$CC_{pq} = [\{(i, j) : i \in C_p , j \in C_q \} \cup \{(i, j) : i \in C_q , j \in C_p \}] \cap J^F$ can be positive-fixed, where the cardinality of CC_{pq} is at least two. These constraints take the following form.

$$\sum_{(i,j)\in CC_{pq}} w_{ij} \leq u\,(CC_{pq}) \tag{5.57}$$

where $u\,(CC_{pq}) = \max \{ u_{ij} : (i, j) \in CC_{pq}\}$.

For every pair of components, we include constraint (5.57) within RLT(Ω) and dualize them in the corresponding Lagrangian dual subproblem. This results in introducing new terms in $\bar{c}_{w_{ij}}$ and in the constant term of the objective function of the dual problem. (Note that every $w_{ij} \in J^F$ appears only once in these constraints.) To obtain these new terms, let Q be the number of the components pairs in the current forest, and let l_q^4 for $q = 1, ..., Q$ be the Lagrange multipliers associated with these constraints. For each $w_{ij} \in J^F$ we examine the component pair $q \in \{1, ..., Q \}$ for which (i, j) belongs to the corresponding cut-set, and we add l_q^4 to $\bar{c}_{w_{ij}}$. Likewise for every q, $u\,(CC_{pq}) * l_q^4$ must be subtracted from the constant term of Lagrangian Dual objective function. Note that as the number of the forest components decreases, (5.57) may lead to the detection of infeasibility, if there exists no basic feasible completion to the current partial solution.

5.5.3 Partitioning Scheme

Given a current node of the branch and bound tree and its associated subproblem, the algorithm selects a variable $w_{pq} \in J^F$ in order to partition the problem into two subproblems associated with fixing w_{pq} either as a positive integer, i.e. $w_{pq} \in J^+$ or as zero, i.e. $w_{pq} \in J^0$. For the subproblem associated with $w_{pq} > 0$, l_{pq} is initialized as 1 and then by the logical tests, the lower and the upper bonds are tightened (for all affected variables), which in turn tightens the related constraints of RLT (Ω).

Note that by fixing w_{pq} to be positive, a new link is introduced or added to the current forest graph. This results in connecting two components of the forest, and therefore, in

order to keep the current forest cycle-free, the arcs of the cut-set corresponding to these two components (other than (p, q)) are zero-fixed.

We employ the following strategy for selecting a branching variable with the motivation to tighten the resulting relaxed node subproblems. Given the solution ($\bar{w}, \bar{a}, \bar{q}, \bar{j}, \bar{g}, \bar{x}, \bar{y}$) of the relaxed lower bounding problem at the current node, we choose a free variable w_{pq} having the most influence in determining the lower bound. In order to select this variable, we first release the demand constraint from the objective function of the aforementioned incumbent dual's Lagrangian subproblem. Accordingly, $\bar{c}_w$ and the constant term of the objective function must be adjusted as follows :

$$\text{new } \bar{c}_{w_{ij}} = \text{old } \bar{c}_{w_{ij}} + \lambda_j^3 \quad \forall (i, j)$$

$$\text{new constant term} = \text{old constant term} - \sum d_j \lambda_j^3 .$$

Now, for this restricted problem, we solve the transportation subproblem using this new $\bar{c}_w$ and the current updated bounds imposed on the variables w . Let $\bar{w}$ be the flow solution thus obtained. By subtracting the old contribution due to the w-subproblems and adding the new one, as well as adjusting the constant as above, yields a tighter lower bound ν_{LB2}, since this is equivalent to finding optimal duals with respect to the demand constraints, given the other duals. According to Proposition 1, we also compute the lower bound ν_{LB1}. Next, we select the lower bound of the current node to be equal to the best of the two computed lower bounds, i.e., $LB = \max\{\nu_{LB1}, \nu_{LB2}\}$. Now, according to this determination of LB, we consider the following branching method.

Branching Variable Selection Rule:

Given the bound LB at the current node, if $LB = \nu_{LB1}$, select Strategy #1, otherwise, select any method in Strategy # 2.

Strategy #1

Determine $(p, q) \in \arg\max_{(i,j)} \{(u_{ij} - l_{ij}) : (i, j) \in J^F\}$, and stop.

Strategy #2

For each $(i, j) \in J^F$, find

$C_{1ij} = \{ \bar{c}_{w_{ij}} \bar{w}_{ij} + \bar{c}_{q_{ij}} \bar{q}_{ij} + \bar{c}_{f_{ij}} \bar{f}_{ij} + \bar{c}_{g_{ij}} \bar{g}_{ij} \}$, and (5.58)

$C_{2ij} = \min\{0, u_{ij} * [\bar{c}_{w_{ij}} + \bar{c}_{q_{ij}} x_i^* + \bar{c}_{f_{ij}} y_i^* + a_{ij}^* \bar{g}_{ij}]\}$, (5.59)

where (x_i^*, y_i^*) is the current best incumbent solution's location coordinates, and

$a_{ij}^* = ((x_i^* - a_j)^2 + (y_i^* - b_j)^2)^{1/2}$.

