Power Grid Visualization by Means of Optimization

Mingyang Di

Department of Industrial Engineering and Management Sciences
Northwestern University
mingyangdi2012@u.northwestern.edu

Diego Klabjan

Department of Industrial Engineering and Management Sciences
Northwestern University
d-klabjan@northwestern.edu

Esa Rantanen

Department of Psychology
Rochester Institute of Technology
emrgsh@rit.edu

March 3, 2016

Abstract

The operations in electric power control centers play a crucial role in ensuring the integrity of the nation's electric grid and present formidable challenges to human operators. One of the primary challenges of information display for a power transmission control room application is the clutter from displaying too much information in too small of a display space. Thus, the task to optimize the layout of visual elements consisting of substations and transmission lines on the display interface is of vital importance. To this end, algorithms using several optimization techniques including Lagrangian relaxation and progressive hedging are proposed in this paper to make the interface less cluttered subject to human perceptual and cognitive capabilities. We conduct extensive computational studies to evaluate and compare the developed algorithms, and report our findings based on a real-world power grid in the U.S.

1 Introduction

Reliability of electric power transmission plays a critical role in almost every aspect of a modern society. Despite the advent of the smart grid and extensive automation associated with it, the reliability of the electric grid is still largely in the hands of human operators who manage it at control centers. Therefore, an effective way to prevent possible disruptions to the supply of electric power is to design a better display to allow human operators to process vast amount of information rapidly and reliably. A major difficulty associated with the design is the clutter from displaying too much information in too small of a display space. And it is further aggravated by the future need to broaden the views to also encompass neighboring control areas and display predicted values. Thus, optimizing the layout of visual elements consisting of substations and transmission lines to make the display interface less cluttered subject to human perceptual and cognitive capabilities is of vital significance.

To reduce clutter, geographically close substations should be spread out and transmission lines between them drawn, for example in a rectilinear pattern, to minimize line bends and intersections. Such patterns optimized with respect to given constraints can be modeled as a multicommodity flow problem with side constraints and a complex objective function over an embedded display grid. Unfortunately, due to the size of such a model, it cannot be solved to a satisfactory quality within a reasonable computational time by any commercial software. Hence, developing effective solution methodologies is the focus of this paper.

A stream of research that is closely related to ours is VLSI (very large scale integration) global routing. As one of the most challenging discrete optimization problems, plenty of research effort has been devoted into this field in past decades and among them, sequential routing, multicommodity flow-based methods and hierarchical methods are the most well-researched solution approaches. Chiang et al. (1990) solve the problem by constructing Steiner min-max trees for each net sequentially. Shragowitz and Keel (1987) were among the first researchers to work on global routing using a multicommodity flow model. The hierarchical method, first proposed by Burstein and Pelavin (1983), is a systematic divide-and-conquer approach to transform the large and complicate overall problem into a series of subproblems. This menthod has been used by Patel et al. (1985), Luk et al. (1987), Hayashi and Tsukiyama (1995) and Heisterman and Lengauer (1991). Hu and Sapatnekar (2001) provide an extensive review of various VLSI global routing models and algorithms.

This paper relies mostly on the hierarchical methods. In particular, we develop two network partitioning-based algorithms - a Lagrangian relaxation algorithm and a progressive hedging algorithm - to decompose the global network into smaller regions and then iteratively solve the subproblem in each region by readjusting penalties for corresponding solutions. They are like the traditional hierarchical methods at first glance, but we also incorporate several novel aspects.

1. We apply clustering techniques such as k-means to capture the geographic closeness and connectivity between substations.

2. Unlike VLSI global routing, the visualization decluttering problem addressed in this paper also considers setting apart closely located substations so that when the transmission lines among them

are routed, extremely cluttered areas can be avoided. Combining this location problem with VLSI greatly expands our model's fields of application, but also poses new computational challenges.

3. The proposed iterative algorithms are based on Lagrangian relaxation and progressive hedging, which has been rarely used for solving VLSI routing problems. Unlike common Lagrangian relaxation procedure that relaxes the most difficult constraints or Zhou and Wong (1999) who relax crosstalk constraints[1], we first partition the network into smaller regions so that the subproblem in each region can be easily solved, and then relax the compatibility requirements among regions. The progressive hedging algorithm works in a similar way by partitioning the network and solving each region.

We also develop a sequential routing approach to route transmission lines one by one based on a revised shortest path algorithm, and conduct extensive computational experiments to evaluate and compare the three proposed solution methodologies. Our findings show that the two network partitioning-based iterative algorithms deliver similar high quality solutions within 5 to 10 hours, while the sequential routing algorithm gives a preliminary result of much lower quality in a much shorter time. We also find out that the output from the iterative algorithms can be implemented immediately without modification, but the solution generated by the sequential approach possesses serious defects and needs further post-processing before being implemented. Thus, the comparison between the iterative and sequential methods is indeed a trade-off between computational resources and the quality of the solution.

The contribution of this paper goes beyond the three methodologies to solve a challenging display visualization problem. Most importantly, we develop a divide-and-conquer approach based on mathematical programming techniques to solve large-scale network optimization problems, and demonstrate its applicability by performing a case study on a real-world power grid in the U.S. With slight adjustments, our work can be applied to VLSI and transportation networks. We also extend the traditional shortest path algorithm to incorporate several unique features such as line bends, crossings and overlaps, which have a spectrum of applications in transportation and logistics systems design.

The rest of the paper is organized as follows. Section 2 presents the model for the decluttering algorithm. Section 3 discusses the revised shortest path algorithm and the sequential routing approach. In Section 4, we formalize the construction and partitioning of the grid network and then present the two network partitioning-based iterative algorithms. Section 5 gives the findings from computational experiments and Section 6 concludes.

2 Mathematical Formulation of the Problem

In order to minimize the clutter on a transmission control room display, a map of transmission lines is drawn in which the latitude and longitude coordinates of the substations are redefined and the transmission lines are rerouted on the lattice so that the following criteria are met.

1. The displayed transmission lines should be piece-wise linear regardless of their true paths on the terrain so that (a) the total number of turns along the lines and (b) the total number of intersections between them are minimized.

2. The overlaps between different substations and transmission lines must be minimized. In other words, all display elements must be as visible as possible to the operators regardless of the display scale.

3. The total deviation from substations' actual geographic locations to the schematic coordinates must be minimized.

[1] In electronics, the signal transmitted on one circuit or channel can create an undesired effect such as coupling in another circuit or channel. This phenomenon is called crosstalk.

4. The relative position between each pair of substations should be preserved as much as possible. That being said, given a substation and another substation south west of it, the algorithm should try to relocate them to points A and B so that B is still south west of A on the new map.

5. The total number of loops should be minimized. We define a *loop* as a transmission line whose length is more than three times longer than the L_1 distance of its substations. An example of a loop is shown in Figure 1, in which the transmission line from s to t is a loop but the one from m to n is not.

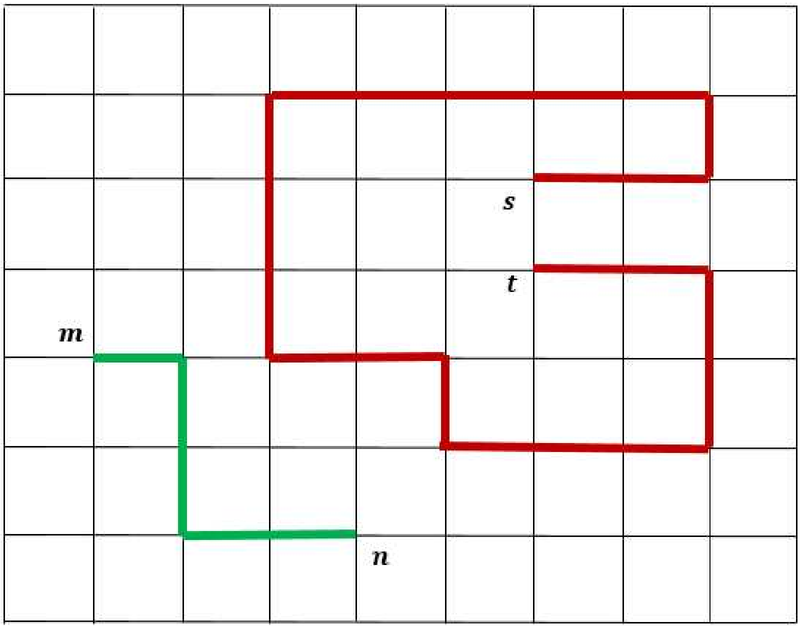

Figure 1: A graphical illustration of loops

The entire concept relies on the notion of an embedded orthogonal grid in which the edges consist of only vertical and horizontal lines. The intersections of edges are referred to as nodes, and they correspond to possible locations of substations. Although it suffices to have undirected edges for the sake of the display grid, modeling the layout requires directed arcs. The input to our model consists of a list of substations with latitude and longitude coordinates, and a list of pairs of substations that must be connected by a transmission line.

The following notation is used in the model.

Sets:

- $G = (N, A)$: the underlying network, in which N is the set of all nodes (all possible locations of substations) and A is the set of all arcs connecting the nodes.
- E: set of all underlying undirected edges. For each $e \in E$, let $e = \{a_1, a_2\}$ denote the two arcs in opposite directions defining e. We assume that each edge and arc is one of the two types. For simplicity, we call one type horizontal and the other type vertical.
- S: set of all substations.
- M: multi-set of all transmission lines. Each $m \in M$ is from $S \times S$. This is a multi-set because a pair of substations can have multiple lines. Pairs in M are treated as ordered.

- $I_1(k)$, $O_1(k)$: set of all incoming, outgoing horizontal arcs of node k.
- $I_2(k)$, $O_2(k)$: set of all incoming, outgoing vertical arcs of node k.

Decision Variables:

$$x_{s,k} = \begin{cases} 1 & \text{if substation } s \in S \text{ is placed at node } k \in N \\ 0 & \text{otherwise} \end{cases}$$

$$y_{m,a} = \begin{cases} 1 & \text{if arc } a \in A \text{ is used to display transmission line } m \in M \\ 0 & \text{otherwise} \end{cases}$$

$$u_{m,k} = \begin{cases} 1 & \text{if transmission line } m \in M \text{ has a turn at node } k \in N \\ 0 & \text{otherwise} \end{cases}$$

$$v_{k,m_1,m_2} = \begin{cases} 1 & \text{if transmission lines } m_1,\ m_2 \in M \text{ have an intersection at node } k \in N \\ 0 & \text{otherwise} \end{cases}$$

$$o_e = \begin{cases} 1 & \text{if edge } e \in E \text{ is used by more than one transmission line} \\ 0 & \text{otherwise} \end{cases}$$

In line with criteria 1 - 5, the objective function of our decluttering model reads as:

$$\begin{aligned} \min \quad & \alpha \sum_{m\in M} \sum_{k\in N} u_{m,k} + \beta \sum_{k\in N} \sum_{m_1\in M} \sum_{m_2\in M} v_{k,m_1,m_2} + \gamma \sum_{e\in E} o_e \\ & + \theta \sum_{s\in S} \sum_{k\in N} x_{s,k} d(s,k) \\ & + \mu \sum_{s_1\in S} \sum_{s_2\in S} \sum_{k_1\in N} \sum_{k_2\in N} 1_{\{x_{s_1,k_1}=x_{s_2,k_2}=1\}} V(s_1,s_2,k_1,k_2) \\ & + \nu \sum_{m\in M} 1_{\{\text{line } m \text{ is a loop}\}} \end{aligned} \tag{2.1}$$

where

- $\alpha, \beta, \gamma, \theta, \mu, \nu$ are penalty terms for turns, intersections, overlapping edges, deviations from actual geographic locations, violations of relative positions, and loops, respectively;
- $d(s,k)$ represents the distance from node k to the actual geographic location of substation s;
- $1_{\{x_{s_1,k_1}=x_{s_2,k_2}=1\}}$ is an indicator variable indicating whether substations s_1 and s_2 are located at nodes k_1 and k_2, and $V(s_1,s_2,k_1,k_2) \in \{0,1\}$ indicates whether locating substations s_1 and s_2 at nodes k_1 and k_2 violates their relative position.

