The Applications of Ameso Optimization in Supply Chains

Katehakis, M. N
Rutgers Business School
Department of MSIS
180 University,
Newark, New Jersey

Chen, Wen
Rutgers Business School
Department of MSIS
180 University,
Newark, New Jersey

Abstract: In this paper we discuss the application of a special class of discrete optimization problems the *ameso* programming problems. For these problems we have established elsewhere that there exits an efficient solution procedure called the Ameso Reduction Procedure (ARP). In this paper it is shown that certain multi-type vehicle shipping problems can be expressed as ameso optimization problems. We apply the ARP to solve these problems and we present detailed numerical studies. Furthermore, the complexity of the two type vehicle problem is give,

Key–Words: Discrete Optimization, Supply Chain Logistics.

1 Introduction

In this paper we discuss the applicationm of a special class of discrete optimization problems the *ameso* programming problems. The main properties they have are: 1) high dimensional ameso optimization problems can be solved by decomposition into lower dimensional ameso optimization problems; 2) for one dimension ameso optimization problems there are simple, to verify, optimality conditions at any optimal point [1].

I section 2 we review the ameso optimization problem and its properties. In section 3, we present an application of the ameso optimization for Multi Type Vehicle Shipping Problems (MTVSP). Firstly we show that MTVSP can be formulated as an ameso optimization problem. Next, we design an algorithm to find the optimal solution for the Two Type Vehicle Shipping Problem. Next, we show that our algorithm will computer an optimal solution in at most q iterations, where q is a constant determined by parameters of the problem. Then we extend our ameso based algorithm to Multi Type Vehicle Shipping Problems. In the end, we give some numerical examples on the MTVSP problem. For related work in supply chain logistics and optimization we refer to [2] [3] [4] and [6]. For another successful solution by decomposition we refer to [5].

2 Review of Ameso(C) Optimization

Given a subset D^n of the n - dimensional integers, $D^n \subseteq \mathbf{Z}^n$, and a real function f defined on D^n, we define the following:

Definitions 1

1. **Ameso Set:** D^n is an ameso set, if it satisfies following condition:

   - $\forall \vec{x}, \vec{y} \in D^n \Rightarrow \left\lceil \frac{\vec{x}+\vec{y}}{2} \right\rceil, \left\lfloor \frac{\vec{x}+\vec{y}}{2} \right\rfloor \in D^n$.

2. **Ameso(C) pair:** (D^n, f) is a ameso(C) pair, if and only if

   - the domain D^n of the function $f(\cdot)$ is an ameso set and
   - $f(\cdot)$ has a lower bound and
   - $\exists C > 0$ such that the following holds for $\forall \vec{x}, \vec{y} \in D^n$

   $$f(\vec{x})+f(\vec{y})+C \geq f(\lceil \frac{\vec{x}+\vec{y}}{2} \rceil)+f(\lfloor \frac{\vec{x}+\vec{y}}{2} \rfloor).$$

3. **Ameso(C) Optimization:** The problem $minimize\ f(\vec{x});\ subject\ to\ \vec{x} \in D^n$ is an Ameso(C) Optimization problem, if (D^n, f), is an Ameso(C) pair.

Definitions 2.

1. **The domain** $\Delta^j_{i_1,...,i_j} \subseteq D^n$ is the set: $\Delta^j_{i_1,...,i_j} = \{(x_{i_1}, ..., x_{i_j}) :$ $\exists (x'_1, ..., x'_n) \in D^n$ with $x'_{i_k} = x_{i_k}, \forall k = 1, ..., j\}$.

2. **The conditional domain** of $\vec{x} \in D^n$ given $x^0_{i_1}, ..., x^0_{i_j}$ to be the set: $\Gamma^{n-j}_{x^0_{i_1},...,x^0_{i_j}}$ $= \{(x_1, ..., x_n) \in D^n : x_{i_k} = x^0_{i_k}\ \forall k = 1, ..., j\}$.

3. **The conditional function**: $f^*_{i_1,\ldots,i_j} : \Delta^j_{i_1,\ldots,i_j} \longrightarrow \Re$
$f^*_{i_1,\ldots,i_j}(x_{i_1},\ldots,x_{i_j}) = min_{\vec{y}\in\Gamma^{n-j}_{x_{i_1},\ldots,x_{i_j}}} f(\vec{y})$.