Method #1: Pick $(p, q) \in$ arglexmin$\{C_{1ij}, C_{2ij}\}$

Method #2: Pick $(p, q) \in$ arglexmin$\{C_{2ij}, C_{1ij}\}$

Method #3: Pick $(p, q) \in$ arglexmin$\{u_{ij} * C_{1ij}, C_{2ij}\}$.

5.5.4 Computation of Upper Bounds and Updating of Incumbent Solution Based Constraints

For node fathoming purposes and for generating and updating the upper bound n^* which is required in the construction of constraint (5.25c) of Problem RLT(Ω), it is important to obtain a good quality incumbent (upper bound) solution early in the branch-and-bound tree. For determining an initial incumbent, given a set of fixed flows, we use the alternating method by first solving for the source locations via the location problem that uses weights in the objective function as given by the specified flows. With these fixed source locations, we compute the weights of objective function of the transportation problem, and hence solve for the corresponding optimal allocation variables. This continues in a sequential fashion until no further improvement is obtained in the objective function of Problem EDLAP. The detailed steps of this algorithm are provided below.

<u>Alternating method</u>

The following steps describe the generation of an upper bound at the initial node of the branch-and-bound algorithm, and the updating of this bound at any intermediate node.

Phase 1 (Performed only at the initial node)

Step 1: Consider the tightest rectangle that bounds the existing facilities, and partition this rectangle into n vertical slices having equal widths along the x-axis. Compute the aggregate demands $D_1,..., D_n$ for the facilities in slices 1,..., n (splitting the demand equally for facilities lying on the boundaries of the sliced regions). Arrange $D_1,..., D_n$ in non-decreasing order and also arrange $s_1,..., s_n$ in non-decreasing order. Match each supply with each demand according to this order. Let $s_{i(1)},...,s_{i(n)}$ be respectively matched with $D_1,..., D_n$. Accordingly, permute the new facilities in a list $S=\{i(1),..., i(n)\}$.

Step 2: Again proceeding from left to right, partition the existing facilities into sets $G_1,...,G_n$, splitting facilities along with their split demands among consecutive sets as required, so that the total demand in set G_k equals $s_{i(k)}$. For each set G_k, denoting d_j^k to be the demand for any existing facility $j \in G_k$ that has been assigned to this k^{th} set, solve the single facility squared Euclidean problem

$$\min\text{imum}_{(x,y)} \sum_{j \in G_k} [(x- a_j)^2 +(y - b_j)^2]\, c_{i(k)j} d_j^k \tag{5.60}$$

to obtain the solution $(x^*_{i(k)}, y^*_{i(k)})$ for each $k = 1, ..., n$. This yields

$$x^*_{i(k)} = \Big(\sum_{j \in G_k} c_{i(k)j} d_j^k\ a_j\Big) / \Big(\sum_{j \in G_k} c_{i(k)j} d_j^k\Big) \tag{5.61}$$

$$y^*_{i(k)} = \Big(\sum_{j \in G_k} c_{i(k)j} d_j^k\ b_j\Big) / \Big(\sum_{j \in G_k} c_{i(k)j} d_j^k\Big) \tag{5.62}$$

for each k =1,..., n. Let (x^*, y^*) be the resulting facility location solution thus obtained. Compute $c_{w_{ij}} = c_{ij}\ \mathsf{a}^*_{ij}\ \ \forall(i, j)$, where $\mathsf{a}^*_{ij} = [(x^*_i - a_j)^2 +(y^*_i - b_j)^2]^{1/2}$, solve the transportation problem min $\{c_w\ w : w \in W\}$, and let w^* be the corresponding optimal allocation for (x, y) fixed at (x^*, y^*). Compute

$$\mathsf{n}_{LR} = \sum_i \sum_j c_{ij} w^*_{ij} [(x^*_i - a_j)^2 +(y^*_i - b_j)^2]^{1/2} \tag{5.63}$$

Step 3: Repeat Step 1 and 2, but this time, proceed from bottom to top according to horizontal slices in a similar fashion. Let n_{BT} be the analogous value thus obtained via

(5.63). Select the better of the solutions η_{LR} and η_{BT}, take the corresponding allocation for this solution, and enter the alternating algorithm (Phase 2) with w fixed at this allocation value.