Notice that the 6th component of the objective function, $\nu \sum_{m\in M} 1_{\{\text{line } m \text{ is a loop}\}}$, is linked to decision variables in a complex way. For this reason, we approximate $\sum_{m\in M} 1_{\{\text{line } m \text{ is a loop}\}}$ by $\sum_{m\in M} \sum_{a\in A} y_{m,a}\text{length}(a)$ to minimize the total length of all the transmission lines. Although this approximation does not necessarily give the minimal number of loops, these two are (strongly) positively correlated.

The following constraints are imposed.

$$\sum_{k\in N} x_{s,k} = 1 \qquad s \in S \tag{2.2}$$

$$\sum_{s\in S} x_{s,k} \leq 1 \qquad k \in N \tag{2.3}$$

$$y_{m,a_1} + y_{m,a_2} \leq 1 \qquad e = \{a_1, a_2\} \in E,\ m \in M \tag{2.4}$$

$$x_{s,k} + \sum_{a \in I_1(k) \cup I_2(k)} y_{m,a} = \sum_{a \in O_1(k) \cup O_2(k)} y_{m,a} + x_{t,k} \qquad k \in N, m = (s,t) \in M \tag{2.5}$$

$$\sum_{a \in I_1(k) \cup I_2(k)} y_{m,a} \leq 1 \qquad k \in N, m \in M \tag{2.6}$$

$$\sum_{a \in O_1(k) \cup O_2(k)} y_{m,a} \leq 1 \qquad k \in N, m \in M \tag{2.7}$$

$$\sum_{m \in M} y_{m,a_1} + \sum_{m \in M} y_{m,a_2} - 1 \leq Lo_e \qquad e = \{a_1, a_2\} \in E, \tag{2.8}$$

$$y_{m,r} + y_{m,h} - 1 \leq u_{m,k} \qquad \begin{array}{r} k \in N,\ m \in M,\ r \in I_1(k), h \in O_2(k) \\ \textit{or}\ r \in I_2(k), h \in O_1(k) \end{array} \tag{2.9}$$

$$y_{m_1,r_1} + y_{m_1,h_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 \leq v_{k,m_1,m_2} \qquad \begin{array}{l} k \in N,\ m_1, m_2 \in M,\ r_1 \in I_1(k), h_1 \in O_1(k), \\ r_2 \in I_2(k), h_2 \in O_2(k) \text{ or} \\ r_1 \in I_2(k), h_1 \in O_2(k), r_2 \in I_1(k), h_2 \in O_1(k) \end{array} \tag{2.10}$$

$$x_{s,k},\ y_{m,a},\ u_{m,k},\ v_{k,m_1,m_2},\ o_e\ \in \{0,1\} \tag{2.11}$$

Constraints (2.2) assign each substation to a node. Constraints (2.3) and (2.4) impose that we can assign at most one substation to a node and no two-arc cycles are allowed in the network. Constraints (2.5) - (2.7) model the transmission lines in the network using the following common principles.

1. There is a single arc emanating from the origin of a transmission line;

2. There is a single arc going into the destination node of a transmission line;

3. For all other nodes, the *flow-in* must equal to the *flow-out*.

Constraints (2.5) become standard flow conservation constraints if $x_{s,k} = x_{t,k} = 0$ for a given $k \in N$. If either is 1, (2.5) - (2.7) combined imply the corresponding node is either the origin or destination. Finally, constraints (2.8) - (2.10) link the decision variables capturing the objectives, where $L = 2|M|$ (or any larger number). The model is NP-hard.

3 A Sequential Routing Algorithm

The oldest and perhaps the most straightforward approach to route multiple lines on a grid is to pick an order and then route them sequentially. In this section, we develop a sequential routing algorithm based on a revised version of the Dijkstra's shortest path algorithm.

3.1 Locating the Substations

As we can see from (2.1) - (2.11), we create far more arc-related variables (i.e., y's and hence the associated u's, v's and o's) than node-related variables (i.e., x's). This implies that a hierarchical solution approach that first locates each substation to a node by solving a partial optimization model with only node-related variables and then routes the transmission lines is likely to reduce the computational load significantly. The partial model is formulated as follows.

$$\min \quad \sum_{s\in S}\sum_{k\in N} x_{s,k} d(s,k) \tag{3.1}$$

subject to

$$\sum_{k\in N} x_{s,k} = 1 \qquad s \in S \tag{3.2}$$

$$\sum_{s\in S} x_{s,k} \leq 1 \qquad k \in N \tag{3.3}$$

$$x_{s_1,k_1} + x_{s_2,k_2} \leq 1 \qquad \begin{array}{l} if\ V\big(s_1,k_1,s_2,k_2\big) = 1,\ s_1, s_2 \in S, \\ k_1 \in neighbor(s_1), \\ k_2 \in neighbor(s_2) \end{array} \tag{3.4}$$

$$x_{s,k} \in \{0,1\} \tag{3.5}$$

We must also ensure that the substations do not deviate too much from their actual geographic coordinates. This can be easily done by setting $x_{s,k} = 0$ for all nodes k with distance $d(s,k) \geq l$, where l is a threshold.

Notice that instead of minimizing pairwise geographical violations in the objective function, a new set of constraints (3.4) are introduced so that for each pair of substations s_1 and s_2, any possible violation caused by relocating s_1 and s_2 within their neighborhoods is forbidden. The neighborhood of substation s, $neighbor(s)$, is defined as the set of grid nodes within a certain distance l' from s. Following this definition, contraints (3.4) prevent most of the violations because according to (3.1), substations would most likely be located within their corresponding neighborhoods. We make this modification to avoid millions of calculations in the objective function to facilitate the solution process, but since it is weaker in terms of forbidding pairwise geographical violations, we only use it in our sequential routing heuristic.

Since the partial model is much simpler than the overall problem, we can afford a relatively fine grid to put substations close to their original coordinates and avoid many pairwise geographical violations. A straightforward choice would be an $|S| \times |S|$ grid by drawing a pair of latitude and longitude lines at the coordinates of each substation. We can certainly create a finer grid, even an arbitrarily fine one to completely rule out intersections and overlaps, but meanwhile the routing process will also become computationally challenging. To maintain the balance, we stick with the $|S| \times |S|$ grid.

3.2 A Revised Dijkstra's Shortest Path Algorithm

Once locating each substation to a node on the grid, the next step is to route the transmission lines to capture the remaining components in the objective function: minimization of turns, intersections, overlapping edges, and the total length of all the transmission lines. To do so, we first introduce the following revised Dijkstra's shortest path algorithm, which performs exactly the same as the general Dijkstra's algorithm, except that each time when we consider all the unvisited neighbors of a visited node and update the tentative distances, we need to trace back to the second-to-last node along the current shortest path to check for turns and intersections. This revised Dijkstra's algorithm exhibited in Algorithm 1 is used in several steps of the overall solution methodologies.

Algorithm 1 Revised Dijkstra's shortest path algorithm

```
1:  Initialize dist[$s$] := 0, prev[s] := $\emptyset$
2:  for each $v \in V - \{s\}$ do
3:      dist[$v$] := $\infty$
4:      prev[s] := $\emptyset$
5:  end for
6:  Set $S := \emptyset$, $Q := V$
7:  while $Q \neq \emptyset$ do
8:      Select $u \in Q$ with the minimum dist[$u$]
9:      Set $S := S \cup \{u\}$
10:     Set $z$ := prev[u]
11:     for each neighbor $v$ of $u$ do
12:         if arcs $(u, v)$ and $(z, u)$ are perpendicular then
13:             if dist[$v$] > dist[$u$] + $w(u, v) + \alpha$ then
14:                 dist[$v$] := dist[$u$] + $w(u, v) + \alpha$
15:                 prev[$v$] := $u$
16:             end if
17:         else
18:             if there exists another already processed line going through $u$ and perpendicular to $(z, v)$ then
19:                 if dist[$v$] > dist[$u$] + $w(u, v) + \beta$ then
20:                     dist[$v$] := dist[$u$] + $w(u, v) + \beta$
21:                     prev[$v$] := $u$
22:                 end if
23:             else
24:                 if dis[$v$] > dist[$u$] + $w(u, v)$ then
25:                     dist[$v$] := dist[$u$] + $w(u, v)$
26:                     prev[$v$] := $u$
27:                 end if
28:             end if
29:         end if
30:     end for
31:     Remove $u$ from $Q$
32: end while
33: return  dist, prev
```

In this algorithm, s is the source node, V is the set of all nodes, $w(u, v)$ denotes the weight of arc (u, v) and prev[u] represents the previous node of u along the current best path from source to u. Turns and intersections are penalized by α and β penalties and we can proportionately penalize the total length of transmission lines by assigning proper weight (e.g., ν) to arcs. Since we only add two steps - checking the occurrence of turns in Steps 11 - 15 and intersections in Steps 17 - 21 and adding penalties accordingly - in each iteration of the original Dijkstra's algorithm, the complexity remains the same.

We should also be aware that this revised Dijkstra's algorithm does not guarantee an optimal solution. An example in which the algorithm fails to find an optimal solution is given in Figure 2. To find the revised shortest path from A to D with the minimum length and penalty, Algorithm 1 follows the following sequence: (1) Starting from node A, we update dist[B] = 2 and dist[B'] = 2. (2) Breaking the tie by randomly choosing node B, we update dist[C] = 5, dist[G] = 6 and dist[E] = 26. (3) At node B', we update dist[F] = 7 but do not update dist[C] as we cannot get a better path traversing from B' to C. So the best path from source node to C at this moment is still $A \rightarrow B \rightarrow C$. (4) Finally, at node C, we update dist[D] = 10 and dist[H] = 10. Since dist[D] cannot be further updated from E or other neighborhood nodes, we get our solution as $A \rightarrow B \rightarrow C \rightarrow D$ with total length and penalty equal to 10. However, a better path $A \rightarrow B' \rightarrow C \rightarrow D$ with length and penalty equal to 9 could be obtained if at node C, we can trace back to node B' rather than B and hence avoid the turn penalty incurred along $B \to C \to D$. Unfortunately, Algorithm 1 cannot achieve this because node B' is not on the best path to C when updating dist$[C]$.

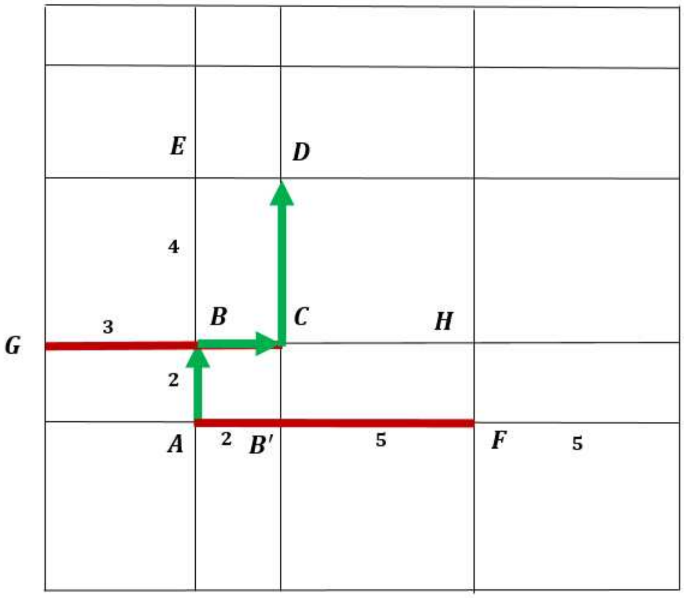

Figure 2: A counter-example of the revised Dijkstra's algorithm. The penalty terms are assigned as $\alpha = 1$, $\beta = 20$ and weights of edges/arcs are as labeled. The bold lines CG and AF refer to the fact that prior lines have already been routed on these edges. The bold line with arrow $A \to B \to C \to D$ denotes the solution obtained from Algorithm 1.