4. **The conditional pair** of ameso(C) pair to be the pair: $(\Delta^j_{i_1,\ldots,i_j}, f^*_{i_1,\ldots,i_j})$.

It has been shown in [1] that the following corollary and proposition hold.

Corollary 1. *If we can find* $x' \in D^1 = [z_s, z_t] \cap \mathbf{Z}, z_1 \in D^1, z_2 \in D^1, z_1 < x', z_2 > x'$ *with* $f(z_1) - f(x') \geq C$ *and* $f(z_2) - f(x') \geq C$
then

$$min_{y\in[z_1,z_2]\cap\mathbf{Z}} f(y) = min_{y\in D^1} f(y).$$

Proposition 2. *The conditional pair* $(\Delta^j_{i_1,\ldots,i_j}, f^*_{i_1,\ldots,i_j})$ *of an n-dimensional ameso(C) pair* (D^n, f) *is a j-dimensional ameso(C) pair.*

Further, in [1] it was shown that the following Ameso Reduction Procedure (ARP) can compute an optimal solution, without necessarily doing an exhaustive search.

Ameso Reduction Procedure (ARP)
Input: Ameso(C) Optimization problem:
$minimize\ f(\vec{x})$; $subject\ to\ \vec{x} \in D^n$
Step 1: $A = \phi, l^+ = 0, l^- = 0$, select a point $l^0 \in \Delta^1_n, l^* = l^0, S = l^0$, then calculate $f^*_n(l^0) = min_{(x_1,\ldots,x_{n-1},l^0)\in\Gamma^{n-1}_{l^0}} f(x)$ †,
Step 2: Update $l^+ = min\{x | x > l^0, x \in \Delta^1_n - A\}$,
then calculate $f^*_n(l^+) = min_{(x_1,\ldots,x_{n-1},l^+)\in\Gamma^{n-1}_{l^+}} f(x)$ †
Step 3: If $f^*_n(l^+) - f^*_n(l^*) \geq C$, go to **step 4**;
If $0 \leq f^*_n(l^+) - f^*_n(l^*) < C$, then $A = A \cup \{l^+\}$ go to **step 2**.
If $f^*_n(l^+) - f^*_n(l^*) < 0$, then $A = A \cup \{l^+\}$, $l^* = l^+$ go to **step 2**.
If $\{x | x \in \Delta^1_n - A, x > l_0\} = \phi$, go to **step 4**;
Step 4: Update l^- to be any integer satisfying $f^*_n(l^-) = max_{\{l | l\in A, l<l^*\}} f^*_n(l)$;
If $f^*_n(l^-) - f^*_n(l^*) \geq C$, go to **output**;
Step 5: Update $l^- = max\{x | x < l^0, x \in \Delta^1_n - A\}$,
then calculate
$f^*_n(l^-) = min_{(x_1,\ldots,x_{n-1},l^-)\in\Gamma^{n-1}_{l^-}} f(x)$ †
Step 6: If $f^*_n(l^-) - f^*_n(l^*) \geq C$, go to **output**;
If $0 \leq f^*_n(l^-) - f^*_n(l^*) < C$, then $A = A \cup \{l^-\}$ go to **step 5**.
If $f^*_n(l^-) - f^*_n(l^*) < 0$, then $A = A \cup \{l^-\}$, $l^* = l^-$ go to **step 5**.
If $\{x | x \in \Delta^1_n - A, x < l_0\} = \phi$, go to **output**;
Output: A, l^*, $f^*_n(l^*)$

†Note : the computation of $f^*_n(l^0)$, $f^*_n(l^+)$ and $f^*_n(l^-)$ above involves the solution of $(n-1)$-dim ameso(C) optimization problems.

3 Multi Type Vehicle Shipping Problems(MTVSP).

3.1 Problem Formulation.

Assume we have $n+1$ types of vehicles to transport a total cargo of weight W. Each type $i = 0, 1, \ldots, n$, has a fixed shipping fee S_i and a fixed capacity C_i. We assume that type 0 vehicle has the smallest capacity. The problem is how to distribute the total cargo W. Let x_i be the number of type i vehicles used in an allocation, $i = 1, \ldots, n$; where we assume that residual cargo is allocated to "type 0" vehicle. The number of "type 0" vehicles that are used is $\left\lceil \left(\frac{W - \sum_{i=1}^n x_i C_i}{C_0} \right)^+ \right\rceil$.

