

Decentralized Dual-Based Algorithm for Computing Optimal Flows in a General Supply Chain

Vladimir Brayman (vbrayman@ee.washington.edu)*

Department of Electrical Engineering, University of Washington, Campus Box 352500, Seattle, WA 98195-2500 and Hynomics Corp.

Zelda Zabinsky (zelda@u.washington.edu)†

Industrial Engineering, University of Washington, Campus Box 352650, Seattle, WA 98195

Wolf Kohn (wk@hynomics.com)

Hynomics Corp., 10632 NE 37th Circle, Building 23, Kirkland, WA 98033-7021

Abstract. This paper addresses the issue of the optimal flow allocation in general supply chains. Our basic observation is that a distribution channel involving several reselling steps for a particular product can be viewed as a route in a supply chain network. The flow of goods or services along each route is influenced by the customer's demand, described by the corresponding utility functions, and prices charged at each node. We develop an optimization algorithm based on the *primal-dual framework* and the *Newton's step* that computes optimal prices at each node (dual problem) and then computes the optimal flow allocation (primal problem) based on these prices. Our main contribution is a discovery that the Newton's step leads to a partially decentralized algorithm which is a first step toward a decentralization schema for computing optimal prices.

Keywords: Supply Chain, Optimization, Primal-Dual Framework

1. Introduction

This paper addresses the issue of the optimal flow allocation in general supply chains. We view the supply chain optimization problem as an optimal allocation of available capacities of the facilities involved in the supply chain network to the customers of the supply chain. The performance measure is then chosen such that the result of the optimization, optimal flow allocation, strikes the right balance between profit maximization and customer satisfaction.

A complex supply chain structure makes centralized decision making impractical. Complete information is difficult to obtain, and often fierce competition leads to situations where companies hesitate to reveal sensitive information. Even if perfect information was available, the computational complexity of a centralized decision maker for a

* Partially supported by Hynomics Corp.

† Partially supported by NSF grant DMI-9820878



general supply chain would be enormous. An important feature of our approach is that it provides an approach toward a practical distributed optimization algorithm.

Supply chain management, as defined in [5], is “... a connected series of activities which is concerned with planning, coordinating and controlling materials, parts, and finished goods from supplier to customer. It is concerned with two distinct flows (material and information) through the organization.” There is growing research devoted to the modeling of supply chains. A comprehensive overview of recent developments can be found in [1] and [5].

From the inception of the supply chain research, it was realized that supply chain management is a distributed problem [4]. In this work, Clark and Scarf outlined a multi-echelon system that can be sequential, as shown in Figure 1, or have a tree-like structure, as in Figure 2. The boxes in Figure 2 represent facilities and arrowed lines indicate a downward flow of material. Clark and Scarf considered a centralized decision maker with a finite planning horizon and perfect information. In their analysis they also assumed that demand originates at the lowest level and at no other point in the system. The holding and shortage costs, at any level, are functions of the inventory at the current level plus all other inventory at lower levels or in transit to a lower level. These assumptions are characteristic of a *multi-echelon* system. An additional assumption was that each echelon backlogs excess demand. For this multi-echelon system, Clark and Scarf obtained an optimal inventory replenishment policy. They used a dynamic programming approach. This optimal policy has been the foundation for later analyses of multi-echelon systems.

Lee and Whang [8] extended this work in several significant ways. First, they realized that decentralization of the decision making is the necessary component of a realistic model of a modern enterprise, where on-site managers are responsible for the decisions they make. Second, Lee and Whang showed that decentralization of the decision making implies decentralization of the information: much of the information relevant to a particular site is not shared due in part to the large quantity of information and/or security reasons. Third, the authors introduced the idea of performance measure alignment. Although the importance of the right performance measures for the optimal behavior of a supply chain was emphasized in [2], Lee and Whang’s contribution is that they showed how a performance measurement scheme (corporate rules) acts as an incentive mechanism that aligns the interests of the on-site managers with the overall performance of the supply chain. The properties of such performance measurement schemes are cost conservation, incentive compatibility, and informational decentralizability. Lee

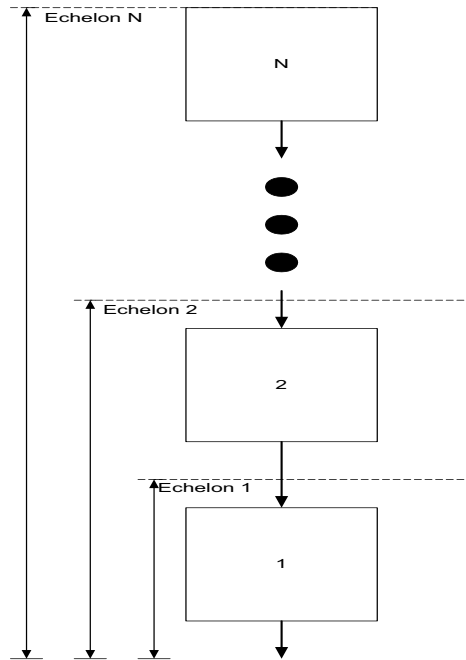


Figure 1. Sequential Multi-Echelon System

and Whang showed that the optimal multi-echelon inventory management policy of Clark and Scarf satisfies these requirements with minor assumptions.

Recent years witnessed a surge of interest in modeling and analysis of supply chains that have a general topology. There are several reasons for this. As markets and technologies become more dynamic, companies, striving for competitive advantage, envision their supply chains as means for achieving this advantage. This fact together with the advances in information technologies (e.g. communication, data mining) and transportation technology (e.g. overnight delivery) lead to appearance of larger supply chains and of more sophisticated topology. Also heightened customer expectations require introducing loops into supply chains due to return of merchandise. Figure 3 gives an example of such a supply chain. Flow of material in Figure 3 is not necessarily downward, as shown in Figure 1 and Figure 2. In this paper we extend the network configuration considered in [4] and [8] to include a general supply chain topology.