Phase 2 (Alternating steps)

Initialization: Let c_L be the vector of interaction weights for the location problem given any allocation, c_w be the vector of cost coefficients for the transportation problem given any set of location, and let k be the iteration number of the alternating method. Initialize with k =1, and go to Step 1 with $\bar{w}$ as the given allocation and use $f_1 = \infty$.

Step 1 (**Solution of the location problem with objective function =** $\sum_i \sum_i c_{L_{ij}} \alpha_{ij}$ **)**

Given the allocation variables $\bar{w}$, compute the weights c_L between the existing demand points as $c_{L_{ij}} = c_{ij} \ \bar{w}_{ij}, \forall(i, j)$. Use the algorithm of Chapter 4 to obtain an optimal solution $(\bar{x}, \bar{y})$, and set f_{k+1} equal to the optimal objective function value for this location problem. If $f_k - f_{k+1} < 10^{-3}$, then stop with the prescribed solution given by $(\bar{w}, \bar{x}, \bar{y})$, and with its objective value f_{k+1} being the incumbent primal solution (upper bound). Otherwise, increment k by one and go to Step 2.

Step 2 (**Solution of the transportation problem)**: Compute $c_{w_{ij}} = c_{ij} \ \bar{\alpha}_{ij} \ \forall(i, j)$, where $\bar{\alpha}_{ij} = \{(\bar{x}_i - a_j)^2 + (\bar{y}_i - b_j)^2\}^{1/2}$. Use the transportation solver to find an optimum $\bar{w}$ to the problem $f_{k+1} = \min \{c_w w : w \in W\}$.

If $f_k - f_{k+1} < 10^{-3}$, then stop with the prescribed solution given by $(\bar{w}, \bar{x}, \bar{y})$, and with its objective value f_{k+1} being the incumbent primal solution (upper bound). Otherwise, increment k by one and return to Step 1.

5.5.5 Other Features of the Branch-and-Bound Algorithm.

Search Strategy

A depth first (LIFO) strategy is adopted in the binary search in which a partial solution list, *PS*, records the history of branching in the branch-and-bound tree using the framework of Geoffrion (1967). A +(i, k) in *PS* indicates that w_{ik} is positive-fixed, -(i, k) in *PS* indicates that $w_{ik} = 0$, and an underlined $\underline{\pm(i, k)}$ indicates that the descendants of the corresponding complementary branch have been fathomed.

Optimality Criteria.

In order to avoid excessive computations involved in sifting through alternative solutions or close to global optimal solutions, the following fathoming criterion is adopted :

$$LB \geq (1- \epsilon') \, \eta^*,$$

where $0<\epsilon'<1$, *LB* is a lower bound on the objective function value computed at the current node of the branch-and-bound tree, and η^* is the current incumbent solution value. Accordingly, at termination of the algorithm, the global minimum of EDLAP is within $\epsilon' \cdot 100\%$ of the current best solution.

<u>Summary of the Branch-and-Bound Algorithm</u>

This procedure largely follows Sherali and Tuncbilek (1992) in its framework.

Step (0) Initialization. Set $PS = \emptyset$, $J^+ = \emptyset$, $J^0 = \emptyset$, $J^F = \{(i, j) : i = 1, \ldots, n, j = 1, \ldots, m\}$. Set $l_{ij} = 0$, $u_{ij} = \min \{s_i , d_j \} \; \forall (i, j) \in J^F$.

Calculate the maximum slack values using (5.55). Initialize the forest with (n+m) nodes and with no arcs, and the component count to $m + n$. Include all the nodes of the transportation graph in the set *NT* of nodes eligible for applying the logical test, and apply the logical tests as described in Step 2. Assuming feasibility of the transportation components, this logical test should end with $NT = \emptyset$. Go to Step 3.

Step (1) Cycle Prevention. Let (p, q) be the arc last added to J^+ , and let C_p and C_q be the two components in the current forest; note that $C_p \neq C_q$. Set $u_{ik} = 0$ $\forall(i, k) \in CC_{pq}$ - J^0 where $CC_{pq} = [\{(i, k) : i \in C_p , k \in C_q \} \cup \{(i, k) : i \in C_q , k \in C_p, (i, k) \neq (p, q)\}]$, and update SU_i and DU_k. If any of the maximum slack values becomes negative, go to Step 5. Otherwise, decrease the component count by 1 after combining the components C_p and C_q . Update J^F , PS , NT , and J^0 as $J^F \leftarrow J^F$ - $(CC_{pq}$ - J^0),
$PS \leftarrow$ PS $\cup \{ -\underline{(i, k)} : (i, k) \in CC_{pq} - J^0 \}$, $NT \leftarrow NT \cup NT_{CCpq}$ (where NT_{CCpq} is the set of nodes incident to the arcs in $(CC_{pq} - J^0)$, and $J^0 \leftarrow J^0 \cup CC_{pq}$. Proceed to Step 2.