Let D^n_{TS} be the domain of $(x_1, \ldots x_n)$, then

$$D^n_{TS} = \{\vec{x} = (x_1, \ldots, x_n) : \vec{x} \in \mathbf{Z}^{n+},\ x_i \leq \lceil \frac{W}{C_i} \rceil,\ \forall i\}. \quad (1)$$

It is easy to see that the domain D^n_{TS} is an ameso set, because if $\vec{x}, \vec{y} \in D^n_{TS}$, then $\left\lceil \frac{\vec{x}+\vec{y}}{2} \right\rceil, \left\lfloor \frac{\vec{x}+\vec{y}}{2} \right\rfloor \in D^n_{TS}$.

And the total shipping fee (TS) is the following function:

$$TS(x_1, \ldots, x_n) = \sum_{i=1}^n x_i S_i + \left\lceil \left(\frac{W - \sum_{i=1}^n x_i C_i}{C_0} \right)^+ \right\rceil S_0. \quad (2)$$

The objective is to solve the following problem:

$$\min TS(\vec{x}); \text{ subject to } \vec{x} \in D^n_{TS}. \quad (3)$$

The following lemma is easy to prove.

Lemma 3. *For any arbitrary real numbers* x_1, x_2, y, *the following is true:*

$$\left\lceil (x_1 + y)^+ \right\rceil + \left\lceil (x_2 - y)^+ \right\rceil \geq \lceil x_1^+ \rceil + \lceil x_2^+ \rceil - max\{|x_1 - x_2|, 1\}. \quad (4)$$

Next we will discuss how to solve this Multi Type Vehicle Shipping problem. First, it is easy to prove, using the lema above, the following theorem.

Theorem 4. *The MTVS problem:*

min $TS(x_1,\ldots,x_n)$*; s.t* $(x_1,\ldots,x_n) \in D_{TS}^n$

is an $n-$*dim ameso(C) optimization problem with* $C = \lfloor \frac{\sum_{i=1}^n C_i}{C_0} \rfloor S_0$.

In the remaining discussion we will assume the following inequalities to hold for all $i, j, 0 \le i < j \le n$:

$$C_i < C_j, \quad S_i < S_j, \quad \frac{S_i}{C_i} > \frac{S_j}{C_j}.$$

These inequalities are often valid in practice, and under this assumption we can have very good algorithmic performance.

3.2 Two-Type-Vehicle Shipping Problem

First, we discuss just two types of vehicles with $C_0 < C_1, S_0 < S_1, (\frac{S_1}{C_1} < \frac{S_0}{C_0})$. The problem becomes:

$$\min TS(x_1) = x_1 S_1 + \lceil \left(\frac{W - x_1 C_1}{C_0} \right)^+ \rceil S_0;$$
$$\text{subject to } x_1 \in D_{TS}^1, D_{TS}^1 = \{0, 1, \ldots, \left\lceil \frac{W}{C_1} \right\rceil\}$$

.

We use corollary 1 to design Algorithm 1, below, to solve the Two-Type-Vehicle shipping problem.

Algorithm 1

Input: $W, C_0, C_1, S_0, S_1,$

Step One: $K = \left\lceil \frac{W}{C_1} \right\rceil, J_+ = \phi, l_1 = K, j = 1, J_+ = J_+ \cup \{l_j\}$, calculate $TS(l_1)$

Step Two: $j = j + 1, l_j = l_{j-1} - 1, J_+ = J_+ \cup \{l_j\}$, calculate $TS(l_j)$

If $j = K + 1$, stop;

If $TS(l_j) - min_{l_i \in J_+} TS(l_i) \ge \lfloor \frac{C_1}{C_0} \rfloor S_0$, stop;

If $TS(l_j) - min_{l_i \in J_+} TS(l_i) < \lfloor \frac{C_1}{C_0} \rfloor S_0$, go to step 2.

Output: $j, J_+, \quad l^*$ where $TS(l^*) = min_{l_i \in J_+} TV(l_i)$.

By construction, the output of the above algorithm, j, J_+, will satisfy:

- either $j = \left\lceil \frac{W}{C_1} \right\rceil + 1, J_+ = \{0, 1, \ldots, \left\lceil \frac{W}{C_1} \right\rceil\}$;
- or $TS(l_j) - min_{l_i \in J_+} TS(l_i) \ge \lfloor \frac{C_1}{C_0} \rfloor S_0, J_+ = \{l_1, l_2, \ldots, l_j\}$.