Our approach towards modeling of the general supply chain is based on the research done in the area of the pricing in the Internet [6]. In this work, Kelly considers end-to-end connections in the Internet and uses

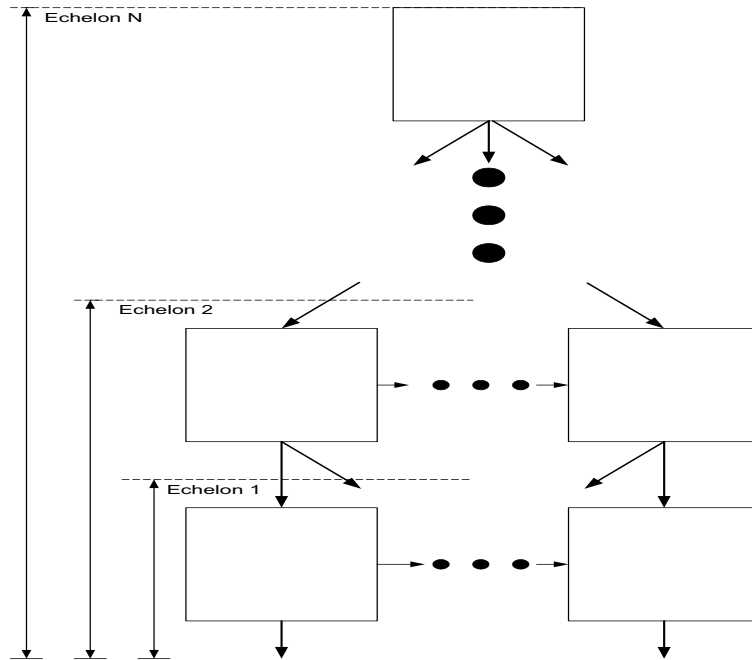


Figure 2. Tree-Structured Multi-Echelon System

an optimization schema based on the Lagrange relaxation technique. He introduces a centralized optimization problem of maximizing a sum of users' utility functions subject to capacity constraints at the resources of the network. He then shows that if each user chooses an increasing, strictly concave and continuously differentiable utility function (this case corresponds to elastic traffic [10]), the overall optimal flows satisfy the proportional fairness criterion. In a later paper, Kelly et al [7] developed decentralized gradient algorithms for computing the optimal flows over the Internet and shadow prices for pricing of bandwidth and proved convergence using appropriate Lyapunov functions.

The present work is, in the most part, concerned with the adaptation of Kelly's approach to the realm of distributed supply chains. Our basic observation is that a distribution channel involving several reselling steps for a particular product can be viewed as a route in a supply chain network. The flow of goods or services along each route is influenced by the customer's demand, described by the corresponding utility functions, and prices charged at each node. We also expand the classic multi-echelon inventory management approach to consider customer behavior. Instead of assuming a given demand, our approach is to incorporate customer satisfaction through customer utility func-

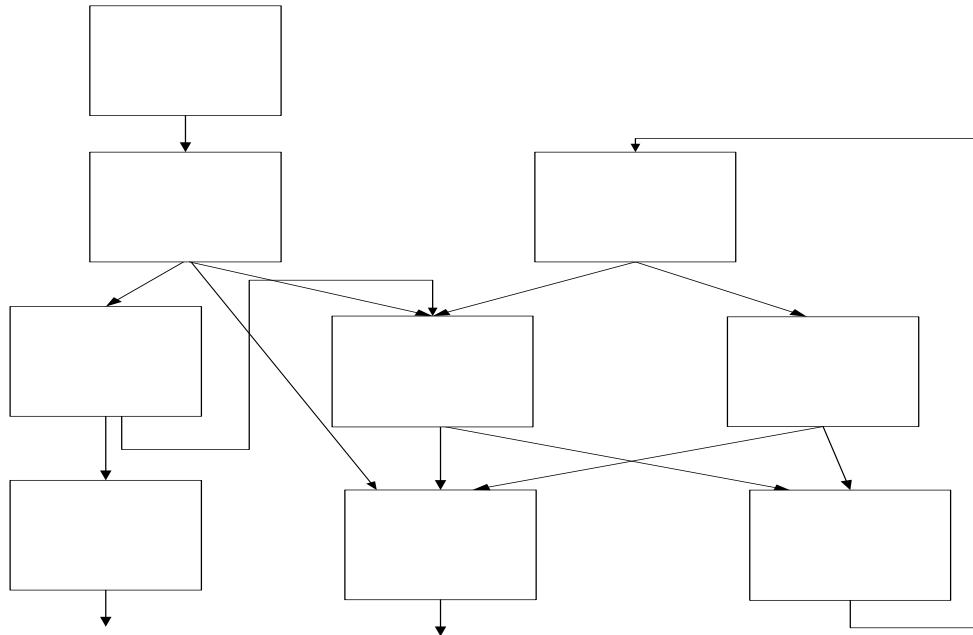


Figure 3. General Supply Chain

tions. We develop an optimization algorithm based on the *primal-dual framework* and the *Newton's step* that computes optimal prices at each node (dual problem) and then computes the optimal flow allocation (primal problem) based on these prices. Our main contribution is a discovery that Newton's step leads to a decentralization schema for computing optimal prices. Although the algorithm developed here is partially decentralized, the approach has a potential for development of a fully decentralized algorithm. Further research is needed to reach this goal.

The rest of the paper is organized as follows. In Section 2, we introduce the notation used in the paper and formulate the primal centralized optimization problem. Section 3 presents the primal-dual framework for solving the optimization problem. In particular, the corresponding Lagrangian is constructed, and the dual problem based on this Lagrangian is formulated. A numerical algorithm for solving the dual problem is developed in Section 4. Computational results are presented in Section 5. Finally, Section 6 discusses directions for future research.

2. Model

Let a general supply chain be represented by a network $G(\mathcal{N}, \mathcal{A})$, where \mathcal{N} is a set of nodes and \mathcal{A} is a set of arcs. Define a route r to be a non-empty ordered subset of \mathcal{N} . We interpret a route r to consist of the sequence of nodes in the supply chain associated with a distribution channel for a particular product. We assume that each route is unique, so there can be no more than one route passing through the same set of ordered nodes (i.e. the routing problem has been solved a priori). We let S be the set of all routes, and let x_r be the average rate of delivery of a product along route $r \in S$. For example, if τ is the time between a customer order and delivery of m units of a product r , then $x_r = \frac{m}{\tau}$.

