Optimizing Strategic Safety Stock Placement in Two-Layer Supply Chains

Ekaterina Lesnaia
Operations Research Center
Massachusetts Institute of Technology
Cambridge, MA 02139
Email: lesnaia@mit.edu

***Abstract*— In this paper, we minimize the holding cost of the safety stock in the supply chain subject to linear constraints on the service times between the nodes of the network. In the problem, the objective function is concave as we assume the demand to be bounded by a concave function. The optimal solutions of the problem belong to the set of extreme points of the polyhedron, specified by the constraints of the problem. We first characterize the extreme points for the two-layer networks and then provide bounds to use in a branch and bound algorithm.**

***Index Terms*— Base-Stock Policy; Dynamic Programming Application; Multi-Stage Supply-Chain Application; Safety Stock Optimization; Branch and Bound Algorithm.**

I. Introduction

Two major issues have to be addressed in the supply chain of a manufacturing firm. On one hand inventory across the chain has to be reduced to provide services cheaply and utilize fewer assets. On the other hand customers expect better services which includes on-time deliveries. A problem to be solved here is where in the chain to hold safety stocks to minimize inventory cost and to provide high level service to end customers.

The problem with similar assumptions was stated in the paper [1]. In particular, the key assumptions are

- we can model the supply chain as a network,
- each stage operates with a periodic-review base-stock policy,
- demand is bounded,
- there is guaranteed service time between every stage and its customer.

The assumption of bounded demand made it possible to formulate the problem as deterministic optimization. An efficient algorithm for a tree structure of the network was developed in the paper.

The purpose of current research is to develop a framework for modelling and solving the problem of placing safety stocks in the supply chain presented by a two-layer network. By two-layer network we mean a graph with two subsets of nodes. One of the subsets we call a subset of supply nodes. The other subset is a subset of demand nodes. Arcs are only possible from the supply nodes to the demand nodes. The objective here is first to describe potentially optimal points in general structure networks and to use the description to specify potentially optimal solutions in a two-layer network. Using this description, we provide an algorithm of solving the problem of placing safety stocks in the two-layer network.

The algorithm we develop in the paper is a branch and bound algorithm. We prove that we need to search an optimal solution only in a finite set of the solutions. Using this fact we specify the branching tree for the problem. For the lower bounds we modify the algorithm, described in [1], to fit into branch and bound framework. We show, how the knowledge of the potential points can help us improve the algorithm to be polynomial.To establish upper bounds we introduce a new algorithm.

II. Assumptions and formulation

A. Assumptions

In this section we introduce basic assumptions of the model. The assumptions were originally presented in [1], therefore we will not justify them here.

- **Multi-stage network.** We can model a supply chain as a network. Each node or stage in the network can be seen as a processing function in the supply chain. The nodes are potential locations for holding a safety-stock inventory of the item processed at the node.
  Let N be the number of nodes and $\mathbb{A}$ be the set of arcs in the graph representing the chain.
- **Production lead-times.** We assume that each node j has deterministic production lead-time T_j. Lead-time is the total time of production, given that all necessary components are available.
  Here we also introduce maximum replenishment time for a node j:
  $$M_j = T_j + \max_i\{M_i | i : (ij) \in \mathbb{A}\}.$$
- **Base-stock replenishment policy.** All stages operate with periodic-review base-stock policy with a common review period.
- **Demand process.** We assume that external demand occurs only in the demand nodes, that is in the nodes with zero outdegree. We denote the set of demand nodes as $\mathbb{D}$. For each demand node j demand $d_j(t)$ comes from a stationary process with average demand per period μ_j. Any other node $i \notin \mathbb{D}$ has only internal demand from its successors. We can calculate the demand in node i at

time t by summing the orders placed by its immediate successors:

$$d_i(t) = \sum_{(i,j)\in\mathbb{A}} \theta_{ij} d_j(t),$$

where a scalar θ_{ij} is associated with each node, representing the number of units of upstream component i required per downstream unit j. From this relationship, we find the average demand rate for the node i to be

$$\mu_i = \sum_{(ij)\in\mathbb{A}} \theta_{ij}\mu_j.$$

The most important assumption of the model is that demand is bounded. In particular, for each node j there exists a function $D_j(F)$ for $F = 1, 2, \ldots, M_j$, such that

1) for any period t and $F = 1, 2, \ldots, M_j$

$$D_j(F) \geq d_j(t-F+1)+d_j(t-F+2)+\ldots+d_j(t);$$

2) $D_j(0) = 0$
3) the function is concave and increasing;
4) $D_j(F) - F\mu_j$ is increasing for $F = 1, \ldots, M_j$.

