Approximate Dynamic Programming Based Approaches for Green Supply Chain Design

Yue Geng, Diego Klabjan
Department of Industrial Engineering and Management Sciences
Northwestern University
yue-geng@northwestern.edu, d-klabjan@northwestern.edu

February 17, 2014

Abstract

Nowhere are low emission operations more important than in logistics to remote pristine locations such as within the Arctic Circle. This study focuses on a long term economic assessment of research sites in Greenland from the logistics perspective. We study the strategic supply chain design problem as a bi-objective optimization problem, where cost and carbon footprint must be controlled. We model the problem as a time-spaced multi-commodity network flow problem with inventory tracking. To solve the multi-year model efficiently, we deploy an approximate dynamic programming (ADP) algorithm based on an approximation of the value function. To establish the efficient frontier between cost and emissions, an ADP based two phase algorithm is designed. We also compare the ADP based algorithms with traditional integer programming based algorithms. Computational results show that the ADP based two phase algorithm is more efficient and robust in determining the efficient frontier. We also show how these algorithms provide guidance for the operations of the logistics network.

1 Introduction

Basic scientific research on Greenland started decades ago. Due to the melting ice and other environmental changes, Greenland has become one of the test beds for scientific studies and explorations. Several remote sites were established on the ice sheet with a pristine landscape and untouched climate. They provide unprecedented opportunities for long term scientific achievements.

Greenland's remote location, vast landscape, and scarce population pose major logistics challenges. There are only limited options to transport goods and personnel. Maritime transport by vessels has an extremely long lead time and only limited service. An alternative is to use a scheduled airline service from Europe. The most widely used mode is chartered flights from the continental U.S. All these options lead to one of the two major towns in Greenland. From there either a chartered plane, helicopter, or a traverse, which is a landline transport by means of a tractor and trailers on year round snow clad surface, is needed to access any remote site. Several transportation modes and Greenland locations are illustrated in Figure 1. Each mode of transportation has attributes such as capacity, speed, cost, emission factors and availability in time. Different transportation modes have different attribute values. Traverse, for example, incurs a big cost and can only operate at a specific time of a year, but has a very low emission factor.

For years National Science Foundation (NSF) has conducted research projects on Greenland. Each year a number of research members from different scientific disciplines land in Greenland with research equipments. To provide them with a working and living environment, construction materials are also transported to build and support infrastructures. To provide energy for these sites, fuel is transported as well. It is used for power generation, heating and other purposes. Fuel can be purchased at several different locations at different prices and be stored in huge tanks at major remote sites.

Due to long lead times and special geographic circumstances, logistics operations are very challenging and costly. A tight budget and uncertain costs of operating a site pose

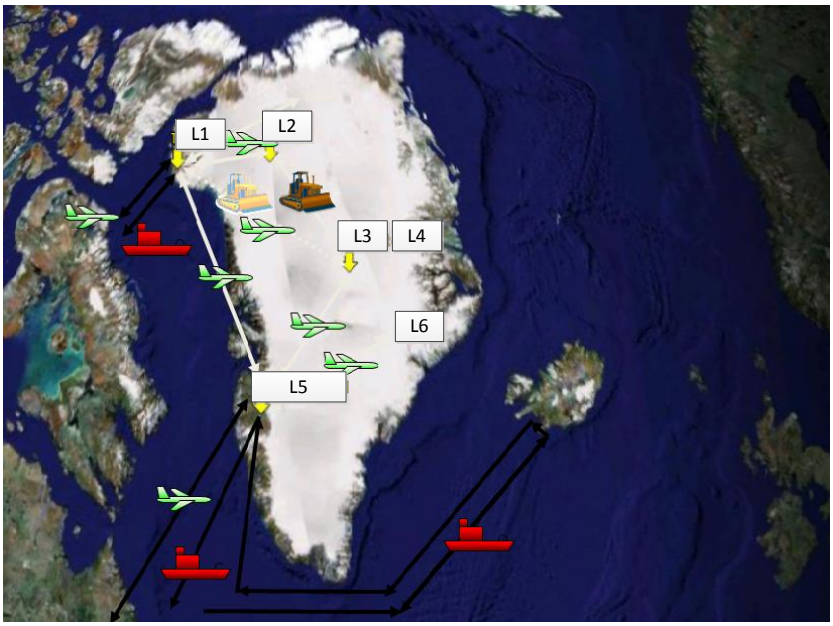


Figure 1: An illustration of Greenland logistics

major strategic decisions for NSF. Logistics costs are an important factor under consideration. Decision makers are faced with a daunting task of strategically sizing a site. The operations and research activities can stay at the same level, expand significantly, or slowly wind down. At the same time, to protect the pristine environment and not to skew the results of scientific experiments, carbon footprint from on-site fuel burning and transportation must be strictly controlled. Therefore, a tactical level guidance on economic and environmentally sustainable operations is also valuable. Over a time period of 10 years or less a decision maker must assess the logistics costs and environmental impact of delivering personnel and goods. Such planning includes detailed logistics decisions in each year such as when to make a transshipment and with what mode. Substantial fuel reserves are carried over from one month to the next.

The decision maker first creates a scenario of future research activities and then the corresponding transportation demand is estimated by using a model presented herein.

The goal of this research is to develop a decision support system that determines the logistics operations and cost over the planning horizon under a given scenario. The

environmental trade-off must be captured.

To achieve these goals, we develop a time-spaced multi-commodity flow model, where the flows of all commodities consisting of personnel, cargo and fuel are taken into account. The output of the model provides the decision makers with key metrics including the long-term cost, emissions and tactical level operations guidance. By specifying different possible scenarios, future feasibility and sizing of the sites can be assessed. The work flow of the full decision support system is depicted in Figure 2. Since to directly forecast the future demand for each commodity is difficult, we forecast the demand at the project level, i.e., the number of projects to be carried out under each scientific discipline at each location. This project level forecasting is then transformed into commodity level input based on fitted probability distributions with parameters estimated from historical data. As a possible scenario, if a new site is to be opened, we can add its location to the network and run the optimization engine to get cost, emissions and operations guidance. Similarly, if the research activity of a specific scientific discipline is to be increased by 10% per year, the optimization engine outputs the key metrics based on the adjusted new demand.

In this paper we focus on the optimization engine part, i.e., the supply chain network design model and the underlying solution methodologies. The problem can be modeled naturally using an integer program formulation. Since it is hard to solve it for the 10 year case, we develop an approximate dynamic programming (ADP) approach to minimize the total cost, where we solve a series of much smaller problems by approximating the future cost as a linear function. To establish the efficient frontier between cost and emissions, we could use IP based multi-objective optimization approaches. However, solving large-scale IPs is inefficient. Thus, we develop a two phase approach. In the first phase, a series of much smaller problems coming from our ADP approach are solved for each period based on the approximation of the future cost. In the second phase, we aggregate the solutions from the first phase to get near Pareto-optimal points over the full time horizon. Computational results show that for the cost minimization problem, the ADP based approach returns a reasonably good solution in a relatively short time. For the problem of minimizing both

cost and emissions, the two phase approach generates a large number of well-distributed near Pareto-optimal points and consumes less time than IP based methods.

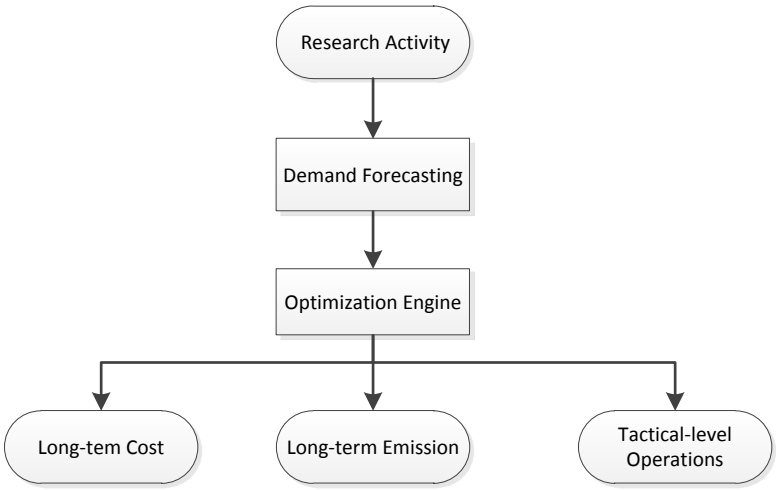


Figure 2: Workflow of the decision support system

Our contributions are twofold. First, we provide a real-world study about the effect of transportation modes on the trade-off between cost and emissions. Second, we provide the first ADP based algorithm for solving large scale multi-period multi-objective optimization problems. The algorithm decomposes the problem by time period and it never scans all states. To the best of our knowledge, our algorithm is the first algorithm for solving multi-period multi-objective problems that does not require a full enumeration of states and thus it scales well.

The rest of the paper is organized as follows. Section 2 provides a literature review. We specifically focus on logistics network design problems, where both cost and emissions are taken into account, and methods to solve multi-period multi-objective optimization problems. We formulate the problem in Section 3. In Section 4 we present the ADP algorithm to minimize the total cost. In Section 5, we develop the two phase algorithm to find the efficient frontier between cost and emissions. Computational results are presented in Section 6. In Section 7 we summarize the research and discuss possible future directions.

2 Literature Review

Our model solves a multi-period supply chain network design problem with fuel inventory tracking. There is abundant literature on supply chain network design. Selected survey articles on the topic are Vidal and Goetschalckx (1997), Melo et al. (2009), Mula et al. (2010). The most common technique is to formulate the problem as an IP and solve it using approaches such as branch-and-bound or heuristics (Wolsey (1998)). Studies on multi-commodity flow models (see Ahuja et al. (1993)) are also relevant to our proposed formulation.

An important feature of our work is to capture emissions. This is of key importance due to the location of the remote sites and the requirements of scientific experiments carried out at the sites. To this end, we discuss how to calculate transportation emissions based on different types of data. Of all greenhouse gas emissions, carbon dioxide (CO_2) contributes the most, in excess of 95%. On the other hand, CO_2 emissions are straightforward to estimate since they are primarily dependent on two factors: the type and quantity of fuel burnt. To estimate the emissions in transportation, either a fuel-based or distance-based methodology can be applied (GHG (2008)). In the fuel-based approach, fuel consumption is multiplied by the CO_2 emission factor for each fuel type. In the distance-based method, emissions can be estimated using distance-based emission factors. It is advantageous to adopt the fuel-based approach whenever fuel consumption data is available, since it can generate a more reliable result. A tool for computing the transportation emission is the FLEET model developed by the Environmental Protection Agency (SmartWay (2010)). It provides spreadsheets to calculate CO_2, particulate matter and nitrogen oxides emissions, based on a pre-defined metric as input unit.

There is also an abundance of research that examines green logistics network designs. Srivastava (2007) gives an extensive survey. Rodrigue et al. (2001) list four basic paradoxes of the green logistics design: reducing costs, shortening transportation time, increasing reliability and reducing the number of warehouses may not reduce the overall environmental

impact. Sbihi and Eglese (2010) discuss some of the problems in the area of green logistics that can be formulated as combinatorial optimization problems. They point out that "classical" vehicle routing and scheduling models aim to minimize cost (usually related to the number of vehicles and distance). Although this will provide some environmental benefit compared with solutions that use unnecessary resources, models that explicitly consider emission as an objective should be considered. We study our problem as a bi-objective optimization problem where both cost objective and emission objective are taken into account.