3.3 The Sequential Routing Algorithm

Before applying Algorithm 1 to the relocated substations, we need to assign weights to the arcs. For each arc $a \in A$, we assign $w(a) := \nu \times length(a)$, which is proportional to its length and effectively imposes a penalty of ν. If the underlying undirected edge has already been used by other transmission lines, $w(a)$ is updated to $w(a) := w(a) + \gamma$ to capture the penalty cost for overlaps. We formalize the overall sequential routing algorithm as follows.

Algorithm 2 The sequential routing algorithm

```
Construct the $|S| \times |S|$ grid
Locate each substation to a grid node by solving (3.1) - (3.5)
Sort to-be-connected transmission lines into descending order with respect to the $L_1$ distance between terminal substations; let $M_s$ denote the set of sorted lines
for $k \in M_s$ do
  Call Algorithm 1 to generate path $P_k$
  for each arc $a \in P_k$ do
    if the underlying undirected edge $e$ has not been used by other lines than $P_k$ then
      Set $w(a) := w(a) + \gamma$
    end if
  end for
end for
```

The primary drawback of this sequential approach is that the quality of the solution highly depends on the order in which the transmission lines are processed, and there is no systematic way of finding a good order. Furthermore, routing of transmission lines and relocating of substations are determined separately with respect to partial objectives, and by doing so, the correlation between these two families of objectives are neglected. Later in Sections 4.2 and 4.3, we develop solution approaches to consider them together.

Despite these drawbacks, since we do not solve the overall problem directly, the running time should be low. This method also allows the cluttering information (e.g., turns, intersections and overlaps) for previously routed lines to be explicitly considered when routing a new line, thus providing a quality feasible solution to the problem formulated in (2.1) - (2.11).

4 Two Network Partitioning-Based Optimization Algorithms

We start this section by constructing an appropriately-sized grid and introducing a partitioning algorithm with which the constructed grid network can be decomposed into smaller solvable regions. We then present two mathematical programming-based algorithms - one Lagrangian relaxation algorithm and one progressive hedging algorithm - to iteratively solve the problem in each region. Although the two algorithms are based on the same (partitioned) network, they rely on completely different assumptions to handle the boundaries.

4.1 Network Construction and Partitioning

Unlike in the sequential routing approach, since all the objectives and transmission lines are considered concurrently in this section, an $|S| \times |S|$ grid would yield intractable subproblems despite the partitioning and is thus unacceptable. At the same time, however, we still need to generate enough grid nodes and arcs to place substations and route transmission lines. Therefore, the algorithm to be exhibited next is developed to find the areas with high substation densities and to create more grid lines within these areas. We start with a sparse grid and the algorithm adjusts the discretization granularity along the way. The next step of the algorithm is to group grid nodes into regions so that:

- Geographically close substations are more likely to be located in the same region;
- Substations with a transmission line connecting between them are more likely to be located in the same region, or equivalently, the algorithm should discourage inter-region transmission lines;
- The number of inter-region arcs should be as small as possible.

We propose the following construction and partitioning approach, with n, n_1, n_2 and n_3 being parameters to control the granularity of the grid discretization.

1. **Group substations into regions and generate the intial grid network.**
   A simple distance function using L_2 distance between substations and *k-means* clustering algorithm are applied to divide the substations into n_1 regions. Within each region, we generate latitude lines according to the latitudes of the north- and south-most substations. And similarly, we generate longitude lines according to the longitudes of the east- and west-most substations. This gives us the initial network grid with $4n_1$ lines (if none of them overlap).

2. **Find areas with the highest substation densities and create a finer granularity grid within these areas.**
   Given the initial network grid, the number of substations in each grid rectangle excluding substations falling on grid edges is calculated. Extra latitude and longitude lines are created in the n_2 grid rectangles with the highest number of substations to divide each of them into 4 equal-sized rectangles. We repeatedly perform this until no grid rectangle contains more than n_3 substations, and convert the final grid network into a directed graph.

3. **Define a distance function for grid nodes.**
   In order to incorporate the connectivity of substations into the partitioning algorithm, we need the following definition of periphery.

   Definition 4.1. *For any substation $s \in S$, the periphery of s is defined by the following three rules:*

   a. *If substation s falls on the intersection of grid arcs, the periphery of s is defined as the eight nodes surrounding s plus the node on which s falls to.*

   b. *If substation s falls on a single grid arc, the periphery of s is defined as the six nodes surrounding s.*

   c. *If substation s does not fall on any grid arc, the periphery of s is defined as the four nodes surrounding s.*

   A graphical illustration of these three rules is given in Figure 3. Before performing clustering on grid nodes, we define a new distance function to capture both the closeness and the connectivity of substations: given any two nodes i and j, the distance between i and j is defined as half of the L_2 distance between them if there exist substations s and t connected by a transmission line such that i is in the periphery of s and j is in the periphery of of t; otherwise, the distance between i and j is simply the L_2 distance between them. Halving L_2 distance effectively brings potential locations of connected substations closer.

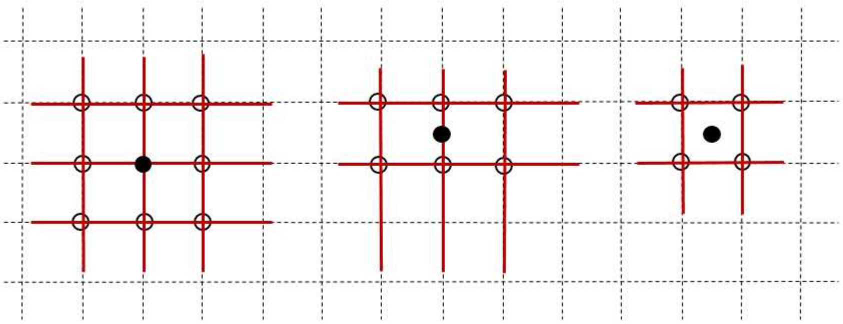

Figure 3: A graphical illustration of periphery: the bold dot represents the geographic location of a substation and the circles represent its periphery nodes

4). **Group grid nodes into regions.**
   Given the distances between grid nodes, *k-means* clustering algorithm is applied to divide the nodes into n regions.

5). **Reduce the number of inter-region arcs.**
   A tabu search algorithm attempting to assign boundary nodes to an adjacent region to reduce the number of inter-region arcs is developed. To learn more about tabu search and its applications, we refer readers to Glover (1989) and Glover (1990).

The above approach produces contiguous regions and an unevenly-spaced grid network.

4.2 The Lagrangian Relaxation Algorithm

Lagrangian relaxation is a common approach for solving difficult problems. The basic principle is that the relaxed problem should be solved very easily for fixed values of Lagrange multipliers. Since applying the algorithm developed in Section 4.1 partitions the network and the subproblem corresponding to each region is of much smaller size, we relax constraints pertaining to the boundaries and iteratively solve the relaxed model in each region. We start by introducing the following region specific notation and adjusting our model to the partitioned grid.

- Let S_i be the set of substations located in region i, $i = 1, ..., n$, $S = S_1 \cup \cdots \cup S_n$.
- Let N_i be the set of interior nodes in region i and N'_i be the set of boundary nodes in region i. The interior nodes are connected only by interior arcs, and the boundary nodes connect to at least one inter-region arc.
- Let A_i be the set of interior arcs in region i and $A_{i,j}$ be the set of inter-region arcs from region i to j, where $j \in adj(i)$ and $adj(i)$ is the set of regions sharing at least one inter-region arc with region i. Let $\bar{A}_i = A_i \cup \bigcup\limits_{j \in adj(i)} A_{i,j} \cup \bigcup\limits_{j \in adj(i)} A_{j,i}$ denote all the arcs related to region i.
- Let E_i be the set of undirected interior edges in region i and let $\bar{E}_i$ denote all the edges related to region i, $\bar{E}_i = E_i \cup \{$the set of inter-region edges incident to nodes of $N'_i\}$.
- Let M_i be the set of interior transmission lines in region i, whose connected substations are both located in i. Let $M_{i,j}$ be the set of inter-region transmission lines from region i to an adjacent region j. Similarly, let $\bar{M}_i = M_i \cup \bigcup\limits_{j \in adj(i)} M_{i,j} \cup \bigcup\limits_{j \in adj(i)} M_{j,i}$ denote all the lines related to region i.
- Let $I_1^i(k), O_1^i(k)$ be the set of incoming, outgoing horizontal arcs of node k within region i.
- Let $I_2^i(k), O_2^i(k)$ be the set of incoming, outgoing vertical arcs of node k within region i.

The decision variables remain the same, and the objective function only needs a slight modification to reflect the partitioning.

$$\begin{aligned}
\min \quad & \alpha \sum_{i}^{n} \sum_{m \in \bar{M}_i} \sum_{k \in N_i \cup N'_i} u_{m,k} + \beta \sum_{i}^{n} \sum_{k \in N_i \cup N'_i} \sum_{m_1 \in \bar{M}_i} \sum_{m_2 \in \bar{M}_i} v_{k,m_1,m_2} \\
& + \gamma \sum_{e \in E} o_e + \theta \sum_{i}^{n} \sum_{s \in S_i} \sum_{k \in N_i \cup N'_i} x_{s,k} d(s,k) \\
& + \mu \sum_{i}^{n} \sum_{s_1 \in S_i} \sum_{s_2 \in S_i} \sum_{k_1 \in N_i \cup N'_i} \sum_{k_2 \in N_i \cup N'_i} 1_{\{x_{s_1,k_2} = x_{s_2,k_2} = 1\}} V(s_1,k_1,s_2,k_2) \\
& + \mu \sum_{i}^{n} \sum_{s_1 \in S_i} \sum_{s_2 \in S \setminus S_i} \sum_{k_1 \in N_i \cup N'_i} x_{s_1,k_1} V'(s_1,k_1,s_2) \\
& + \nu \sum_{m \in M} \sum_{a \in A} y_{m,a} \text{length}(a)
\end{aligned} \tag{4.1}$$

where, $V'(s_1, k_1, s_2)$ is an indicator of whether locating substation s_1 at node k_1 violates the relative position between s_1 and s_2.

In (4.1), we see that the 5th term of the original objective function has been replaced by two components to indicate: (1) whether locating substations s_1 and s_2 at nodes k_1 and k_2 in the same region i violates their relative positions; and (2) whether locating substation s_1 at node k_1 in region i violates the relative positions between s_1 and the other substations outside i.

To facilitate the relaxation of boundary constraints, we need to model the interior network and the boundaries separately. Constraints (2.2) - (2.4) and (2.6) - (2.8) can be easily adapted within each region, and to model the flow of transmission lines, especially at boundaries, we split (2.5) into two cases.

