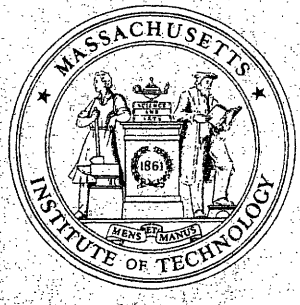
OPERATIONS RESEARCH CENTER

working paper



MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Progressive Equilibration Algorithms: The Case of Linear Transaction Costs

by

A. Eydeland and A. Nagurney

OR 198-89 June 1989

Progressive Equilibration Algorithms:

The Case of Linear Transaction Costs

Alexander Eydeland
Department of Mathematics and Statistics
University of Massachusetts
Amherst, Massachusetts 01003

and

Anna Nagurney
Department of General Business & Finance
School of Management
University of Massachusetts
Amherst, Massachusetts 01003

for correspondence and/or offprints:

Anna Nagurney
Transportation Systems Division
Room 1-174
MIT
Cambridge, Massachusetts 02139

September, 1988

revised January, 1989

to appear in Computer Science in Economics and Management

1. Introduction

The main issues in the study of large-scale market equilibrium problems are the development of models and the construction of accompanying algorithms for the efficient computation of the steady-state production, consumption, and trade patterns.

Recently, convergent progressive cyclic equilibration algorithms for the solution of classical market equilibrium problems with fixed transaction costs were introduced by Dafermos and Nagurney (1987). These algorithms - supply market (producer) equilibration, and demand market (consumer) equilibration - are of the relaxation-type, in that they attempt to equilibrate the whole system by successively equilibrating in a cyclic fashion each supply market or each demand market, until the entire system is equilibrated. They exploit the problem structure fully, in that each restricted equilibrium problem is solved *exactly* in closed form, rather than iteratively (Nagurney (1987a)). Subsequently, Nagurney (1988a) considered the case of linear increasing transaction costs, and extended these equilibration algorithms to this more general problem which, in turn, is isomorphic to the constrained matrix problem.

In this paper our principal goals are twofold: first, to introduce two new classes of progressive equilibration algorithms and, second, to provide a thorough theoretical analysis, including the rate of convergence and computational complexity results, for the entire family of progressive equilibration algorithms. The theoretical analysis provides assistance to both the theoretician and practitioner in selecting the appropriate algorithm for a specific application. In addition, it can aid other researchers involved in computational comparisons to determine whether or not their implementation of these algorithms is, indeed, "good". The selection of the best "algorithm", hence, need not be programmer, machine, or language dependent; but, instead, based on a sound theoretical framework.

Each of the two new classes of algorithms differs from the cyclic algorithms in the manner in which the next market to be equilibrated is selected. The first class uses the locally optimal market as the next market; whereas the second selects that market which is "good enough" in the sense that the objective function will be improved sufficiently. At the same time the new progressive equilibration algorithms retain the simplicity of the cyclic algorithms in that, at each step, the restricted equilibrium subproblem for a given market is solved exactly in closed form.

The models which can be handled by these algorithms fit into the following framework. A certain commodity is produced by "supply markets" (producers) and is consumed by "demand markets" (consumers); each supply market has a linear supply function which depends only upon the supply price at this market, and each demand market has a linear demand function which depends only on the demand price at this market. The commodity can be traded between any pair of supply and demand markets at a linear transaction cost. The problem is to find the supply and demand prices at each supply and demand market and the commodity shipments between all pairs of supply and demand markets that determine an equilibrium with the following property: a supply market trades with a demand market, provided that the sum of the supply price and the transaction cost equals the demand price. The market equilibrium models which can be solved by progressive equilibration, hence, need only to have a separable quadratic programming formulation subject to supply and demand constraints of the transportation-type and nonnegativity assumptions on the commodity shipments. These include, in particular, virtually the entire spectrum of classical spatial price equilibrium models, including distinct supply and demand price models, as well as, the simpler, single-price net import and net export models (see, e.g., Samuelson (1952), Takayama and Judge (1971), Dafermos and Nagurney (1987), Nagurney (1987c), and the references, therein.)

This contribution is also applicable to more general multicommodity market equilibrium models, since they can be formulated as variational inequality problems and then solved as sequences of classical models, using the progressive equilibration algorithms at each step (see, e.g., Florian and Los (1982), Dafermos (1986), Nagurney and Kim (1988)).

In Section 2 we briefly review the classical market equilibrium model with distinct supply and demand prices. In Section 3 we introduce the two new classes of progressive equilibration algorithms - equilibration algorithms using locally optimal markets, and those which are "good enough", and we review the cyclic equilibration algorithms.

In Section 4 we describe theoretical results as to the relative performance of the algorithms via the introduction of a theoretical algorithm. In Section 5 we then discuss implementation issues and provide suggestions and guidance. In Section 6 we present computational results to illuminate and illustrate the theoretical analysis. We conclude with a discussion in Section 7.

2. The Market Equilibrium Model with Distinct Supply and Demand Prices

In this Section we briefly review the single commodity market equilibrium model with distinct supply and demand price functions considered recently in Nagurney (1988a). This model allows for distinct supply and demand prices at equilibrium at a market and permits linear, rather than fixed, transaction costs between markets. Hence, the implication is that the producers' adjustment process is independent from that of the consumers. Single price market equilibrium models and variants, including the simpler net-import model and the alternative net-export model are special cases of this model.

In particular, we consider a market equilibrium problem in which a certain commodity is produced by m supply markets (producers) and is consumed by n demand markets (consumers). The typical supply market (producer) will be denoted by i, and the typical demand market (consumer) by j.

We let s_i denote the nonnegative commodity output produced by supply market i, and we let d_j denote the nonnegative demand for the commodity by demand market j. The nonnegative shipment from supply market i to j will be denoted by X_{ij}.

The following feasibility conditions must hold:

$$s_i = \sum_j X_{ij}, \quad i = 1, \ldots, m \tag{1}$$

$$d_j = \sum_i X_{ij}, \quad j = 1, \ldots, n \tag{2}$$

$$\sum_i s_i = \sum_j d_j = \sum_i \sum_j X_{ij}. \tag{3}$$

We associate with each supply market i a nonnegative supply price π_i and with each demand market j a nonnegative demand price ρ_j. Each unit of the commodity which is traded between supply market i and demand market j is surcharged with a unit nonnegative transaction cost c_{ij}, which we assume to be a linear function

$$\hat{c}_{ij}(X_{ij}) = g_{ij}X_{ij} + h_{ij}. \tag{4}$$

We assume also that the quantity supply function $\hat{s}_i(\pi_i)$ is a linear function

$$\hat{s}_i(\pi_i) = \gamma_i \pi_i + \theta_i \tag{5}$$

with inverse

$$\hat{\pi}_i(s_i) = \eta_i s_i + \psi_i \tag{6}$$

where $\eta_i = \frac{1}{\gamma_i}$ and $\psi_i = -\frac{\theta_i}{\gamma_i}$. Also, we assume that the demand function $\hat{d}_j(\rho_j)$ is a linear function

$$\hat{d}_j(\rho_j) = \alpha_j - \beta_j \rho_j \tag{7}$$

with inverse

$$\hat{\rho}_j(d_j) = \lambda_j - \omega_j d_j \tag{8}$$

where $\lambda_j = \frac{\alpha_j}{\beta_j}$ and $\omega_j = \frac{1}{\beta_j}$.

Assuming perfect competition, we seek the market equilibrium with the following property which must hold for all pairs of supply and demand markets. A supply market trades with a demand market provided that the sum of the supply price and the transaction cost equals the demand price. Mathematically, this state is characterized by the following equilibrium conditions which must hold for all supply and demand market pairs (i, j):

$$\pi_i + c_{ij} \begin{cases} = \rho_j, & \text{if } X_{ij} > 0 \\ \geq \rho_j, & \text{if } X_{ij} = 0. \end{cases} \tag{9}$$

As is well-known (see Takayama and Judge (1971)), the equilibrium conditions are equivalent to the solution of a minimization problem in $m \times n$ variables, which, in view of the functions (6), (8), and (4) is a quadratic programming problem:

$$\text{Min} \quad \Phi([X_{ij}]) =$$

$$\text{Min} \sum_{i=1}^{m} \left(1/2\eta_i (\sum_j X_{ij})^2 + \psi_i \sum_j X_{ij} \right) + \sum_{i=1}^{m} \sum_{j=1}^{n} 1/2 g_{ij} X_{ij}^2 + h_{ij} X_{ij} +$$

$$\sum_{j=1}^{n} \left(1/2\omega_j (\sum_i X_{ij})^2 - \lambda_j \sum_i X_{ij} \right) \tag{10}$$

$$\text{subject to } X_{ij} \geq 0, i = 1, \ldots, m; j = 1, \ldots, n, \tag{11}$$

where the constraints (1) and (2) have been incorporated directly into the objective function.

Assuming that $\eta_i, \psi_i, \lambda_j, \omega_j$ and g_{ij}, h_{ij} are > 0, the problem (10) is a strictly convex programming problem with unique solution $[X_{ij}]$. This market equilibrium model is

isomorphic to the constrained matrix problem in which row and column total estimates, s_i and d_j and the matrix estimates $[X_{ij}]$ are to be computed (For details, see Nagurney (1988a)). For the above model, Nagurney (1987b) first proposed iterative equilibration algorithms and then adapted the Dafermos-Nagurney (1987) cyclic equilibration algorithms in Nagurney (1988a), which had originally been proposed for the model with fixed transaction costs, that is, for the case where $c_{ij} = h_{ij}$, for all $i = 1, \ldots, m; j = 1, \ldots, n$.