In our bi-objective optimization problem, we construct the efficient frontier, which can be viewed as the trade-off curve between cost and emissions. Each point on the curve is a Pareto optimal point, i.e., a non-dominated solution. For an introduction to multi-objective optimization methods, readers are referred to Marler and Arora (2004). These optimization methods can be divided into two main categories: weighting methods where a new objective function is formed by aggregating each single objective function based on the given preference, and Pareto-based algorithms which establish relationships among solutions according to the Pareto-dominance concept (Banos et al. (2011)). Cohon (2003) provides an introduction to the weighting method. Work on Pareto-based methods is extensive. Methods such as goal programming (Romero (1991)) and the ε-constraint method (Miettinen (1999)) find a single Pareto point in a single run while evolutionary approaches such as genetic algorithms can return a set of Pareto optimal points. The latter attracted great attention from researchers in recent years. Coello et al. (2007) provide a comprehensive introduction to evolutionary approaches for solving multi-objective problems, which include local search, genetic algorithms, simulated annealing, tabu search, etc.

Multi-objective approaches have been used to study logistics network design problems where both cost and environmental impacts are considered. Wang et al. (2011) introduce a green supply chain network design model which is based on the classical facility location problem. Different from our problem, their problem is single-period with no procurement decisions considered. The model is solved by a normalized constraint method with the

CPLEX solver to obtain a well-distributed Pareto frontier. Each run returns a single Pareto point. Bouzembrak et al. (2011) propose a single-period supply chain model that lists both cost and emissions as objectives. A commercial solver is adopted to obtain Pareto optimal points. Ramudhin et al. (2008) is a similar work where a single-period green supply chain network design model is proposed. The IP model explicitly integrates carbon prices and total cost. The trade-off between cost and emissions are also computed using a goal programming approach and the model is solved by CPLEX. Paksoy et al. (2010) present a closed-loop supply chain network model that considers both transportation cost and emissions. The study also takes into account the benefit of using recyclable products. All these articles adopt a single-period logistics network model and the problem scale is small or medium, whereas our problem is multi-period and large scale, which makes solving directly the IP model difficult.

Methods for solving multi-period multi-objective optimization problems can be divided into three categories. The first category is to solve the model directly by using commercial solvers without decomposing the problem on the time dimension. The strategy is weight-based in order to derive a simple objective function. Under this category, Ustun and Demirtas (2008) propose a supplier selection model in a lot-sizing environment. Meza et al. (2007) adopt an analytical hierarchy process to find the Pareto optimal points in the context of power generation expansion. Wang and Liang (2004), Liang (2008), Torabi and Hassini (2008) use fuzzy multi-objective programming models to solve multi-period supply chain planning problems, including transportation of commodities. Silverman et al. (1988) use the interactive augmented weighted Tchebycheff method to solve a multi-period manpower planning problem. Since these methods require a subroutine to solve the full IP, the problem size handled is usually small or medium while we are facing a large scale problem. The second category uses evolutionary based algorithms. Hatzakis and Wallace (2006) provide an evolutionary type algorithm named the queueing multi-objective optimizer, combined with a feed-forward forecasting strategy, to solve multi-objective dynamic problems. Since their focus is more on the underlying dynamic environment, the algorithm

only finds the efficient frontier in the last time period. The third category adopts dynamic programming based algorithms which decompose the problem over the time dimension. Chankong et al. (1981) integrate a single-objective dynamic programming method with a surrogate worth trade-off method (Haimes and Freedman (1975)) to solve a capacity expansion problem. Kim and Ah (1993) adapt a preference-order dynamic programming approach to solve a power generation-expansion planning problem. Both of these algorithms require scanning all states.

In summary, known multi-period multi-objective algorithms either require solving many times an entire IP model (in essence, neglecting the multi-period aspect), or find the efficient frontier only in the last time period, which clearly does not meet our needs, or they decompose by time but iterate through all states. In our case due to the sheer size of our problem, the first and third type of algorithms would be intractable. The third type does not apply to our problem due to the large state space. To circumvent this, we develop an ADP (Powell (2007)) based two phase algorithm, which returns a set of Pareto optimal points efficiently.

3 Problem Formulation

In this section, we formulate the problem. We first show how to transform the problem to a network flow graph, then we present a dynamic programming (DP) and integer programming (IP) formulation. Both formulations are based on the constructed network flow graph.

To start, we state the specifications of a scenario, which is the input of the optimization engine. This includes the geographic locations, vehicle modes, and for each vehicle mode the vehicle capacity, cost and emission factors, and travel time among the locations (a vehicle mode may only serve a given pair of locations), fuel purchase locations and the associated prices for a gallon of fuel, and forecasted demands. Forecasted demands consist of the weight, location and time window for each commodity, including personnel, cargo and

fuel. For personnel and cargo, the time window refers to the time window requirement for dispatching the commodity at its origin and the time window for arriving at its destination. For example, scientific equipment is ready within a certain period at a given location and it must be delivered at the site of the project within a time period. For fuel, the demand is forecasted at each location that has fuel usage, and there is no explicit time window for the purchase or transportation of fuel.

3.1 Model as a network flow problem

To capture flows of personnel, cargo and fuel over time, we adopt a multi-period multi-commodity network flow model. We create L nodes to represent L geographically different locations. To take time dimension into account, each location node is duplicated H times if there are H weeks in the planning horizon (for simplicity we assume that a week is the smallest appropriate time unit). So each node represents a combination of location and time and the number of such regular nodes equals LH. An arc from node i to node j is created if an available transportation mode can start from the location and time represented by node i and arrive at the location and time denoted by node j. Each arc has one and only one associated mode of transportation. There is also a special type of arcs which connect the same location and extend from one week to the next. These arcs denote the inventory carried to the next week at this location.

All commodities, including personnel, cargo and fuel, are also captured in the graph. A commodity is associated with an origin and a destination, and at the source and sink time windows are imposed. A given weight of a commodity must be shipped and received within the given time window. To capture this, $2K - 1$ dummy nodes are created where K is the number of different commodities (note that we do not create a dummy node for fuel demand since fuel demand is imposed at each node). For example, if commodity A is to be sent out at location a during weeks 1 or 2, then two dummy arcs are created. They start from the dummy node representing commodity A, and end at the two nodes which represent location a in week 1 and location a in week 2 (see Figure 3). An arc can be a

transportation arc associated with a transportation mode, or an inventory arc tracking the inventory of commodities, or a dummy arc. Note that the total number of nodes equals $LH + 2K - 1$ and each dummy node is associated with either a supply or demand.

Figure 3 is a demonstration of the constructed directed graph. Vertically three locations a, b and c are shown. A time horizon of eight weeks is presented in the horizontal direction. Each regular commodity has time windows imposed at both its source and sink location. By using dummy node "Commodity Fuel" to represent fuel supply, fuel purchase decisions can also be handled. The amount of flow from this "Commodity Fuel" node to node $a1$ denotes fuel purchased at location a in week 1. Note that there can be multiple fuel purchase locations. Fuel purchased at an earlier time can be stored in inventory and carried over to a later time via the horizontal inventory arcs. Our goal is to find the minimum cost flow so that the transportation cost plus fuel purchase cost is minimized and the carbon footprint is controlled.

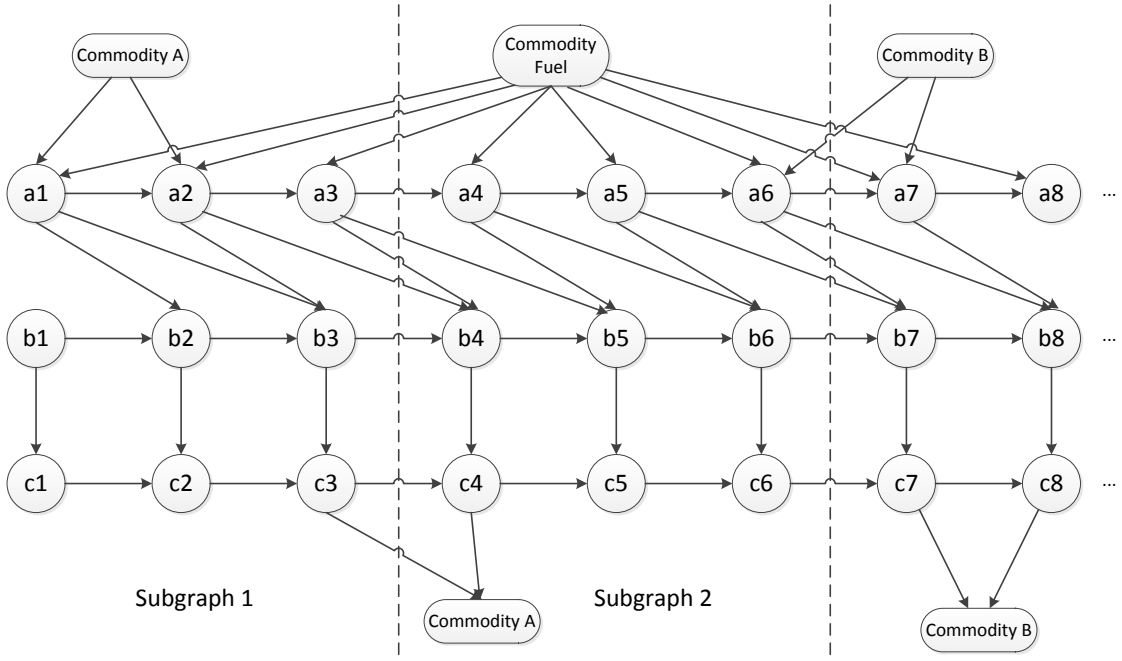


Figure 3: Construction of the full network flow graph

When a long time horizon is considered, for example, ten years, the number of nodes and arcs would increase dramatically, and modeling this network flow problem in one shot would lead to a huge model. To make the problem more tractable, a natural approach is to decompose the full graph into smaller subgraphs. Based on the observation that there are relatively few arcs reaching to a distant node in time due to limited lead times, we decompose the graph in time dimension and formulate the problem as a dynamic program. As shown in Figure 3, the full graph can be decomposed into subgraphs by the dashed lines according to the defined length of a time period, which is set to three weeks in Figure 3. The length of a time period should be chosen carefully so that both the number of the across-graph arcs and the size of each subgraph are kept small. Note that if the length of the time period equals the total number of weeks, then we have only one subgraph which is also the full graph. To maintain tractability, we only allow fuel to link subgraphs and we confine shipments to be within a single time period (there are some special cases, which are discussed in Section 4).

These subgraphs are shown in Figure 4. Note that the dummy fuel source node is replicated if necessary. The across-subgraph arcs are modified and across-subgraph commodities such as commodity A are re-assigned. The rules for these steps are detailed in Section 4. We next present the mathematical formulation.

3.2 Dynamic program

Based on the graphs constructed, we formulate the problem in a DP manner. Each subgraph defines a single-period problem. We use $\mathbf{N}_t$ and $\mathbf{A}_t$ to denote the node set and arc set for period t, respectively, and we use $\mathbf{S}_t$ and $\mathbf{U}_t$ to denote the node set representing the first and the last week in period t, respectively. Let F_t denote the dummy fuel source node in period t, and we use d_i^k to denote demand for commodity k ($1 \leq k \leq K$) at node i. Note that commodity K denotes fuel. All parameters are listed in Appendix A.

As an example for the node set notation, in Figure 4 we have $\mathbf{S}_2 = \{a4, b4, c4\}$ and $\mathbf{U}_2 = \{a6, b6, c6\}$. Note that if the length of one time period is set to 1 week, then $\mathbf{S}_t = \mathbf{U}_t$.