We adopt a philosophy that supply chain performance should be measured in terms of the degree of customers' satisfaction. A customers' satisfaction is characterized by a utility function $U(x) = \sum_{r \in S} U_r(x_r)$.

We assume that $U_r(x_r)$ is strictly concave and monotonically increasing. This type of utility function corresponds to a customer behavior characterized by a tolerance to delays and decreasing marginal improvement in satisfaction due to incremental increases in the rate. Later in this paper we assume the utility function has a logarithmic form, $U_r(x_r) = \omega_r \log(x_r)$ for ω_r a positive constant. A logarithmic form was used by Kelly [6] in his Internet pricing framework, and is a special case of an increasing strictly concave continuously differentiable utility function, which corresponds to elastic traffic [10]. We also assume that each node i of the supply chain has a throughput capacity C_i .

The centralized supply chain problem can now be stated, as maximizing customer satisfaction of delivery rate of products subject to capacity constraints.

PROBLEM 1 (Centralized Primal).

$$\begin{aligned} & \max_{x \geq 0} \sum_{r \in S} U_r(x_r) & (1) \\ \text{subject to} & \quad Ax \leq C \end{aligned}$$

where $x = (x_1, \dots, x_{|S|})$, A is an $|\mathcal{N}| \times |S|$ matrix with the ir -th element defined as follows $A_{ir} = \begin{cases} 1 & \text{if } i \in r \\ 0 & \text{otherwise} \end{cases}$, and vector $C = (C_i | i \in \mathcal{N})$ where C_i is a given throughput capacity of node i . Note that concavity of $U_r(x_r)$ ensures that an optimal solution exists.

EXAMPLE 1. Consider the network shown in Figure 4.

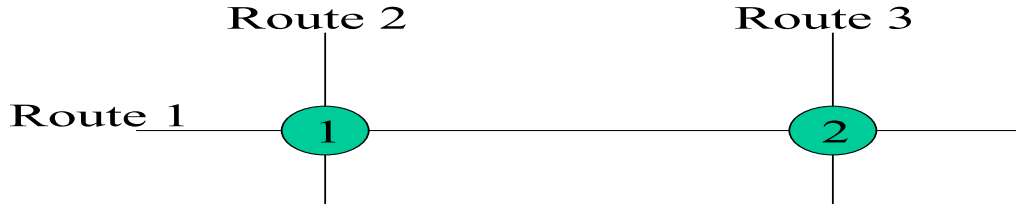


Figure 4. Network for Example 1

The route assignment matrix for this network, with two nodes and three routes, is

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

and the capacity vector is

$$C = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix}.$$

3. Primal/Dual Framework

In order to analyze Problem 1, we proceed as follows. Define the Lagrangian

$$L(x, \eta) = \sum_{r \in S} L_r(x_r, \eta) + \sum_{i \in \mathcal{N}} \eta_i C_i, \quad (2)$$

where $\eta = (\eta_i | i \in \mathcal{N})$ is the vector of Lagrange multipliers and for all routes $r \in S$,

$$L_r(x_r, \eta) = U_r(x_r) - \sum_{i \in \mathcal{N}} \eta_i A_{ir} x_r. \quad (3)$$

Then the Kuhn-Tucker conditions in the Lagrangian form state the necessary conditions for vectors $x = (x_r | r \in S)$ and $\eta = (\eta_i | i \in \mathcal{N})$ to be locally optimal [9]:

$$\frac{\partial L(x, \eta)}{\partial x_r} = \frac{\partial U_r(x_r)}{\partial x_r} - \sum_{i \in \mathcal{N}} \eta_i A_{ir} \leq 0 \quad \text{if } x_r = 0 \quad (4)$$

$$= 0 \quad \text{if } x_r > 0 \quad (5)$$

$$-\frac{\partial L(x, \eta)}{\partial \eta_i} = \sum_{r \in S} A_{ir} x_r - C_i \leq 0 \quad \text{if } \eta_i = 0 \quad (6)$$

$$= 0 \quad \text{if } \eta_i > 0. \quad (7)$$

Written differently, conditions (4)-(7) imply that there exist optimal vectors $\bar{x} = (\bar{x}_r | r \in S)$ and $\bar{\eta} = (\bar{\eta}_i | i \in \mathcal{N})$ that satisfy

$$\bar{x} \geq 0, \quad \bar{\eta}^T A \geq \nabla_x U(\bar{x}), \quad (\nabla_x U(\bar{x}) - \bar{\eta}^T A) \bar{x} = 0 \quad (8)$$

$$\bar{\eta} \geq 0, \quad A \bar{x} \leq C, \quad \bar{\eta}^T (A \bar{x} - C) = 0. \quad (9)$$

From relations (8) and (9), we interpret the quantity η_i as the price per unit flow through node i , and $\eta^T A$ as a vector of prices per unit flow for each route.

3.1. DUAL PROBLEM

We use the Lagrange relaxation technique [3] for the Lagrangian in (2). The dual problem of optimization associated with Problem 1 is as follows:

PROBLEM 2 (Centralized Dual).

$$\max_{\eta \geq 0} g(\eta)$$

where

$$g(\eta) := \inf_{x \geq 0} \left\{ - \sum_{r \in S} U_r(x_r) + \sum_{r \in S} \sum_{i \in \mathcal{N}} \eta_i A_{ir} x_r - \sum_{i \in \mathcal{N}} \eta_i C_i \right\}. \quad (10)$$