We can now state the following:

Theorem 5. *The output* l^* *of Algorithm 1 is an optimal solution.*

Next we discuss the complexity of Algorithm 1. Let $q = min \left\{ \left\lceil \frac{W}{C_1} \right\rceil + 1, \left\lceil \frac{2}{1 - \frac{S_1/C_1}{S_0/C_0}} \right\rceil + 1 \right\}$.

Theorem 6. *Algorithm 1 will stop in at most* q *iterations.*

Proof: First, we only have $\left\lceil \frac{W}{C_1} \right\rceil + 1$ feasible solutions for x_1, so the algorithm has to stop in at most $\left\lceil \frac{W}{C_1} \right\rceil + 1$ iterations, in the worst case that it will try all feasible solutions. In this case $j = \left\lceil \frac{W}{C_1} \right\rceil + 1$.

Next we will show algorithm has to stop in at most $\left\lceil \frac{2}{1 - \frac{S_1/C_1}{S_0/C_0}} \right\rceil + 1$ iterations. We have $\frac{S_1}{C_1} < \frac{S_0}{C_0} \Rightarrow \frac{S_1}{S_0} < \frac{C_1}{C_0} \Rightarrow \frac{S_1}{S_0} \frac{C_0}{C_1} < 1$.

Let $m = \left\lceil \frac{2}{1 - \frac{S_1/C_1}{S_0/C_0}} \right\rceil$. Then $m \ge \frac{2}{1 - \frac{S_1/C_1}{S_0/C_0}} = \frac{\frac{2C_1}{C_0}}{\frac{C_1}{C_0} - \frac{S_1}{S_0}}$. Hence, $m(\frac{C_1}{C_0} - \frac{S_1}{S_0}) \ge \frac{2C_1}{C_0}$. And then $m\frac{C_1}{C_0} - \frac{C_1}{C_0} - m\frac{S_1}{S_0} \ge \frac{C_1}{C_0}$. Then $\lceil m\frac{C_1}{C_0} - \frac{C_1}{C_0} \rceil - m\frac{S_1}{S_0} \ge \lfloor \frac{C_1}{C_0} \rfloor$. And because $l_1 = \left\lceil \frac{W}{C_1} \right\rceil \Rightarrow W - l_1 c_1 > -c_1 \Rightarrow \left\lceil m\frac{C_1}{C_0} + \frac{W - l_1 C_1}{C_0} \right\rceil \ge \left\lceil m\frac{C_1}{C_0} - \frac{C_1}{C_0} \right\rceil$, then $\left\lceil m\frac{C_1}{C_0} + \frac{W - l_1 C_1}{C_0} \right\rceil - m\frac{S_1}{S_0} \ge \lfloor \frac{C_1}{C_0} \rfloor$.

That is to say

$$\left\lceil \frac{W - (l_1 - m)C_1}{C_0} \right\rceil S_0 - mS_1 \ge \lfloor \frac{C_1}{C_0} \rfloor S_0 \qquad (5)$$

Since $l_1 = \left\lceil \frac{W}{C_1} \right\rceil$, and $l_i = l_{i-1} - 1, \ \forall i > 1$ it follows that $l_{m+1} = l_1 - m, \frac{W - (l_1 - m)C_1}{C_0} \ge 0$, and $\frac{W - l_1 C_1}{C_0} \le 0$. Thus,

$$\begin{aligned} &TS(l_{m+1}) - T(l_1) \\ &= (l_1 - m)S_1 + \lceil \tfrac{W - (l_1 - m)C_1}{C_0} \rceil S_0 - l_1 S_1 - 0 \\ &= \lceil \tfrac{W - (l_1 - m)C_1}{C_0} \rceil S_0 - mS_1 \ge \lfloor \tfrac{C_1}{C_0} \rfloor S_0 \end{aligned}$$

(because of (5))

Let $J_+ = \{l_1, l_2, \ldots, l_{m+1}\}$, then $|J_+| = m + 1$.

And it is easy to see that $TS(l_1) \ge min_{l_i \in J_+} TS(l_i)$ because of $l_1 \in J_+$, so

$$\begin{aligned} &TS(l_{m+1}) - min_{l_i \in J_+} TS(l_i) \\ &\ge TS(l_{m+1}) - TS(l_1) \ge S_0, \end{aligned}$$

satisfies the one stop condition of the algorithm. That is to say, the algorithm would stop in at most $|J_+| = m + 1$ iterations.