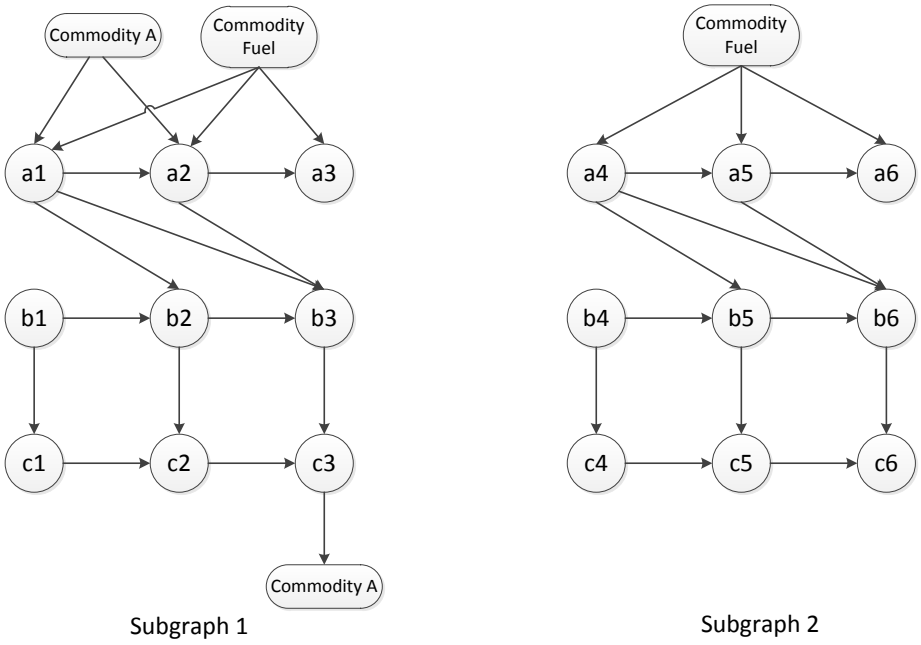


Figure 4: Construction of subgraph 1 and subgraph 2 from Figure 3

Demand d_i^k is part of a scenario and its forecasting is presented in Section 4. Demand d_i^k is positive if it is a "true" demand node, negative if it is a supply node and zero if it is a transshipment node. Decision variables in period t read

- $x_{t,a}^k$: weight of commodity k transported on arc $a \in \mathbf{A}_t$,
- $y_{t,a}$: number of vehicles used on arc $a \in \mathbf{A}_t$,
- f_{F_t}: amount of fuel purchased in time period t.

Note that each transportation arc is associated with a unique vehicle mode, which has the following attributes: capacity, cost and emission factors, available time periods, and travel time between the two locations. Variable f_{F_t} denotes the fuel purchased in period t, i.e., the amount of fuel flow out of the dummy fuel node F_t. It has a negative value since it provides a negative fuel demand. Note that we can purchase fuel at an earlier time and store it as inventory, which can be carried over to the next time period. Thus

although the total fuel demand is known over the full time horizon, the amount of fuel to purchase in each time period remains unknown. Using the above notation, the problem can be formulated as a DP. For period t, the system state is the fuel inventory level at the beginning of this period at each location i, denoted by $I_{t,i}$. We reiterate that all shipments occur within a single time period. The system dynamics are described by the following equations and they include only fuel.

$$I_{t,i} + \sum_{a=(j,i)\in\mathbf{A}_t} x^K_{t,a} = \sum_{a=(i,j)\in\mathbf{A}_t} x^K_{t,a} + d^K_i \qquad i \in \mathbf{S}_t \tag{1}$$

$$\sum_{a=(i,j)\in\mathbf{A}_t} x^K_{t,a} = I_{t+1,j} + \sum_{a=(j,i)\in\mathbf{A}_t} x^K_{t,a} + d^K_j \qquad j \in \mathbf{U}_t \tag{2}$$

$$\sum_{a=(j,i)\in\mathbf{A}_t} x^K_{t,a} = \sum_{a=(i,j)\in\mathbf{A}_t} x^K_{t,a} + d^K_i \qquad i \in \mathbf{N}_t \setminus (\mathbf{U}_t \cup \mathbf{S}_t). \tag{3}$$

Action space $\mathcal{F}_t$ is defined as follows.

$$f_{F_t} = -\sum_{a=(F_t,j)\in\mathbf{A}_t} x^K_{t,a} \tag{4}$$

$$\sum_{a=(j,i)\in\mathbf{A}_t} x^k_{t,a} = \sum_{a=(i,j)\in\mathbf{A}_t} x^k_{t,a} + d^k_i \qquad i \in \mathbf{N_t}, k \in \{1, ..., K-1\} \tag{5}$$

$$\sum_{k\in\mathbf{K}} x^k_{t,a} \le C_a y_{t,a} \qquad a \in \mathbf{A}_t \tag{6}$$

$$\sum_{a=(j,i)\in\mathbf{A}_t} x^K_{t,a} \le \bar{I}_i \qquad i \in \mathbf{N}_t \setminus \mathbf{S}_t \tag{7}$$

$$I_{t,i} + \sum_{a=(j,i)\in\mathbf{A}_t} x^K_{t,a} \le \bar{I}_i \qquad i \in \mathbf{S}_t \tag{8}$$

$$\sum_{a=(i,j)\in\mathbf{A}_t} x^K_{t,a} \ge \underline{\mathrm{I}}_i \qquad i \in \mathbf{N}_t \setminus (\mathbf{U}_t \cup \{F_t\}) \tag{9}$$

$$I_{t+1,i} \ge \underline{\mathrm{I}}_i \qquad i \in \mathbf{U}_t \tag{10}$$

$$x^k_{t,a} = 0 \qquad (k,a) \in \mathbf{R}. \tag{11}$$

Since the system state is the fuel inventory level, constraints (1) - (3) in system dynamics capture the fuel flow balance constraints. Constraint (4) states that the fuel supply from

the dummy fuel node F_t equals the total fuel purchased over all location j. Constraint (5) enforces the flow balance for other commodities, while constraint (6) states that the amount of flow on any arc cannot exceed the capacity provided by the vehicles assigned to this arc. Constraints (7) - (10) confine the fuel inventory level due to limited storage availability, where $\bar{I}_i$ is the capacity of the fuel tank at location i, and $\underline{I}_i$ addresses the safety fuel stock at location i. Constraint (11) states that certain commodities cannot be transported on certain arcs since the modes of these arcs are not appropriate, e.g., we never use a vessel to transport research personnel due to its long travel time.

To capture both cost and emissions we present two different objective functions. Let c_a denote the cost of one vehicle to travel on arc a, let p_i denote per unit weight purchase cost of fuel at node i, and let g_a denote emissions produced by one vehicle traveling on arc a, where g_a can be computed based on emission factors of the given transportation mode. The objective function for period t is either minimizing cost

$$\min \sum_{a \in \mathbf{A}_t} c_a y_{t,a} + \sum_{a=(F_t,j) \in \mathbf{A}_t} p_j x^K_{t,a}, \tag{12}$$

or minimizing emissions

$$\min \sum_{a \in \mathbf{A}_t} g_a y_{t,a}. \tag{13}$$

Note that on-site fuel burning is another major source for emission. However, it can be estimated by computing on-site fuel consumption using demand data, therefore, it can be excluded from the optimization model.

Let $V_t(\mathbf{I}_t)$ denote the value function for state $\mathbf{I}_t$, where $\mathbf{I}_t$ is a vector with elements $\{I_{t,i}\}_i$. The cost minimization Bellman equation reads:

$$V_t(\mathbf{I}_t) = \min_{(x^k_{t,a}, y_{t,a}) \in \mathcal{F}_t} \{ \sum_{a \in \mathbf{A}_t} c_a y_{t,a} + \sum_{a=(F_t,j) \in \mathbf{A}_t} p_i x^K_{t,a} + V_{t+1}(\mathbf{I}_{t+1}) \}. \tag{14}$$

Value function $V_1(\mathbf{I}_1)$ denotes the minimum cost starting in time period 1 in state $\mathbf{I}_1$,

which is also the optimal cost for our problem. This DP formulation provides a foundation to the algorithm developed in Section 4.

Note that if we do not separate the graph into subgraphs, we can also formulate the problem based on the full graph. Alternatively, if we let the length of the DP period equal the total number of weeks, it gives rise to an IP formulation.

4 An Approximate Dynamic Programming Based Approach for Cost Minimization

In a strategic decision problem, decision makers usually need to foresee the operations over a long period, which in our case involves the next ten years. Under such situations, the IP model usually falls short due to the large size of the problem. Therefore, we develop an ADP algorithm to obtain quality solutions efficiently.

We first describe the demand forecasting procedure. With forecasted demand, we present the ADP algorithm. We first show a simple yet effective approximation for the value function, then we discuss how to update value function coefficients and how to handle special cases including across-graph commodities and across-graph arcs.

4.1 Demand Forecasting

As shown in Figure 2, the optimization engine needs to take the future demands of all commodities as input. To construct the graphs, we also need to have the source, destination and time windows for each commodity. We describe in this section a method to estimate demands based on historical data. The basic idea is to fit probability distributions at the project level and to generate all commodities based on these distributions.

The decision makers do not deal with commodities which are part of logistics operations. Instead they are much more comfortable with the notion of projects. Discerning the need of an individual future project is also too much to handle, especially for projects many years in

the future. The decision makers however can make high level predictions at the level of the number of projects in a given discipline. This is our starting point. The forecasting module takes these estimates and creates commodity level forecasts. The number of projects to carry out under each discipline can be regulated by a trend parameter, which is an input of the decision maker. The decision maker can specify these numbers for each year and discipline. This also allows what-if analyses.

We need to transport fuel, personnel and other regular commodities. To forecast personnel and other regular commodities, we divide historical shipments into projects that belong to different scientific disciplines. It has been observed that each scientific discipline has its own patterns for locations, project group size, arrival time, duration of experiments, etc. Therefore, we fit distributions for personnel arrival time and length of stay, cargo arrival time and length of stay, personnel group size and cargo weight for each project at each location for each scientific discipline. The goodness of fit for each distribution is established by the chi-square test.

For fuel, the demand can be decomposed into two parts: base level consumption and activity based consumption. The former denotes the fuel that is used for basic infrastructure, which can be treated as constant year round. The latter accounts for human activities and is determined by the number of personnel on site. Therefore, we regress fuel demand against the number of personnel. The constant part can be treated as the base level and the linear coefficient represents the per person consumption. The base level can be adjusted to accommodate possible future infrastructure changes.

4.2 Approximate Dynamic Programming

In Section 3.2 we formulated the DP, however, the large state space makes the traditional backward DP method prohibitive. To overcome this, we adopt the ADP approach with an approximated value function (Powell (2007)).