- **Guaranteed service times.** We assume that node j provides 100% service and promises a guaranteed service time S_j for each node j. That means that demand $d_j(t)$ arrived at time t must be filled at $t+S_j$. Note, we assume that for any non demand node j and any downstream node $i : (j, i) \in \mathbb{A}$ the guaranteed service time of j is the same. Also, we impose bounds on the service times for the demand nodes, i.e., $S_j \leq s_j, j \in \mathbb{D}$, where $s_j < T_j$ is a given input representing the maximum service time for demand node j.
- **Inbound service times.** Let SI_j be inbound service time for node j. Inbound service time is the time for node j to get all of its supplies from nodes $i : (i, j) \in \mathbb{A}$ and to commence production.

B. Formulation

Suppose B_j is the base stock level for a node j and $I_j(t)$ is inventory in j at time t. Then the finished inventory at stage j at the end of period t is

$$I_j(t) = B_j - d_j(t - SI_j - T_j, t - S_j),$$

where $d_j(a, b)$ denotes the demand at stage j over the time interval $(a, b]$.

To provide 100% service level $I_j(t) \geq 0$. Therefore, we set $B_j = D_j(SI_j + T_j - S_j)$. Hence, expected inventory at stage j is

$$D_j(SI_j + T_j - S_j) - (SI_j + T_j - S_j)\mu_j,$$

which represents safety stock held at stage j.

Now, we formulate the problem $\mathcal{P}$ of finding optimal guaranteed outbound service times $\{S_j, j = 1, \ldots, N\}$ and inbound service times $\{SI_j, j = 1, \ldots, N\}$ in order to minimize total safety stock in the chain.

$$\begin{array}{ll} \text{minimize} & \sum_{j=1}^{N} h_j\{D_j(F_j) - F_j\mu_j\} \\ \text{s.t.} & F_j = SI_j + T_j - S_j, \quad j = 1, \ldots, N \\ & F_j \geq 0, \quad j = 1, \ldots, N \\ & F_j \leq M_j, \quad j = 1, \ldots, N \\ & S_i \leq SI_j, \quad (i, j) \in \mathbb{A} \\ & S_j \leq s_j, \quad j \in \mathbb{D} \\ & S_j \geq 0, \quad j = 1, \ldots, N \\ & SI_j \geq 0, \quad j = 1, \ldots, N, \end{array}$$

where h_j denotes the per-unit holding cost for inventory at stage j.

This is a problem of minimizing a concave function over a polyhedron, which in general is NP-hard (see [2] and [3]). It is not proved whether problem $\mathcal{P}$ is or is not NP-hard. Therefore, in what follows we will show some characteristics of the potentially optimal solutions and establish bounds on the optimal cost.

III. Necessary conditions

In this section we describe solutions, satisfying necessary optimality conditions. First, we show that optimal points can be found among solutions of special structure.

Let $\mathbb{S}$ be the set of supply nodes, i.e., the nodes with zero outdegree.

Lemma 1. *There always exists an optimal solution to the problem, such that all the inbound service times of the supply nodes are 0:*

$$SI_j = 0, j \in \mathbb{S}$$

and the remaining inbound service times are equal to a maximal service time of the upstream nodes:

$$SI_j = \max_i\{S_i : (i, j) \in \mathbb{A}\}, j \notin \mathbb{S}.$$

Proof:

Let us first consider a supply node $j \in \mathbb{S}$. Since $M_j = T_j$, we have $SI_j + T_j - S_j \leq M_j = T_j$. Therefore, $SI_j \leq S_j$. Consider a new solution:

$SI'_j = 0$
$S'_j = S_j - SI_j$.
The solution is feasible and

$$F'_j = SI'_j + T_j - S'_j = SI_j + T_j - S_j = F_j,$$

which implies that the cost of the new solution is the same as that of the original solution. Therefore, for any feasible solution to the problem there exists a solution with $SI_j = 0, j \in \mathbb{S}$ and with the same cost.