If $\frac{S_1/C_1}{S_0/C_0}$ becomes smaller, then $\left\lceil \frac{2}{1 - \frac{S_1/C_1}{S_0/C_0}} \right\rceil + 1$ will be smaller, and this will result in a smaller number of iterations in Algorithm 1. The proof is complete. □

Next, we will show how to solve Multi Type Vehicle Shipping Problem.

3.3 Algorithm for Multi-Vehicle-Shipping Problem

Using proposition 2, and results of [1] regarding the ARP algorithm, it follows that we can solve the $(n+1)-$type-vehicle shipping problem, with algorithm 2 below.

Algorithm 2

Input: $W, C_0, \ldots, C_n, S_0, \ldots, S_n,$

Step One: $K_n = \left\lceil \frac{W}{C_n} \right\rceil, J_+ = \phi, l_1 = K_n, j = 1, J_+ = J_+ \cup \{l_j\}$

calculate $f_n^*(l_1) = TS(0, \ldots, 0, l_1)$

Step Two: $j = j+1, l_j = l_{j-1} - 1, J_+ = J_+ \cup \{l_j\}$,

Let $W' = W - l_j C_n$, then calculate $f_n^*(l_j)$ $f_n^*(l_j) = min_{(x_1,\ldots,x_{n-1},l_j)\in\Gamma_{l_j}^{n-1}} \sum_{i=1}^n x_i S_i + \left\lceil \left(\frac{W' - \sum_{i=1}^{n-1} x_i C_i}{C_0} \right)^+ \right\rceil S_0$.

It is a n-type-vehicle shipping problem and

it is a $(n-1)$-dim ameso(C') optimization problem ($C' = \lfloor \frac{\sum_{i=1}^{n-1} C_i}{C_0} \rfloor S_0$)

Step Three: If $j = \left\lceil \frac{W}{C_n} \right\rceil + 1$, stop;

If $f_n^*(l_j) - min_{l_i \in J_+} f_n^*(l_i) \geq \lfloor \frac{\sum_{i=1}^n C_i}{C_0} \rfloor S_0$, stop;

If $f_n^*(l_j) - min_{l_i \in J_+} f_n^*(l_i) < \lfloor \frac{\sum_{i=1}^n C_i}{C_0} \rfloor S_0$, go to step 2.

Output: j, J_+, l^*: $f_n^*(l^*) = min_{l_i \in J_+} f_n^*(l_i)$.

$(x_1*, \ldots, x_{n-1}*, l^*)$: $f_n^*(l^*) = TS(x_1*, \ldots, x_{n-1}*, l^*)$

We can now state the following

Theorem 7. *The output $(x_1^*, \ldots, x_{n-1}^*, l^*)$ of Algorithm 2 generates an optimal solution for $(n+1)-$type vehicle shipping problem.*

3.4 Numerical study

Example 3: We consider $W = 152257$. We assume that the "type 0 " vehicle of the discussion above is the one with the smallest capacity. The capacities and shipping fees listed in table 3.1. We did a numerical study for the above example. The entire algorithm was implemented in Matlab 6.5 by PC IBM X32: Intel Pentium M, processor 1.80GHZ. We started with two types of vehicles only and we increased the number of available types of vehicles, solving each case with the same algorithm. Computational results are summarized in Table 3.2. below. (capacity: c_i(unit: ton), shipping fee s_i(unit:\$1K))

Table 3.1

type i	1	2	3	4	5	6	7
c_i	23	27	31	37	44	49	54
s_i	46	48.6	49.6	51.8	52.8	53.8	54

Table 3.2

Available type	$\lfloor \frac{\sum_{i\neq 1} C_i}{C_1} \rfloor$	Opt. value (\$1K)	Number of Iterations	Running time
1,7	2	152280	4	< 0.01s
1,4,7	3	152277.8	9	< 0.01s
1,2,4 7	5	152277.8	12	< 0.01s
1,2,4, 6,7	7	152277.7	2352	0.04s
1,2,4, 5,6,7	9	152277.6	3127	0.05s
1-7	10	152275.6	4013	0.10s