1. Interior transmission lines whose connected substations locate in the same region $i = 1, 2, ..., n$ are modeled in the same way as (2.5).

$$x_{s,k} + \sum_{a \in I_1^i(k) \cup I_2^i(k)} y_{m,a} = \sum_{a \in O_1^i(k) \cup O_2^i(k)} y_{m,a} + x_{t,k} \qquad k \in N_i \cup N_i',\ m = (s,t) \in M_i \tag{4.2}$$

2. Let $c = (p, q) \in A_{i,j}$ be an inter-region arc from region i to j and $c' = (q, p) \in A_{j,i}$ be the corresponding reverse arc from region j to i, where $p \in N_i'$ and $q \in N_j'$. To handle an inter-region transmission line whose connected substations locate in two adjacent regions i and j, we should first route the two ends separately in i and j to some boundary nodes p and q and then connect the two segments through the inter-region arc across the boundary.

$$x_{s,k} + \sum_{a \in I_1^i(k) \cup I_2^i(k)} y_{m,a} = \sum_{a \in O_1^i(k) \cup O_2^i(k)} y_{m,a} \qquad k \in N_i,\ m = (s,t) \in \bigcup_{j \in adj(i)} M_{i,j} \tag{4.3}$$

$$x_{t,k} + \sum_{a \in O_1^i(k) \cup O_2^i(k)} y_{m,a} = \sum_{a \in I_1^i(k) \cup I_2^i(k)} y_{m,a} \qquad k \in N_i,\ m = (s,t) \in \bigcup_{j \in adj(i)} M_{j,i} \tag{4.4}$$

$$x_{s,p} + \sum_{a \in I_1^i(p) \cup I_2^i(p)} y_{m,a} = y_{m,c} + \sum_{a \in O_1^i(p) \cup O_2^i(p)} y_{m,a} \qquad m = (s,t) \in \bigcup_{j \in adj(i)} M_{i,j} \tag{4.5}$$

$$x_{t,p} + \sum_{a \in O_1^i(p) \cup O_2^i(p)} y_{m,a} = y_{m,c'} + \sum_{a \in I_1^i(p) \cup I_2^i(p)} y_{m,a} \qquad m = (s,t) \in \bigcup_{j \in adj(i)} M_{j,i} \tag{4.6}$$

$$y_{m,c} + \sum_{a \in O_1^i(p) \cup O_2^i(p)} y_{m,a} \leq 1 \qquad m = (s,t) \in \bigcup_{j \in adj(i)} M_{i,j} \tag{4.7}$$

$$y_{m,c'} + \sum_{a \in I_1^i(p) \cup I_2^i(p)} y_{m,a} \leq 1 \qquad m = (s,t) \in \bigcup_{j \in adj(i)} M_{j,i} \tag{4.8}$$

The routing is modeled by constraints (4.3) and (4.4), and the connection is accomplished through constraints (4.5) and (4.6).

Finally, we have constraints capturing turns and intersections. To model turns and intersections which do not involve inter-region arcs, the constraints remain the same as (2.9) and (2.10). On the other hand, modeling turns and intersections constructed by inter-region arcs would need a substitution of an outgoing arc (i.e., h in (2.9) and h_1 in (2.10)) by the corresponding inter-region arc.

$$y_{m,c} + y_{m,r} - 1 \leq u_{m,p} \qquad m \in \bigcup_{j \in adj(i)} M_{i,j},\ r \in I_1^i(p) \text{ or } I_2^i(p) \tag{4.9}$$

$$y_{m,c'} + y_{m,r} - 1 \leq u_{m,p} \qquad m \in \bigcup_{j \in adj(i)} M_{j,i},\ r \in O_1^i(p) \text{ or } O_2^i(p) \tag{4.10}$$

$$\begin{aligned} y_{m_1,c} + y_{m_1,r_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 \leq v_{p,m_1,m_2} \qquad & m_1 \in \bigcup_{j \in adj(i)} M_{i,j},\ m_2 \in \bar{M}_i, \\ & r_1 \in I_1^i(p),\ r_2 \in I_2^i(p),\ h_2 \in O_2^i(p) \text{ or} \\ & r_1 \in I_2^i(p),\ r_2 \in I_1^i(p),\ h_2 \in O_1^i(p) \end{aligned} \tag{4.11}$$

$$\begin{aligned} y_{m_1,c'} + y_{1m,r_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 \leq v_{p,m_1,m_2} \qquad & m_1 \in \bigcup_{j \in adj(i)} M_{j,i},\ m_2 \in \bar{M}_i, \\ & r_1 \in O_1^i(p),\ r_2 \in I_2^i(p),\ h_2 \in O_2^i(p) \text{ or} \\ & r_1 \in O_2^i(p),\ r_2 \in I_1^i(p),\ h_2 \in O_1^i(p) \end{aligned} \tag{4.12}$$

The complete formulation ready for Lagrangian relaxation is presented in Appendix A. Notice that this formulation is unable to capture the transmission lines going across more than two regions, and hence is not equivalent to the original model. We call such transmission lines as *jump lines* to indicate that they can *jump* over one or several regions. Fortunately, in our New York State case, the jump lines only account for less than 5% of all the transmission lines, so they can be taken care of by a sequential routing heuristic after most others are processed. But if this percentage is too high[2], post-processing jump lines may not be a viable option.

With this formulation, all the constraints related to the inter-region arcs are relaxed. For each inter-region edge $e = \{c, c'\}$, λ_e is the Lagrange multiplier for constraint (2.8) and $\lambda^{1(i)}_{m,c}$, $\lambda^{1(i)}_{m,c'}$, $\lambda^{2(i)}_{m,c}$, $\lambda^{2(i)}_{m,c'}$ are for constraints (4.5) - (4.8). Moreover, multipliers $\lambda^{3(i)}_{m,c}$, $\lambda^{4(i)}_{m,c}$, $\lambda^{3(i)}_{m,c'}$, $\lambda^{4(i)}_{m,c'}$, $\lambda^{5(i)}_{m,c}, \lambda^{6(i)}_{m,c}$ $\lambda^{5(i)}_{m,c'}$, $\lambda^{6(i)}_{m,c'}$ are for constraints (4.9) - (4.11) since each of them consists of two different cases.[3] An iterative algorithm is then developed to solve the Lagrangian relaxation, in which we improve the handling of inter-region transmission lines through the notion of penalties that encourage these lines to be more compatible at boundaries in each iteration. Having the grid nodes grouped into n regions, the Lagrangian relaxation algorithm is described as follows.

1. **Group substations into n regions.**
   Notice that the partitioning algorithm is performed on the grid network (i.e., nodes) rather than on substations. Therefore, in order to assign each substation into a region, we have to establish a correspondence between substations and nodes. We use (3.1) - (3.5) from Section 3.1 to initially locate each substation s to a node k. Then s can be assigned to the region which k belongs to.

2. **Solve the relaxed problem.**
   Since all the constraints pertaining to the inter-region arcs are relaxed, the problem decomposes into n smaller subproblems. They are well-defined and can be solved independently. For fixed values of Lagrange multipliers, the values at the boundaries (i.e., solution for $y_{m,c}$) are optimally determined by the following rule:

$$y_{m,c} = \begin{cases} 0 & \text{if } \lambda_e - (\lambda^{1(i)}_{m,c} + \lambda^{1(j)}_{m,c}) + \left(\sum_{k=2}^{6} \lambda^{k(i)}_{m,c} + \sum_{k=2}^{6} \lambda^{k(j)}_{m,c}\right) + \nu \text{ length}(c) > 0 \\ 1 & \text{otherwise} \end{cases}.$$

   Intuitively, we are assigning *easy-going* arcs to the boundaries because the above *if*- statement sums up all the Lagrange multipliers associated with c and hence effectively indicates how difficult it is to go across this inter-region arc for any transmission line. We then compute the gradients within each region. For example, in region i, the gradient of constraint (4.9) is computed as $y_{m,c}+y_{m,r}-1-u_{m,p}$ after the subproblem is solved.

3. **Find a primal feasible solution based on a Lagrangian solution.**
   The paths obtained from solving the relaxed subproblems are unlikely to be feasible for the original problem. However, we can readily find a feasible solution from these paths by the following heuristic.

   a. For any transmission line $m = (s, t)$ with both substations s and t in the same region, we locate s, t and route m according to the solution from the subproblem.

   b. For any transmission line $m = (s, t)$ with substations s and t in different regions, we first locate s, t according to the solution from the subproblem and then find the shortest path between s and t using Algorithm 1. The weight of interior arc is set as ν times its length, and to avoid routing the line across boundaries frequently, the weight of any inter-region arc a is assigned as $w(a) := L' \times \nu \times \text{length}(a)$. Here L' is an amplifying parameter.

[2] This is unusual because long transmission lines would cause severe loss of energy and are thus discouraged in the field.

[3] If edge e falls on the borderline of the entire grid network, each of the constraints (4.9) - (4.11) consists of only one case.

The order of processing or routing the lines is determined by $\sum_{c} \Big[\Big(\sum_{k=2}^{6} \lambda_{m,c}^{k(i)} + \sum_{k=2}^{6} \lambda_{m,c}^{k(j)} \Big) - \big(\lambda_{m,c}^{1(i)} + \lambda_{m,c}^{1(j)} \big) \Big]$, which indicates the level of *difficulty* to pass through the boundaries for each transmission line $m \in M$. c denotes any inter-region arc. To be specific, if

$$\sum_{c} \Big[\Big(\sum_{k=2}^{6} \lambda_{m_1,c}^{k(i)} + \sum_{k=2}^{6} \lambda_{m_1,c}^{k(j)} \Big) - (\lambda_{m_1,c}^{1(i)} + \lambda_{m_1,c}^{1(j)}) \Big] > \sum_{c} \Big[\Big(\sum_{k=2}^{6} \lambda_{m_2,c}^{k(i)} + \sum_{k=2}^{6} \lambda_{m_2,c}^{k(j)} \Big) - (\lambda_{m_2,c}^{1(i)} + \lambda_{m_2,c}^{1(j)}) \Big],$$

m_1 will be routed before m_2 in the above heuristic.

4. **Update the Lagrange multipliers.**
   Stepsize $t^{(n)} = a_0 \times a_1^n$ at the n^{th} iteration is used, where a_0 and a_1 are randomly selected between 0 and 2. The Lagrange multipliers are then updated by $\lambda^{(n+1)} = \lambda^{(n)} + t^{(n)}(\text{gradient})$.

The following Algorithm 2 summarizes the overall procedure.

Algorithm 3 Lagrangian relaxation algorithm

```
 1: Construct and partition the network grid into $n$ regions
 2: Assign substations into each region by solving (3.1) - (3.5)
 3: Temporarily remove all the jump lines
 4: Initialize Lagrange multipliers, $LB := -\infty$, $UB := \infty$, and $a_0$, $a_1$
 5: Set $k := 1$
 6: for $i = 1, ..., n$ do
 7:     Solve the relaxed subproblem in region $i$
 8:     Compute the gradients related to the subproblem in region $i$
 9: end for
10: Update $LB$
11: Construct a feasible solution, and update $UB$
12: Set $t^{(k)} := a_0 \times a_1^k$
13: if stopping criteria are not met (e.g., $UB - LB > \epsilon$) then
14:     Update Lagrange multipliers: $\lambda^{(k+1)} := \lambda^{(k)} + t^{(k)}$(gradient)
15:     Set $k := k + 1$
16:     Go to step 6
17: end if
18: Call Algorithm 1 to route jump lines sequentially
```

4.3 The Progressive Hedging Algorithm

Progressive hedging (PH) proposed by Rockafellar and Wets (1991) is a decomposition type method for solving multistage stochastic programming problems. The basic idea is to iteratively solve individual scenario problems, perturbed in a certain sense, and to aggregate the scenario-dependent solutions into an overall implementable solution. Under certain assumptions, the sequence of the implementable solutions converges to the solution of the stochastic program. However, in the presence of discrete variables, PH is only a heuristic (see Fan and Liu (2010) and Listes and Dekker (2005)).