The idea of ADP is to move forward in time by iteratively solving the Bellman equation, and updating the approximated value function at the same time. Note that if $V_{t+1}(\mathbf{I}_{t+1})$ is

known for every t, (14) could be solved efficiently since it is an IP with a much smaller size. We approximate $V_{t+1}(\mathbf{I}_{t+1})$ by a linear function. Note that as fuel inventory level increases, the future purchase cost for fuel would decrease, and the transportation cost would also decrease since less fuel needs to be transported. It was observed in several applications that for monotone problems, linear functions are good approximations. Let $\alpha_{t+1,i}$ denote the linear coefficient associated with state $I_{t+1,i}$ (the value of the intercept is irrelevant). Then we have

$$V_t(\mathbf{I}_t) = \min_{x^k_{t,a}, y_{t,a} \in \mathcal{F}_t} \{ \sum_{a \in \mathbf{A}_t} c_a y_{t,a} + \sum_{a=(F_t,j) \in \mathbf{A}_t} p_i x^K_{t,a} + \sum_{i \in \mathbf{L}} \alpha_{t+1,i} I_{t+1,i} \}. \tag{15}$$

Thus in period t, we solve the following IP,

$$\min_{x^k_{t,a}, y_{t,a}} \sum_{a \in \mathbf{A}_t} c_a y_{t,a} + \sum_{a=(F_t,j) \in \mathbf{A}_t} p_i x^K_{t,a} + \sum_{i \in \mathbf{L}} \alpha_{t+1,i} I_{t+1,i} \tag{16}$$

$$\text{subject to constraints (1) - (11).}$$

Next we develop a cost minimization ADP based on this formulation. We follow the framework from Powell (2007).

The ADP has two levels of loops. The outer loop updates the step size. The inner loop iterates from period 1 to T and updates coefficients of the value function. We present the algorithmic framework in Algorithm 1.

The loop in steps 4 to 9 iterates forward in time based on the current value function approximation. Step 5 solves the Bellman equation as the IP while step 7 handles some special cases which are described later. In step 8, we use the sum of the dual values of the root node LP relaxation corresponding to the constraints that contain $I_{t,i}$, to update the current slopes. Additional details are presented later. Step 10 is the so-called McClain's formula (McClain (1974)), for updating the step size. Parameter $\bar{w}$ is often set to a small positive number, e.g., 0.1. The following section gives more detailed descriptions of key

1: Initialize linear coefficients $\alpha_{t,i}$ for $t = 1, ..., T$, $i \in \mathbf{L}$
2: **for** $n = 1, ...$ **do**
3: Set initial fuel inventory level $\mathbf{I}_1$
4: **for** $t = 1, ..., T$ **do**
5: Solve the IP with objective function (16) and constraints (1) - (11). Record the dual values of the root node LP relaxation.
6: Update system state, i.e., the value of $\mathbf{I}_{t+1}$ by using (2) based on the solution $(x^k_{t,a}, y_{t,a})$ from the IP
7: Handle special cases, including across-graph commodities, arcs and vehicle number upper bounds
8: Update slopes: $\alpha_{t,i} = w_n \lambda_{t,i} + (1 - w_n)\alpha_{t,i}$ for each $i \in \mathbf{L}$, where $\lambda_{t,i}$ is the sum of the dual values corresponding to constraints (1) and (8)
9: **end for**
10: $w_{n+1} = \frac{w_n}{1+w_n-\bar{w}}$, where $\bar{w}$ is a parameter
11: **end for**

Algorithm 1: ADP algorithm for cost minimization

steps.

4.2.1 Implementation

A careful initialization of the value function coefficients is very important for a successful ADP algorithm. It enables the algorithm to converge to a better solution in less time. Linear coefficients $\alpha_{t,i}$ indicate the cost saved when an additional unit of fuel is stored in inventory. To this end, we first compute the average transportation cost to send a unit of fuel from each fuel supply location s to location i, by specifying a most commonly used vehicle type. For example, if one unit of fuel only takes up one percent of the flight capacity, then the charge would be one percent of the charge of the entire flight. Next we add this transportation cost to the purchase cost of a unit of fuel to obtain the total cost for a unit of fuel at location i, supplied from location s. By averaging the number over all the fuel supply locations, an estimation of $\alpha_{T,i}$ is obtained. We let $\alpha_{t,i} = \delta \cdot \alpha_{T,i}/T \cdot t$, for $t = 1, ..., T-1$ and $i \in \mathbf{L}$, where δ is a scale parameter.

To update $\alpha_{t,i}$, the dual values from the root node LP relaxation are needed. In step 8 of Algorithm 1, we have $\lambda_{t,i} = \pi_{t,i,1} + \pi_{t,i,2}$, where $\pi_{t,i,1}$ and $\pi_{t,i,2}$ correspond to constraints for node i among constraint sets (1) and (8), respectively. This is a reasonable update

since a dual value represents the change in the objective value if the right hand side of the corresponding constraint changes by a small amount. In our case it measures how much does the cost change if we have one more unit of fuel in inventory. Thus it provides an approximation to the current slopes. Step size w_n weighs the previous estimation of slopes against the slopes obtained in the current iteration. Note that as n increases, the step size approaches $\bar{w}$, which is a small number. Thus the newly generated slopes have a diminishing effect and the slopes finally stabilize.

In the following we discuss how to handle special cases that destroy the DP structure, such as shipments that span more than one subgraph.

As shown in Figure 3, some arcs span more than one subgraph. If it is an arc representing a transportation mode, we call it an *across-graph arc* (across-graph inventory arcs are already handled by state variable $I_{t,i}$). Each commodity except fuel has the corresponding source and sink nodes. If either its source or sink node has arcs connected to different subgraphs, we call this commodity an *across-graph commodity* (fuel only has one source node, which is replicated in each subgraph as discussed in Section 3). If time periods are large enough, there would be relatively few across-graph arcs and thus they can be disposed when constructing subgraphs.

The case of across-graph commodities is more complicated. If a commodity's source or sink node only has arcs within a single subgraph, then this source or sink node simply belongs to this subgraph. Otherwise, the node belongs to the subgraph that has more arc connections to it (if it is a tie, then break up strategies need to be designed). For an across-graph commodity, if both its source and sink nodes belong to the same subgraph, then no special treatment is needed; otherwise, the commodity has only the source in the time period and the flow is routed in the usual way except that it can be accumulated at transition locations in the last week of the time period. Such accumulations are carried over to the time period where the sink resides and are matched. Note that this is an approximation since the carry over amounts are not part of the state space or the value function.

Another special requirement is that there is a limited number of certain types of vehicles in a year, which is handled approximately. For example, a traverse can only be used once per year within a specified set of locations during a limited time period. For such a case, when stepping forward, being at the end of a time period, we record the remaining number of times that the vehicle type can be used. If the number decreases to zero, we disallow the type until the period corresponding to the next year.

5 A Two Phase Approach for Pareto Frontier

The application considered herein requires a strict control of the emission level. Therefore, the trade-off between cost and emissions becomes very important for the decision maker to consider. To this end, Pareto-optimal regions are of interest where cost is kept within a certain range of the minimum cost.

Emissions are composed of on-site and transportation emissions. While the former can be estimated directly based on the demand, the latter largely depends on the underlying logistics plan. In this section, we introduce an ADP-based method for constructing the efficient frontier by obtaining enough points approximately on it. The algorithm has two phases. In phase I, the single-period trade-offs are calculated. In phase II, a dynamic programming method is adopted to synthesize the single-period results. Note that for a small to medium problem, an IP solver can be used directly within a goal-programming, or weighting method framework, or other multi-objective optimization frameworks. However, when the problem size becomes large, our approach is much more reliable and scalable.

In the following we use z_t and e_t to represent the cost and emissions for period t, and Z_t and E_t represent the cost and emissions from period 1 to period t. Furthermore, let (Z_T^*, E_T^*) denote a Pareto-optimal solution given a fixed initial inventory level.

5.1 Divide and conquer

In the ε-constraint method for multi-objective optimization, one of the objective functions is selected as the objective, and all other objective functions are converted into constraints by setting an upper bound on each one of them. Assuming that each single-period problem is solved by this method given an initial inventory level (ideally this initial inventory level equals the ending inventory level of the previous period), we can obtain a trade-off curve rapidly by setting (12) as the objective function and using constraint

$$\sum_{a \in \mathbf{A}_t} g_a y_{t,a} \leq \varepsilon_t \tag{17}$$

as an additional constraint for emissions. By varying the upper bound ε_t, we can obtain enough points on the efficient frontier.

Unfortunately, summing up these single-period Pareto-optimal points does not lead to global Pareto-optimal solutions since all future contributions need to be taken into account. To amend this, we use coefficients α obtained from Algorithm 1 and optimize (16) instead. The resulting IP formulation for period t is

$$\min_{x^k_{t,a}, y_{t,a}} \sum_{a \in \mathbf{A}_t} c_a y_{t,a} + \sum_{a=(F_t,j) \in \mathbf{A}_t} p_i x^K_{t,a} + \sum_{i \in \mathbf{L}} \alpha_{t+1,i} I_{t+1,i} \tag{18}$$

$$\text{subject to constraints (1) - (11) and (17).}$$

We call this an *emission-constrained problem*.

The rational behind our approach is that the global Pareto-optima can be reached if we can allocate the total allowed emission in an appropriate way to each period. On one hand, the objective function (16) accounts for the cost from period t to T. Thus, an optimal solution $(x^k_{t,a}, y_{t,a})$ of the emission-constrained problem can be interpreted as a globally optimal cost-driven action. On the other hand, any E^*_T of a Pareto-optimal solution (Z^*_T, E^*_T) can be written as $E^*_T = \sum_{t=1}^{T} e^*_t$, where e^*_t is the emission allowed in period t. Thus, if we can optimally allocate the total emissions allowed into the emission allowed

for each period, under the assumption that constraint (17) does not affect coefficients α_{t+1}, then solving the above emission-constrained problem provides a global optimal cost-emission pair (z_t^*, e_t^*) for period t. Note that an approximation has been made since constraint (17) actually skews α_{t+1}, but in practice we are interested in the region of the Pareto-optimal solutions where the cost values are only slightly increased from the minimum cost, and thus this assumption is reasonable. An extreme case is when the emission upper bound ε_t is large, resulting in actually minimizing the total cost without considering emissions, which yields exactly the same α as the one obtained from Algorithm 1.

The remaining issues are that E_T^* is unknown and the rule to allocate E_T^* to time periods unclear. To circumvent these difficulties, we increase ε_t from the smallest to the maximum possible value and solve the emission-constrained problem multiple times. We can sum over all these single period cost-emission pairs to restore Z_T^* and E_T^*. In order to obtain the maximum possible value for ε_t, we minimize cost function (12) with constraint set (1) - (11) and obtain the maximum emission value $\bar{e}_t$ based on the solution. To obtain the minimum emission $\underline{e}_t$, we minimize emission function (13) subject to the same set of constraints.

Note that the emission-constrained problem deals with a single period. Therefore, we must guarantee that the ending inventory level of the previous period is the same as or within a certain range of the beginning inventory level of the current period. To achieve this, we solve the emission-constrained problem for a subset of initial inventory levels. We divide the inventory range at location i into K_i buckets, and let $\mathcal{I} = \{(I_1^{k_1}, I_2^{k_2}, ..., I_L^{k_L}) | 0 \leq k_i \leq K_i, ..., 0 \leq k_L \leq K_L\}$ denote the set of all possible inventory level combinations. Therefore, for each emission-constrained problem, we can obtain a four-dimensional vector $(\mathbf{I}_t, \mathbf{I}_{t+1}, z_t, e_t)$ which later needs to be matched with the solution of its previous period and its following period. With all these components, we present the divide-and-conquer algorithm in Algorithm 2, which finds all such four-tuples.