Models in which a single price at equilibrium for each market in the economy is expected to prevail are termed single-price models and include both net-import and net-export models, in which the number of producers must be equal to the number of consumers. These models can also be formulated as quadratic programming problems under the assumption of linear supply and demand price functions and linear transaction costs and solved via the algorithms we describe in the next Section. In these models, hence, the consumers' demand quantity adjustment process is dependent on that of the suppliers. Consumers and producers react to a uniform market price, that at which the supply price is equal to the demand price in each market. For specialized algorithms for single-price models, see Asmuth, Eaves, and Peterson (1979), Glassey (1978), Jones, Saigal, and Schneider (1986), Güder (1987), and Nagurney (1988b).

Generalizations of the above single commodity model include multicommodity market equilibrium models for which equivalent optimization formulations no longer exist (see, e.g., Florian and Los (1982), Dafermos (1986), Nagurney (1987a, b), Nagurney and Kim (1988)). Such models can then be formulated and solved as variational inequality problems using variational inequality algorithms. Such algorithms require the efficient solution of the embedded mathematical programming problem encountered at each step, which, in turn, can take on the form of problem (10). Hence, the ability to solve large-scale multicommodity market equilibrium problems depends on the development of efficient algorithms for single commodity problems. Applications of multicommodity models include agricultural and energy markets, international trade, and financial markets (see, e.g., Judge and Takayam (1973) and Nagurney and Kim (1988)).

3. Progressive Equilibration Algorithms

In this Section, we describe a family of *progressive equilibration algorithms* for computing an equilibrium for the market model described in Section 2. The algorithms are distinguished, firstly, by whether they equilibrate on the demand or on the supply side. Hence, they are categorized into demand equilibration and supply equilibration algorithms. Within a category, we present three distinct equilibration algorithms, which differ in the manner in which each selects the next market to be equilibrated. The first equilibration algorithm in each category selects that market which is locally optimal; the second – that market which is "good enough", and the third – simply the next market in the cycle. The cyclic algorithms were introduced by Dafermos and Nagurney (1987) for the solution of market equilibrium models with distinct supply and demand price functions and with fixed unit transaction costs.

These progressive equilibration algorithms are of the *relaxation type*, i.e., they attempt to equilibrate the whole system by equilibrating successively each demand market (consumer), or each supply market (producer). The noteworthy feature of the algorithms is that, due to the special structure of the problem, the restricted equilibrium for each demand market (or supply market) can be obtained explicitly in closed form. We first present the demand market (consumer) equilibration algorithms and then the supply market (producer) equilibration algorithms. These algorithms can also be easily adapted to solve the simpler single-price net-import and net-export models.

The Demand Market (Consumer) Equilibration Algorithms

The first category of algorithms is comprised of progressive demand market (consumer) equilibration methods; each computes a sequence of feasible shipments $[X_{ij}^0], [X_{ij}^1], \ldots$, which converges to an equilibrium by the following procedure:

Step 0: Start with an arbitrary feasible shipment $[X_{ij}^0]$.

Step t $(t = 1, 2, ...)$: Starting from the feasible shipment $[X_{ij}^{t-1}]$ (computed at step $t-1$), construct a new feasible shipment $[X_{ij}^t]$ by modifying $X_{il}^{t-1}, i = 1, \ldots, m$ in such a way that the equilibrium conditions (9) are satisfied for the particular demand market (consumer) l, where l is selected in a specific manner, depending upon the particular equilibration algorithm, as follows.

Selection of Market l

(I).Demand Equilibration Algorithm Using Locally Optimal Markets

Let $l = \max_j \sum_{i=1}^m \mathrm{P}_{ij}^2$, where P_{ij} is the i-th element of the projection of $\frac{\partial \Phi}{\partial X_j}$, $\mathrm{P}\frac{\partial \Phi}{\partial X_{ij}}$, which we define below and where the vector $X_j \equiv (X_{1j}, \ldots, X_{mj})^T$.

(II). Good Enough Demand Equilibration Algorithm

Let $l = j : \mathrm{P}\| \frac{\partial \Phi}{\partial X_j} \|^2 \geq 1/nC_{lold}^2$, where

$$C_{lold} \equiv \frac{\Phi(X^t) - \Phi(X^{t-1})}{\|X_l^t - X_l^{t-1}\|} = \frac{\Phi(X^t) - \Phi(X^{t-1})}{\sqrt{\sum_i (X_{il}^t - X_{il}^{t-1})^2}}.$$

If, for all j, no such l is found, let $l = j|\max \|\mathrm{P}\frac{\partial \Phi}{\partial X_j}\|^2$

(III). Cyclic Demand Equilibration Algorithm (Dafermos and Nagurney (1987), Nagurney (1988a, b))

Let $l = (t-1)(\mathrm{mod} n) + 1$.

Each of the restricted equilibrium problems at Step t is computed exactly. The restricted equilibrium which is computed at each step for each of the three equilibration algorithms above may be attained in closed form as follows via a procedure we describe now and developed in Dafermos and Nagurney (1987). We note, however, that there may be other procedures which can also solve the restricted equilibrium problem exactly.

Let $[X'_{ij}]$ denote the given feasible shipment which has to be modified into a new feasible shipment $[X_{ij}]$ so as to attain equilibration with respect to demand market (consumer) l. This new feasible shipment $[X_{ij}]$ will be obtained by preserving all shipments to all demand markets (consumers) other than l, i.e.,

$$X_{ij} = X'_{ij}, \quad j \neq l, \quad i = 1, \ldots, m \tag{12}$$

and modifying only $X'_{il}, i = 1, \ldots, m$, so as to satisfy market equilibrium conditions (9) for demand market (consumer) l which take for the market equilibrium model described in Section 2 the following general form:

$$\hat{g}_i X_{il} + \hat{h}_{il} \begin{cases} = -\hat{\omega}_l d_l + \hat{\lambda}_l, & \text{if } X_{il} > 0 \\ \geq -\hat{\omega}_l d_l + \hat{\lambda}_l, & \text{if } X_{il} = 0, \end{cases} \tag{13}$$

$$i = 1, \ldots, m$$

where d_l satisfies (2). In the case of the market equilibrium model with distinct supply and demand price functions, where the supply price functions are given by (6), the demand price functions given by (8), and the transaction cost functions given by (4),

$$\hat{g}_i = \eta_i + g_{il} \quad \hat{h}_{il} = \eta_i(\sum_{j \neq l} X'_{ij}) + \psi_i + h_{il} \tag{14}$$

$$-\hat{\omega}_l = -\omega_l \quad \hat{\lambda}_l = \lambda_l. \tag{15}$$

For fixed transaction costs, the above relationships still apply, with the proviso that the g_{il}'s are set equal to zero.

In view of the above, we note that

$$\mathrm{P}\frac{\partial \Phi}{\partial X_{ij}} \begin{cases} = \hat{g}_i X_{ij} + \hat{h}_{ij} + \hat{\omega}_j d_j - \hat{\lambda}_j, & \text{if } X_{ij} > 0 \\ = \min\{0, \hat{g}_i X_{ij} + \hat{h}_{ij} + \hat{\omega}_j d_j - \hat{\lambda}_j\}, & \text{if } X_{ij} = 0 \end{cases} \tag{16}$$

As noted in the Introduction, these relaxation/equilibration algorithms require that the restricted equilibrium problem (13) encountered at each step be solved exactly.

We now describe such an exact procedure, developed by Dafermos and Nagurney (1987), and adapted by Nagurney (1988a) for the market equilibrium model with linear transaction costs and by Nagurney (1988b) for the net import model. We may, without loss of generality, rewrite (13) as follows:

$$\hat{g}_1 X_{1l} + \hat{h}_{1l} = \hat{g}_2 X_{2l} + \hat{h}_{2l} = \ldots = \hat{g}_s X_{sl} + \hat{h}_{sl} = \rho_l = \hat{\lambda}_l - \hat{\omega}_l y_l$$

$$\leq \hat{g}_{s+1} X_{s+1,l} + \hat{h}_{s+1,l} \leq \ldots \leq \hat{g}_m X_{ml} + \hat{h}_{ml}, \tag{17}$$

$$X_{il} > 0, \quad i = 1, \ldots, s \tag{18}$$

$$X_{il} = 0, \quad i = s+1, \ldots, m.$$

Provided the critical s is known, X'_{il} may be calculated through

$$X'_{il} = \frac{\rho_l - \hat{h}_{il}}{\hat{g}_i}, \quad i = 1, \ldots, s$$

$$X'_{il} = 0, \quad i = s+1, \ldots, m. \tag{19}$$

We now give a procedure for the calculation of the critical s.

(i) Sort the $\hat{h}_{il}$'s, $i = 1, \ldots, m$ in nondescending order and relabel the $\hat{h}_{il}$'s accordingly.

(ii) If $\hat{\lambda} < \hat{h}_{1l}$, stop; $s = 0$. Otherwise, set $q = 1$ and go to (iii).

(iii) Compute

$$\rho_l^q = \frac{\sum_{i=1}^{q} \hat{h}_{il}/\hat{g}_i + \beta_l}{\sum_{i=1}^{q} 1/\hat{g}_i + \alpha_l}. \tag{20}$$

If $\hat{h}_{ql} < \rho_l^q \leq \hat{h}_{q+1,l}$, then stop; $s = q$. Otherwise, set $q = q + 1$, and go to (iii).