In order to proceed further, we need to specify the utility functions $U_r(x_r)$. As we discussed earlier, logarithmic utility functions possess nice properties that simplify the derivations:

$$U_r(x_r) = w_r \log x_r, \quad r \in S, \quad (11)$$

where w_r 's are positive constants. The analytical expression for $g(\eta)$ can be obtained by finding the vector

$$\bar{x}(\eta) = \arg \min_{x>0} \left\{ - \sum_{r \in S} U_r(x_r) + \sum_{r \in S} \sum_{i \in \mathcal{N}} \eta_i A_{ir} x_r - \sum_{i \in \mathcal{N}} \eta_i C_i \right\}, \quad (12)$$

and then substituting its entries \bar{x}_r into (10). The following observation simplifies the derivation and justifies the choice of the utility function (11). Observe that $U_r(0) = -\infty$ and $\frac{\partial U_r(x_r)}{\partial x_r}(0) = \frac{w_r}{x_r} |_{x_r=0} = \infty$ and thus values $x_r = 0$, $r \in S$ can not be considered as candidates for the optimal solution. Also observe that the expression in braces in (12) is the negative of Lagrangian (2). Then from (5), each entry of this vector is

$$\bar{x}_r(\eta) = \frac{w_r}{\sum_{i \in \mathcal{N}} \eta_i A_{ir}}. \quad (13)$$

Substitute (11) and (13) into (10) to get

$$g(\eta) = \sum_{r \in S} w_r \log \left(\sum_{i \in \mathcal{N}} \eta_i A_{ir} \right) - \sum_{i \in \mathcal{N}} \eta_i C_i - \text{const}, \quad (14)$$

where the constant term is $\text{const} = \sum_{r \in S} (w_r \log w_r - w_r)$. In what follows, we drop the constant term and consider Problem 2 with the objective function

$$g(\eta) = \sum_{r \in S} w_r \log \left(\sum_{i \in \mathcal{N}} \eta_i A_{ir} \right) - \sum_{i \in \mathcal{N}} \eta_i C_i. \quad (15)$$

In general, this problem can be solved in a centralized manner. This assumes existence of a certain central agent that possesses all the information about the supply chain and performs the computation. However, such an agent might not exist in a supply chain. Thus the computation of the optimal prices must be distributed among the nodes.

REMARK 1. *Observe that $g(\eta)$ is not decomposable by node because of the summation over all routes. However, the gradient of $g(\eta)$, $\nabla g(\eta) = \left(\frac{\partial g(\eta)}{\partial \eta_i} |_{i \in \mathcal{N}} \right)$, is decomposable. The i -th entry of $\nabla g(\eta)$ is given by*

$$\frac{\partial g(\eta)}{\partial \eta_i} = \sum_{r \in S_i} \frac{w_r}{\sum_{j \in \mathcal{N}} \eta_j A_{jr}} - C_i, \quad i \in \mathcal{N}. \quad (16)$$

where S_i is the set of routes passing through node i . Equation (16) suggests that a decentralized gradient-based algorithm can be developed for solving Problem 2.

4. Algorithm

In this section we develop a distributed numerical algorithm based on the Newton's step for solving dual Problem 2. First we decompose the gradient (16) into two parts: $\nabla g(\eta) = \nabla \tilde{g}(\eta) - C$, where, by definition,

$$\nabla \tilde{g}(\eta) := \left(\sum_{r \in S_i} \frac{w_r}{\sum_{j \in \mathcal{N}} \eta_j A_{jr}} \mid i \in \mathcal{N} \right) \quad (17)$$

and $C := (C_i \mid i \in \mathcal{N})$. Next we define the Hessian of $g(\eta)$ to be the symmetric negative-definite matrix $H(\eta)$ with each entry, $i, j \in \mathcal{N}$:

$$H_{ij}(\eta) = \frac{\partial^2 g(\eta)}{\partial \eta_i \partial \eta_j} = \frac{\partial (\nabla g(\eta))_i}{\partial \eta_j} = - \sum_{r \in S_i \cap S_j} \frac{w_r}{\left(\sum_{k \in \mathcal{N}} \eta_k A_{kr} \right)^2}. \quad (18)$$

Notice that the constant C portion of the gradient (16) is not present in the Hessian (18).

THEOREM 1. *Given $\nabla \tilde{g}(\eta)$ and $H(\eta)$ defined above, the following identity holds*

$$H^{-1}(\eta) \nabla \tilde{g}(\eta) = -\eta, \quad (19)$$

where $H^{-1}(\eta)$ is the inverse of $H(\eta)$. It is assumed that $H^{-1}(\eta)$ exists.

Proof. Observe that the gradient of $g(\eta)$ can be written in matrix form as follows

$$\nabla g(\eta) = A \left(\text{diag} \left(A^T \eta \right) \right)^{-1} w - C = \nabla \tilde{g}(\eta) - C, \quad (20)$$

where $\text{diag} \left(A^T \eta \right)$ is an $|\mathcal{N}| \times |\mathcal{N}|$ diagonal matrix and the vector $A^T \eta$ is along the main diagonal, yielding

$$\nabla \tilde{g}(\eta) = A \left(\text{diag} \left(A^T \eta \right) \right)^{-1} w. \quad (21)$$

Also the Hessian $H(\eta)$ in matrix form is

$$H(\eta) = -A \left(\text{diag} \left(A^T \eta \right) \right)^{-2} \text{diag}(w) A^T. \quad (22)$$

Now, we would like to prove identity (19) or, which is the same, we need to show that

$$\nabla \tilde{g}(\eta) = -H(\eta)\eta. \quad (23)$$

But

$$\begin{aligned}
-H(\eta)\eta &= A \left(\text{diag} \left(A^T \eta \right) \right)^{-2} \text{diag}(w) A^T \eta \\
&= A \left(\text{diag} \left(A^T \eta \right) \right)^{-1} \left(\text{diag} \left(A^T \eta \right) \right)^{-1} \text{diag} \left(A^T \eta \right) w \\
&= A \left(\text{diag} \left(A^T \eta \right) \right)^{-1} w.
\end{aligned} \tag{24}$$

Compare (24) and (21) to infer (23).