In step 1 of Algorithm 2, the inventory range at each location i is divided into K_i

1: Divide inventory level $I_{t,i}$ evenly into K_i buckets $[I_i^0, I_i^1], ..., [I_i^{K_i-1}, I_i^{K_i}]$
2: **for** $t = 1, ..., T$ **do**
3: **for** $\mathbf{I}_t \in \mathcal{I}$ **do**
4: Obtain $\underline{e}_t$ by solving the IP with objective (13) and constraint set (1) - (11)
5: Obtain $\bar{e}_t$ by solving the IP with objective (12) and constraint set (1) - (11)
6: Divide $[\underline{e}_t, \bar{e}_t]$ into N_t buckets $[e_{t,0}, e_{t,1}], ..., [e_{t,N_t-1}, e_{t,N_t}]$, where $e_{t,0} = \underline{e}_t$ and $e_{t,N_t} = \bar{e}_t$
7: **for** $\varepsilon_t = e_{t,0}, e_{t,1}, ..., e_{t,N_t}$ **do**
8: Solve the emission-constrained problem (18)
9: Obtain cost z_t^* and emission e_t^* based on the solution $(x_{t,a}^k, y_{t,a})$
10: **end for**
11: **end for**
12: Obtain set Q_t of non-dominated solutions $(\mathbf{I}_t^*, \mathbf{I}_{t+1}^*, z_t^*, e_t^*)$
13: **end for**

Algorithm 2: Phase I: divide-and-conquer

buckets. As the number of buckets increases, the algorithm returns more accurate solutions but with longer running time. In steps 4 and 5, the lower and upper bounds of ε_t are computed. In step 9, after solving the emission-constrained IP, the actual emission e_t is recomputed since allowed emission ε_t may be excessive. In step 12, only the non-dominated cost-emission pairs are left to be explored in Phase II. The dominance relation is defined as follows. Given two solutions $(\mathbf{I}_t^1, \mathbf{I}_{t+1}^1, z_t^1, e_t^1)$ and $(\mathbf{I}_t^2, \mathbf{I}_{t+1}^2, z_t^2, e_t^2)$, $(\mathbf{I}_t^1, \mathbf{I}_{t+1}^1, z_t^1, e_t^1)$ is dominated by $(\mathbf{I}_t^2, \mathbf{I}_{t+1}^2, z_t^2, e_t^2)$ if

$$\sum_{i=1}^{L} |I_{t,i}^1 - I_{t,i}^2| \leq \Delta \quad \text{and} \quad \sum_{i=1}^{L} |I_{t+1,i}^1 - I_{t+1,i}^2| \leq \Delta \quad \text{and} \quad z_t^1 \geq z_t^2 \quad \text{and} \quad e_t^1 \geq e_t^2, \quad (19)$$

where Δ is a parameter specifying the maximum distance between two inventory levels.

5.2 Synthesize by dynamic programming

With all the potential global optimal single-period cost-emission pairs, we next design a procedure to find overall Pareto-optimal solutions (Z_T^*, E_T^*). The idea is to keep adding the cost and emissions of the single-period problems, under the condition that the ending inventory of the previous period should be consistent with the beginning inventory of the next period, while moving forward in time. As an example, in period 1, consider non-

dominated solution $\mathbf{j} = (\mathbf{I}_1, \mathbf{I}_2, z_1, e_1)$ (note that initial inventory $\mathbf{I}_1$ is given), and we want to stitch solution $\mathbf{q} = (\mathbf{I}_2^*, \mathbf{I}_3^*, z_2^*, e_2^*)$ from period 2 with it. We must guarantee that $I_2 \approx I_2^*$. The result of this operation is stored as a triple $(\mathbf{I}_3, Z_2, E_2)$, where $\mathbf{I}_3 = I_3^*$, $Z_2 = z_1 + z_2^*$ and $E_2 = e_1 + e_2^*$. As the algorithm moves forward in time, it maintains non-dominated triples $(\mathbf{I}_{t+1}^*, Z_t^*, E_t^*)$ at the end of each period t. In such a triple, I_{t+1}^* is the inventory level at the beginning of period $t+1$, Z_t^* and E_t^* correspond to the cost and emissions from period 1 to t, respectively. Here the dominance relation between two triples $(I_{t+1}^1, Z_t^1, E_t^1)$ and $(I_{t+1}^2, Z_t^2, E_t^2)$ is defined as follows. Triple $(I_{t+1}^1, Z_t^1, E_t^1)$ is dominated by triple $(I_{t+1}^2, Z_t^2, E_t^2)$ if

$$\mathbf{I}_{t+1,i}^1 \leq \mathbf{I}_{t+1,i}^2 \text{ for each } i \quad \text{and} \quad Z_t^1 \geq Z_t^2 \quad \text{and} \quad E_t^1 \geq E_t^2. \tag{20}$$

In the last period, Pareto-optimal solutions (Z_T^*, E_T^*) can be extracted from $(\mathbf{I}_{T+1}^*, Z_T^*, E_T^*)$ by dropping the first dimension. The algorithm is presented in Algorithm 3.

1: Find set J_1 of non-dominated triples $(\mathbf{I}_2^*, Z_1^*, E_1^*)$ from Q_1
2: **for** $t = 2, ..., T$ **do**
3: **for** each $(\mathbf{I}_t^*, Z_{t-1}^*, E_{t-1}^*)$ in J_{t-1} **do**
4: **for** each $(\bar{\mathbf{I}}_t, \bar{\mathbf{I}}_{t+1}, \bar{z}_t, \bar{e}_t)$ in Q_t **do**
5: **if** $\mathbf{I}_t^* \geq \bar{\mathbf{I}}_t$ **then**
6: Compute triple $(\mathbf{I}_{t+1}, Z_t, E_t) = (\bar{\mathbf{I}}_{t+1}, \bar{z}_t, \bar{e}_t) + (0, Z_{t-1}^*, E_{t-1}^*)$
7: **end if**
8: **end for**
9: **end for**
10: Obtain set J_t of non-dominated triples $(\mathbf{I}_{t+1}^*, Z_t^*, E_t^*)$ from all computed $(\mathbf{I}_{t+1}, Z_t, E_t)$
11: **end for**
12: Obtain all non-dominated solutions (Z_T^*, E_T^*) from J_T

Algorithm 3: Phase II: synthesize by dynamic programming

At the beginning, an initial set of non-dominated triples is found in step 1. As we move forward in time, in period t, we add each single-period solution in set Q_t to each accumulated solution in set J_{t-1}, as shown in steps 3 and 4. We require the ending inventory of the accumulated solution from period 1 to period $t-1$ to be no less than the beginning inventory of the single-period solution in period t, in order to maintain feasibility,

this is shown in step 5. In step 6, the single-period solution for period t is added to the accumulated solution for periods 1 to $t-1$. At the end of each period, only non-dominated triples are maintained, as shown in step 10. In step 12, final Pareto-optimal solutions are collected from set J_T.

6 Computational Results

In this section, we present the computational results of the proposed algorithms. We have run all the computational experiments on a 900MHz 4 CPU Linux server. IP solver Gurobi version 4.61 is used wherever an IP is solved, which utilizes all 4 CPUs. We first present the results of the cost-minimization ADP algorithm.

6.1 Computational results of the cost-minimization ADP

We set the values of ADP parameters as follows. The period length is set to 4 weeks. The step size limit $\bar{w}$ is set to 0.15. A maximum of 30 iterations or a time limit of 10 hours is imposed. The scaling parameter δ in the initialization step is set to 3. No infrastructure expansions or shrinkage over the planning horizon is assumed. To guarantee the correctness of the ADP solutions, we insert the solutions back into the IP formulation by fixing all the $\mathbf{y}$ variables. In all cases the solution was feasible.

The performances of the ADP compared with IP are presented in Figures 5 to 8. These figures show the results for one-, three-, seven- and ten-year cases, respectively.

For each figure, the horizontal axis denotes time. For the one and three-year cases, we let the IP solver run for 4 hours. For seven and ten-year cases, 10 hours are allowed. The figures also show events happening at some specific time points. These events include the ADP reaching its best solution, IP reaching its best solution, ADP terminating (due to reaching the time limit or maximum iterations) and IP terminating (due to reaching the time limit). The percentages shown in squares are the optimality gaps. All the gaps are calculated as $(V - V^*_{IP})/V^*_{IP}$, where V is the current best objective value and V^*_{IP} is

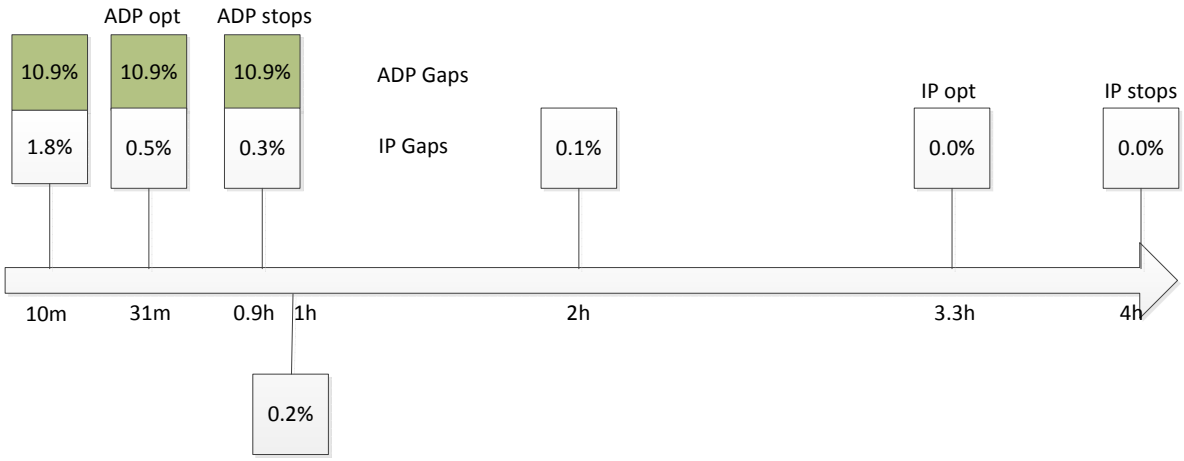


Figure 5: ADP versus IP for one-year case

the best objective value provided by the IP. For the IP part, the next to the last square showing the gap of 0% corresponds to the time when the IP found the best solution (not provably optimal - indeed none of the solutions found are provably optimal within the time limit). In Figures 7 and 8, the gray portion on the horizontal axis highlights the period when ADP outperforms IP.

The following are the main observations from these figures.

(i) At the end of the time limit, the IP always outperforms ADP. The gap is approximately 10% for small- and mid-size problems while it drops to 3.3% in the ten-year case.

(ii) In the one- and three-year cases the IP performs better at any point in time.

(iii) In the seven- and ten-year cases the ADP finds better solutions earlier in the optimization process. In the first 4 to 5 hours, the ADP yields a better solution.

