tArc;
    end
end

% Capacity equals kTon/train * trains/arc
for i = 1:size(Edges.arcCapacity, 2)
   tracksArc = Edges.arcCapacity(i).nmbrTracks;

   %fill in trains per arc data, calculate kg per arc per commodity k
   for k = 1:Edges.commodityCount + 1
           Edges.trainCapacity(k) * 1.5);
       if isnan(Edges.arcCapacity(i).kg(k))
```

```
            Edges.arcCapacity(i).kg(k) = 0;
        end
    end
end
```

A.1.8 Export SwRail Data

```
%export inflow, position, and station name from Nodes
NodesE = struct();

NodesE.pos_0 = Nodes.pos_0;
NodesE.pos_1 = Nodes.pos_1;
NodesE.node = [1:Nodes.nodeCount].';
NodesE.station = Nodes.station;

for k = 1:Nodes.commodityCount
    fieldName = sprintf('inflow_%d',k);
    NodesE.(fieldName) = Nodes.inflow(:,k);
end

struct2csv(NodesE, 'SwRailMatlabNodesExport.csv');

%export arc i,j, and capacity from Edges
EdgesE = struct();

EdgesE.arc_i = Edges.arc_ij(:,1);
EdgesE.arc_j = Edges.arc_ij(:,2);

for i = 1:size(Edges.arcCapacity, 2)

    for k = 1:Edges.commodityCount + 1
        fieldName = sprintf('capacity_%d',k);
        EdgesE.(fieldName)(i) = Edges.arcCapacity(i).kg(k);

    end

end

for k = 1:Edges.commodityCount + 1
    fieldName = sprintf('capacity_%d',k);
    EdgesE.(fieldName) = EdgesE.(fieldName).';
end

struct2csv(EdgesE, 'SwRailMatlabEdgesExport.csv');
```

Appendix A.2 Network Model

```
def optMCNF(G):

    ##Model the data
    #copy the node data
    nodes_h = deepcopy(nx.get_node_attributes(G, 'inflow'))
    nodes = deepcopy(G.nodes())

    #Number of commodities, extracted from lenght of 'inflow' list -
    #IMPORTANT for future code always ensure it matches intended number of
    #COMMODITIES
    N = 0

    #for each node, have a list of N commodities length attached to give[][]
    #index feature for i nodes and h commodities
    for i in nodes_h.iterkeys():
        count = 0
        j = nodes_h[i]
        for k in xrange(len(j)):
            nodes_h[i][k]=k

        if N == 0:
            N = len(j)

    #for each arc i,j add index h for N commodities as third index of tuple
    #list = N * number arcs in length
    larcs = G.edges()
    arcs_h = []
    arcs = tuplelist(G.edges())
    for i in xrange(len(larcs)):
        for k in xrange(N):
            ik = list(larcs[i])
            ik.extend([k])
            arcs_h.append(tuple(ik))


    arcs_h = tuplelist(arcs_h)


    #supply and demand dictionairies
    supply = {}
    demand = {}

    #remove super source/sink nodes from nodelist
    inflow = nx.get_node_attributes(G, 'inflow')

    #loop through attribute data
    for i,j in inflow.iteritems():
        for k in xrange(len(j)):
            if j[k] > 0:
                supply[(i, k)] = j[k]
                nodes_h.pop(i)
                break
            if j[k] < 0:
                demand[(i, k)] = -j[k]
                nodes_h.pop(i)
                break

```

```
    #determine capacity

    #seperate capacity for arc specific and link capacity
    capacity_h = {}
    capacity_arc = {}


    #k index for commodities
    c = nx.get_edge_attributes(G, 'capacity')

    #extract capacity from networkx graph
    for i, j in c.iteritems():
        for k in xrange(len(j)):
            if k < N:
                x = tuple([k],)
                capacity_h[i + x] = j[k]
            else:
                capacity_arc[i] = j[k]


    # Create optimization model
    m = Model('netflow')

    # Create decision variables, flow of commodity k across arc ij
    flow = {}

    for i,j,k in arcs_h:
        flow[i,j,k] = m.addVar(name ='flow_%s_%s_%s' % (i, j, k))


    m.update()


    # Arc capacity constraints, flow >= 0
    # seperate for loops for commodity specific capacity and arc capacity
    for i,j,k in arcs_h:
        #flow of commodity h across arc ij <= capacity of commodity h of
        #arcij
        m.addConstr(flow[i,j,k] <= capacity_h[i,j,k],
                    'cap_%s_%s_%s' % (i, j, k))

        #all flow across arc ij of commodity h is >= 0
        m.addConstr(flow[i,j,k] >= 0)

    for i,j in arcs:
        #flow of all commodities across arc ij <= capacity of arc ij
        m.addConstr(quicksum(flow[i,j,k] for i,j,k in arcs_h.select(i,j,'*'))
                    <= capacity_arc[i,j], 'arccap_%s_%s' % (i,j))


    # Flow conservation constraints
    #loop over every commdodity flowing through every node h
    for j, h in nodes_h.iteritems():
        for k in h:
            #flow in = flow out of node for every node and k within node
            m.addConstr(
                quicksum(flow[i,j,k] for i,j,k in arcs_h.select('*',j,k)) ==
                quicksum(flow[j,i,k] for j,i,k in arcs_h.select(j,'*',k)),
                'node_%s_%s' % (j, k))