EXAMPLE 2. We continue with Example 1. From (15),

$$g(\eta) = w_1 \log(\eta_1 + \eta_2) + w_2 \log(\eta_1) + w_3 \log(\eta_2) - \sum_{i=1}^2 \eta_i C_i. \tag{25}$$

The gradient of $g(\eta)$ in (25) is

$$\nabla g(\eta) = \left(\frac{\partial g(\eta)}{\partial \eta_1}, \frac{\partial g(\eta)}{\partial \eta_2} \right) = \left(\frac{w_1}{\eta_1 + \eta_2} + \frac{w_2}{\eta_1} - C_1, \frac{w_1}{\eta_1 + \eta_2} + \frac{w_3}{\eta_2} - C_2 \right) \tag{26}$$

and the Hessian of it is as follows

$$H(\eta) = - \begin{bmatrix} \frac{w_1}{(\eta_1 + \eta_2)^2} + \frac{w_2}{\eta_1^2} & \frac{w_1}{(\eta_1 + \eta_2)^2} \\ \frac{w_1}{(\eta_1 + \eta_2)^2} & \frac{w_1}{(\eta_1 + \eta_2)^2} + \frac{w_3}{\eta_2^2} \end{bmatrix}. \tag{27}$$

The inverse of Hessian in (27) is given as follows

$$H^{-1}(\eta) = \frac{1}{\det H} \begin{bmatrix} H_{22}(\eta) & -H_{12}(\eta) \\ -H_{12}(\eta) & H_{11}(\eta) \end{bmatrix}, \tag{28}$$

where $H_{ij}(\eta)$, $i, j = 1, 2$ is given by (18) and (27), and the determinant of H is

$$\begin{aligned}
\det H &= H_{11}H_{22} - (H_{12})^2 \\
&= \left(\frac{w_1}{(\eta_1 + \eta_2)^2} + \frac{w_2}{\eta_1^2} \right) \left(\frac{w_1}{(\eta_1 + \eta_2)^2} + \frac{w_3}{\eta_2^2} \right) - \left(\frac{w_1}{(\eta_1 + \eta_2)^2} \right)^2 \\
&= \frac{w_1}{(\eta_1 + \eta_2)^2} \frac{w_3}{\eta_2^2} + \frac{w_2}{\eta_1^2} \frac{w_1}{(\eta_1 + \eta_2)^2} + \frac{w_2}{\eta_1^2} \frac{w_3}{\eta_2^2}.
\end{aligned} \tag{29}$$

From (17) and (28),

$$H^{-1}(\eta) \nabla \tilde{g}(\eta) = \frac{1}{\det H} \begin{bmatrix} H_{22}(\eta) (\nabla \tilde{g}(\eta))_1 - H_{12}(\eta) (\nabla \tilde{g}(\eta))_2 \\ -H_{12}(\eta) (\nabla \tilde{g}(\eta))_1 + H_{11}(\eta) (\nabla \tilde{g}(\eta))_2 \end{bmatrix}. \tag{30}$$

We carry out the calculation of the first term in the square brackets explicitly

$$\begin{aligned}
& H_{22}(\eta) (\nabla \tilde{g}(\eta))_1 - H_{12}(\eta) (\nabla \tilde{g}(\eta))_2 \\
&= - \left(\frac{w_1}{(\eta_1 + \eta_2)^2} + \frac{w_3}{\eta_2^2} \right) \left(\frac{w_1}{\eta_1 + \eta_2} + \frac{w_2}{\eta_1} \right) + \frac{w_1}{(\eta_1 + \eta_2)^2} \left(\frac{w_1}{\eta_1 + \eta_2} + \frac{w_3}{\eta_2} \right) \\
&+ \frac{w_1}{(\eta_1 + \eta_2)^2} \frac{w_1}{\eta_1 + \eta_2} + \frac{w_1}{(\eta_1 + \eta_2)^2} \frac{w_3}{\eta_2} \\
&= - \frac{w_1}{(\eta_1 + \eta_2)^2} \frac{w_2}{\eta_1} - \frac{w_3}{\eta_2^2} \frac{w_1}{\eta_1 + \eta_2} - \frac{w_3}{\eta_2^2} \frac{w_2}{\eta_1} + \frac{w_1}{(\eta_1 + \eta_2)^2} \frac{w_3}{\eta_2} \\
&= - \frac{w_1}{(\eta_1 + \eta_2)^2} \frac{w_2}{\eta_1} - \frac{w_3}{\eta_2^2} \frac{w_2}{\eta_1} - \frac{w_1}{(\eta_1 + \eta_2)^2} \frac{w_3}{\eta_2^2} \eta_1 \\
&= -\eta_1 \left(\frac{w_1}{(\eta_1 + \eta_2)^2} \frac{w_2}{\eta_1^2} + \frac{w_3}{\eta_2^2} \frac{w_2}{\eta_1^2} + \frac{w_1}{(\eta_1 + \eta_2)^2} \frac{w_3}{\eta_2^2} \right). \tag{31}
\end{aligned}$$

Note that (29) and (31) imply that

$$\frac{1}{\det H} (H_{22}(\eta) (\nabla \tilde{g}(\eta))_1 - H_{12}(\eta) (\nabla \tilde{g}(\eta))_2) = -\eta_1. \tag{32}$$

It can be also shown that

$$\frac{1}{\det H} (-H_{12}(\eta) (\nabla \tilde{g}(\eta))_1 + H_{11}(\eta) (\nabla \tilde{g}(\eta))_2) = -\eta_2. \tag{33}$$

Identities (30), (32), and (33) imply (19). This illustrates the application of Theorem 1.

REMARK 2. Theorem 1 suggests a partially distributed algorithm with the following Newton step

$$\eta_i^{(t+1)} = \eta_i^{(t)} + \eta_i^{(t)} + H_i^{-1}(\eta^{(t)}) C, \quad i \in \mathcal{N}, \tag{34}$$

where $H_i^{-1}(\eta)$ is the i -th row of $H^{-1}(\eta)$. Notice that the second term in (34) follows from identity (19). If an approximation of the last term in (34) can be developed so that it depends on $\eta_i^{(t)}$ only, then a fully distributed algorithm can be achieved. This is a subject of further research.