This behavior is expected if we examine the problem size. ADP efficiently decomposes the problem into subproblems with much smaller size. For example, in the one year case, the IP has a size of around 30 thousand variables (including 2 thousand integer variables) and around 9 thousand constraints. For the ten year case, the IP grows to over 350 thousand variables (including around 20 thousand integer variables) and around 100

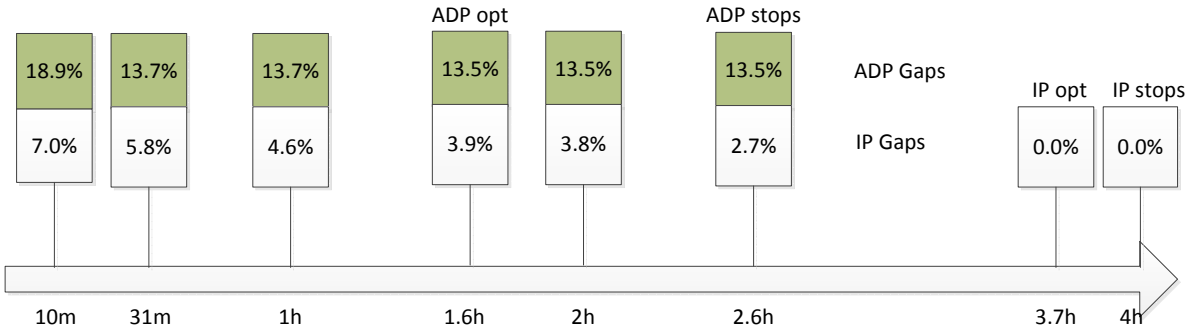


Figure 6: ADP versus IP for three-year case

thousand constraints. However, the subproblem size of the ADP remains steady, with the largest IP size around 8 thousand variables and 2 thousand constraints. To make the ADP even faster, we allow the IP solver to terminate when the IP gap is within 6% if we detect that the problem size is relatively big. With the length of the time horizon increasing linearly, the size of the full IP size increases linearly. With a linear increase in the IP size, the time needed to obtain a near-optimal solution by the IP solver tends to increase exponentially. With the ADP algorithm, however, we expect a linear increase in computing time. This is because the number of subproblem IPs increases linearly and the size of each subproblem IP remains almost the same.

From these figures we draw the following recommendations.

(1) For problems with a small number of time periods, IP is the methodology to use regardless of the available running time.

(2) For bigger problems with regard to the number of time periods,

- if the running time is limited, ADP should be used;

- otherwise, if substantial time is available, the IP finds better solutions.

To further back up the observations, in Figure 9 we show the cost value versus the number of ADP iterations for the three-year case. This figure is representative since the best objective value is usually found between iterations 10 and 20.

We next provide interesting facts about solutions. Figures 10 and 11 show the number

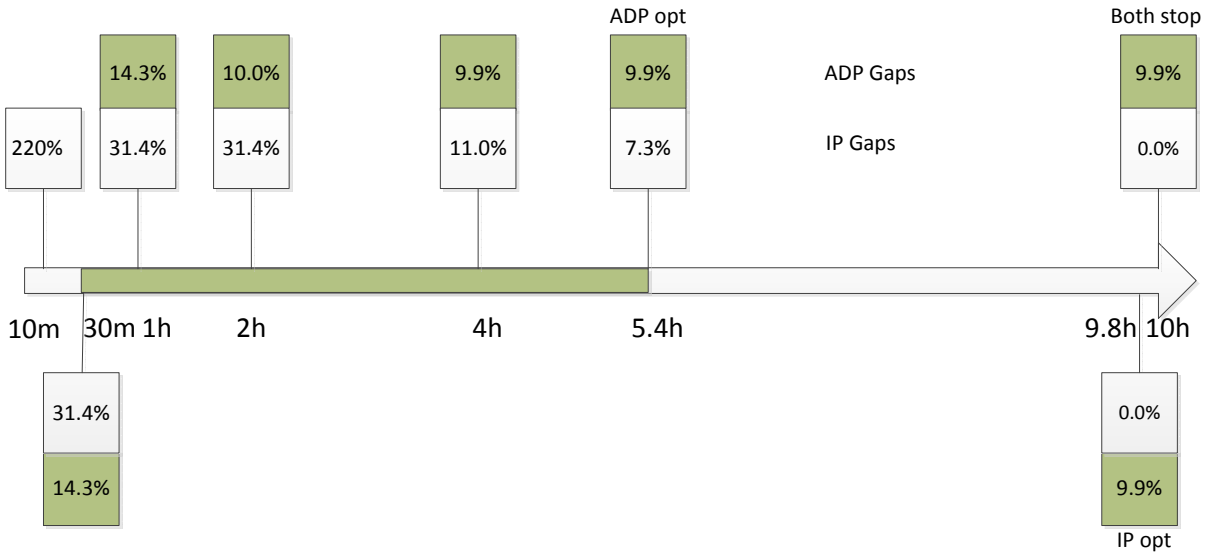


Figure 7: ADP versus IP for seven-year case

of projects carried out and the average number of persons staying per day for each period (i.e., 4 weeks) across all the locations and disciplines for a period of ten years. Figure 12 shows the total number of vehicles used for transportation across all locations for each period for the same horizon. The numbers shown are aggregated over all modes. Figure 13 shows the fuel inventory level at the end of each period at a major site, where a strict requirement on fuel inventory level is present. From these figures we can observe a seasonal trend in operations. Usually few operations are carried out in winters while summers are much more active. Figure 13 shows that after fuel inventory level reaches a peak, it usually keeps decreasing until a new cycle appears. Towards the end of the ten-year period, fuel inventory accumulated is usually below average, which is an economical inventory plan considering the finite time horizon.

6.2 Computational results of the two phase algorithm

In this section we present the results for the ADP based two phase algorithm. We compare the efficient frontiers obtained by the two phase algorithm with the efficient frontiers derived from the ε-constraint method, which is formulated by setting an upper bound on

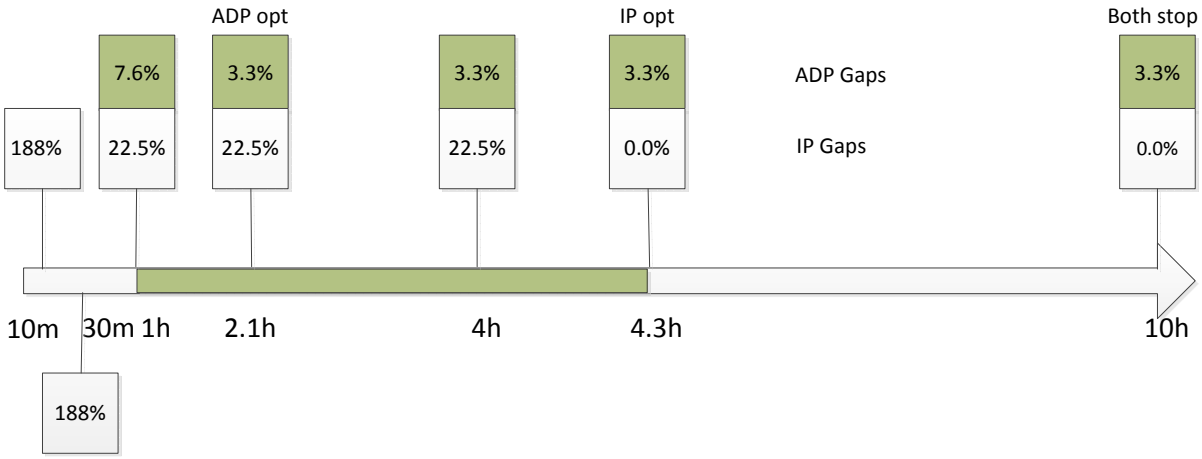


Figure 8: ADP versus IP for ten-year case

the total emission allowed, modeled as an additional constraint in the IP formulation.

When implementing the two phase algorithm, the number of inventory buckets is set to 15 and the number of emission buckets is set to 10. Since only one location has an inventory range that is wide enough to be divided into buckets, vector $\mathbf{I}_t$ has 15 possible values in step 3 of Algorithm 2. When implementing the IP based ε-constraint method, we perform warm starts, where the total emission is minimized first and the solution obtained is then passed to the IP solver as the initial solution. The reason to perform this procedure is that minimizing emission can be solved much faster than minimizing cost due to its simplified objective function. This warm start procedure greatly reduces the computational time.

We are particularly interested in the Pareto frontier where the cost does not increase more than 20%. To compare the performance of the two phase algorithm with the IP based ε-constraint method, the efficient frontiers obtained from both algorithms are shown in Figure 14(a) - 14(d). The key observations from these figures are summarized in Table 1.

As shown in the figures, at least fifteen Pareto-optimal points are computed in order to draw the efficient frontier. Note that the two phase method actually provides many more points, as seen in column "No. Points" in Table 1. As more years are involved, the gap between the two phase method and IP with ε-constraint method decreases dramatically.

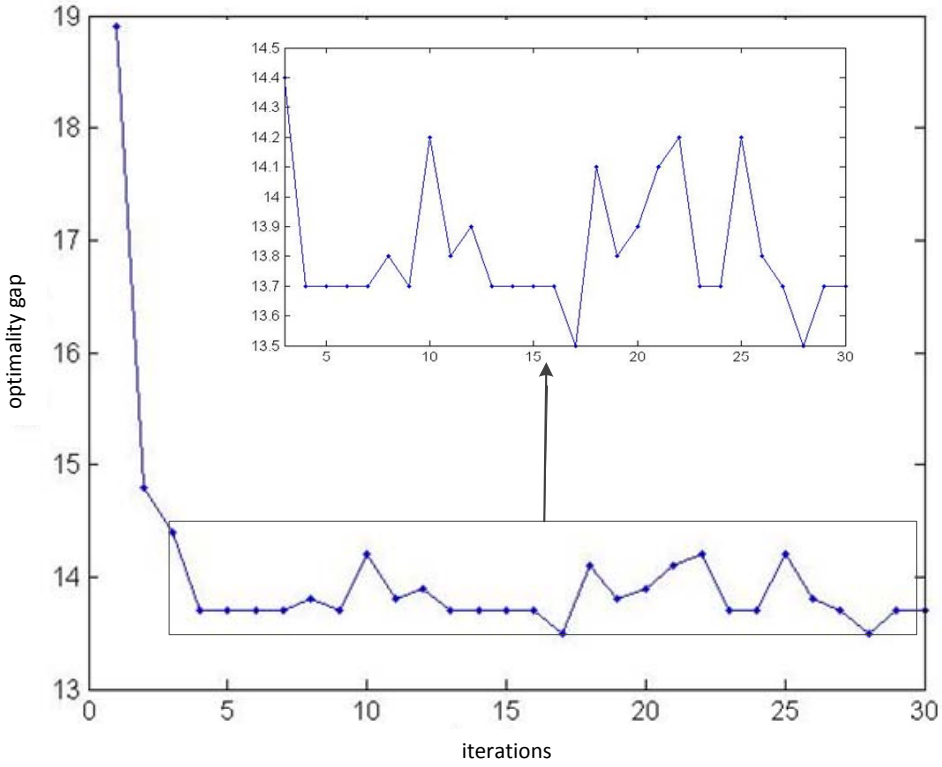


Figure 9: ADP optimality gap versus iterations for three-year case

The gap is computed as the average distance of the fifteen points on the IP curve to the curve produced by the two phase approach. This measure is used in Deb et al. (2002) to quantify distance between Pareto optimal solutions obtained from different approaches. Note that for Figure 14(d), if an IP point is dominated by a two-phase point, the sign of the gap is changed to negative. Values of these computed gaps are shown in the last column of Table 1.