In this section, we present a PH algorithm by considering the subproblem in each region as a scenario. Unlike the Lagrangian relaxation approach in which inter-region arcs do not belong to any region, the PH approach partitions the network in such a way that each region includes inter-region arcs either originating from or going into it.

The notation and decision variables are defined in the same way as Section 4.2, except that within each region the interior nodes and boundary nodes are no longer differentiated. Thus, N_i denotes all the nodes in region i. In addition, since each inter-region arc now belongs to both regions it connects to, we have two scenario-dependent solutions obtained from solving for the arc in different regions. In order to consolidate the scenario-dependent solutions into an implementable solution, we must force them to be equal.

The objective function is different from (4.1) since we have to take scenarios/regions into consideration. It reads as

$$\min \sum_{i}^{n} Q_1^i + 0.5 \sum_{i}^{n} Q_2^i \tag{4.13}$$

where

$$\begin{aligned} Q_1^i = {} & \alpha \sum_{m \in \bar{M}_i} \sum_{k \in N_i} u_{m,k} + \beta \sum_{k \in N_i} \sum_{m_1 \in \bar{M}_i} \sum_{m_2 \in \bar{M}_i} v_{k,m_1,m_2} \\ & + \gamma \sum_{e \in E_i} o_e + \theta \sum_{s \in S_i} \sum_{k \in N_i} x_{s,k} d(s,k) \\ & + \mu \sum_{s_1 \in S_i} \sum_{s_2 \in S_i} \sum_{k_1 \in N_i} \sum_{k_2 \in N_i} 1_{\{x_{s_1,k_1} = x_{s_2,k_2} = 1\}} V(s_1,k_1,s_2,k_2) \\ & + \mu \sum_{s_1 \in S_i} \sum_{s_2 \in S \setminus S_i} \sum_{k_1 \in N_i} x_{s_1,k_1} V'(s_1,k_1,s_2) \\ & + \nu \sum_{m \in \bar{M}_i} \sum_{a \in A_i} y_{m,a} \text{length(a)}, \\ Q_2^i = {} & \gamma \sum_{e \in \bar{E}_i \setminus E_i} o_e + \nu \sum_{m \in \bar{M}_i \setminus M_i} \sum_{a \in \bar{A}_i \setminus A_i} y_{m,a} \text{length(a)}. \end{aligned}$$

As we can see from (4.13), the summation is over all regions. In each region $i = 1, 2, ..., n$, Q_1^i consists of the objective components that are independent of inter-region arcs, while Q_2^i has objectives related to those arcs. Each inter-region arc connects exactly 2 regions, and thus the term 0.5 is in the objective function as a probability.

Constraints (2.2) - (2.4) and (2.6) - (2.10) are adapted for each region $i = 1, 2, ..., n$ in a straightforward way, and constraints (4.2) remain the same for modeling interior transmission lines. The following two constraints are unique to the PH approach.

$$x_{s,k} + \sum_{a \in I_1^i(k) \cup I_2^i(k)} y_{m,a} = \sum_{a \in O_1(k) \cup O_2(k)} y_{m,a} \qquad k \in N_i,\ m = (s,t) \in \bigcup_{j \in adj(i)} M_{i,j} \tag{4.14}$$

$$x_{t,k} + \sum_{a \in O_1^i(k) \cup O_2^i(k)} y_{m,a} = \sum_{a \in I_1(k) \cup I_2(k)} y_{m,a} \qquad k \in N_i,\ m = (s,t) \in \bigcup_{j \in adj(i)} M_{j,i} \tag{4.15}$$

Together, they connect the two ends of an inter-region transmission line to some inter-region arcs at the boundary. For each pair of adjacent regions i and j, we also need to impose the following constraints to make the scenario-dependent solutions implementable.

$$y_{m,a}^i = y_{m,a}^j \qquad j \in adj(i),\ a \in A_{i,j},\ m \in \bigcup_j M_{i,j}, \tag{4.16}$$

where $y_{m,a}^k$ is the value of $y_{m,a}$ from solving scenario k, $k = i, j$. Similar to Section 4.2, this new formulation cannot take jump lines into explicit consideration either.

Like the Lagrangian relaxation approach, we first partition the network into n regions and assign each substation into a region by solving (3.1) - (3.5). Next, due to the model's intrinsic inability of handling jump lines, these lines are temporarily removed before the start of the iterative algorithm and then processed after most other lines have been routed. We now formalize our PH algorithm as follows, taking ρ (penalty factor) as an input parameter.

Algorithm 4 Progressive hedging algorithm

```
1: Set $k := 0$
2: Initialize $\bar{y}_{m,a}^{(0)}$ and $(w_{m,a}^i)^{(0)}$, $i = 1, 2, ..., n$ for each inter-region arc $a$ and each inter-region line $m$
3: Set $k := k + 1$
4: for $i = 1, 2, ..., n$ do
5:     Solve subproblem $i$ and obtain
           $(x^i, y^i, o^i, u^i, v^i)^{(k)} := \arg min \Bigg( Q_1^i + Q_2^i + \sum_{m \in \bar{M}_i \setminus M_i} \sum_{a \in \bar{A}_i \setminus A_i} (w_{m,a}^i)^{(k-1)} y_{m,a} + \frac{\rho}{2} \sum_{m \in \bar{M}_i \setminus M_i} \sum_{a \in \bar{A}_i \setminus A_i} \|y_{m,a} - \bar{y}_{m,a}^{(k-1)}\|^2 \Bigg)$   (4.17)
6: end for
7: for any pair of adjacent regions $i$ and $j$ do
8:     for each inter-region arc $a$ and each inter-region line $m$ between $i$ and $j$ do
9:             $\bar{y}_{m,a}^{(k)} = 0.5(y_{m,a}^i)^{(k)} + 0.5(y_{m,a}^j)^{(k)},$
               $(w_{m,a}^i)^{(k)} = (w_{m,a}^i)^{(k-1)} + \rho[(y_{m,a}^i)^{(k)} - \bar{y}_{m,a}^{(k)}]$
10:    end for
11: end for
12: if stopping criteria are not met then
13:     Go to step 3
14: end if
15: Construct a feasible solution based on the current solution
16: Call Algorithm 1 to route jump lines sequentially
```

In the above algorithm, ρ is a pre-selected constant which is a common practice. However, a thorough observation indicates that an effective ρ value should be close in magnitude to $\nu \times length(a)$, which is the unit cost of $y_{m,a}$. Otherwise, $w_{m,a}^i$ would yield a small fraction of (4.17) and the per-iteration change in the penalty terms $(w_{m,a}^i)^{(k-1)} y_{m,a}$ would also be small. Small changes in the penalty terms would yield little improvements in $y_{m,a}$ which in turn trigger slow PH convergence. Therefore in this paper, we use an arc-specific value $\rho_{m,a} := \rho_0 \times (\nu \times length(a))$.

5 Computational Results

5.1 Implementation

All computational experiments in this section are performed on a server with 8GB RAM amd 2.8GHz 2220 SE Dual Core AMD Opteron Processor. The subproblems are solved by Gurobi optimization software, and the data we use to test the algorithms is derived from the New York ISO Electrical System Map, which represents the network of existing and proposed substations and transmission lines. In total, there are 692 substations and 809 transmission lines.

Through extensive computational experiments, we set $n = 180, n_1 = 35, n_2 = 10$ and $n_3 = 10$ to create a 72×72 grid network and then partition it into 180 regions. Under this setting, the major computational difficulty is caused by the large number of intersection-related variables. In Section 2, a binary variable v_{k,m_1,m_2} is defined to capture whether two transmission lines m_1 and m_2 intersect at a particular node k. Due to these variables, even the much smaller size subproblems sometimes become too big to be handled in a reasonable computational time. Therefore, a hierarchical approach is employed to *first* minimize the number of turns and *then* intersections so that each subproblem can be approximately solved in seconds. To be specific, the minimization of intersections is excluded from the objective function when we solve the relaxed subproblem in each region, and is addressed later when a feasible solution is constructed.

To reduce memory needs, we use a common Lagrange multiplier for a set of similar constraints. Referring back to Section 4.2, $\lambda^{3(i)}_{m,c}$, $\lambda^{4(i)}_{m,c}$ and $\lambda^{3(j)}_{m,c}$, $\lambda^{4(j)}_{m,c}$ are replaced by $\eta^{1}_{m,c}$ because the corresponding constraints capture turns related to the same inter-region arc. Similarly, $\lambda^{5(i)}_{m,c}$, $\lambda^{6(i)}_{m,c}$, $\lambda^{5(j)}_{m,c}$ and $\lambda^{6(j)}_{m,c}$ are replaced by $\eta^{2}_{m,c}$. The new gradient is computed by taking the average over each component's gradient.

Since this is a large-scale problem, we focus exclusively on obtaining good quality solutions, but not on the optimality gap or other possible measures. For this reason, we stop our solution processes after 10 iterations of execution as running additional iterations does not lead to obvious further convergence. This usually results in a running time of less than 8 hours, which is considered well acceptable according to operators at NYISO control centers. Furthermore, instead of seeking an optimal solution from each subproblem, we stop solving it when the optimality gap is less than 30% or the execution time reaches 120 seconds.

5.2 Results

Given the above parameter estimates and implementation assumptions, Figure 4 shows an output display from the Lagrangian relxation algorithm with penalties $\alpha = \beta = \gamma = \theta = \mu = 1$ and $\nu = 1000$. It is directly drawn on a New York State map in our web-based GUI, and the geographic features on the terrain is ignored. The transmission lines are also drawn with different widths and colors to improve visibility and expedite operator's response. To validate our work, we conduct a subjective evaluation by asking 5 operators at NYISO control centers to compare our output with the current interface they are using. Three of them agreed that the reduction of clutter is apparent and the transmission lines become more traceable by human eyes with our layout. The other two voted neutral, saying that they thought the new layout is as good as the existing one. All of them liked the widely spread out substations.

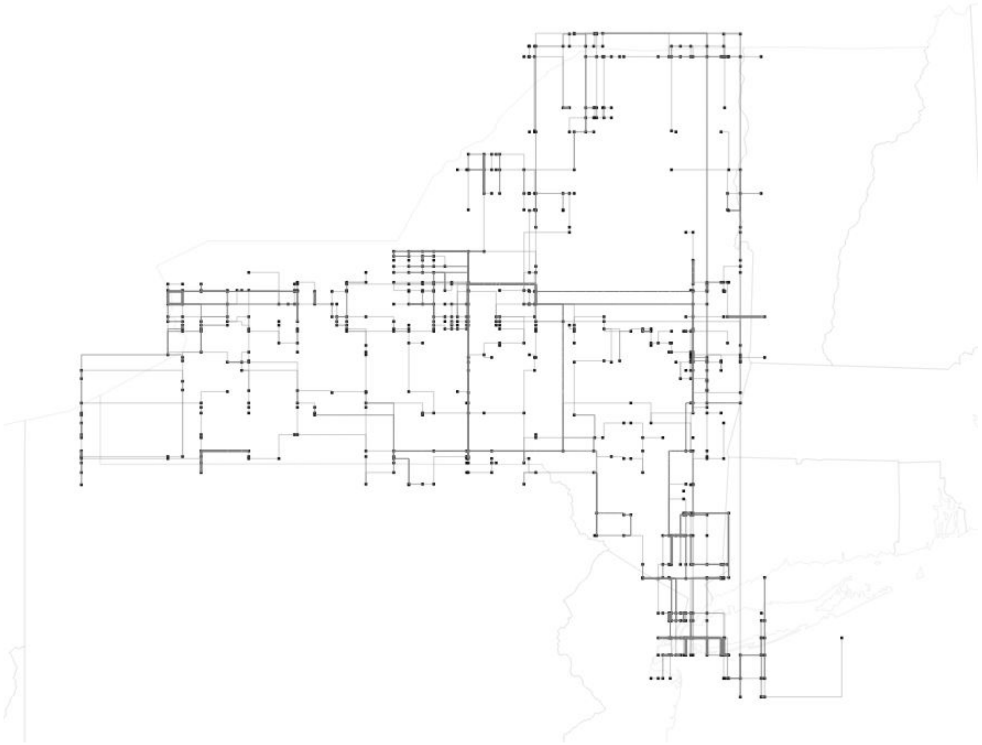

Figure 4: An output display from Algorithm 2

We perform computational experiments with various combinations of penalties. In the rest of this section, SR, LR and PH stand for the sequential routing, Lagrangian relaxation and progressive hedging algorithms respectively. We start by varying ν which is the penalty for loops. From the results given in Table 1, we can see that SR grants a significant reduction in running time compared with the iterative algorithms. It also requires less memory by avoiding a vast number of variables created in the process of solving the integer program. Therefore, SR is recommended when time and computational power are highly limited.