Observe that Theorem 1 assumes that the inverse of the Hessian $H(\eta)$ exists. At the same time, (22) suggests that the invertibility of $H(\eta)$ depends on the invertibility of A . We state the obvious proposition without a proof.

PROPOSITION 1. Hessian (22) has an inverse if and only if A has a right inverse.

Note that Proposition 1 implies that the number of rows of A (nodes of the network) must be less than the number of columns (routes). The Gauss-Jordan procedure can be applied in order to assure the invertibility of A . Alternatively, by adding dummy nodes or routes the original network can be transformed into a network with invertible A . This must be done before running the algorithm. We also note that Proposition 1 refers to the structural invertibility of $H(\eta)$. There might be values of η^t while running the algorithm such that $H(\eta^t)$ is not invertible. Special care must be taken in this case, and on-going research is investigating ways to assure invertibility.

We are now ready to state the algorithm. It is assumed that the vector of throughput capacities C , and vector $H_i^{-1}(\eta^t)$ are known to every node. Then, for each node i :

Algorithm (Distributed Dual)

Initialization Initialize: $\eta_i^0 > 0$.

Step 1 Compute vector $H_i^{-1}(\eta^{(t)}) C$

Step 2 Set the next point to

$$\eta_i^{(t+1)} = \eta_i^{(t)} + \eta_i^{(t)} + H_i^{-1}(\eta^{(t)}) C, \quad i \in \mathcal{N},$$

Step 3 Check the stopping criterion. If it is satisfied, stop. Otherwise, go to Step 1.

The convergence proof of this algorithm is the same as the convergence proof of the Newtonian algorithm and can be found elsewhere [3].

5. Computational Results

In this section we present the computational results of the run of the algorithm on two examples. The first example has been introduced in Example 1. The second sample is derived from the real supply chain of the HP DeskJet production facility presented in [11]. We performed several tests using the Distributed Dual Algorithm (DDA) presented in Section 4, standard Newton algorithm, and Matlab constraint optimization function *fmincon*. The comparison shows that DDA converges to the same solution as the standard Newton algorithm. This solution is within a predefined numerical accuracy interval from the true optimum obtained with the Matlab routine *fmincon*. The comparison also shows that DDA exhibits linear rate of convergence of the standard

Newton algorithm which is slightly better than the rate of convergence of *fmincon*.

5.1. NUMERICAL EXAMPLE 1

The numerical values in Table I provide the parameters used in this example.

Table I. Parameter Values for Numerical Example 1.

| Parameter | Numerical Value |
|----------------------------------|-----------------|
| $w_r, r = 1, 2, 3$ | 1 |
| $C_i, i = 1, 2$ | 1 |
| Initial point η^0 | [1, 1] |
| Stopping criterion ε | 10^{-3} |

Performance results of the three algorithms are presented in Table II. One can observe that both DDA algorithm and Standard Newton algorithm converge in 4 iterations with 4 functions evaluation. The computed optimal value of η within the tolerance interval from the true optimum $\eta = 1.5$. This demonstrates that the DDA algorithm is comparable with Newton's algorithm in computational performance and moreover, has the potential to be distributed.

Table II. Performance Results for Numerical Example 1.

| Algorithm | # of Iterations | # of Func. Evaluations | Final Values of η | Optimal Flows x |
|-----------------------|-----------------|------------------------|--|--|
| DDA | 4 | 4 | $\begin{bmatrix} 1.499999975 \\ 1.499999975 \end{bmatrix}$ | $\begin{bmatrix} 0.33 \\ 0.67 \\ 0.67 \end{bmatrix}$ |
| Standard Newton | 4 | 4 | $\begin{bmatrix} 1.499999975 \\ 1.499999975 \end{bmatrix}$ | $\begin{bmatrix} 0.33 \\ 0.67 \\ 0.67 \end{bmatrix}$ |
| Matlab <i>fmincon</i> | 2 | 9 | $\begin{bmatrix} 1.5 \\ 1.5 \end{bmatrix}$ | $\begin{bmatrix} 0.33 \\ 0.67 \\ 0.67 \end{bmatrix}$ |

5.2. NUMERICAL EXAMPLE 2

Consider a supply chain shown in Figure 5. This is a simplified version of the HP DeskJet printers supply chain, see [11]. HP DeskJet printer division is located in Vancouver, Washington. This is a high volume, high production rate facility with total factory cycle of about one week. The Vancouver division is connected with its suppliers, most of which are other HP divisions, and the distribution centers, located in the US, Europe, and Asia. Overseas sales require localization, i.e. equipment of printers with the appropriate power supplies and manuals. Also they lead to a significant transportation times between the factory and the distribution centers (DC's) in Europe and Asia. Customers of HP (resellers) request a high level of availability of the printers due to highly competitive market. In response, HP decided to operate the three DC's in a make-to-stock mode, i.e. maintaining the target inventories at the levels of the forecasted sales plus some safety stock. As a consequence of the long transportation time, the European and Asian DC had to keep high levels of safety stocks in order to respond to fluctuations in the demand for the different versions of the printer.