    #flow of commodity k out of node j in supply <= supply of commodity k at
```

```
    #node j,
    for j, k in supply:
        m.addConstr(
            quicksum(flow[j,i,k] for j,i,k in arcs_h.select(j,'*',k)) <=
            supply[j,k],
            'supply_%s_%s' % (j, k))
        #flow into supply node j = 0
        m.addConstr(
            quicksum(flow[i,j,k] for i,j,k in arcs_h.select('*', j,k)) == 0)


    #flow of commodity k into node j in demand <= demand of commodity k at
    #node j,
    for j,k in demand:
        m.addConstr(
            quicksum(flow[i,j,k] for i,j,k in arcs_h.select('*',j,k)) <=
            demand[j,k],
            'demand_%s_%s' % (j,k))
        #flow out of demand node j = 0
        m.addConstr(
            quicksum(flow[j,i,k] for j,i,k in arcs_h.select(j,'*',k)) == 0)

    m.update()

    #set the objective, minimize unmet demand
    unmetDemand = LinExpr()

    for j,k in demand:
        #select all arcs flowing into demand node j
        flow_demand = arcs_h.select('*', j, k)
        for x, y, z in flow_demand:
            unmetDemand.addTerms(-1/demand[j,k], flow[x,y,z])
        unmetDemand.addConstant(1)

    m.setObjective(unmetDemand, GRB.MINIMIZE)

    m.update()

    m.optimize()

    flow_solution = m.getAttr('x', flow)

    return flow_solution
```

Appendix A.3 TOPSIS

```
function C_ideal = TOPSIS(X, w)


%Step 1(a) of TOPSIS: calculate sum(x^2(i,j))^1/2 for each column
step_1a = sum(X.^2,1).^(0.5);

%calculate rij
for i = 1:size(X, 2)
    if step_1a(i) == 0
        step_1b(:,i) = 0;
    else
        step_1b(:,i) = X(:,i)/step_1a(i);
    end
end

%step 2: multiply each column by weight to get vij
for i = 1:size(X, 2)
   step_2(:,i) = step_1b(:,i)*w(i);

end

%step 3: determine ideal and anti-ideal solution
for i = 1:size(X, 2)
    step_3a(i) = max(step_2(:,i));
    step_3b(i) = min(step_2(:,i));
end

%step 4: determine seperation from ideal, Euclidean distance
for i = 1:size(X, 2)
    step_4a(:,i) = (step_3a(i) - step_2(:,i)).^2;
    step_4b(:,i) = (step_3b(i) - step_2(:,i)).^2;
end

%sum across industry to complete step 4
S_ideal = sum(step_4a, 2);
S_anti = sum(step_4b, 2);

%calculate relative closeness to the ideal solution
for i = 1:length(S_ideal)
   C_ideal(i) = S_anti(i)/(S_ideal(i) + S_anti(i));
end


end
```

Appendix A.4 TOPSIS Ranking Spread

Appendix A.4.1. TOPSIS rank spread of unmet demand percentage performance with SIOT weightings.

TOPSIS Unmet Demand SIOT	Count of Rank SIOT Weights
0-0.04	1382
0.04-0.08	27
0.08-0.12	17
0.12-0.16	5
0.2-0.24	3
0.28-0.32	1
0.32-0.36	1
0.56-0.6	1
0.64-0.68	1

Appendix A.4.2. TOPSIS rank spread of unmet demand percentage performance with equal weightings.

TOPSIS Unmet Demand Equal Weights	Count of Rank Equal Weights
0-0.01	1310
0.01-0.02	46
0.02-0.03	27
0.03-0.04	11
0.04-0.05	14
0.05-0.06	4
0.06-0.07	6
0.07-0.08	2
0.08-0.09	1
0.09-0.1	2
0.1-0.11	1
0.11-0.12	2
0.12-0.13	3
0.13-0.14	4
0.14-0.15	1
0.16-0.17	1
0.17-0.18	1
0.18-0.19	1
0.21-0.22	1

Appendix A.4.3. TOPSIS rank spread of link usage > 90% performance with SIOT weights.

TOPSIS Link Usage SIOT	Count of Rank SIOT Weights
0.05-0.1	16
0.1-0.15	53
0.15-0.2	110
0.2-0.25	115
0.25-0.3	146
0.3-0.35	423
0.35-0.4	128
0.4-0.45	94
0.45-0.5	90
0.5-0.55	61
0.55-0.6	72
0.6-0.65	48
0.65-0.7	40
0.7-0.75	21
0.75-0.8	11
0.8-0.85	3
0.85-0.9	4
0.9-0.95	3

Appendix A.4.4. TOPSIS rank spread of link usage > 90% performance with equal weights.

TOPSIS Link Usage Eqal Weights	Count of Rank Equal Weights
0.15-0.25	46
0.25-0.35	730
0.35-0.45	461
0.45-0.55	161
0.55-0.65	36
0.65-0.75	4

Appendix A.4.5. TOPSIS ranking with combined importance metrics performance with top 5 SIOT weights.

TOPSIS Combined, SIOT Top 5	Count of Rank
0-0.1	1375
0.1-0.2	52
0.2-0.3	7
0.3-0.4	1
0.4-0.5	1
0.5-0.6	1
0.6-0.7	1