Now, consider a node $j \notin \mathbb{S}$. Suppose

$$(S_1, \ldots, S_N, SI_1, \ldots, SI_N)$$

is an optimal solution. Let

$$\delta_j = SI_j - \max_{i:(i,j)\in\mathbb{A}} S_i > 0.$$

We define a new solution
$SI'_j = SI_j - \delta_j$,
$S'_j = S_j - \min\{\delta_j, S_j\}$.

Suppose $\delta_j \leq S_j$. Then the new solution is feasible and

$$F'_j = SI'_j + T_j - S'_j = F_j.$$

This implies that the new solution has the same optimal cost and satisfies the lemma.

Let $\delta_j > S_j$. Then the new solution is feasible, but

$$F_j = SI'_j + T_j - S'_j = SI_j + T_j - S_j - (\delta_j - S_j) < F_j.$$

Since the inventory at stage j decreases as F_j decreases, the cost of the new solution is strictly less than the cost of the optimal solution. Therefore, δ_j is always no greater than S_j, which proves the lemma.

□

By lemma 1, there might be multiple optimal solutions, at least one of which has the properties, described in the lemma. In what follows, we will look for the solution with this property.

Lemma 2. *Let j be a demand node. In an optimal solution,*

$$S_j = s_j.$$

Proof:

The lemma follows from

$$h_j\{D_j(SI_j + T_j - S_j) - (SI_j + T_j - S_j)\mu_j\}$$

being decreasing in S_j and $S_j \leq s_j$. Note, that $s_j < T_j$, therefore no matter what the value of SI_j is, $SI_j + T_j - s_j > 0$. Hence, $S_j = s_j$ is always feasible.

□

The two lemmas, provide characterization of optimal solutions for the problem. It is optimal to have service times for the demand nodes to be equal to the maximum guaranteed service times. It is also optimal for an optimal inbound service time for any non supply node to be equal to the maximum outbound service time of its upstream nodes.

The results are rather intuitive. Postponing delivery of a product to the end customers till the latest possible moment gives greater flexibility in the earlier stages of the chain, and therefore more opportunities to minimize the total cost of the safety stock. The intuition behind the result of the first lemma might be as follows. In order to avoid unnecessary inventory, the inbound service time of a node should be no greater than the largest guaranteed service time of its suppliers.

In the next section we will concentrate on the networks, consisting of supply and demand nodes only. We will see how the previous analysis can be applied to identify an optimal solution.

IV. Two-layer networks

A. Necessary conditions

In this section we consider a special case of supply chain networks, that have only two subsets of nodes. The first layer is the set of supply nodes and the second is the set of demand nodes. Arcs are only possible between the layers with each arc going from a supply node to a demand node. Let the number of supply nodes be m and the number of demand nodes be l.

From the general case we know that for any demand node j:

- $S_j = s_j$;
- $SI_j = \max\limits_{i:(i,j)\in\mathbb{A}} S_i$.

Knowing this, we can consider the objective function as a function of guaranteed service times $\{S_i, i \in \mathbb{S}\}$. Remember, that $SI_i = 0, \forall i \in \mathbb{S}$.

Lemma 3. *If j is a supply node in a two-layer network, there are three possibilities for the optimal service time S_j:*

1) $S_j = 0$;
2) $S_j = T_j$;
3) $S_j = T_k$ *for* $T_k < T_j$.

Moreover, $S_j = T_k, T_k < T_j$ only if $S_k = T_k$.

Proof:

Let us order the nodes such that supply nodes get numbers from 1 to m in the order of their outbound service times, i.e.,

$$S_1 \leq S_2 \leq \ldots \leq S_m.$$

Let these outbound service times be optimal.

Suppose for some supply node i we know that $S_i < SI_j$ for all demand nodes $j : (i, j) \in \mathbb{A}$. The only other condition on S_i is $S_i \leq T_i$. Since the inventory function of stage j decreases as S_i increases and S_i is optimal, it can not happen that $T_i \geq SI_j$. Indeed, if $T_i \geq SI_j$, then we can increase the value of S_i up to SI_j without violating any constraints. This decreases the objective function value, which contradicts the optimality of $S_1, \ldots, S_m$. Hence, $T_i < SI_j$. By similar argument, $S_i = T_i$ in an optimal solution.