Step (2) Logical Tests. All the arcs incident to the nodes in NT are eligible for applying the logical tests of subsection (5.5.1). Also, when testing an arc (p, q), where $p \in NT$, and $q \notin NT$, if any maximum slack value of node q changes then q is included in NT. The same argument applies for an arc (p, q) such that $q \in NT$ and $p \notin NT$.
Let (i, k) be the arc being tested. Update l_{ik} and u_{ik} using (5.56). If l_{ik} has changed, update SL_i and DL_k using (5.55). If u_{ik} has changed, update SU_i and DU_k using (5.55). Now, consider the following situations :

(a) If any maximum slack value becomes negative, go to Step 5.

(b) If $l_{ik} > 0$ where $(i, k) \in J^F$, update J^+ , J^F and PS, as $J^+ \leftarrow J^+ \cup \{(i, k)\}$, $J^F \leftarrow J^F$ -$\{(i, k)\}$, and $PS \leftarrow PS \cup \{\underline{(i, k)}\}$. Return to Step 1.

(c) If $l_{ik} = u_{ik} = 0$ where $(i, k) \in J^F$, update J^0 , J^F and PS, as $J^0 \leftarrow J^0 \cup \{(i, k)\}$, $J^F \leftarrow J^F$ - $\{(i, k)\}$, and $PS \leftarrow$ PS $\cup \{-\underline{(i, k)}\}$. Continue the logical tests on the other arcs.

If none of the bounds change for all the arcs incident to a node, drop that node from NT. Continue until $NT = \emptyset$. At this stage, if $|PS|$ is less than the total number of variables, then proceed to Step 3. Otherwise, by Proposition 3, set $\overline{w}_{ik} = u_{ik}$ $(=l_{ik})$ $\forall(i, k)$ in the graph.

Step 3 : Bounding Step. Using the current lower and upper bounds (l, u) on the variables in addition to the transportation constraints, solve the Lagrangian relaxation problem, and let η_{LB2} be the associated optimal objective value obtained. (Use the method

in subsection (5.5.3) to obtained the tightened version of η_{LB2}.) Compute η_{LB1} as in (5.48) (of Section 5.4) . Select $LB = \max\{\eta_{LB1}, \eta_{LB2}\}$. If $LB = \eta_{LB2}$ then starting with the allocation $\bar{w}$ determined while deriving η_{LB2}, use the alternating method to compute an upper bound on the problem. If the incumbent upper bound improves, update ν^* and the incumbent solution. If LB $\geq (1-\epsilon')\,\nu^*$ go to Step 5. Else, store the best upper bound on the current node and go to Step 4.

Step 4 : Branching Step. Select the branching variable w_{pq} , where $(p, q) \in J^F$, using the rule of subsection (5.5.3). Update J^+, J^F and PS, as $J^+ \leftarrow J^+ \cup \{(p, q)\}$, $J^F \leftarrow J^F - \{(p, q)\}$, and $PS \leftarrow PS \cup \{(p, q)\}$. Accordingly, set $l_{pq} = 1$, and update SL_p and SL_q using (5.55). Also, include nodes p and q in NT, and return to Step 1.

Step 5 : Fathoming Step. Let (p, q) be the right-most non-underlined entry in PS such that $LB_{pq} < (1 - \epsilon')\,\nu^*$. If such an entry does not exist, then stop; the incumbent solution is within $100\epsilon'$ % of optimality. Otherwise, set $l_{ik} = 0$ and $u_{ik} = \min\{s_i, d_k\}$ $\forall (i, k) \in PS^A_{pq} \cup J^F$, where PS^A_{pq} is the set arcs included in PS after (p, q). Drop the corresponding arcs in $PS^A_{pq} \cup \{(p, q)\}$ from J^+ and J^0, and include arcs in PS^A_{pq} in J^F.

Set $l_{ij} = 1$ and $u_{ij} = \min\{s_i , d_j\}$ $\forall (i, j) \in PS^B_{pq} \cap J^+$,

where $PS^B_{pq} = PS - [PS^A_{pq} \cup \{(p, q)\}]$. Also, noting that currently $\{+(p, q)\} \in PS$ by the branching process, set $l_{pq} = u_{pq} = 0$, and update PS and J^0 as $PS \leftarrow PS^B_{pq} \cup \{-(\underline{p, q})\}$, and $J^0 \leftarrow J^0 \cup \{(p, q)\}$.

Include all the nodes of the transportation graph in the set NT. Since at least one positive-fixed variable has changed its status, the current forest is no longer valid. Let AC be the total number of affected components in the current forest. If AC is equal to the component count, then rebuild the forest by using the current J^+. Otherwise, rebuild only the components which are affected. Update the component count and return to Step 2.