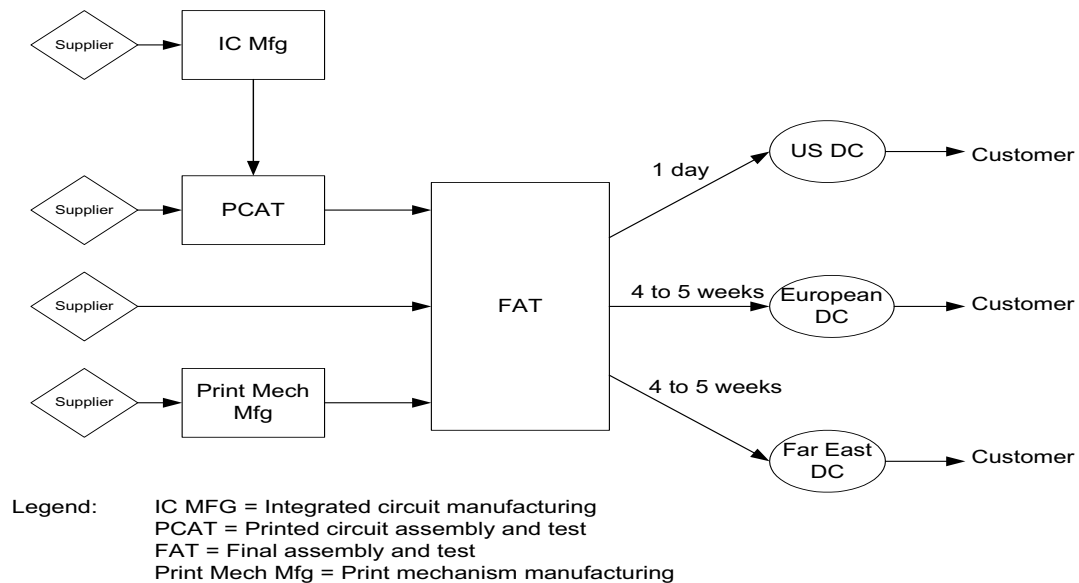


Figure 5. HP DeskJet Printer Supply Chain

In this paper, we consider the distribution process of this supply chain only. Assume that there are two versions of the printer, AU and BU, that target the US market, two versions, AE and BE, that target the European market, and one version, AA, that targets the Asian market. Figure 6 shows the distribution network.

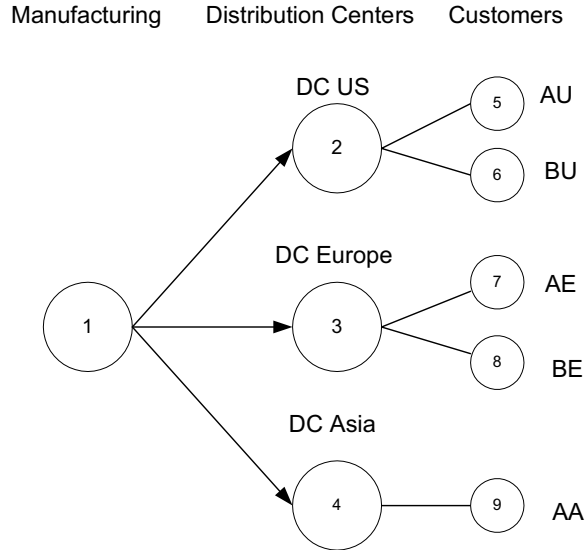


Figure 6. Distribution Process Graph

For the example on hand, the set of routes is as follows:

$$S = \left\{ \begin{array}{l} (1, 2, 5), (1, 2, 6), (1, 3, 7), (1, 3, 8), (1, 4, 9), \\ (1, 2), (1, 3), (1, 4), (2, 5), (2, 6), (3, 7), (3, 8), (4, 9) \end{array} \right\},$$

and the route assignment matrix is:

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Table III. Parameter Values for Numerical Example 2.

| Parameter | Numerical Value |
|----------------------------------|--|
| $w_r, r = 1, \dots, 13$ | 70000 |
| $C_i, i = 1, \dots, 9$ | [2000, 1500, 1500, 900, 600.6, 605.1, 600.7, 610.3, 581.4] |
| Initial point η^0 | [50, 50, 50, 50, 50, 50, 50, 50, 50] |
| Stopping criterion ε | 10^{-3} |

The problem parameters are given in Table III.

The performance results are summarized in Table IV. The three algorithms converged to very similar solutions, and DDA took the same amount of computation as the standard Newton algorithm, which was less than the Matlab routine.

The values of η after each iteration of the Distributed Dual Algorithm are presented in Table V to illustrate the speed of convergence. The linear nature of the convergence rate of the algorithm is shown in Figure 7. After only two iterations the intermediate value is very close to the optimal solution. This implies that DDA can be used for large supply chain systems.

6. Conclusions

We presented a primal-dual framework for computing optimal flow allocation in a general supply chain. The main feature of our approach is that it leads to a practical distributed algorithm. Numerical results show that the linear convergence of the algorithm is the same as the centralized Newton algorithm. In order to reach complete decentralization of the algorithm, we need to find a suitable approximation to $H^{-1}(\eta)$. This is a topic for further research.

7. Acknowledgments

The authors would like to acknowledge an appreciation to Professor Paul Tseng for valuable discussions during writing of this paper, and to Professor Jim Ritcey for recommending insightful readings.