A major drawback of this sequential approach is the significant number of loops it generates - up to 25% of all the transmission lines. These loops would bring in unnecessary turns and congestions and thus make the transmission lines more difficult to trace on the display interface. In reality, such solutions are considered of bad quality and post-processing algorithms to eliminate these loops should be developed. Also, the quality of the solution depends on the order in which the transmission lines are routed, and unfortunately, it is hard to find a good ordering. According to Abel (1972), there is no single ordering technique that consistently performs better than any other ordering method.

The numbers with asterisks in Table 1, which correspond to the best (smallest) objective values among the three algorithms, suggest that as the penalty for loops gradually increases, the iterative algorithms start to outperform SR.[4] When $\nu \leq 100$, SR gives the smallest objective value and is thus superior to the iterative ones. As ν keeps increasing over 200, the iterative algorithms dominate. The case when $\nu = 140$ is the trickiest since PH outperforms SR while LR does not. A thorough observation reveals that the objective values output by the three algorithms are indeed close, and thus $\nu = 140$ can be treated as a cutoff point of choosing the iterative algorithms over the sequential routing approach in our NYISO case. We expect such pattern to hold in general, but the exact values are likely to change. Therefore, if a practically implementable solution with less loops is desired, we recommend the iterative algorithms.

Table 1: Comparison of the three solution algorithms

	Performance Measure								
	SR			**LR**			**PH**		
ν	**time (hr)**	**loop**	**obj**	**time (hr)**	**loop**	**obj**	**time (hr)**	**loop**	**obj**
1	0.45	201	8,650*	5.27	8	20,319	4.16	8	17,410
10	0.44	200	10,467*	5.24	3	23,794	4.20	6	21,103
100	0.41	200	28,527*	6.73	1	37,820	4.90	6	33,177
140	0.41	200	36,520	7.15	0	37,158	5.21	10	35,963*
200	0.41	200	48,508	7.40	0	37,436	5.16	9	36,381*
1,000	0.40	199	208,518	8.17	0	37,609*	5.92	2	41,522

Next, we compare the two iterative algorithms based on the performance measure given in Tables 2 and 3. In each row of these two tables, we assign a large penalty 1,000 to one particular objective component, and leave the other objectives less penalized with penalty 1. The columns show the results in terms of each objective. A notable remark is that the total length of all the transmission lines is not a real objective, but an approximation to the total number of loops to simplify our model.

Table 2: Performance analysis of LR

	Performance Measure								
penalty	**turn**	**intersec**	**overlap**	**dev (mi)**	**vio**	**len (mi)**	**loop**	**obj**	**time (hr)**
$\alpha = 1000$	244*	132	764	4,985	12,394	14,064	0*	262,275	6.27
$\beta = 1000$	821	5*	891	5,516	16,036	15,866	12	28,276	5.49
$\gamma = 1000$	1,040	453	315*	4,930	12,690	16,351	20	334,133	6.91
$\mu = 1000$	774	130	794	5,215	12,212*	15,133	9	12,218,922	5.16
$\nu = 1000$	529	178	834	7,434	28,634	13,663*	0*	37,609	8.17

[4] The number of loops from the PH algorithm is not monotonically decreasing as ν increases. This is because we approximate it by the total length of transmission lines to facilitate our solution process, and certainly these two are not equivalent. Fortunately, such inequivalency does not affect our conclusion and the total length of transmission lines is indeed monotonically decreasing.

Table 3: Performance analysis of PH

	Performance Measure								
penalty	**turn**	**intersec**	**overlap**	**dev (mi)**	**vio**	**len (mi)**	**loop**	**obj**	**time (hr)**
$\alpha = 1000$	216*	134	708	$4,994$	$12,420$	$14,941$	2*	$234,258$	5.19
$\beta = 1000$	759	8*	806	$4,804$	$11,110$	$14,477$	10	$25,489$	4.19
$\gamma = 1000$	874	419	330*	$4,780$	$11,566$	$15,470$	18	$347,657$	7.32
$\mu = 1000$	683	151	780	$5,214$	$10,220$*	$15,006$	10	$10,226,838$	4.09
$\nu = 1000$	471	167	854	$7,804$	$30,226$	$13,653$*	2*	$41,522$	5.92

As indicated by the numbers with asterisks in Tables 2 and 3, assigning a larger penalty to a particular objective component gives a solution with smaller value in that regard. The only exception happens for the number of loops because clearly there is a tie between $\alpha = 1000$ and $\nu = 1000$. This implies that the approximation we used is not equivalent to the original objective. Fortunately, despite the tie, $\nu = 1000$ still gives a solution with the least number of loops, and hence we can say that such inequivalency does not harm the applicability of our approximation.

In Tables 2 and 3, for the same combination of penalties, LR and PH output very similar results. It is not easy to distinguish them by a clear cutoff value as what we did between SR and the iterative algorithms. However, a thorough observation still reveals a few guidelines regarding how to select an appropriate algorithm based on the computational results at hand. As shown in the first, second and fourth rows of each table, when a large penalty is assigned to *turns*, *intersections* or *violations*, PH delivers a better solution within a shorter time, and thus outperforms LR. Similarly, when a large penalty is assigned to *overlaps*, LR performs better. When loops are penalized with $\nu = 1000$, a trade-off between the quality of the solution and running time is involved as LR reaches a better solution while PH runs faster. We do not consider the case when $\theta = 1000$, because the total deviations had already been minimized when we assigned each substation into a region by solving (3.1) - (3.5) no matter what combination of penalties is selected to solve the subproblems. Thus, the total deviations in the final solution is not completely determined by the ranking of the penalties, and should not be compared in the same way as the other objective components.

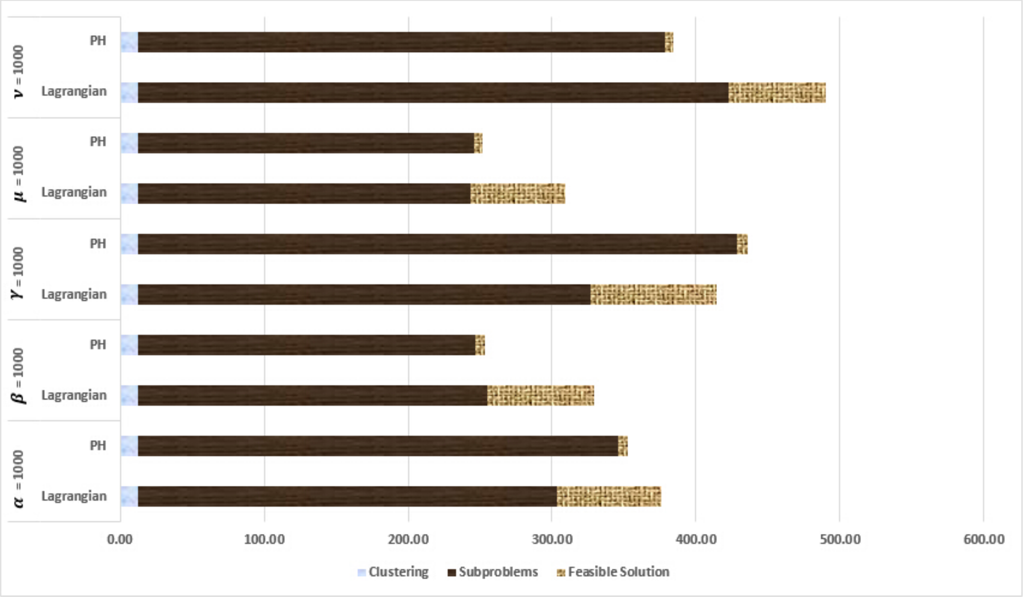

Figure 5: Running time of major components in the solution process

As we can see from Figure 5, the majority of the running time is spent on solving the independent subproblems. Therefore, parallel computing has a great potential to reduce the running time by solving the subproblems simultaneously. Moreover, as PH spends a larger proportion of time on subproblems, it would benefit more from parallel computing.

6 Summary and Future Research

In conclusion, we have proposed a mixed-integer model to optimize the layout of visual elements on a power transmission display map and developed three algorithms to effectively solve this NP-hard problem. Besides direct applications in power transmission control centers or more generally in display technology, our modeling and solution approaches can be extended to other complex networks such as transportation and supply chain systems. An interesting example is drone delivery. To make it possible in the future, logistics companies have to design routes to avoid intersections and overlaps to mitigate risk of collision and avoid loops to save time and energy costs. They also need to decide the candidate locations to build control centers and parking facilities.

There are many related topics worth of further effort. First, our iterative algorithms cannot explicitly handle transmission lines which are routed across more than two regions. Thus, a more general approach that is able to resolve this difficulty should be potentially considered. Second, we propose a heuristic shortest path algorithm to take the minimization of line turns and intersections into consideration. Although a virtual layer method like Deza et al. (2012) and Terlaky et al. (2005) could be applied to handle the minimization of turns in lattice graphs, to the best of authors' knowledge, an optimal approach to take care of both does not exist and is thus left for future research.

Acknowledgement

This research was supported by the New York State Energy Research and Development.

References

L. C. Abel. On the ordering of connections for automatic wire routing. *IEEE Transactions on Computers*, 100(11):1227–1233, 1972.

C. Albrecht. Provably good global routing by a new approximation algorithm for multicommodity flow. In *Proceedings of the 2000 International Symposium on Physical Design*, pages 19–25. ACM, 2000.

M. Burstein and R. Pelavin. Hierarchical wire routing. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 2(4):223–234, 1983.

R. C. Carden, J. Li, and C. Cheng. A global router with a theoretical bound on the optimal solution. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15(2):208–216, 1996.

C. Chiang, M. Sarrafzadeh, and C. Wong. Global routing based on Steiner min-max trees. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 9(12):1318–1325, 1990.

C. Chiang, C. Wong, and M. Sarrafzadeh. A weighted Steiner tree-based global router with simultaneous length and density minimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13(12):1461–1469, 1994.

A. Deza, C. Dickson, T. Terlaky, A. Vannelli, and H. Zhang. Global routing in VLSI design algorithms, theory, and computational practice. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 80:71, 2012.

Y. Fan and C. Liu. Solving stochastic transportation network protection problems using the progressive hedging-based method. *Networks and Spatial Economics*, 10(2):193–208, 2010.

M. L. Fisher. The Lagrangian relaxation method for solving integer programming problems. *Management science*, 27(1):1–18, 1981.

M. Formann, T. Hagerup, J. Haralambides, M. Kaufmann, F. T. Leighton, A. Symvonis, E. Welzl, and G. Woeginger. Drawing graphs in the plane with high resolution. *SIAM Journal on Computing*, 22(5): 1035–1052, 1993.