The Supply Market (Producer) Equilibration Algorithms

The second category of algorithms consists of progressive supply market (producer) equilibration methods and is, of course, the "dual" of the demand algorithms; each computes a sequence of feasible shipments $[X_{ij}^0], [X_{ij}^1], \ldots$, which converges to an equilibrium by the following procedure:

Step 0: Start with an arbitrary feasible pattern $[X_{ij}^0]$.

Step $t (t = 1, 2, \ldots)$: Starting from the feasible shipment $[X_{ij}^{t-1}]$ (computed at step $t-1$), construct a new feasible shipment $[X_{ij}^t]$ by modifying $X_{kj}^{t-1}, j = 1, \ldots, n$ in such a way that the equilibrium conditions (9) are satisfied for the particular supply market (producer) k, where k is again selected in a specific manner depending upon the particular equilibration algorithm.

Selection of market k:

(I). Supply Equilibration Algorithm Using Locally Optimal Markets

Let $k = \max_i \sum_{j=1}^{n} \mathrm{P}_{ij}^2$, where P_{ij} is the j-th element of the projection of $\frac{\partial \Phi}{\partial X_i}$, $\mathrm{P}\frac{\partial \Phi}{\partial X_{ij}}$, which is defined below, where now, for convenience, we let $X_i \equiv (X_{i1}, \ldots, X_{in})^T$.

(II). Good Enough Supply Equilibration Algorithm

Let $k = i : \|\mathrm{P}\frac{\partial \Phi}{\partial X_i}\|^2 \geq 1/mC_{k^{old}}^2$, where

$$C_{k^{old}} \equiv \frac{\Phi(X^t) - \Phi(X^{t-1})}{\|X_k^t - X_k^{t-1}\|} = \frac{\Phi(X^t) - \Phi(X^{t-1})}{\sqrt{\sum_j (X_{kj}^t - X_{kj}^{t-1})^2}}.$$

If, for all i, no such k is found, let $k = i|\max\|\mathrm{P}\frac{\partial \Phi}{\partial X_i}\|^2$.

(III). Cyclic Supply Equilibration Algorithm (Dafermos and Nagurney (1987), Nagurney (1988b))

Let $k = (t-1)(\mathrm{mod} m) + 1$.

The restricted market equilibrium which is computed at each step of each of the above three algorithms again should be computed exactly. Although alternative exact procedures may be available, we present only one, which was developed by Dafermos and Nagurney (1987).

Let $[X'_{ij}]$ denote the given feasible shipment which has to be modified into a new feasible shipment $[X_{ij}]$ so as to attain equilibration with respect to supply market (producer) k. The new feasible shipment $[X_{ij}]$ will be obtained by preserving all shipments to all supply markets (producers) other than k; i.e.,

$$X_{ij} = X'_{ij}, \quad i \neq k, \quad j = 1, \ldots, n \tag{21}$$

and modifying only $X'_{kj}, j = 1, \ldots, n$, so as to satisfy conditions (9) for supply market (producer) k which for the model described in Section 2 take now the general form

$$\hat{\eta}_k s_k + \hat{\psi}_k \begin{cases} = -\bar{\hat{g}}_j X_{kj} + \bar{\hat{h}}_{kj}, & \text{if } X_{kj} > 0 \\ \geq -\bar{\hat{g}}_j X_{kj} + \bar{\hat{h}}_{kj}, & \text{if } X_{kj} = 0 \end{cases} \tag{22}$$

$$j = 1, \ldots, n.$$

where s_k satisfies (1).

In the case of the market equilibrium model with distinct supply and demand prices with the supply price functions given by (6), the demand price functions given by (8), and the transaction cost functions given by (1)

$$\hat{\eta}_k = \eta_k \quad \hat{\psi}_k = \psi_k \tag{23}$$

$$-\bar{\hat{g}}_j = -\omega_j - g_{kj} \quad \bar{\hat{h}}_{kj} = -\omega_j(\sum_{i \neq k} X'_{ij}) + \lambda_j - h_{kj}. \tag{24}$$

In view of the above, we note that

$$\Pr \frac{\partial \Phi}{\partial X_{ij}} \begin{cases} = \hat{\eta}_i s_i + \hat{\psi}_i + \bar{\hat{g}}_i X_{ij} - \bar{\hat{h}}_{ij}, & \text{if } X_{ij} > 0 \\ = \min\{0, \hat{\eta}_i s_i + \hat{\psi}_i + \bar{\hat{g}}_i X_{ij} - \bar{\hat{h}}_{ij}\}, & \text{if } X_{ij} = 0 \end{cases} \tag{25}.$$

We now describe a procedure for the exact solution of (22). We may without loss of generality, rewrite (22) as

$$\bar{\hat{g}}_1 X_{k1} + \bar{\hat{h}}_{k1} = \bar{\hat{g}}_2 X_{k2} + \bar{\hat{h}}_{k2} = \ldots = \bar{\hat{g}}_{s'} X_{ks'} + \bar{\hat{h}}_{ks'} = -\pi_k = -\hat{\eta}_k x_k - \hat{\psi}_k$$

$$\leq \bar{\hat{g}}_{s'+1} X_{k,s'+1} + \bar{\hat{h}}_{k,s'+1} \leq \ldots \leq \bar{\hat{g}}_n X_{kn} + \bar{\hat{h}}_{kn} \tag{26}$$

where

$$X_{kj} > 0; \quad j = 1, \ldots, s' \tag{27}$$

$$X_{kj} = 0; \quad j = s'+1, \ldots, n.$$

Having introduced the necessary notation, we now state the procedure for the determination of the critical s'.

In particular, the steps are:

(i) Sort the $-\bar{\hat{h}}_{kj}$'s, $j = 1, \ldots, n$ in nondescending order and relabel the $\bar{\hat{h}}_{kj}$'s accordingly.

(ii) If $-\hat{\psi}_k \leq -\bar{\hat{h}}_{kj}$, stop; $s' = 0$. Otherwise, set $q = 1$ and go to (iii).

(iii) Compute

$$\pi_k^q = \frac{\sum_{j=1}^{q} -\bar{\hat{h}}_{kj}/\bar{\hat{g}}_j + \theta_k}{\sum_{j=1}^{q} 1/\bar{\hat{g}}_j + \gamma_k}. \tag{28}$$

if $-\bar{\hat{h}}_{kq} < \pi_k^q \leq -\bar{\hat{h}}_{k,q+1}$, then stop; $s' = q$. Otherwise, set $q = q + 1$ and go to (iii).

X'_{kj} may then be calculated as follows:

$$X'_{kj} = \frac{\pi_k - \bar{\hat{h}}_{kj}}{\bar{\hat{g}}_j}, \quad j = 1, \ldots, s' \tag{29}$$

$$X'_{kj} = 0, \quad j = s'+1, \ldots, n.$$

We now prove convergence of the demand market (consumer) equilibration algorithms. Convergence of the supply market (producer) equilibration algorithms can be shown using similar arguments.

Convergence of the Demand Market (Consumer) Equilibration Algorithms

The algorithms have the following features:

(a) The sequence $\{[X_{ij}^0], [X_{ij}^1], [X_{ij}^2], \ldots\}$ contains convergent subsequences since all shipments $[X_{ij}]$ remain in a bounded set of R^{mn}.

(b) Recall that the equilibrium conditions (9) are equivalent to the solution of the convex programming problem:

$$\text{Min } \Phi([X_{ij}]) =$$

$$\text{Min} \sum_i \int_0^{\sum_j X_{ij}} \hat{\pi}_i(x)dx + \sum_{ij} \int_0^{X_{ij}} \hat{c}_{ij}(z)dz - \sum_j \int_0^{\sum_i X_{ij}} \hat{\rho}_j(y)dy \tag{30}$$

$$\text{subject to} \quad X_{ij} \geq 0, \quad i = 1, \ldots, m; \quad j = 1, \ldots, n. \tag{31}$$

As is then familiar and easy to verify, starting from the feasible shipment $[X_{ij}^{t-1}]$ and determining the feasible shipment $[X_{ij}^{t}]$ by modifying $X_{il}^{t-1}, i = 1, \ldots, m$ in such a way that the equilibrium conditions (9) are satisfied for the particular demand market (consumer) l, is equivalent to finding a minimum of (10) over a restricted feasible set, i.e.

$$\text{Min } \Phi([X_{ij}^{t}]) =$$

$$\text{Min} \left\{ \sum_i \int_0^{X_{il}^t + \sum_{j \neq l} X_{ij}^{t-1}} \hat{\pi}_i(x)dx + \int_0^{X_{il}^t} \hat{c}_{il}(z)dz - \int_0^{\sum_i X_{il}^t} \hat{p}_l(y)dy \right\} \tag{32}$$

$$\text{subject to} \quad X_{il}^{t} \geq 0, \quad i = 1, \ldots, m. \tag{33}$$

Hence, it follows that $\Phi([X_{ij}^{t}]) \leq \Phi([X_{ij}^{t-1}])$.

(c) If throughout a cycle of n subsequent iterations Φ remains constant, then equilibrium conditions (9) are satisfied for all supply markets and all demand markets.