Suppose now $S_i = SI_j = a > 0$. Consider the connected component C_a (disregarding the directions of the arcs), such that

- $i \in C_a$;
- *for any* $j \in C_a \bigcap \mathbb{D}$, $SI_j = a$;
- *for any* $j \in C_a \bigcap \mathbb{S}$, $S_j = a$.

Let $c_1 = \max(j : j \in C_a \bigcap \mathbb{S})$ and $c_2 = \min(j : j \in C_a \bigcap \mathbb{S})$.

The total inventory function of the nodes from C_a can be considered as a function of a. We note, that since $a = S_j, j \in C_a \bigcap \mathbb{S}$, we have $a \leq T_j, j \in C_a \bigcap \mathbb{S}$. The function is concave, therefore, it reaches its minimum in the end points of the interval $[S_{c_1-1}, \min(T_j, j \in C_a \bigcap \mathbb{S}, S_{c_2+1})]$. Let

$$k = \arg\min(T_j : j \in C_a \bigcap \mathbb{S}).$$

If the minimum of the function is in S_{c_1-1} then we had to include $c_1 - 1$ into C_a. The same is true if $T_k > S_{c_2+1}$. Therefore, $a = \min(T_j, j \in C_a \bigcap \mathbb{S})$.

Suppose now, that $a = S_1$. The we again define C_a. This time the total inventory function on C_a is concave in a and defined on $[0, \min(T_j : j \in C_a \bigcap \mathbb{S})$. Therefore, the optimal a must be 0 or $\min(T_j : j \in C_a \bigcap \mathbb{S})$, which completes the proof of the lemma.

□

B. Algorithm

In this section we describe a branch and bound algorithm for the problem. The algorithm uses lower and upper bounds developed in the later sections of the paper.

We know that the objective function can be presented as a function of outbound service times for the supply nodes only. By lemma 3, the outbound service times can take only finite number of values in an optimal solution. Therefore, the most natural branching step is to take variable S_j and try all possible values for it. However, we can simplify the branching tree using lemma 3. In particular, we know that $S_j = T_k < T_j$ only if $S_k = T_k$. Therefore, we can eliminate the branch with $S_j = T_k < T_j$ if $S_k \neq T_k$.

Taking advantage of lemma 3, we can present the branching tree as follows. First, we order the supply nodes in the order of increasing lead-times:

$$T_1 \ldots, T_m.$$

At the first level of branching we let $S_1 = 0$ or $S_1 = T_1$. At the ith level of branching we let $S_i = \{0, T_1, \ldots, T_i\}$, making sure that $S_i = T_k < T_i$ only if $S_k = T_k$ earlier in the branch.

Now, we can search the tree using depth first search or breadth first search. The algorithm obviously gives an optimal solution [4].

In the worst case, the algorithm is no better than the complete enumeration of all possible solutions. Nevertheless, the implementation of the algorithm gives good results due to the bounds we use. The branching tree stays relatively small which makes the algorithm run relatively fast for the problems with the number of supply nodes up to 100.

C. Lower Bounds

To construct a lower bound on the optimal solution, we relax some constraints of the problem $\mathcal{P}$. In particular, we remove a number of constraints of the form $S_i \leq SI_j, (i,j) \in \mathbb{A}$, which is the same as removing corresponding arcs (i,j) from the arc set $\mathbb{A}$. Our goal here is to remove so many arcs from the graph, that the resulting graph becomes a spanning tree. We define the set of arcs of the spanning tree as $\mathbb{A}_T$.

For the spanning tree we can apply the algorithm described in [1] to obtain an optimal solution and therefore a lower bound on the optimal cost of $\mathcal{P}$. Since here we develop an algorithm for two-layer networks, we adjust the algorithm from [1] to an algorithm for the spanning trees, obtained from the two-layer networks.