One difficulty with the IP based ε-constraint method is that although the emission upper bound is varied evenly, the Pareto-optimal points may not be evenly distributed. For example, a higher emission upper bound can lead to an even lower actual emission usage under limited running time. On the contrary, the two phase approach maintains diversity in the solution space. To characterize this property, we exhibit the column "Fill Rate" in Table 1, which is computed as follows. We divide the cost region evenly into 10

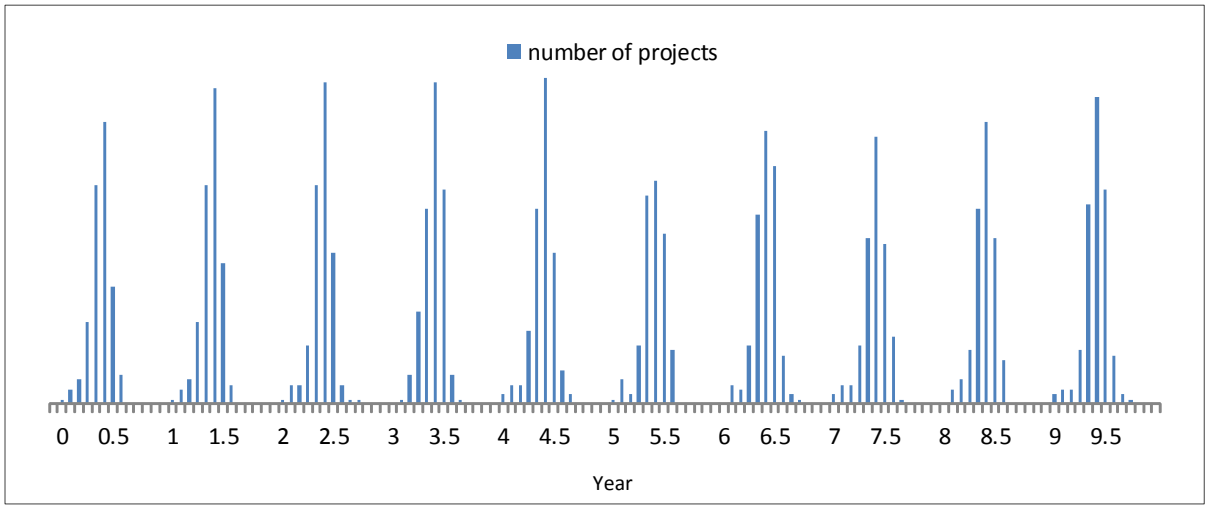


Figure 10: Total number of projects for each period (4 Weeks) over 10 years

buckets, and count the percentage of times a bucket contains at least one Pareto optimal point. The IP based ε-constraint method has a relatively low fill rate while the two phase approach fills all these buckets. We can also observe this phenomenon in Figure 14(a) - 14(d), where the two phase approach provides much smoother trade-off curves. Table 1 also shows the time needed to compute the efficient frontiers. To compute one Pareto optimal point using IP, we allow 0.5 hour of warm up time plus 4 hours for solving the emission-constrained IP for the 1-year case. For the 3, 7 and 10-year cases, these time limits are set to 1 hour plus 6 hours, 1 hour plus 6 hours, and 2 hours plus 10 hours, respectively. The total time for computing all the points on a curve is shown in Table 1. From the table, we observe that the two phase approach provides more near-optimal points in less computing time. Furthermore, we point out that it is not difficult to parallelize Phase I of the two phase approach, which would greatly reduce the running time.

7 Conclusions and Future Research

To support the operations for remote locations in Greenland, we study a logistics network design problem where both cost and emissions are considered. We formulate the problem

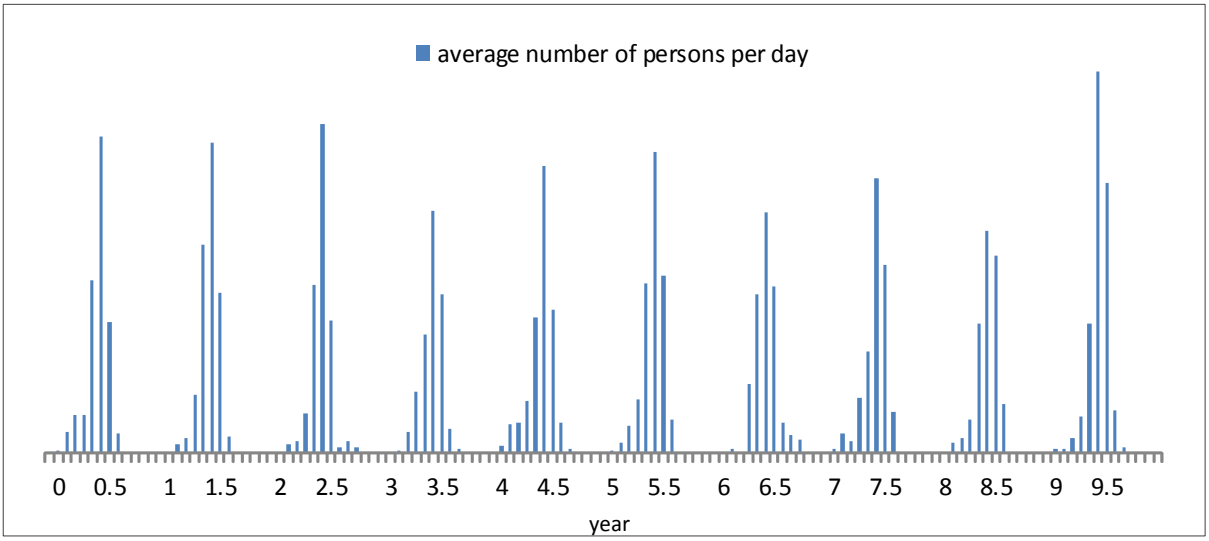


Figure 11: Average number of person per day for each period (4 Weeks) over 10 years

Table 1: ADP based two phase algorithm versus IP based ε-constraint method

	IP Based ε-Constraint			ADP based Two Phase			
	Time (hr)	No. Points	Fill Rate	Time (hr)	No. Points	Fill Rate	Gap
1-year	67	15	90%	9	32	100%	11.5%
3-year	105	15	40%	32	182	100%	14.4%
7-year	105	15	50%	90	606	100%	8.8%
10-year	180	15	20%	128	1,274	100%	0.7%

by means of a time-spaced multi-commodity network flow framework, with additional features including fuel inventory tracking, availability and delivery time windows. The cost is composed of the per vehicle charges and fuel purchase cost. To assist the decision makers to strategically size the sites under the current budget, we first study the cost minimization problem. We develop an ADP algorithm based on a linear approximation of the value function. The output of the algorithm also provides a guidance for the tactical level operations. Next we take emissions into account by modeling the problem as a bi-objective optimization problem. A novel ADP based two phase algorithm is developed, which enables the decomposition along the time dimension. Computational results show

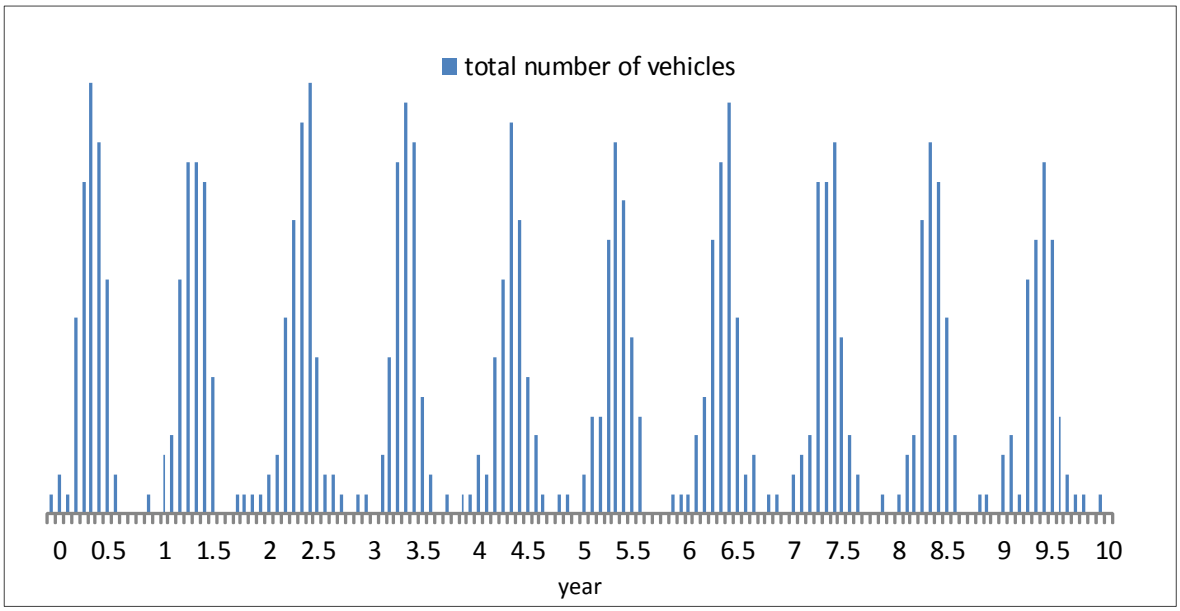


Figure 12: Total number of vehicles for each period (4 weeks) over 10 years

the efficiency of both the cost minimization ADP algorithm and the two phase algorithm. Compared with traditional methods based on solving the full IP, the two phase algorithm generates a sufficient number of well-distributed, feasible and near Pareto-optimal points. Since each IP involved in the two phase algorithm has a much smaller size, the algorithm is more robust and maintains diversity in the solution sets.

As discussed in Section 5, in the two phase algorithm, after adding the emission upper bound constraint (17), the coefficients for fuel inventories are actually skewed from those obtained from the cost minimization ADP. This effect becomes more pronounced if we consider the Pareto-optimal solution region where the cost deviates more from the minimum cost. This calls for a mechanism for updating fuel coefficients in the two phase approach. However, such an updating procedure would require additional iterations and thus would consume additional computing time. An efficient approach to conquer this issue is desirable.

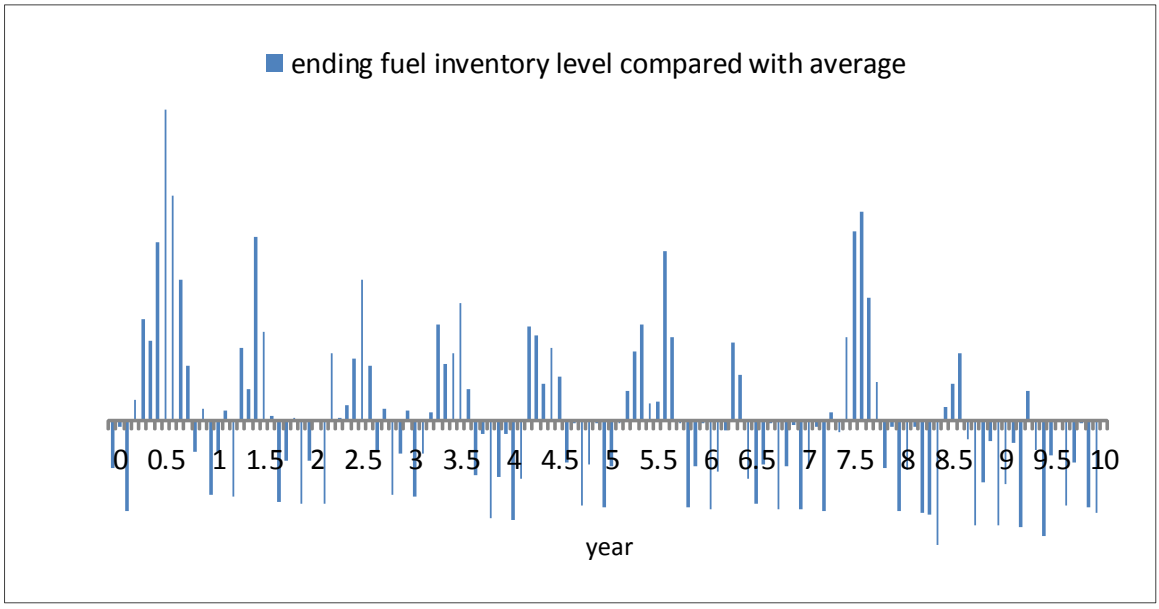


Figure 13: Fuel inventory level at the end of each period (4 weeks) for a major location over 10 years

Acknowledgment

We acknowledge NSF funding ARC-1010147 and are grateful to Patrick Haggerty and Renee Crain from NSF for their support. We thank Jason Buenning and Sandy Starkweather from CH2M Hill for providing data and support, and we are much obliged for Jason's arrangement to visit Greenland, which greatly helped our understanding of the problem. His company in Greenland was much appreciated.