It is clear that adding more vehicle types would bring a lower optimal value, but it required more iterations and computation times. We still see the running time is not very large even having 7 types of vehicles available. Figure 1 shows the relation between the number of types, $\lfloor \frac{\sum_{i\neq 1} C_i}{C_1} \rfloor$ and the optimal values. More available types will require solving a higher dimension algorithm, and bigger $\lfloor \frac{\sum_{i\neq 1} C_i}{C_1} \rfloor$ will require more iterations of the same dimensional algorithm. In table 3.2 we let both of them to increase. Figure 2 shows the relation between running time and number of iterations. About four thousand iterations can be performed in 0.1 seconds in a modern PC, because the computation of each iteration takes minimal time. Figure 3 shows the relation between the number of available types and the optimal values. It is interesting that adding more available types does not guarantee to reduce costs (sometimes, the result cannot get better), but adding more available types guarantees to increase iterations.

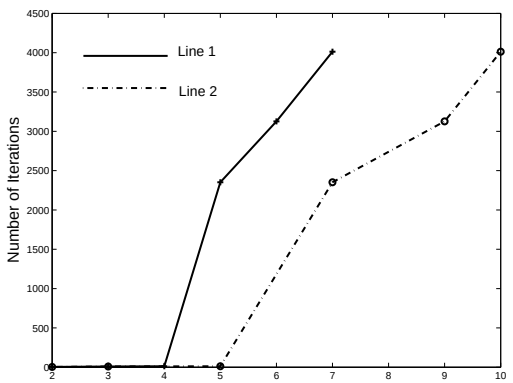

Figure 1 Line 1: No. of types vs. No. of Iterations
Line 2: $\lfloor \frac{\sum_{i\neq 1} C_i}{C_1} \rfloor$ vs. No. of Iterations

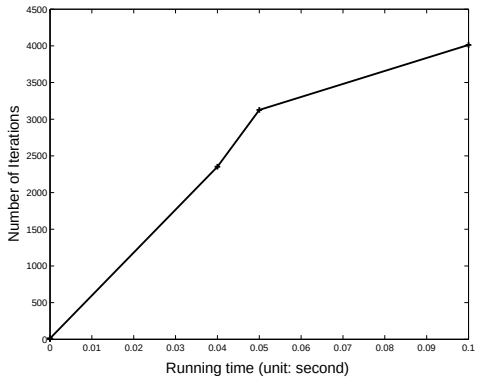

Figure 2 Running Time vs. No. of Iterations

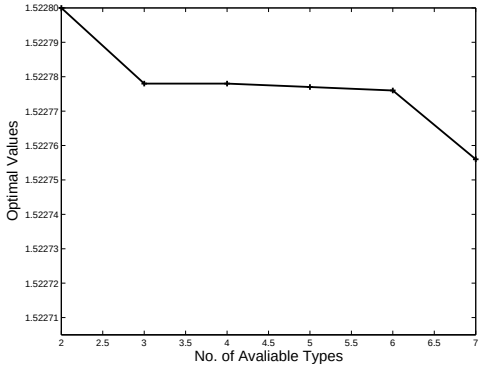

Figure 3 No. of types vs. Optimal Values

Example 4: In this example we demonstrate the relation between the dispersion of the capacities or shipping fees and the running times, or the number of iterations. We assume that the "type 0 " vehicle of the discussion above is the one with the smallest capacity. We consider shipping 152265 tons of products, with available vehicles as listed in table 3.3 below. The running results are also in this table. The entire algorithm was implemented in Matlab 6.5 on an IBM X32 PC with Intel Pentium M, processor 1.80GHZ.

Table 3.3

Available Types	$\frac{C_{max}-C_{min}}{C_{avg}}$	$\frac{S_{max}-S_{min}}{S_{avg}}$
$C_0 = 23, C_1 = 31, C_2 = 39$ $S_0 = 46, S_1 = 60.2, S_2 = 73.1$	0.516	0.453
$C_0 = 23, C_1 = 32, C_2 = 41$ $S_0 = 46, S_1 = 62.1, S_2 = 76.7$	0.563	0.498
$C_0 = 23, C_1 = 33, C_2 = 43$ $S_0 = 46, S_1 = 64.0, S_2 = 80.3$	0.606	0.541
$C_0 = 23, C_1 = 34, C_2 = 45$ $S_0 = 46, S_1 = 65.9, S_2 = 83.9$	0.647	0.581
$C_0 = 23, C_1 = 35, C_2 = 47$ $S_0 = 46, S_1 = 67.8, S_2 = 87.5$	0.686	0.618
$C_0 = 23, C_1 = 36, C_2 = 49$ $S_0 = 46, S_1 = 69.7, S_2 = 91.1$	0.722	0.654
$C_0 = 23, C_1 = 37, C_2 = 51$ $S_0 = 46, S_1 = 71.6, S_2 = 94.7$	0.756	0.688
$C_0 = 23, C_1 = 38, C_2 = 53$ $S_0 = 46, S_1 = 73.5, S_2 = 98.3$	0.789	0.720