We first renumber the nodes of the tree, using the following procedure:

1) Start with all nodes in the unlabeled set, U.
2) Set $k := 1$.
3) Find a node $i \in U$ such that node i is adjacent to at most one other node in U. That is, the degree of node i is 0 or 1 in the subgraph with node set U and arc set $\mathbb{A}_T$ defined on U.
4) Remove node i from set U and insert into the labeled set L; label node i with index k.
5) Stop if U is empty; otherwise set $k := k+1$ and repeat steps 3-4.

We will use the notation $p(k)$ for the node of the network, connected to node k such that $p(k) > k$. Note, that for each node $k < N$, there exists only one $p(k)$ according to the described procedure.

Following paper [1], we introduce the functions to use in the algorithm.

$$c_k(S, SI) = h_k\{D_k(SI + T_k - S) - (SI + T_k - S)\mu_k\}$$
$$+ \sum_{(i,k)\in A_T, i<k} f_i(S_i^*(SI)) + \sum_{(k,j)\in A_T, j<k} g_j(SI_j^*(S)),$$

where

$$f_k(S) = \min_{SI}\{c_k(S, SI)\},$$

for $\max(0, S - T_k) \leq SI \leq M_k - T_k$ and SI integer;

$$S_i^*(SI) = \arg\min\{f_i(S) : 0 \leq S \leq \min(SI, M_i), S \text{ integer}\};$$

$$g_k(SI) = \min_{S}\{c_k(S, SI)\},$$

for $0 \leq S \leq SI + T_k$ and S integer; If node k is a demand node, $S \leq s_k$;

$$SI_i^*(S) = \arg\min\{g_i(SI) : S \leq SI \leq M_i - T_i, SI \text{ integer}\}.$$

The algorithm evaluates $f_k(S)$ for $S = 0, \ldots, M_k$ if $p(k)$ is downstream of k. And it evaluates $g_k(SI)$ for $SI = 0, \ldots, M_k - T_k$ if $p(k)$ is upstream of k.

In section III we proved that for any demand node j, $S_j = s_j$ and for any supply node i, $SI_i = 0$ in an optimal solution. Since for a supply node i, $p(i)$ is always downstream of i, the algorithm evaluates

$$f_i(S) = \min_{SI}\{c_i(S, SI), SI \in [\max(0, S - T_i), M_i - T_i],$$
$$SI \text{ integer}\} = c_i(S, 0)$$

taking into account that $M_i = T_i$ for supply node k.

For a demand node j, $p(j)$ is always upstream. Therefore, the algorithm always evaluates

$$g_j(SI) = \min_{SI}\{c_j(S, SI), S \in [0, \min(s_j, SI + T_j)],$$
$$S \text{ integer}\} = c_i(s_j, SI).$$

In section IV-A we proved that optimal outbound service times of the supply nodes, and therefore inbound service times of the demand nodes, belong to the set $\{0\} \bigcup \{T_i, i \in D\}$. Moreover, if $i, k \in D$, then $S_i = T_k$ only if $T_i \geq T_k$. Hence, we only need to evaluate $f_k(S)$ for lead times of some demand nodes $S = 0, T_1, \ldots, T_i$, and $g_k(SI)$ for $SI = 0, T_1, \ldots, T_j$, such that $T_j = M_k - T_k$.

In the contexts of the branch and bound algorithm we described earlier, we need to obtain a lower bound when we know a part of the solution. In particular, suppose we know outbound service times of a subset $\mathbb{S}' \subseteq \mathbb{S}$ of supply nodes for the original problem. Then for any demand node j, such that $(i,j) \in \mathbb{A}, i \in \mathbb{S}'$, we can bound the inbound service times from below:

$$SI_j \geq \max_{i\in S'} S_i = si_j.$$

Now we can simply remove $\mathbb{S}'$ from the node set together with the singletons $\mathbb{D}' \in \mathbb{D}$ from the demand nodes, that might appear after removing $\mathbb{S}'$. For the demand nodes from

the removed set $\mathbb{D}'$ it is optimal to put $SI_j^* = si_j, j \in \mathbb{D}'$, since the function

$$h_j\{D_j(SI_j + T_j - s_j) - (SI_j + T_j - s_j)\mu_j\}$$

is increasing in SI_j. For the remaining network we can obtain a lower bound by removing some arcs to make a spanning tree and running a modified algorithm for the spanning tree. Note, that $si_j = 0, j \in \mathbb{D}$ if it is not connected to any removed supply node $i \in \mathbb{S}'$. For generality, we will still denote the set of demand nodes as $\mathbb{D}$ and the set of supply nodes as $\mathbb{S}$.