F. Glover. Tabu search-part I. *ORSA Journal on computing*, 1(3):190–206, 1989.

F. Glover. Tabu searchpart II. *ORSA Journal on computing*, 2(1):4–32, 1990.

M. Hayashi and S. Tsukiyama. A hybrid hierarchical approach for multi-layer global routing. In *Proceedings of the 1995 European Conference on Design and Test*, page 492. IEEE Computer Society, 1995.

J. Heisterman and T. Lengauer. The efficient solution of integer programs for hierarchical global routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(6):748–753, 1991.

J. Hu and S. S. Sapatnekar. A survey on multi-net global routing for integrated circuits. *INTEGRATION, the VLSI Journal*, 31(1):1–49, 2001.

T. C. Hu and M. Shing. *A decomposition algorithm for circuit routing.* Springer, 1985.

M. R. Kramer and J. Van Leeuwen. *Wire-routing is NP-complete.* Department of Computer Science, University of Utrecht Utrecht, 1982.

U. Lauther. Top down hierarchical global routing for channelless gate arrays based on linear assignment. In *Proceedings of the IFIP International Conference on VLSI*, pages 141–151, 1987.

O. Listes and R. Dekker. A scenario aggregation–based approach for determining a robust airline fleet composition for dynamic capacity allocation. *Transportation Science*, 39(3):367–382, 2005.

W. K. Luk, P. Sipala, M. Tamminen, D. Tang, L. S. Woo, and C. Wong. A hierarchical global wiring algorithm for custom chip design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 6(4):518–533, 1987.

B. Pampel and U. Brandes. Orthogonal-ordering constraints are tough. *Journal of Graph Algorithms and Applications*, 17(1):1–10, 2013.

A. M. Patel, N. L. Soong, and R. K. Korn. Hierarchical VLSI routing–an approximate routing procedure. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 4(2):121–126, 1985.

P. Raghavan and C. D. Thompson. Multiterminal global routing: a deterministic approximation scheme. *Algorithmica*, 6(1-6):73–82, 1991.

D. Richards. Complexity of single-layer routing. *IEEE Transactions on Computers*, (3):286–288, 1984.

R. T. Rockafellar and R. J. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147, 1991.

E. Shragowitz and S. Keel. A global router based on a multicommodity flow model. *INTEGRATION, the VLSI journal*, 5(1):3–16, 1987.

T. Terlaky, A. Vannelli, and H. Zhang. On routing in VLSI design and communication networks. In *Algorithms and Computation*, pages 1051–1060. Springer, 2005.

J. P. Watson and D. L. Woodruff. Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems. *Computational Management Science*, 8(4):355–370, 2011.

K. Winter and D. A. Mlynski. Hierarchical loose routing for gate arrays. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 6(5):810–819, 1987.

H. Zhou and D. F. Wong. Global routing with crosstalk constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(11):1683–1688, 1999.

Appendices

A Complete Formulation for Lagrangian Relaxation

Given the notation and decision variables defined in Sections 2 and 4.2, the complete formulation on a partitioned network for Lagrangian relaxation algorithm reads as follows.

$$\begin{aligned}\min \quad & \alpha \sum_{i}^{n} \sum_{m \in \bar{M}_i} \sum_{k \in N_i \cup N_i'} u_{m,k} + \beta \sum_{i}^{n} \sum_{k \in N_i \cup N_i'} \sum_{m_1 \in \bar{M}_i} \sum_{m_2 \in \bar{M}_i} v_{k,m_1,m_2} \\ & + \gamma \sum_{e \in E} o_e + \theta \sum_{i}^{n} \sum_{s \in S_i} \sum_{k \in N_i \cup N_i'} x_{s,k} d(s,k) \\ & + \mu \sum_{i}^{n} \sum_{s_1 \in S_i} \sum_{s_2 \in S_i} \sum_{k_1 \in N_i \cup N_i'} \sum_{k_2 \in N_i \cup N_i'} 1_{\{x_{s_1,k_1} = x_{s_2,k_2} = 1\}} V(s_1,k_1,s_2,k_2) \\ & + \mu \sum_{i}^{n} \sum_{s_1 \in S_i} \sum_{s_2 \in S \setminus S_i} \sum_{k_1 \in N_i \cup N_i'} x_{s_1,k_1} V'(s_1,k_1,s_2) + \nu \sum_{m \in M} \sum_{a \in A} y_{m,a} \text{length}(a)\end{aligned} \tag{A.1}$$

For each $i = 1, 2, ..., n$, we have the following region-wise constraints.

$$\sum_{k \in N_i \cup N_i'} x_{s,k} = 1 \qquad s \in S_i \tag{A.2}$$

$$\sum_{s \in S_i} x_{s,k} \leq 1 \qquad k \in N_i \cup N_i' \tag{A.3}$$

$$y_{m,a_1} + y_{m,a_2} \leq 1 \qquad e = \{a_1, a_2\} \in E_i,\ m \in \bar{M}_i \tag{A.4}$$

$$x_{s,k} + \sum_{a \in I_1^i(k) \cup I_2^i(k)} y_{m,a} = \sum_{a \in O_1^i(k) \cup O_2^i(k)} y_{m,a} + x_{t,k} \qquad k \in N_i \cup N_i',\ m = (s,t) \in M_i \tag{A.5}$$

$$\sum_{a \in I_1^i(k) \cup I_2^i(k)} y_{m,a} \leq 1 \qquad k \in N_i \cup N_i', m \in \ M_i \tag{A.6}$$

$$\sum_{a \in O_1^i(k) \cup O_2^i(k)} y_{m,a} \leq 1 \qquad k \in N_i \cup N_i', m \in \ M_i \tag{A.7}$$

$$x_{s,k} + \sum_{a \in I_1^i(k) \cup I_2^i(k)} y_{m,a} = \sum_{a \in O_1^i(k) \cup O_2^i(k)} y_{m,a} \qquad k \in N_i,\ m = (s,t) \in \bigcup_{j \in adj(i)} M_{i,j} \tag{A.8}$$

$$x_{t,k} + \sum_{a \in O_1^i(k) \cup O_2^i(k)} y_{m,a} = \sum_{a \in I_1^i(k) \cup I_2^i(k)} y_{m,a} \qquad k \in N_i,\ m = (s,t) \in \bigcup_{j \in adj(i)} M_{j,i} \tag{A.9}$$

$$\sum_{m \in \bar{M}_i} y_{m,a_1} + \sum_{m \in \bar{M}_i} y_{m,a_2} - 1 \leq L o_e \qquad e = \{a_1, a_2\} \in E_i \tag{A.10}$$

$$y_{m,r} + y_{m,h} - 1 \leq u_{m,k} \qquad \begin{aligned} k \in N_i \cup N_i',\ m \in \bar{M}_i,\ r \in I_1^i(k), h \in O_2^i(k) \\ or\ r \in I_2^i(k), h \in O_1^i(k) \end{aligned} \tag{A.11}$$

$$y_{m_1,r_1} + y_{m_1,h_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 \leq v_{k,m_1,m_2} \qquad \begin{aligned} & k \in N_i,\ m_1, m_2 \in \bar{M}_i,\ r_1 \in I_1^i(k), h_1 \in O_1^i(k) \\ & r_2 \in I_2^i(k), h_2 \in O_2^i(k) \text{ or} \\ & r_1 \in I_2^i(k), h_1 \in O_2^i(k), r_2 \in I_1^i(k), h_2 \in O_1^i(k) \end{aligned} \tag{A.12}$$

Let $c = (p, q) \in A_{i,j}$ be an inter-region arc going from region i to j and $c' = (q, p) \in A_{j,i}$ be the corresponding inter-region arc going from region j to i, $p \in N'_i$, $q \in N'_j$.

$$x_{s,p} + \sum_{a \in I_1^i(p) \cup I_2^i(p)} y_{m,a} = y_{m,c} + \sum_{a \in O_1^i(p) \cup O_2^i(p)} y_{m,a} \qquad m = (s,t) \in \bigcup_{j \in adj(i)} M_{i,j} \tag{A.13}$$

$$x_{t,p} + \sum_{a \in O_1^i(p) \cup O_2^i(p)} y_{m,a} = y_{m,c'} + \sum_{a \in I_1^i(p) \cup I_2^i(p)} y_{m,a} \qquad m = (s,t) \in \bigcup_{j \in adj(i)} M_{j,i} \tag{A.14}$$

$$y_{m,c} + \sum_{a \in O_1^i(p) \cup O_2^i(p)} y_{m,a} \le 1 \qquad m = (s,t) \in \bigcup_{j \in adj(i)} M_{i,j} \tag{A.15}$$

$$y_{m,c'} + \sum_{a \in I_1^i(p) \cup I_2^i(p)} y_{m,a} \le 1 \qquad m = (s,t) \in \bigcup_{j \in adj(i)} M_{j,i} \tag{A.16}$$

$$y_{m,c} + y_{m,r} - 1 \le u_{m,p} \qquad m \in \bigcup_{j \in adj(i)} M_{i,j},\ r \in I_1^i(p) \text{ or } I_2^i(p) \tag{A.17}$$

$$y_{m,c'} + y_{m,r} - 1 \le u_{m,p} \qquad m \in \bigcup_{j \in adj(i)} M_{j,i},\ r \in O_1^i(p) \text{ or } O_2^i(p) \tag{A.18}$$

$$\begin{aligned} y_{m_1,c} + y_{m_1,r_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 \le v_{p,m_1,m_2} \qquad & m_1 \in \bigcup_{j \in adj(i)} M_{i,j},\ m_2 \in \bar{M}_i, \\ & r_1 \in I_1^i(p),\ r_2 \in I_2^i(p),\ h_2 \in O_2^i(p) \text{ or} \\ & r_1 \in I_2^i(p),\ r_2 \in I_1^i(p),\ h_2 \in O_1^i(p) \end{aligned} \tag{A.19}$$

$$\begin{aligned} y_{m_1,c'} + y_{m_1,r_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 \le v_{p,m_1,m_2} \qquad & m_1 \in \bigcup_{j \in adj(i)} M_{j,i},\ m_2 \in \bar{M}_i, \\ & r_1 \in O_1^i(p),\ r_2 \in I_2^i(p),\ h_2 \in O_2^i(p) \text{ or} \\ & r_1 \in O_2^i(p),\ r_2 \in I_1^i(p),\ h_2 \in O_1^i(p) \end{aligned} \tag{A.20}$$

The following global constraints are imposed on all the regions.