Consequently, it follows from standard theory (see, e.g., Zangwill (1969)) that every convergent subsequence of the sequence $\{[X_{ij}^{0}], [X_{ij}^{1}], \ldots\}$ converges to a solution of equilibrium conditions (9).

4. The Theoretical Algorithm

In this Section we provide a theoretical analysis of the progressive equilibration algorithms in the strictly convex case. In order to address the analysis, we first prove the convergence of the following theoretical algorithm, which is demand-based. Similar arguments follow for its supply counterpart. We discuss the rate of convergence and the complexity. We then relate this theoretical algorithm to algorithms I, II, and II of Section 3. Without any loss of generality, we consider the problem with distinct supply and demand prices, with objective function given by (10).

Algorithm:

Given X^0. For $t = 0, 1, 2, \ldots$, and $l \in \{1, \ldots, n\}$

$$Y_l^{t+1} = \min_{Y_l \geq 0} \Phi(X_1^t, \ldots, Y_l, \ldots, X_n^t) \tag{34a}$$

$$X_{il}^{t+1} = \max\{\delta^{t+1}, Y_{il}^{t+1}\} \quad \text{if} \quad Y_{il}^{t+1} \neq 0, X_{il}^{t+1} = 0, \quad \text{if} \quad Y_{il}^{t+1} = 0 \tag{34b}$$

$$X_j^{t+1} = X_j^t, \quad j \neq l. \tag{34c}$$

(The monotone decreasing to zero sequence of numbers $\delta^t, t = 0, 1, \ldots$, will be determined later.)

$$l = \max_{1 \leq j \leq n} \|P^m(\frac{\partial \Phi}{\partial X_j}(X^t))\|^2 \tag{34d}$$

where the vector $\frac{\partial \Phi}{\partial X_j} = (\frac{\partial \Phi}{\partial X_{1j}}, \ldots, \frac{\partial \Phi}{\partial X_{mj}})^T$ and P^m is a projection onto the positive orthant R_+^m, where

$$P^m(\frac{\partial \Phi}{\partial X_j}(X^t)) \in R_+^m$$

$$P^m(\frac{\partial \Phi}{\partial X_{ij}}(X^t)) = \begin{cases} \frac{\partial \Phi}{\partial X_{ij}} & \text{if } X_{ij} > 0 \\ \frac{\partial \Phi}{\partial X_{ij}} & \text{if } X_{ij} = 0 \text{ and } \frac{\partial \Phi}{\partial X_{ij}} \leq 0 \\ 0 & \text{if } X_{ij} = 0 \text{ and } \frac{\partial \Phi}{\partial X_{ij}} > 0. \end{cases} \tag{35}$$

We begin by first proving the following auxiliary lemma:

Lemma 1: If $X^* \geq 0$ is the point of minimum of the function $\Phi(X)$ in R_+^{mn} then

$$\Phi(Z) - \Phi(X^*) \leq \frac{1}{\lambda_{min}} \|P^{mn}(\nabla \Phi(Z))\|^2 \tag{36}$$

where $Z \in R_+^{mn}$; λ_{min} is the smallest eigenvalue of the constant matrix $D^2\Phi = \{\frac{\partial^2 \Phi}{\partial X_{ij} \partial X_{il}}\}$.

Proof: By convexity

$$\Phi(Z) - \Phi(X^*) \leq \nabla\Phi(Z)(Z - X^*). \tag{37}$$

We notice now that

$$\nabla\Phi(Z)(Z - X^*) \leq P^{mn}(\nabla\Phi(Z))(Z - X^*). \tag{38}$$

Indeed, if $Z_{ij} > 0$ or $Z_{ij} = 0$ and $\frac{\partial\Phi}{\partial X_{ij}}(Z) \leq 0$, then by definition (35)

$$\frac{\partial\Phi}{\partial X_{ij}}(Z)(Z_{ij} - X^*_{ij}) = P(\frac{\partial\Phi}{\partial X_{ij}}(Z))(Z_{ij} - X^*_{ij}).$$

If $Z_{ij} = 0$ and $\frac{\partial\Phi}{\partial X_{ij}}(Z) > 0$, then

$$\frac{\partial\Phi}{\partial X_{ij}}(Z)(0 - X^*_{ij}) \leq 0 = P^{mn}(\frac{\partial\Phi}{\partial X_{ij}}(Z))(Z_{ij} - X^*_{ij}).$$

Thus, inequality (38) holds. Combining it with (37), we obtain via the Cauchy-Schwarz inequality

$$\Phi(Z) - \Phi(X^*) \leq \|P^{mn}(\nabla\Phi(Z))\|\|Z - X^*\|. \tag{39}$$

Now, we use the relations

$$\nabla\Phi(Z)(Z-X^*) \geq \nabla\Phi(Z)(Z-X^*) - \nabla\Phi(X^*)(Z-X^*) = < (Z-X^*), D^2\Phi(Z-X^*) > \tag{40}$$

$$\geq \lambda_{min}\|Z - X^*\|^2$$

and (38) to obtain the estimate

$$\frac{1}{\lambda_{min}}\|P^{mn}(\nabla\Phi(Z))\| \geq \|Z - X^*\|.$$

This estimate, combined with (39), yields the desired inequality (36).

We now can evaluate the rate of convergence of the algorithm.

Theorem 1:

If $\delta^t = t^{1-\epsilon}$ where ϵ is a small number, $0 < \epsilon < 1$, then

$$\Phi(X^t) - \Phi(X^*) \leq \frac{C}{n}\frac{1}{t^{1-\epsilon}}, t = 1, \ldots \tag{41a}$$

where C is a constant.

Moreover, if the set I^* of indices (i,j) such that $X^*_{ij} = 0$, $\frac{\partial \Phi}{\partial X_{ij}} = 0$ is empty,

$$\Phi(X^t) - \Phi(X^*) \leq C_1 e^{-\frac{c_2}{n}t} \tag{41b}$$

(Observe that in the case (41b) the algorithm is identical to Demand Algorithm I described in Section 3. This is known as the strict complementarity case.)

Proof:

At the point X^t we introduce the vector

$$\eta_l = -P^m(\frac{\partial \Phi}{\partial X_l}(X^t))/\|P^m(\frac{\partial \Phi}{\partial X_l}(X^t))\|.$$

By (34a)

$$\Phi(X_1^t, \ldots, Y_l^{t+1}, \ldots, X_n^t) \leq \Phi(X_1^t, \ldots, X_l^t + \tau\eta_l, \ldots, X_n^t) \tag{42}$$

for all τ such that $0 < \tau < \delta^t$. (Note that this feasible range of τ follows from (34b) and from the fact that $\|\eta_l\| = 1$.) From (42), (34b), (34c), it follows that

$$\Phi(X_1^{t+1}, \ldots, X_l^{t+1}, \ldots, X_n^{t+1}) \leq \Phi(X_1^t, \ldots, X_l^t + \tau\eta_l, \ldots, X_n^t) + \lambda_{max}\delta^{t+1^2} \tag{43}$$

where λ_{max} is the maximum eigenvalue of $D^2\Phi$.

Hence, using Taylor's expansion

$$\Phi(X^{t+1}) \leq \Phi(X^t) + \tau\nabla\Phi(X^t)\eta_l + \tau^2\lambda_{max} + \delta^{t+1^2}\lambda_{max} \tag{44}$$

$$= \Phi(X^t) - \tau\|P^m(\frac{\partial \Phi}{\partial X_l}(X^t))\| + (\tau^2 + \delta^{t+1^2})\lambda_{max}.$$

By (34d), $\|P^m(\frac{\partial \Phi}{\partial X_l}(X^t))\| \leq \frac{1}{\sqrt{n}}\|P^{mn}(\nabla\Phi(X^t))\|$. Therefore, by (44) and (36),

$$\Phi(X^{t+1}) \leq \Phi(X^t) - \frac{\tau}{\sqrt{n}}\sqrt{\lambda_{min}}\sqrt{\Phi(X^t) - \Phi(X^*)} + (\tau^2 + \delta^{t+1^2})\lambda_{max}. \tag{45}$$

Let $\mu^t = \Phi(X^t) - \Phi(X^*)$. Then (45) implies

$$\mu^{t+1} \leq \mu^t - \frac{\sqrt{\lambda_{min}}}{\sqrt{n}}\tau\sqrt{\mu^t} + (\tau^2 + \delta^{t+1^2})\lambda_{max} \tag{46}$$

Introducing $\alpha = \frac{\sqrt{\lambda_{min}}}{\sqrt{n}}$ and $\beta = min(\frac{\alpha}{2\lambda_{max}}, \frac{1}{\sqrt{\mu^0}})$, and substituting $\tau = \delta^t \beta \sqrt{\mu^t}$ into (46) we obtain

$$\mu^{t+1} \leq \mu^t (1 - \frac{\alpha\beta}{2}\delta^t) + \delta^{t^2}\lambda_{max}. \tag{47}$$

(In (47) we used the fact that $\delta^t \geq \delta^{t+1}, \beta\sqrt{\mu^t} \leq 1$ and $\frac{\alpha\beta}{2} \geq \beta^2\lambda_{max}$.) By the standard argument one can show that if $\delta = \frac{d}{t}$, where $d = \frac{1}{min\{\frac{1}{4nK}, \frac{\alpha}{2\sqrt{\mu^0}}\}}$, and K is the condition number of $D^2\Phi$, then

$$\mu^{t+1} \leq C^1 \frac{lnt}{t} \tag{48}$$

with C^1 depending on $\lambda_{min}, \lambda_{max}$, and n. For obvious pratical purposes, to avoid the estimates of the condition number K, we suggest to use a sequence $\delta^t = \frac{1}{t^{1-\epsilon}}, \epsilon > 0$. This choice yields the following estimate for μ^{t+1}

$$\mu^{t+1} \leq \frac{C}{n}\frac{1}{t^{1-\epsilon}} \tag{48a}$$

with C depending *only* on $\lambda_{min}, \lambda_{max}$, and n. This estimate is only slightly worse than (48) but substantially easier to work with. Thus, (41a) is proved. In order to prove (41b) we notice that if I^* is empty, starting from some $T > 0$, all $X_l^t = Y_l^t$. Hence, without changing anything, we can set $\delta^t = 0$ for $t = T$. Then for $t > T$ (46) immediately implies (41b) if $\tau = \frac{\sqrt{\mu^t}\sqrt{\lambda_{min}}}{2\lambda_{max}\sqrt{n}}$.