Now, we state the algorithm for a spanning tree, adjusted for the two-layer networks. We use the algorithm as a part of the branch and bound algorithm to obtain a lower bound on a branch.

Two-Layer Spanning Tree Algorithm

1. For $k =: 1$ to N
   2. If k is a supply node, evaluate $f_k(S) = c_k(S, 0)$ for $S \in \{0\} \bigcup \{T_i, T_i \leq T_k, i \in \mathbb{S}\}$.
   3. If k is a demand node, evaluate $g_k(SI) = c_k(s_k, SI)$ for $SI \in \{si_k\} \bigcup \{T_i, si_k \leq T_i \leq \max(si_k, M_k - T_k), i \in \mathbb{S}\}$.
2. If $N \in \mathbb{S}$ $z^* = \min_S f_N(S)$;
3. If $N \in \mathbb{D}$ $z^* = \min_{SI} g_N(SI)$.

This algorithm finds the optimal objective value z^* of the relaxed problem, and therefore it obtains a lower bound on the branch, analyzed by the branch and bound algorithm. We can find the service times by the backtracking procedure.

We note that the running time of the algorithm does not depend on the size of $M_j, j = 1, \ldots, m$, as it does in [1]. We evaluate a function $f_k(S)$ or $g_k(SI)$ for each node k in no more than m points, which makes the algorithm run in polynomial time.

D. Upper Bounds

Any feasible solution of the problem can be an upper bound. For example, the lower bound solution we obtained in the previous subsection can be changed so that it becomes feasible. The lower bound is characterized by the set of outbound service times of the supply nodes. The set of inbound service times of the demand nodes, that correspond to the spanning tree solution, is feasible for the spanning tree we have chosen, but it might be infeasible, if we consider the set of all arcs $\mathbb{A}$. In order to make the solution feasible, we let the inbound service times for a demand node to be equal to the maximum outbound service times of the of its upstream supply nodes, considering constraints on all arcs in $\mathbb{A}$. The solution we get is an upper bound on the optimal solution and we will call it the fixed solution.

The gap between the bound and the optimal cost can be very large. How large the gap is, can depend on the number of constraints on the arcs in $\mathbb{A} \setminus \mathbb{A}_T$ violated by the spanning tree solution, and the structure of the cost function.

Here, we provide a simple algorithm to tighten the upper bound given by a feasible solution to problem $\mathcal{P}$. We will search for the best solution among the solutions that have the same order of the outbound service times as the upper bound solution.

Suppose we know a priori the order of the outbound service times in an optimal solution, i.e.:

$$S_1 \leq S_2 \leq \ldots \leq S_m.$$

Note, we may reorder the supply nodes. We call the problem the ordered problem. Then we can solve the problem optimally by the following dynamic program.

I) **Define subsets of demand nodes.**
   1. Let $L = \{1, \ldots, l\}$
   2. For $i := m$ to 1
      $R_i = \{j : (i, j) \in \mathbb{A}, j \in L\}$;
      $L = L \setminus R_i$.

II) **Find the optimal solution of the ordered problem.**
   1. For $k := 1$ to m
      2. If $k = 1$
         $f_1(S_1) = h_1\{D_1(T_1 - S_1) - (T_1 - S_1)\mu_1\} + \sum_{i \in R_1} h_i\{D_i(S_1 + T_i - s_i) - (S_1 + T_i - s_i)\mu_i\}$, $S_1 \in [0, T_1]$.
      3. $f_k(S_k) = h_k\{D_k(T_k - S_k) - (T_k - S_k)\mu_k\} + \sum_{i \in R_k} h_i\{D_i(S_k + T_i - s_i) - (S_k + T_i - s_i)\mu_i\} + \min_S\{f_{k-1}(S), S \leq \min(T_{k-1}, S_k\}$, $S_k \in [0, T_k]$.
   2. Minimize $f_m(S_m)$, $S_m \in [0, T_m]$ to obtain the optimal objective function value.