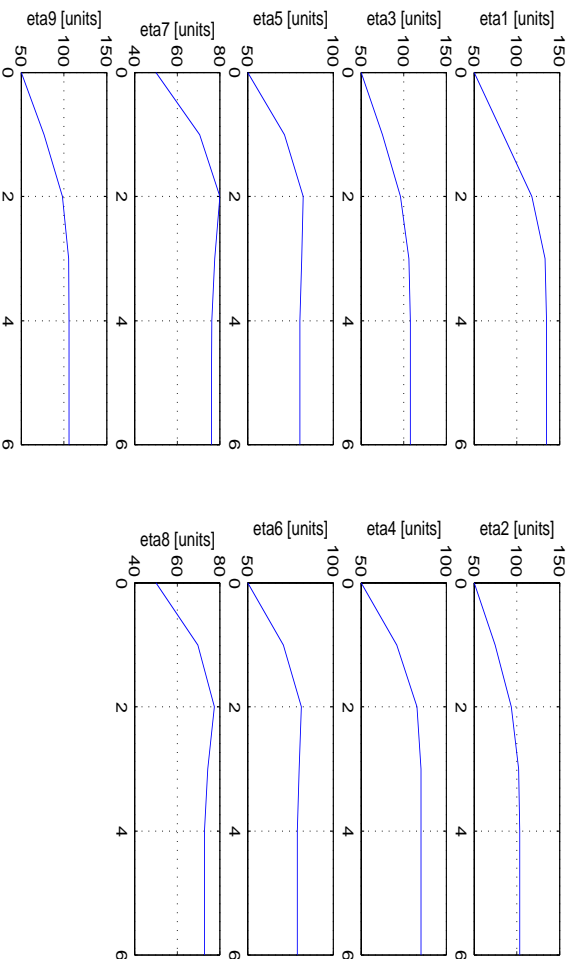


Figure 7. Convergence of the Distributed Dual Algorithm

References

1. Beamon, B. M.: 1998, 'Supply Chain Design and Analysis: Models and Methods'. *International Journal of Production Economics* **55**, 281–294.
2. Beamon, B. M.: 1999, 'Measuring supply chain performance'. *International Journal of Operations and Production Management* **19**(3), 275–292.
3. Bertsekas, D. P.: 1995, *Nonlinear Programming*. Belmont, MA: Athena Scientific.
4. Clark, A. J. and H. Scarf: 1960, 'Optimal Policies for a Multiechelon Inventory Problem'. *Management Science* **6**, 465–490.
5. Ganeshan, R., E. Jack, M. J. Magazine, and P. Stephens: 1999, 'A Taxonomic Review of Supply Chain Management Research'. In: S. Tayur, R. Ganeshan, and M. Magazine (eds.): *Quantitative Models for Supply Chain Management*. Boston, Massachusetts: Kluwer Academic Publishers.
6. Kelly, F.: 1997, 'Charging and rate control for elastic traffic'. *European Transactions on Telecommunications* **8**, 33–37.
7. Kelly, F., A. Mauloo, and D. Tan: 1998, 'Rate control for communication networks: shadow prices, proportional fairness and stability?'. *Journal of the Operational Research Society* **49**(3), 237–52.
8. Lee, H. and S. Whang: 1999, 'Decentralized Multi-Echelon Supply Chains: Incentives and Information'. *Management Science* **45**(5), 633–640.
9. Rockafellar, R. T.: 1998, *Fundamentals of Optimization, class notes*. Seattle, WA: University of Washington.

10. Shenker, S.: 1995, 'Fundamental Design Issues for the Future Internet'. *IEEE Journal on Selected Areas in Communications* **13**(7), 1176–1188.
11. Simchi-Levi, D., P. Kaminsky, and E. Simchi-Levi: 2000, *Designing and Managing the Supply Chain*. Boston: McGraw-Hill.

Table IV. Performance Results for Numerical Example 2.

| Algorithm | # of Iterations | # of Func. Evaluation | Final Values of η | Optimal Flows x |
|-----------------------|-----------------|-----------------------|--|---|
| DDA Algorithm | 6 | 6 | $\begin{bmatrix} 134.4928 \\ 103.3598 \\ 107.7218 \\ 85.2185 \\ 80.5025 \\ 78.8857 \\ 76.1043 \\ 72.6868 \\ 105.83997 \end{bmatrix}$ | $\begin{bmatrix} 388.0081651 \\ 366.3800650 \\ 219.8802743 \\ 221.0026696 \\ 219.9052858 \\ 222.2918349 \\ 215.0199347 \\ 294.2999998 \\ 289.0000008 \\ 318.6000004 \\ 380.7197244 \\ 384.0973281 \\ 380.7947160 \end{bmatrix}$ |
| Standard Newton | 6 | 6 | $\begin{bmatrix} 134.4928 \\ 103.3598 \\ 107.7218 \\ 85.2185 \\ 80.5025 \\ 78.8857 \\ 76.1043 \\ 72.6868 \\ 105.83997 \end{bmatrix}$ | $\begin{bmatrix} 388.0081651 \\ 366.3800650 \\ 219.8802743 \\ 221.0026696 \\ 219.9052858 \\ 222.2918349 \\ 215.0199347 \\ 294.2999998 \\ 289.0000008 \\ 318.6000004 \\ 380.7197244 \\ 384.0973281 \\ 380.7947160 \end{bmatrix}$ |
| Matlab <i>fmincon</i> | 23 | 310 | $\begin{bmatrix} 134.4919 \\ 103.3560 \\ 107.7304 \\ 85.2214 \\ 80.5079 \\ 78.8873 \\ 76.0921 \\ 72.6743 \\ 105.8379 \end{bmatrix}$ | $\begin{bmatrix} 388.0081651 \\ 366.3800650 \\ 219.8802743 \\ 221.0026696 \\ 219.9052858 \\ 222.2918349 \\ 215.0199347 \\ 294.2999998 \\ 289.0000008 \\ 318.6000004 \\ 380.7197244 \\ 384.0973281 \\ 380.7947160 \end{bmatrix}$ |

Table V. Convergence of the Distributed Dual Algorithm.

| Iter. 1 | Iter. 2 | Iter. 3 | Iter. 4 | Iter. 5 | Iter. 6 |
|---|--|---|---|---|---|
| $\begin{bmatrix} 83.7070 \\ 74.2732 \\ 75.0307 \\ 70.8005 \\ 71.3623 \\ 70.9171 \\ 70.5950 \\ 69.6452 \\ 76.7348 \end{bmatrix}$ | $\begin{bmatrix} 117.5343 \\ 93.4966 \\ 95.9107 \\ 82.8263 \\ 82.4232 \\ 81.2524 \\ 79.9830 \\ 77.4963 \\ 98.1880 \end{bmatrix}$ | $\begin{bmatrix} 132.8836 \\ 101.9444 \\ 105.9138 \\ 85.1234 \\ 81.5760 \\ 80.0005 \\ 77.5714 \\ 74.2382 \\ 105.3795 \end{bmatrix}$ | $\begin{bmatrix} 134.4814 \\ 103.3327 \\ 107.6850 \\ 85.2166 \\ 80.5290 \\ 78.9124 \\ 76.1405 \\ 72.7234 \\ 105.8398 \end{bmatrix}$ | $\begin{bmatrix} 134.4928 \\ 103.3598 \\ 107.7217 \\ 85.2185 \\ 80.5025 \\ 78.8857 \\ 76.1043 \\ 72.6868 \\ 105.8399 \end{bmatrix}$ | $\begin{bmatrix} 134.4928 \\ 103.3598 \\ 107.7218 \\ 85.2185 \\ 80.5025 \\ 78.8857 \\ 76.1043 \\ 72.6868 \\ 105.8399 \end{bmatrix}$ |