In particular, if, indeed, the set I^* is empty, then

$$\mu^t = \Phi(X^t) - \Phi(X^*) \leq (\Phi(X^0) - \Phi(X^*))e^{\frac{-\lambda_{min}t}{4\lambda_{max}n}}.$$

Hence, if for some fixed $\epsilon > 0$, we use as the stopping criterion $\mu^t < \epsilon$, we obtain the following estimate for the number of iterations t:

$$t = [\frac{-4\lambda_{max}}{\lambda_{min}} ln \frac{\epsilon}{\mu^0}]n. \tag{49}$$

If, on the other hand, $\|X^t - X^*\|$ is used as the stopping criterion, we have that:

$$t = [\frac{-4\lambda_{max}}{\lambda_{min}} ln(\frac{\epsilon}{\lambda_{min}\mu^0})]n. \tag{50}$$

Observe that, irregardless of the stopping criterion, the number of iterations is proportional to n, i.e.,

$$t = Cn,$$

where C varies depending upon the criterion.

Moreover, we can infer from both (49) and (50), that as ϵ gets smaller, the number of iterations grows additively, rather than multiplicatively.

We now utilize this result to analyze the total number of operations required by the algorithm.

Once a stopping criterion is chosen we can construct the total number of operations required by the algorithm (by operations we mean elementary arithmetical operations, assuming that they take equal time to be performed in the computer.)

The total number of operations per iteration consists of:

(1). The number of operations for the choice of l in (34) (partially, these computations can be used for the stopping criterion based on (13)):

$$N_1 = 9mn + 3m + n,$$

where

$$7mn \text{ (operations)}$$

for computing all $\frac{\partial \Phi}{\partial X_{ij}}$ (provided that all $\hat{g}_i, \hat{\lambda}_j$ and $\hat{\omega}_j$ from (14)) are stored in advance),

$$2mn(\text{ operations})$$

for computing $\|\frac{\partial \Phi}{\partial X_{ij}}\|^2$,

$$n \text{ (operations)}$$

for computing $\max_j \|\frac{\partial \Phi}{\partial X_j}\|^2$ (where we assume the previous s_i's and d_j's have been stored.), and

$$3m \text{ operations}$$

to update d_l and the s_i's.

(2). The number of operations necessary for sorting all h'_{il}'s, $i = 1, \ldots, m$,

$$N_2 = m \ln m$$

(see Press, et al. (1986))

(3). Computing ρ_l^q takes

$$N_3 = 7n \text{ operations}$$

(4). Computing $x'_{il}, i = 1, \ldots, m$ in the worst case requires

$$N_4 = 2m \text{ operations.}$$

Thus, the total number of operations for one iteration of the above Demand Algorithm is:

$$N_D = 9mn + mlnm + 8n + 5m,$$

or for large n and m

$$N_D = 9nm.$$

Since the number of iterations required to satisfy a given stopping criterion is Cn (see (49) and (50)), the overall number of iterations for the demand algorithm is:

$$N_D = C_1 n^2 m, \quad C_1 = 9C \tag{51}.$$

It is clear that by analogous argument, one can obtain an estimate for the supply algorithm:

$$N_S = C_1 m^2 n, \quad C_1 = 9C. \tag{52}$$

The estimates (51) and (52) give one an opportunity to choose between different implementations of the algorithm:

Guidance:

Choose demand algorithm if $m > n$ and supply algorithm, if $m < n$.

Of course, n or m should be sufficiently large so that the second order terms in (51) and (52) become insignificant. The above suggestions hold if one is "cautious", that is, if one wishes to select the implementation as if the worst case behavior will predominate. However, the algorithm can behave as mn, in which case the supply and demand implementations are essentially equivalent in the nm terms. There, however, the $m\ln m$ terms can dominate , in which the opposite rule would hold.

In order to emphasize the significance of the above analysis, we now highlight two critical points. First, observe from (50), that as ϵ gets smaller, decreasing ϵ by a factor results in an additive change (increase) in the number of iterations t. Second, the algorithm is of the order 3/2, in that if $m = n = N$ (as is often assumed in spatial price models), and if the number of variables is doubled, then the total number of operations (and CPU time)

increases by only a factor of $2^{3/2} \sim 3$. Specifically, if $m \sim n$, then $N_D = O(V^{3/2}) \approx C_1 V^{3/2}$ where V is the total # of unknowns: $V = nm \sim n^2$. Hence, the total time is equal to: $N_D \approx C_1 n^3 = C_1 V^{3/2}$.

Although we have conducted our analysis in the framework of the above theoretical algorithm, the discussion is relevant to Demand Equilibration Algorithms I, II, and III. For example, in the case that strict complementarity holds at the equilibrium solution, then the theoretical algorithm is identical to I. Algorithm II, in this case, has the features of I, in that one chooses an "optimal" direction, and switches to Algorithm I towards the end of the scheme. The infrequent use of the computation of (34d), on the other hand, results in a # of operations/iteration of order m, rather than mn. Algorithm II is, hence, better than n^2m; between nm and n^2m. In the limit, its behavior is like that of I. Algorithm III also takes m operations per iteration. However, the estimate of the number of iterations (49) no longer holds.

In summary, for algorithms I and II, the number of iterations grows linearly as the number of supply markets m increases, for a fixed number of demand markets n. We will show in Section 6 that ,empirically, this result also holds for the cyclic algorithm III. The number of operations per iteration for both II and III also grows linearly, theoretically. We will provide numerical evidence supporting the theory in Section 6.

Our recommendations for practical use are as follows: to begin with Algorithm III because of its simplicity and ease of implementation. If satisfactory performance is obtained for the particular application, then one can stop here. If better performance is desired, with only minor expansion and modification of the code for III, the good enough algorithm II can be implemented to select the next demand market l to be equilibrated in a more strategic fashion. If a cautious approach is then desired, then the implementation with the δ^t constructs as described in this Section can be incorporated. We emphasize that the theoretical algorithm constructed in this Section is useful for qualitative analysis only and should not be used in practice.

In the subsequent Section we present implementation suggestions to ensure that one obtains a good implementation of III and/or I.

5. Implementation Considerations

As cannot be over-emphasized, a good implementation is crucial to realizing the potential efficiency of any algorithm. Towards this end we present in this Section detailed comments on the effective implementation of progressive equilibration algorithms, so that the theoretical qualities of performance are encountered in practice. Again, we continue our discussion, without loss of generality, in the framework of the demand equilibration algorithms.

Recall that demand equilibration algorithms I, II, and III, at each iteration must solve a restricted equilibrium problem of the form (13) exactly. We first, hence, address the implementation of the exact procedure described in (20).

At each iteration, the $\hat{h}_{il}$'s, $i = 1, \ldots, m$ (cf.(14)) should be computed and stored in an array. An appropriate sorting procedure should be implemented. Given that the market equilibrium problems we consider here are large-scale, where the number of supply markets and the number of demand markets exceed 100 units, the recommended sorting procedure is heapsort (for details, see Press, et al. (1986)). In order to efficiently implement the computation of ρ_l^q (cf. (20)), the number of unnecessary additions and divisions should be minimized. We recommend that two variables be used to accumulate the value of the numerator and denominator, respectively, at each q. All the supplies and demands at all markets should be stored in two distinct arrays. At each iteration of the demand algorithms we further recommend that another array be used to store the supply at each supply market i minus the shipment from i to the demand market l to be equilibrated. In this manner, the new supplies can be easily recomputed following the exact procedure in m operations.

In order to complete the implementation of the cyclic algorithm III, a clever implementation of a convergence criterion should be incorporated. In particular, we recommend that, before equilibration of, say, demand market l is to take place, the criterion $|\pi_i + c_{il} - \rho_l| \leq \epsilon$, if $X_{il} > 0$; or $\pi_i + c_{il} - \rho_l \geq -\epsilon$, if $X_{il} = 0$; should be verified for all $i = 1, \ldots, m$. This can be incorporated within the $\hat{h}_{il}$ computation loop. If l is indeed equilibrated then one jumps past the exact procedure where a counter is accumulated. If the market has not yet reached equilibrium, the counter is set equal to zero. If the counter is equal to the number of demand markets n, then III can be considered terminated.