The solution obtained is optimal by the principle of optimality for dynamic programming.

The algorithm gives an optimal solution to the problem $\mathcal{P}$ if we have the right order of the outbound service times for the supply nodes. It is still unclear how to get the order and further research can be done in this direction. Nevertheless, for the purpose of establishing an upper bound, any order of the service times can be used.

Suppose now, we know outbound service times of a subset $\mathbb{S}' \subseteq \mathbb{S}$ as we did in the previous subsection for the purpose of developing upper bounds for the branch and bound algorithm. We define $si_j, j \in \mathbb{S}$ the same way as before. We can again remove $\mathbb{S}'$ and $\mathbb{D}'$ from the set of nodes and run the algorithm to find an upper bound using functions:

- $\forall i \in R_k, k = 1, \ldots, m$
  $$v_i(S_k) = \begin{cases} h_i\{D_i(S_k + T_i - s_i) - (S_k + T_i - s_i)\mu_i\} \\ \text{if } S_k \geq si_i \\ h_i\{D_i(si_i + T_i - s_i) - (si_i + T_i - s_i)\mu_i\} \\ \text{if } S_k < si_i \end{cases}$$
- $f_1(S_1) = h_1\{D_1(T_1 - S_1) - (T_1 - S_1)\mu_1\} + \sum_{i \in R_1} v_i(S_1)$
- $k = 2, \ldots, m$
  $f_k(S_k) = h_k\{D_1(T_k - S_k) - (T_k - S_k)\mu_k\} + \sum_{i \in R_k} v_i(S_k)$
  $+ \min_S\{f_{k-1}(S), s < \min(T_{k-1}, S_k)\}$

The modified function are used in the branch and bounds algorithm, when the ordered problem is solved to obtain an upper bound for a branch. Since we do not know the optimal order of the inbound service times for the supply nodes, we

suggest to use the order of the fixed solution. The solution we get from the algorithm will certainly be no worse that the fixed solution, since the latter has the same order used in the algorithm.

V. Conclusion

While the problem of optimizing safety stocks in general network is complicated, we were able to provide characterization of the potential optimal solutions to some extend. In particular, we saw that it is optimal to set outbound service times for the demand nodes to be equal to the maximum guaranteed service times. It is also optimal to have inbound service times of a non supply node to be equal to the maximum outbound service time of its upstream suppliers. We proved that we can always assume inbound service times of the supply nodes to be 0.

Using the results and analyzing the constraint polyhedron for the two-layer networks, we concluded that in this case of a supply chain, the optimal cost is a function of the outbound service times for the supply nodes. We also concluded that the optimal outbound service time of a supply node should be equal to 0 or to the lead-times of supply nodes, which are no greater than the lead-time of the node.

Using the structure of the optimal solution, we saw how to do branching in the branch and bound algorithm. To bound the solutions we used the algorithm from [1], which we modified to adjust to the purposes of the branching and simplified according to the structure of an optimal solution. We introduced a new algorithm for an upper bound for a branch.

For the further research, it is interesting to explore how to order the outbound service times for the supply nodes to get better upper bounds.

It is also interesting to find a way to tighten the lower bounds. For example, a subgradient method can be applied for a Lagrangian relaxation of the problem. For each set of the Lagrange multipliers an algorithm similar to the spanning tree algorithm might be developed.

More computational test and theoretical analysis might be done in order to evaluate the performance of the algorithm.

Finally, we are interested in extending the technique of solving the problem of optimizing the safety stock in a general supply chain.

References

[1] Graves, S., and Willems, S. 2000. *Optimizing Strategic Stock Placement in Supply Chains*,Manufacturing & Service Operations Management 2000, Vol.2 No.1, Winter 2000, pp. 68-83.

[2] Chung, S.-J. 1989. *NP-completness of the linear complementarity problem*. Journal of Optimization Theory and Applications, 60:393-399.

[3] Mangasarian, O.L., 1997. *Minimum-Support Solutions of Polyhedral Concave Programs*. Technical report 97-05.

[4] Korte, B., Vygen, J. 2002. *Combinatorial Optimization: Theory and Algorithms (Algorithms and Combinatorics, 21)*, Springer Verlag; 2nd edition.