$$\sum_{m \in M} y_{m,a_1} + \sum_{m \in M} y_{m,a_2} - 1 \le Lo_e \qquad e = \{a_1, a_2\} \in E \backslash \bigcup_i E_i \tag{A.21}$$

$$x_{s,k},\ y_{m,a},\ u_{m,k},\ v_{k,m_1,m_2},\ o_e\ \in \{0,1\} \tag{A.22}$$

All the constraints pertaining to the inter-region edge $e = \{c, c'\}$ are relaxed. The Lagrangian dual is

$$\begin{aligned} \max_{\lambda} \min \quad & \alpha \sum_i^n \sum_{m \in \bar{M}_i} \sum_{k \in N_i \cup N'_i} u_{m,k} + \beta \sum_i^n \sum_{k \in N_i \cup N'_i} \sum_{m_1 \in \bar{M}_i} \sum_{m_2 \in \bar{M}_i} v_{k,m_1,m_2} \\ & + \gamma \sum_{e \in E} o_e + \theta \sum_i^n \sum_{s \in S_i} \sum_{k \in N_i \cup N'_i} x_{s,k} d(s,k) \\ & + \mu \sum_i^n \sum_{s_1 \in S_i} \sum_{s_2 \in S_i} \sum_{k_1 \in N_i \cup N'_i} \sum_{k_2 \in N_i \cup N'_i} 1_{\{x_{s_1,k_1} = x_{s_2,k_2} = 1\}} V(s_1, k_1, s_2, k_2) \\ & + \mu \sum_i^n \sum_{s_1 \in S_i} \sum_{s_2 \in S \backslash S_i} \sum_{k_1 \in N_i \cup N'_i} x_{s_1,k_1} V'(s_1, k_1, s_2) + \nu \sum_{m \in M} \sum_{a \in A} y_{m,a} \text{length}(a) \end{aligned} \tag{A.23}$$

$$
\begin{aligned}
&+ \sum_{e\in E\setminus\bigcup\limits_{i} E_i} \lambda_e\Big(\sum_{m\in M} y_{m,c} + \sum_{m\in M} y_{m,c'} - 1 - Lo_e\Big) \\
&+ \sum_{i}^{n} \sum_{m\in \bigcup\limits_{j\in adj(i)} M_{i,j}} \sum_{c\in \bigcup\limits_{j\in adj(i)} A_{i,j}} \lambda^{1(i)}_{m,c}\Big(x_{s,p} + \sum_{a\in I_1^i(p)\cup I_2^i(p)} y_{m,a} - y_{m,c} - \sum_{a\in O_1^i(p)\cup O_2^i(p)} y_{m,a}\Big) \\
&+ \sum_{i}^{n} \sum_{m\in \bigcup\limits_{j\in adj(i)} M_{j,i}} \sum_{c'\in \bigcup\limits_{j\in adj(i)} A_{j,i}} \lambda^{1(i)}_{m,c'}\Big(x_{t,p} + \sum_{a\in O_1^i(p)\cup O_2^i(p)} y_{m,a} - y_{m,c'} - \sum_{a\in I_1^i(p)\cup I_2^i(p)} y_{m,a}\Big) \\
&+ \sum_{i}^{n} \sum_{m\in \bigcup\limits_{j\in adj(i)} M_{i,j}} \sum_{c\in \bigcup\limits_{j\in adj(i)} A_{i,j}} \lambda^{2(i)}_{m,c}\Big(y_{m,c} + \sum_{a\in O_1^i(p)\cup O_2^i(p)} y_{m,a} - 1\Big) \\
&+ \sum_{i}^{n} \sum_{m\in \bigcup\limits_{j\in adj(i)} M_{j,i}} \sum_{c'\in \bigcup\limits_{j\in adj(i)} A_{j,i}} \lambda^{2(i)}_{m,c'}\Big(y_{m,c'} + \sum_{a\in I_1^i(p)\cup I_2^i(p)} y_{m,a} - 1\Big) \\
&+ \sum_{i}^{n} \sum_{m\in \bigcup\limits_{j\in adj(i)} M_{i,j}} \sum_{c\in \bigcup\limits_{j\in adj(i)} A_{i,j}} \sum_{r\in I_1^i(p)} \lambda^{3(i)}_{m,c}(y_{m,c} + y_{m,r} - 1 - u_{m,p}) \\
&+ \sum_{i}^{n} \sum_{m\in \bigcup\limits_{j\in adj(i)} M_{i,j}} \sum_{c\in \bigcup\limits_{j\in adj(i)} A_{i,j}} \sum_{r\in I_2^i(p)} \lambda^{4(i)}_{m,c}(y_{m,c} + y_{m,r} - 1 - u_{m,p}) \\
&+ \sum_{i}^{n} \sum_{m\in \bigcup\limits_{j\in adj(i)} M_{j,i}} \sum_{c'\in \bigcup\limits_{j\in adj(i)} A_{j,i}} \sum_{r\in O_1^i(p)} \lambda^{3(i)}_{m,c'}(y_{m,c'} + y_{m,r} - 1 - u_{m,p}) \\
&+ \sum_{i}^{n} \sum_{m\in \bigcup\limits_{j\in adj(i)} M_{j,i}} \sum_{c'\in \bigcup\limits_{j\in adj(i)} A_{j,i}} \sum_{r\in O_2^i(p)} \lambda^{4(i)}_{m,c'}(y_{m,c'} + y_{m,r} - 1 - u_{m,p}) \\
&+ \sum_{i}^{n} \sum_{m_1\in \bigcup\limits_{j\in adj(i)} M_{i,j}} \sum_{m_2\in \bar{M}_i} \sum_{c\in \bigcup\limits_{j\in adj(i)} A_{i,j}} \sum_{\substack{r_1\in I_1^i(p),\\ r_2\in I_2^i(p),\\ h_2\in O_2^i(p)}} \lambda^{5(i)}_{m,c}(y_{m_1,c} + y_{m_1,r_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 - v_{p,m_1,m_2}) \\
&+ \sum_{i}^{n} \sum_{m_1\in \bigcup\limits_{j\in adj(i)} M_{i,j}} \sum_{m_2\in \bar{M}_i} \sum_{c\in \bigcup\limits_{j\in adj(i)} A_{i,j}} \sum_{\substack{r_1\in I_2^i(p),\\ r_2\in I_1^i(p),\\ h_2\in O_1^i(p)}} \lambda^{6(i)}_{m,c}(y_{m_1,c} + y_{m_1,r_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 - v_{p,m_1,m_2}) \\
&+ \sum_{i}^{n} \sum_{m_1\in \bigcup\limits_{j\in adj(i)} M_{j,i}} \sum_{m_2\in \bar{M}_i} \sum_{c'\in \bigcup\limits_{j\in adj(i)} A_{j,i}} \sum_{\substack{r_1\in O_1^i(p),\\ r_2\in I_2^i(p),\\ h_2\in O_2^i(p)}} \lambda^{5(i)}_{m,c'}(y_{m_1,c'} + y_{m_1,r_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 - v_{p,m_1,m_2}) \\
&+ \sum_{i}^{n} \sum_{m_1\in \bigcup\limits_{j\in adj(i)} M_{j,i}} \sum_{m_2\in \bar{M}_i} \sum_{c'\in \bigcup\limits_{j\in adj(i)} A_{j,i}} \sum_{\substack{r_1\in O_2^i(p),\\ r_2\in I_1^i(p),\\ h_2\in O_1^i(p)}} \lambda^{6(i)}_{m,c'}(y_{m_1,c'} + y_{m_1,r_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 - v_{p,m_1,m_2})
\end{aligned}
\tag{A.24}
$$

subject to

$$\text{Constraints } (A.2) - (A.12) \text{ and } (A.22)$$

B Complete Formulation for Progressive Hedging

Similar to Appendix A, given the notation and decision variables defined in Sections 2 and 4.3, the complete formulation for the progressive hedging (PH) algorithm is listed below. Unlike the Lagrangian relaxation approach in which inter-region arcs do not belong to any region, the PH approach partitions the network in such a way that each region includes inter-region arcs either originating from or going into it.

$$\min \sum_{i}^{n} Q_1^i + 0.5 \sum_{i}^{n} Q_2^i \tag{B.1}$$

where

$$\begin{aligned}
Q_1^i = \alpha & \sum_{m\in\bar{M}_i}\sum_{k\in N_i} u_{m,k} + \beta \sum_{k\in N_i}\sum_{m_1\in\bar{M}_i}\sum_{m_2\in\bar{M}_i} v_{k,m_1,m_2} \\
& + \gamma \sum_{e\in E_i} o_e + \theta \sum_{s\in S_i}\sum_{k\in N_i} x_{s,k} d(s,k) \\
& + \mu \sum_{s_1\in S_i}\sum_{s_2\in S_i}\sum_{k_1\in N_i}\sum_{k_2\in N_i} 1_{\{x_{s_1,k_1}=x_{s_2,k_2}=1\}} V(s_1,k_1,s_2,k_2) \\
& + \mu \sum_{s_1\in S_i}\sum_{s_2\in S\setminus S_i}\sum_{k_1\in N_i} x_{s_1,k_1} V'(s_1,k_1,s_2) \\
& + \nu \sum_{m\in\bar{M}_i}\sum_{a\in A_i} y_{m,a}\text{length(a)}, \\
Q_2^i = \gamma & \sum_{e\in\bar{E}_i\setminus E_i} o_e + \nu \sum_{m\in\bar{M}_i\setminus M_i}\sum_{a\in\bar{A}_i\setminus A_i} y_{m,a}\text{length(a)}.
\end{aligned}$$

For each $i = 1, 2, ..., n$, we have the following region-wise constraints.

$$\sum_{k\in N_i} x_{s,k} = 1 \qquad s\in S_i \tag{B.2}$$

$$\sum_{s\in S_i} x_{s,k} \le 1 \qquad k\in N_i \tag{B.3}$$

$$y_{m,a_1} + y_{m,a_2} \le 1 \qquad e=\{a_1,a_2\}\in E_i,\ m\in\bar{M}_i \tag{B.4}$$

$$x_{s,k} + \sum_{a\in I_1^i(k)\cup I_2^i(k)} y_{m,a} = \sum_{a\in O_1^i(k)\cup O_2^i(k)} y_{m,a} + x_{t,k} \qquad k\in N_i,\ m=(s,t)\in M_i \tag{B.5}$$

$$x_{s,k} + \sum_{a\in I_1^i(k)\cup I_2^i(k)} y_{m,a} = \sum_{a\in O_1(k)\cup O_2(k)} y_{m,a} \qquad k\in N_i,\ m=(s,t)\in \bigcup_{j\in adj(i)} M_{i,j} \tag{B.6}$$

$$x_{t,k} + \sum_{a\in O_1^i(k)\cup O_2^i(k)} y_{m,a} = \sum_{a\in I_1(k)\cup I_2(k)} y_{m,a} \qquad k\in N_i,\ m=(s,t)\in \bigcup_{j\in adj(i)} M_{j,i} \tag{B.7}$$

$$\sum_{a\in O_1(k)\cup O_2(k)} y_{m,a} \le 1 \qquad k\in N_i,\ m\in\bar{M}_i \tag{B.8}$$

$$\sum_{a \in I_1(k) \cup I_2(k)} y_{m,a} \leq 1 \qquad k \in N_i,\ m \in \bar{M}_i \tag{B.9}$$

$$\sum_{m \in \bar{M}_i} y_{m,a_1} + \sum_{m \in \bar{M}_i} y_{m,a_2} - 1 \leq Lo_e \qquad e = \{a_1, a_2\} \in \bar{E}_i, \tag{B.10}$$

$$y_{m,r} + y_{m,h} - 1 \leq u_{m,k} \qquad k \in N_i,\ m \in \bar{M}_i,\ r \in I_1(k), h \in O_2(k) \text{ } or\ r \in I_2(k), h \in O_1(k) \tag{B.11}$$

$$y_{m_1,r_1} + y_{m_1,h_1} + y_{m_2,r_2} + y_{m_2,h_2} - 3 \leq v_{k,m_1,m_2} \qquad k \in N_i,\ m_1, m_2 \in \bar{M}_i,\ r_1 \in I_1(k), h_1 \in O_1(k) \text{ } r_2 \in I_2(k), h_2 \in O_2(k) \text{ or } r_1 \in I_2(k), h_1 \in O_2(k), r_2 \in I_1(k), h_2 \in O_1(k) \tag{B.12}$$

The following global constraints are imposed on all the regions. For each $i = 1, 2, ..., n$, we need to impose the following conditions to make the scenario-dependent solutions implementable. We have

$$y^i_{m,a} = y^j_{m,a} \qquad j \in adj(i),\ a \in A_{i,j},\ m \in \bigcup_j M_{i,j}, \tag{B.13}$$

where $y^k_{m,a}$ is the value corresponding to $y_{m,a}$ from solving the subproblem in region k, $k = 1, 2$.

$$x_{s,k},\ y_{m,a},\ u_{m,k},\ v_{k,m_1,m_2},\ o_a\ \in \{0, 1\} \tag{B.14}$$