The implementation of the Good Enough Demand Equilibration Algorithm II requires that l be chosen so that a large enough decrease of the objective function is attained at each iteration. We now describe how the computation of C_{lold} can be accomplished with the minimum number of computations. Recall that $C_{lold} \equiv \frac{\Phi(X^t)-\Phi(X^{t-1})}{\|X_l^t - X_l^{t-1}\|}$. For convenience, we let $\Phi_{new} \equiv \Phi(X^t)$, $\Phi_{old} \equiv \Phi(X^{t-1})$; $S_{old} \equiv \sum_i 1/2g_i(s_i^{t-1})^2 + \sum_i h_i s_i^{t-1}$ (before equilibration of l), and $S_{new} \equiv \sum_i 1/2g_i(s_i^t)^2 + \sum_i h_i s_i^t$ (after equilibration of l at t). Then

$$\Phi_{new} = \Phi_{old} - S_{old} + S_{new} - \sum_i (1/2g_{il}((X_{il}^{t-1})^2 - (X_{il}^t)^2))$$

$$- \sum_i h_{il}(X_{il}^{t-1} - X_{il}^t) - 1/2m_l((d_l^{t-1})^2 - (d_l^t)^2)) + q_l(d_l^{t-1} - d_l^t).$$

For algorithm II so as not to introduce any inconsistencies, the following convergence criterion is appropriate: $P\|\frac{\partial \Phi}{\partial X_l}\|^2 \leq \epsilon$. Similarly, for I: $\sum_{i=1}^m P_{ij}^2 \leq \epsilon$ is appropriate.

In terms of computer memory requirements, the implementation of algorithm III for the model with distinct supply and demand price functions,requires on the order of $mn+n$ storage elements, that is, the storage is in the dimension of the number of variables. Algorithm II requires only slightly more storage. As is well-known, iterative algorithms tend to be conservative in terms of storage (as opposed to direct algorithms).

6. Illustrative Numerical Results

In this Section we provide numerical results vis a vi's the performance of progressive equilibration algorithms. The computational experience is presented to highlight and illuminate the theoretical contributions in Sections 3 and 4.

All of the subsequent numerical experiments were conducted on a mainframe – the IBM 4381-14 at the Cornell National Supercomputer Facility. The progressive equilibration algorithms were coded in FORTRAN and the CPU times are reported exclusive of input/output.

The data for the large-scale market equilibrium examples was generated as described in Nagurney (1987b). For completeness, we now describe the data generation. The number of demand markets n and the number of supply markets m in any given experiment were fixed accordingly. All of the examples were generated randomly and uniformly over the ranges as follows. For all the examples the supply price, demand price, and transaction cost function slopes and intercepts (cf.(6), (8), and (4)) were whole numbers within the following ranges: $\eta_i \in [3, 10], \psi_i \in [10, 25]$, $-\omega_j \in [-1, -5], \lambda_j \in [150, 650], g_{ij} \in [1, 15], h_{ij} \in [10, 25]$; $i = 1, \ldots, m; j = 1, \ldots, n$.

The initial commodity shipment X^0 was identical for all (i, j) pairs in a given example; where $X^0_{ij} = (\frac{\lambda_j}{\omega_j} + 1)/m$. Hence, all of the X^0_{ij} values were positive. For these experiments for purposes of consistency across all algorithms, the convergence criterion utilized was the one suggested in Section 5 for Algorithm III, in particular, $|\pi_i + c_{il} - \rho_l| \leq 1$, if $X_{il} > 0$; $\pi_i + c_{il} - \rho_l \geq -1$, if $X_{il} = 0$, for all i and l. (Note, however, that for *practical use* the suggestions in Section 5 regarding the appropriate criterion for an individual algorithm, should be adhered to.)

In the first set of experiments, reported in Table 1 and Figure 1, we fixed the number of demand markets at 100, and varied the number of supply markets from 100 to 300, in increments of 100. In Table 1 we report both the CPU time and the number of iterations t, for each of the demand progressive equilibration algorithms I, II, and III. Observe that I required the fewest number of iterations and III the greatest number, with II lying somewhere in between. However, as also predicted by the theory, I required the most CPU time and II the least. Note also that for all the algorithms, although there were many iterations t, each iteration was very inexpensive computationally. In Figure 1, we then

proceeded to plot the total CPU time in seconds reported in Table 1 for the examples. Empirically the behavior of the original progressive equilibration algorithm III was similar to that of II and I in that the CPU time grew linearly as the number of supply markets was increased.

In the second set of tests, results of which are tabulated in Table 2 and depicted graphically in Figures 2 and 3, we fixed the number of demand markets at 500 and varied the number of supply markets from 100 to 400, again in increments of 100, and we studied the performance of demand algorithms II and III (which are for practical use). As can be seen from Figure 2, the CPU time again increased linearly, as the number of supply markets was increased. In Figure 3, we then plotted the CPU time per iteration for the same examples as in Figure 2 for both algorithms II and III. Note that the CPU time per iteration is a measure of the number of operations per iteration. For algorithm III, the numerical data illustrated the theoretical results which predicted a linear relationship. For algorithm II, the numerical data also supported the theoretical prediction which stated that the number of operations per iteration would vary between m and nm.

In the final set of experiments, reported in Figures 4 and 5, we compared the relative performance of Demand Algorithm III and Supply Algorithm III. In Figure 4, we fixed the number of demand markets at 500 and varied the number of supply markets from 100 to 500, whereas in Figure 5, we fixed the number of supply markets at 500 and varied the number of demand markets from 100 to 500 in increments of 100. Observe that in Figure 4, the demand equilibration algorithm outperformed the supply equilibration algorithm, whereas in Figure 5 the opposite relative performance (as expected) was exhibited. Note, also, that in the case $m = n = 500$, the relative performance was essentially equivalent. Here the relative efficiencies can be explained by the effect that the sorting component takes, which, recall, because of the use of heapsort, is $k\ln k$, with k representing the number of variables sorted.

In summary, the numerical results reported here have been presented with the intention of providing insight into the qualitative results. The results are for large-scale market equilibrium problems and the examples computed in this Section are the largest of this kind reported. For numerical experience on the simpler, special case net import model for problems with as many as 800 markets, see Nagurney (1988b). For the use of equilibration

algorithms embedded in variational inequality decomposition algorithms for multicommodity problems and implemented on serial and parallel computers, see Nagurney and Kim (1988).

7. Summary and Discussion

In this paper we considered the solution of large-scale market equilibrium problems which can be formulated as quadratic programming problems in quantity variables. Such market equilibrium problems include virtually the entire spectrum of classical spatial price equilibrium problems and are encountered when more general multicommodity market equilibrium problems are solved iteratively (see, e. g., Nagurney and Kim (1988)). Indeed, the overall efficiency of, for example, variational inequality algorithms applied to compute the solution of multicommodity market problems depends critically on the algorithm used for the solution of the single commodity problems encountered at each step. Hence, the development of efficient algorithms for single commodity problems has a direct effect on the ability to compute efficiently multicommodity market equilibria.

We introduced two new classes of progressive equilibration algorithms – locally optimal and good enough algorithms, which differ from the cyclic algorithms in the strategic manner in which the next market to be equilibrated is selected. A theoretical framework for the entire family of progressive equilibration algorithms was then developed in the case of linear increasing transaction costs. The theory provided the rate of convergence, as well as, the complexity of the algorithms. Although the original cyclic algorithms had been supported by extensive computational experience, a rigorous theoretical framework had been lacking. Moreover, the theory serves as a background for implementations of such algorithms for both researchers and practitioners. Indeed, these algorithms are intuitive and straightforward to implement and their behavior is now explained.

Finally, suggestions for implementation and numerical results to illustrate and highlight the theory were provided. As expected from the theory, the new "good-enough" algorithm was found to be the most efficient computationally, and especially well-suited for large-scale problems. As a consequence, the "best" algorithm need no longer be programmer, machine, or computer dependent, but, rather, based on a sound theoretical framework.

Acknowledgements

The authors are indebted to the referees for their careful reading of the paper and their helpful suggestions which added substantially to the presentation of this work.

The first author's reasearch was supported by NSF Grant: DMS - 8602316.

The second author's research was supported in part by NSF Grant: SES-8702831 and in part by NSF Grant: RII-8800361 under the sponsorship of the NSF VPW program, while the author was visiting MIT. The cordiality and hospitality of the Center for Transportation Studies and the Operations Research Center are warmly appreciated.

This research was conducted on the Cornell National Supercomputer Facility, a resource of the Center for Theory and Simulation in Science and Engineering at Cornell University, which is funded in part by the National Science Foundation, New York State, and IBM Corporation.

References

Asmuth, R., Eaves, B. C., and Peterson, E. L., 1979, "Computing Economic Equilibria on Affine Networks with Lemke's Algorithm," *Mathematics of Operations Research*4: 209-214.

Dafermos, S., 1986, "Isomorphic Multiclass Spatial Price and Multimodal Traffic Network Equilibrium Models," *Regional Science and Urban Economics* **16**: 197-209 .

Dafermos, S. and Nagurney, A., 1987," Supply and Demand Equilibration Algorithms for a Class of Market Equilibrium Problems," forthcoming in *Transportation Science*.

Florian, M. and Los, M., 1982, "A New Look at Static Spatial Price Equilibrium Models," *Regional Science and Urban Economics* **12**: 579-597 .

Glassey, C. R., 1978,"A Quadratic Network Optimization Model for Equilibrium Single Commodity Trade Flows," *Mathematical Programming* **14**: 98-107.

Güder, F., 1987,"Pairwise Reactive SOR Algorithm for Quadratic Programming of Net Import Spatial Equilibrium Models" , forthcoming in *Mathematical Programming*.