A Dynamic Programming Notation

- **T**: set of periods, with index $1, ..., T$
- **L**: set of locations, with index $1, ..., L$
- **K**: commodity set, with index $1, ..., K$, and commodity K denoting fuel
- **M**: vehicle type (mode) set, with index $1, ..., M$

- $\mathbf{N}$: node set of the full graph
- $\mathbf{N}_t$: node set for period $t \in \mathbf{T}$, i.e., the t-th subgraph
- $\mathbf{S}_t$: node set representing the first week in period $t \in \mathbf{T}$ ($\mathbf{S}_t \subseteq \mathbf{N}_t$)
- $\mathbf{U}_t$: node set representing the last week in period $t \in \mathbf{T}$ ($\mathbf{U}_t \subseteq \mathbf{N}_t$)
- F_t: dummy fuel source node in period $t \in \mathbf{T}$ ($F_t \in \mathbf{N}_t$)
- $\mathbf{A}$: arc set for the full graph (each arc is associated with a unique transportation mode $m \in \mathbf{M}$)
- $\mathbf{A}_t$: arc set for period $t \in \mathbf{T}$, i.e., the t-th subgraph (each arc is associated with a unique transportation mode $m \in \mathbf{M}$)
- $\mathbf{R}$: set of pairs (k, a), where $k \in \mathbf{K}$ and $a \in \mathbf{A}$, and it denotes that commodity k is prohibited to be transported by the associated mode of arc a
- d_i^k: demand for commodity $k \in \mathbf{K}$ at node $i \in \mathbf{N}_t$ (when $i = F_t$ and $k = K$, d_i^k is unknown thus it is a variable)
- C_a: capacity of a single vehicle assigned to arc a
- c_a: cost of a single vehicle to travel on arc $a \in \mathbf{A}_t$
- g_a: emission produced by a single vehicle assigned to arc $a \in \mathbf{A}_t$
- p_i: purchase cost per unit weight fuel purchased at node $i \in \mathbf{N}$
- $\bar{I}_i$: fuel inventory upper bound at node $i \in \mathbf{N}$
- $\underline{I}_i$: fuel inventory lower bound at node $i \in \mathbf{N}$

References

2008. *GHG Protocol Tool for Mobile Combustion Version 2.0*. World Resources Institute.

Ahuja, Ravindra K., Thomas L. Magnanti, James B. Orlin. 1993. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall.

Banos, Raul, Francisco Manzano-Agugliaro, Maria G. Montoya, Consolacion Gil, Alfredo Alcayde, Julio Gómez. 2011. Optimization methods applied to renewable and sustainable energy: A review. *Renewable and Sustainable Energy Reviews* **15** 1753 – 1766.

Bouzembrak, Yamine, Hamid Allaoui, Gilles Goncalves, Hanen Bouchriha. 2011. A multiobjective green supply chain network design. *Proceedings of fourth International Conference on Logistics (LOGISTIQUA)*. Lille, France.

Chankong, Vira, Yacov Y. Haimes, David M. Gemperline. 1981. A multiobjective dynamic programming method for capacity expansion. *IEEE Transactions on Automatic Control* **26** 1195–1207.

Coello, Carlos A. Coello, Gary B. Lamont, David A. van Veldhuizen. 2007. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer.

Cohon, Jared L. 2003. *Multiobjective Programming and Planning*. Dover Publications.

Deb, Kalyanmoy, Amrit Pratap, Sameer Agarwal, T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation* **6** 182 – 197.

Haimes, Yacov Y., Herbert T. Freedman. 1975. *Multiobjective optimization in water resources systems: The surrogate worth trade-off method*. Elsevier.

Hatzakis, Iason, David Wallace. 2006. Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach. *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. Seattle, Washington, USA.

Kim, Young-Chang, Byong-Hun Ah. 1993. Multicriteria generation-expansion planning with global environmental considerations. *IEEE Transactions on Engineering Management* **40** 154–161.

Liang, Tien-Fu. 2008. Fuzzy multi-objective production/distribution planning decisions with multi-product and multi-time period in a supply chain. *Computers and Industrial Engineering* **55** 676–694.

Marler, Timothy, Jasbir Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization* **26** 369 – 395.

McClain, John O. 1974. Dynamics of exponential smoothing with trend and seasonal terms. *Management Science* **20** 1300–1304.

Melo, Maria T., Stefan Nickel, Francisco Saldanha da Gama. 2009. Facility location and supply chain management - a review. *European Journal of Operational Research* **196** 401–412.

Meza, Jose L. Ceciliano, Mehmet Bayram Yildirim, Abu S. M. Masud. 2007. A model for the multiperiod multiobjective power generation expansion problem. *IEEE Transactions on Power Systems* **22** 871–878.

Miettinen, Kaisa M. 1999. *Nonlinear multiobjective optimization*. Kluwer Academic Publishers.

Mula, Josefa, David Peidro, Manuel Díaz-Madroñero, Eduardo Vicens. 2010. Mathematical programming models for supply chain production and transport planning. *European Journal of Operational Research* **204** 377–390.

Paksoy, Turan, Eren Ozceylan, Gerhard-Wilhelm Weber. 2010. A multi objective model for

optimization of a green supply chain network. *Proceedings of the 3rd Global Conference on Power Control and Optimization*. Gold Coast, Australia.

Powell, Warren B. 2007. *Approximate Dynamic Programming - Solving the Curses of Dimensionality*. Wiley.

Ramudhin, Amar, Amin Chaabane, Mourad Kharoune, Marc Paquet. 2008. Carbon market sensitive green supply chain network design. *IEEE International Conference on Industrial Engineering and Engineering Management*. Montreal, QC, Canada.

Rodrigue, Jean-Paul, Brian Slack, Claude Comtois. 2001. Green logistics. A. M. Brewer, K. J. Button, D. A. Hensher, eds., *Handbook of Logistics and Supply-Chain Management*, chap. 21. Elsevier Ltd.

Romero, Carlos M. 1991. *Handbook of Critical Issues in Goal Programming*. Pergamon Press.

Sbihi, Abdelkader, Richard W. Eglese. 2010. Combinatorial optimization and green logistics. *Annals of Operations Research* **175** 159–175.

Silverman, Joe, Ralph E. Steuer, Alan W. Whisman. 1988. A multi-period, multiple criteria optimization system for manpower planning. *European Journal of Operational Research* **34** 160–170.

SmartWay, EPA. 2010. *SmartWay Transport Partnership Shipper/Logistics Model User Guide Version 3.4d*. U.S. Environmental Protection Agency.

Srivastava, Samir K. 2007. Green supply-chain management: A state-of-the-art literature review. *International Journal of Management Reviews* **9** 53–80.

Torabi, S. Ali, Elkafi Hassini. 2008. An interactive possibilistic programming approach for multiple objective supply chain master planning. *Fuzzy sets and systems* **159** 193–214.

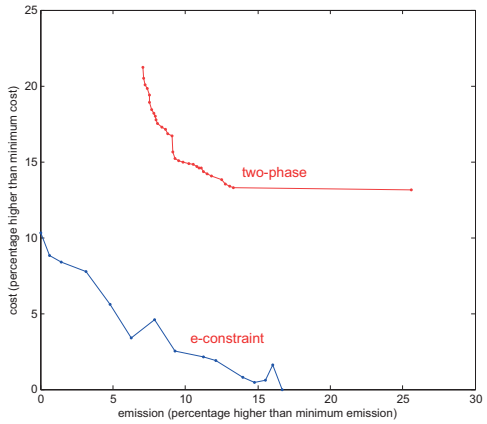
Ustun, Ozden, Ezgi Aktar Demirtas. 2008. An integrated multi-objective decision-making process for multi-period lot-sizing with supplier selection. *Omega* **36** 509–521.

Vidal, Carlos J., Marc Goetschalckx. 1997. Strategic production-distribution models: A critical review with emphasis on global supply chain models. *European Journal of Operational Research* **98** 1–18.

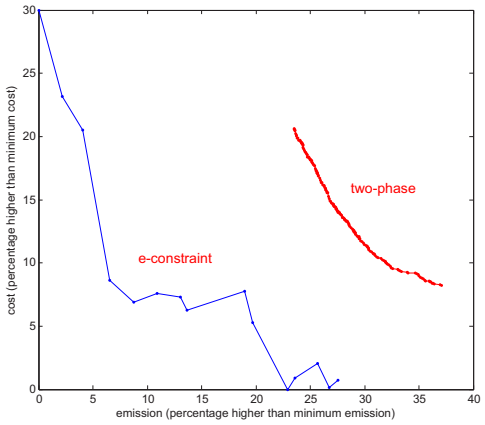
Wang, Fan, Xiaofan Lai, Ning Shi. 2011. A multi-objective optimization for green supply chain network design. *Decision Support Systems* **51** 262 – 269.

Wang, Reay-Chen, Tien-Fu Liang. 2004. Application of fuzzy multi-objective linear programming to aggregate production planning. *Computers and Industrial Engineering* **46** 17–41.

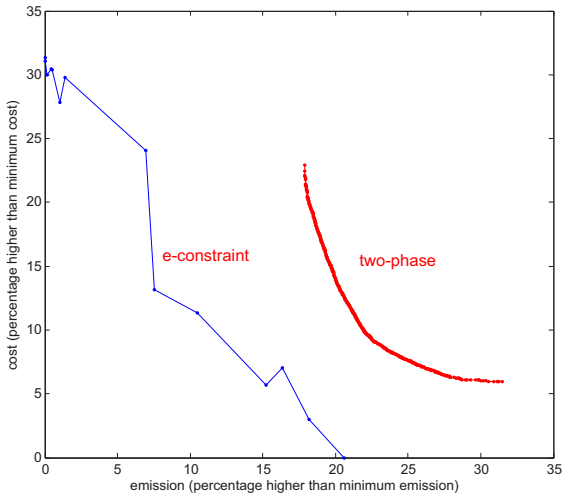
Wolsey, Laurence A. 1998. *Integer Programming*. John Wiley & Sons.



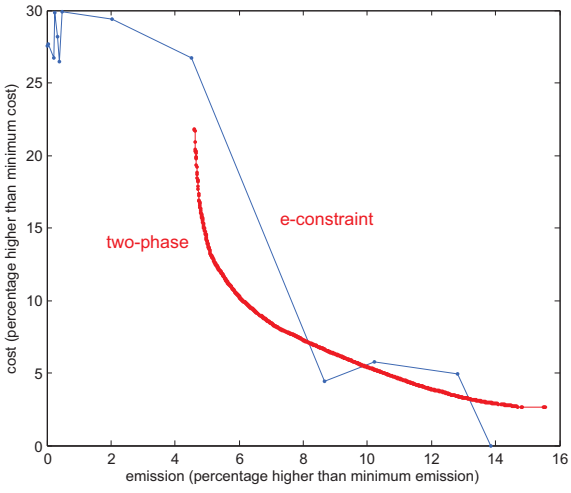
(a) Efficient frontiers for the one-year case



(b) Efficient frontiers for the three-year case



(c) Efficient frontiers for the seven-year case



(d) Efficient frontiers for the ten-year case

Figure 14: Efficient frontiers of the two phase algorithm and ε-constraint method