Jones, P. C., Saigal, R., and Schneider, M. H., 1986, "A Variable Dimension Homotopy on Networks for Computing Linear Spatial Equilibria," *Discrete Applied Mathematics* **13**: 131-156.

Judge, G. G., and T. Takayama, 1973, **Studies in Economic Planning over Space and Time**, North-Holland Publishing Co.

Nagurney, A., 1987a, " An Algorithm for the Classical Spatial Price Equilibrium Problem," *Operations Research Letters* **6, no. 2**: 93-98 .

Nagurney, A., 1987b, "Computational Comparisons of Spatial Price Equilibrium Methods," *Journal of Regional Science* **27**: 55-76 .

Nagurney, A., 1987c,"Competitive Equilibrium Problems, Variational Inequalities, and Regional Science, " *Journal of Regional Science*, **27**: 503-514 .

Nagurney, A., 1988a, "An Algorithm for the Solution of a Quadratic Programming Problem with Application to Constrained Matrix and Spatial Price Equilibrium Problems,"

Environment & Planning A, in press.

Nagurney, A., 1988b, "Import and Export Equilibration Algorithms for the Solution of the Net Import Model," *Journal of Cost Analysis*,", in press.

Nagurney, A. and Aronson, J., 1988, "A General Dynamic Spatial Price Equilibrium Model: Formulation, Solution, and Computational Results, " *Journal of Computational and Applied Mathematics*, 359-377 .

Nagurney, A. and Kim, D. S., 1988, "Parallel and Serial Variational Inequality Decomposition Algorithms for Multicommodity Market Equilibrium Problems," *The International Journal of Supercomputer Applications*, in press.

Press, W. H., Flannery, B. H., Teukolsky, S. A., and Vetterling, W. T., 1986, **Numerical Recipes**, Cambridge University Press.

Samuelson, P. A., 1952, "Spatial Price Equilibrium and Linear Programming,"*American Economic Review* **42**: 283-303 .

Takayama, T. and Judge, G. G., 1971, **Spatial and Temporal Price and Allocation Models**, North-Holland, Amsterdam.

Zangwill, W. I., 1969, **Nonlinear Programming - A Unified Approach**, Prentice Hall.

Table 1. Computational Experience for the Demand Equilibration Algorithms I, II, and III

# of demand markets n fixed at 100

CPU time in seconds(# of iterations t)

# of supply markets	I	II	III
100	35.46(294)	5.58(543)	7.61(1044)
200	111.55(458)	19.87(797)	24.93(1833)
300	224.46(613)	43.85(1029)	49.96(2552)

Table 2: Computational Experience for Demand Equilibration Algorithms II and III

# of demand markets n fixed at 500

CPU time in seconds(# of iterations t)(CPU time/t)

# of supply markets m	II	III
100	23.99(1764)(.0136)	45.73(3450)(.0133)
200	62.21(2075)(.0300)	126.25(4646)(.0272)
300	101.25(2111)(.0480)	211.63(5713)(.0370)
400	172.27(2082)(.0827)	263.32(5466)(.04817)

Figure 1: Demand Algorithms I,II,III
100 Demand Markets

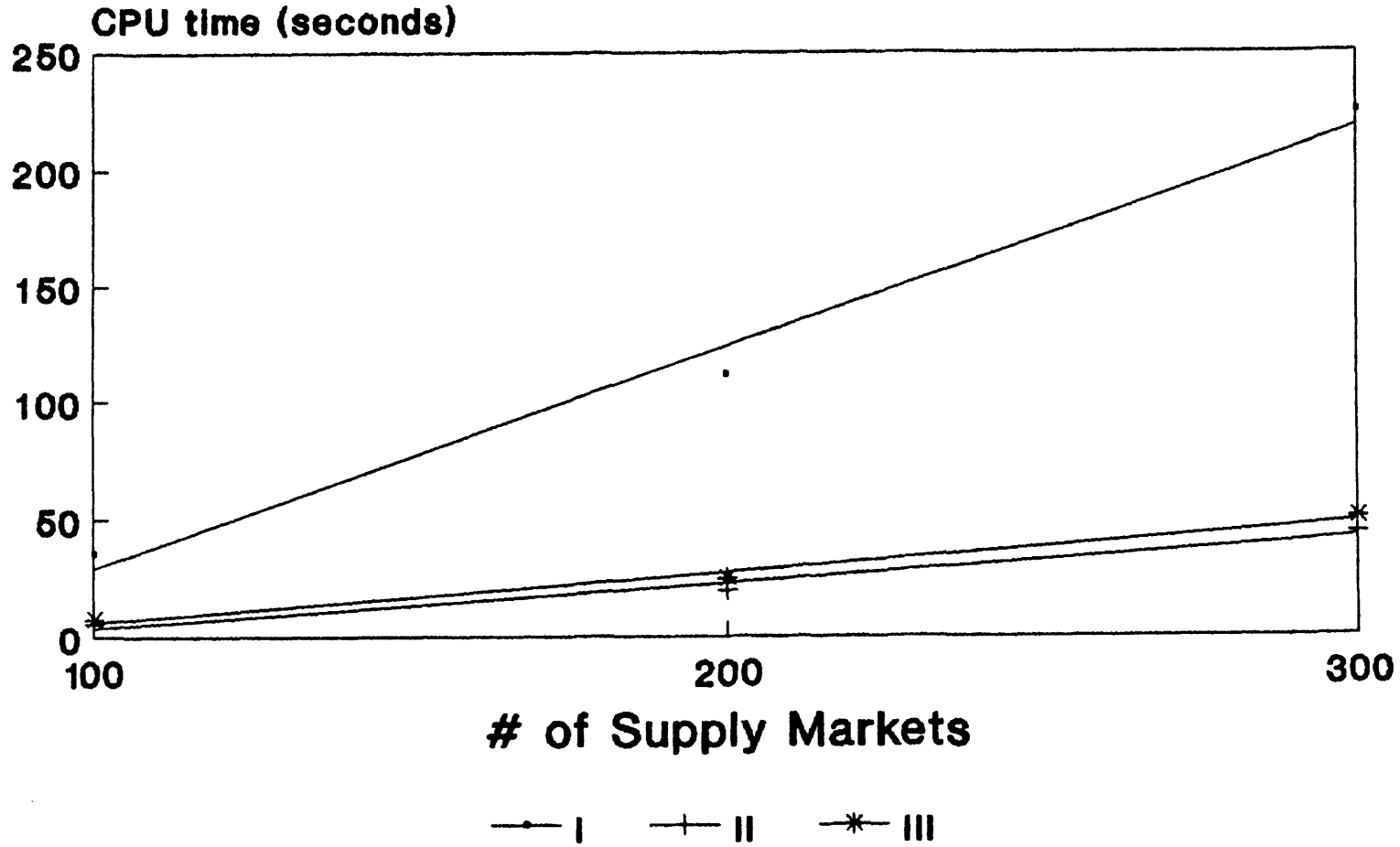


Figure 2: Demand Algorithms II,III
500 Demand Markets

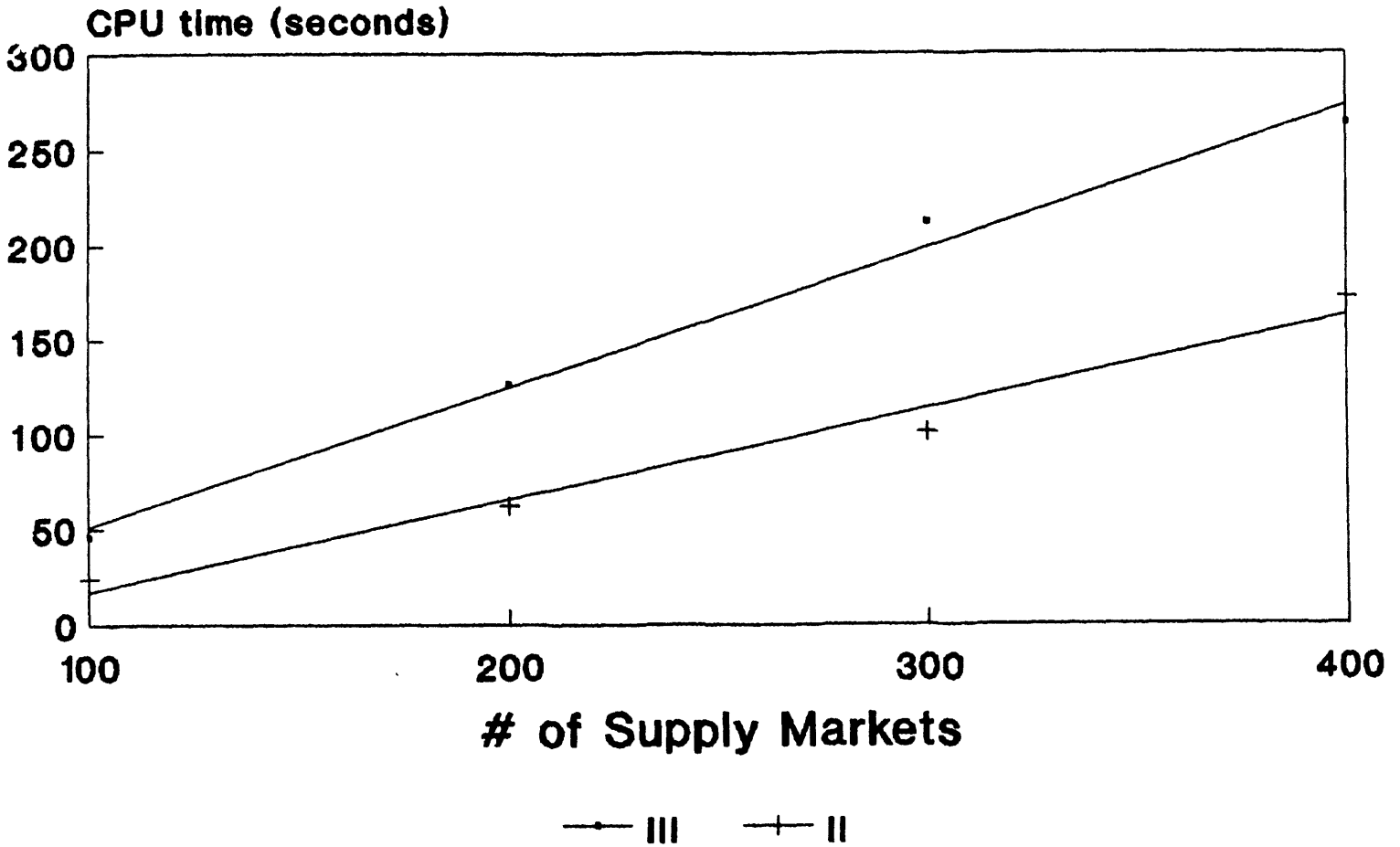


Figure 3: CPU time/it. for Demand II,III
500 Demand Markets

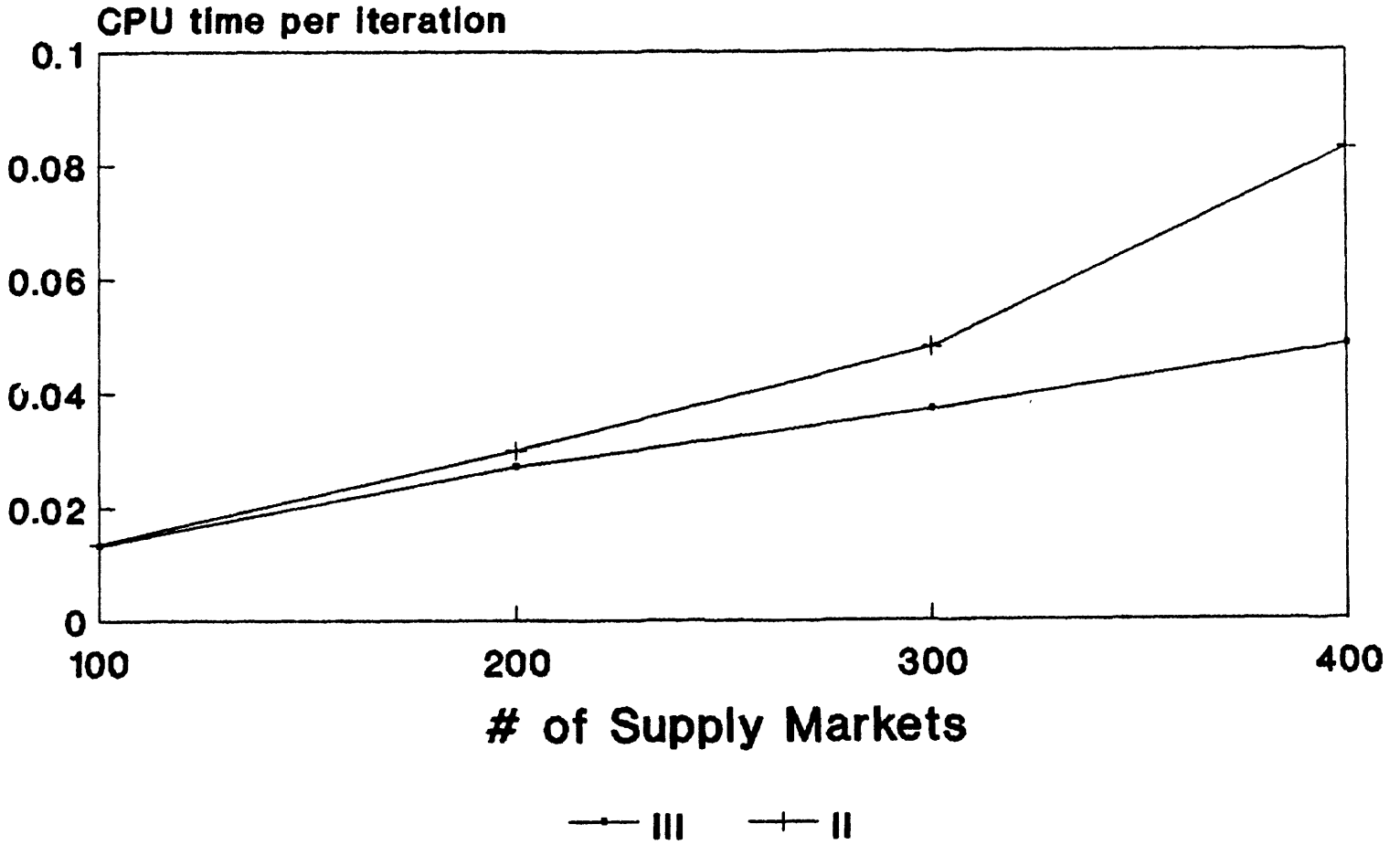


Fig 4.Performance Demand & Supply III
500 Demand Markets

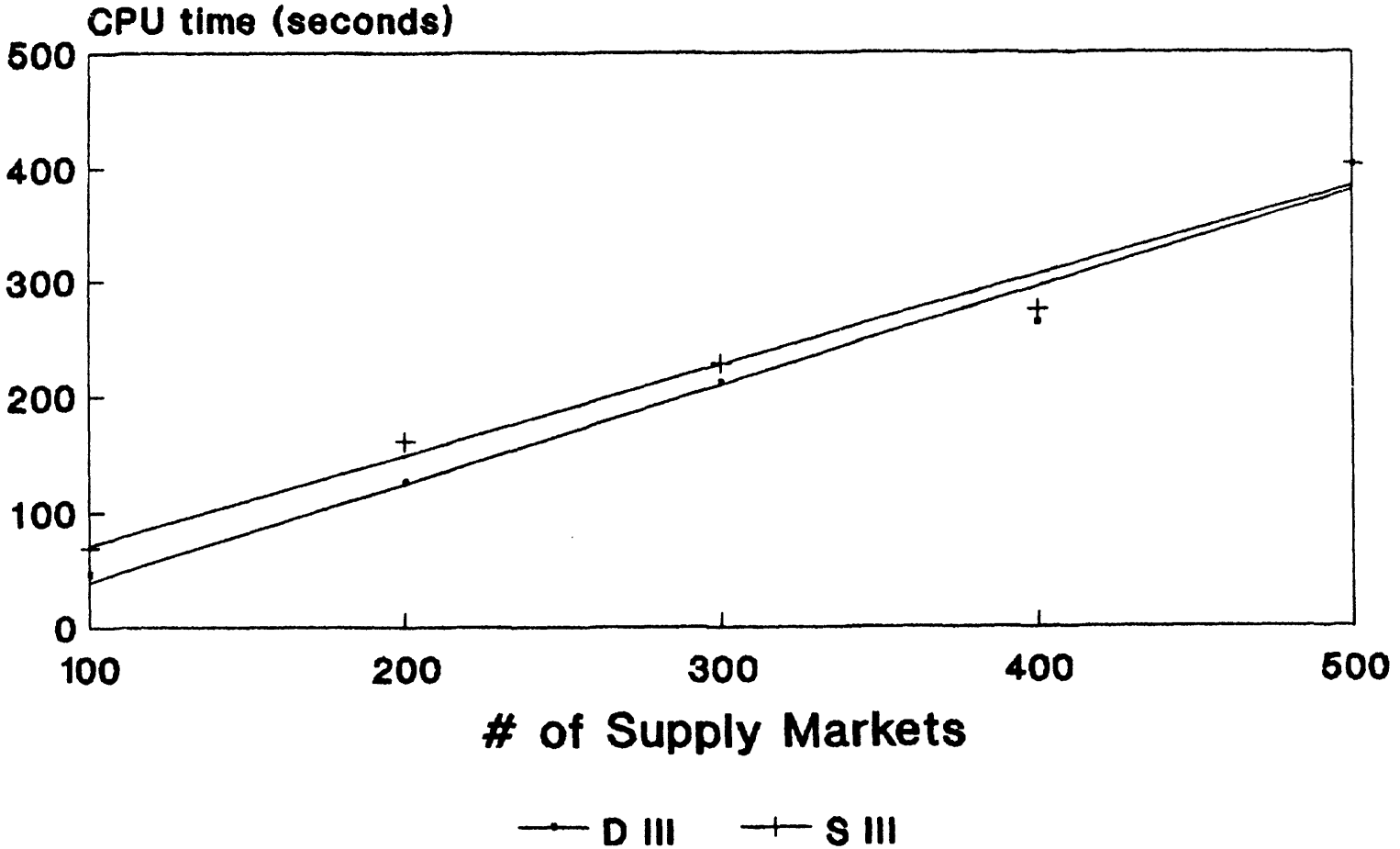


Fig 5: Performance Demand & Supply III
500 Supply Markets