Every case in table 3.3 has $\lfloor \frac{C_1+C_2}{C_0} \rfloor S_0 = 3S_0 = 138$. Hence each case is a 3-dim ameso(138) optimization problem.

Figure 4 shows the relation between $\frac{C_{max}-C_{min}}{C_{avg}}$,$\frac{S_{max}-S_{min}}{S_{avg}}$ and the number of iterations. When $\frac{C_{max}-C_{min}}{C_{avg}}$ (or $\frac{S_{max}-S_{min}}{S_{avg}}$) become smaller, the number of iterations increases. When the differences become smaller, the algorithm needs more steps to compute which one is better and more iterations are generated. Figure 5 shows the relation of the optimal values for the five cases. We constructed the data in table 3.3, so that S_0/C_0 in each group is the same, and S_2/C_2 and S_3/C_3 increase gradually. This means the second and the third vehicle type in each group is replaced by two more economical types in the next group.

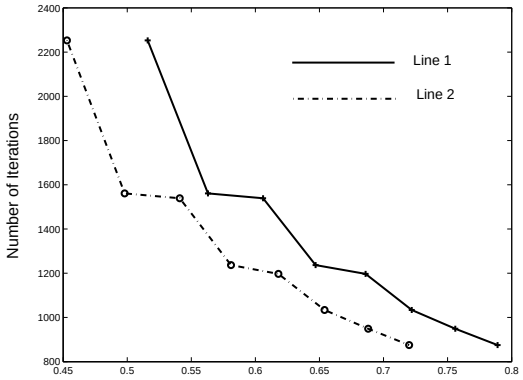

Figure 4 $\frac{c_{max}-c_{min}}{c_{avg}}$ vs. No. of Iterations

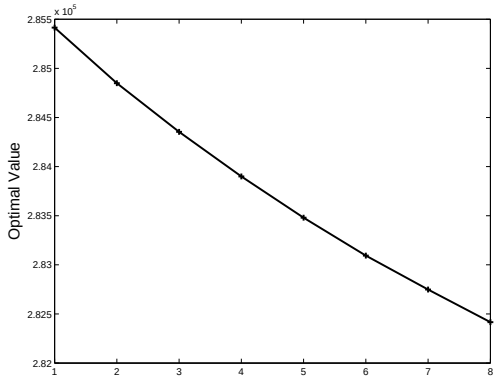

Figure 5 Case No. vs. Optimal Value

4. Conclusion

In this paper, we have discussed the application of Ameso optimization to certain multi- type vehicle shipping problems (MTVS), when they are viewed as Ameso optimization problems. The later is a very interesting class of discrete optimization problems that have the following properties. Their solution can be computed by identifying some subset of their domain and a high dimensional problem can by be decomposed into solving lower dimensional problems. In this paper we have shown how to design an efficient algorithm for the MTVS problems and we have given numerical studies that illustrate the performance of the algorithm.

Acknowledgements: The research was supported by the Rutgers University Business School Research Resources Committee.

References:

[1] Katehakis M. N. and Chen Wen (2007). "Ameso Optimization and its Properties", *Working paper.*

[2] Julien Bramel, David Simchi-Levi (1997) *The Logic of Logistics : Theory, Algorithms, and Applications for Logistics Management*, Springer Series in Operations Research.

[3] Jing-Sheng Song and David D. Yao (2001) *Supply Chain Structures: Coordination, Information and Optimization* Kluwer Academic Publishers.

[4] Yinyu Ye (1997)*Interior-Point Algorithms: Theory and Analysis*, John Wiley & Sons.

[5] Katehakis M. N. and A. F. Veinott Jr. (1987). "The Multi-Armed Bandit problem: decomposition and computation", *Mathematics of Operations Research*, 22 (2), 262–268.

[6] Van Ryzin, Garrett; McGill, Jeff (2000) "Revenue Management Without Forecasting or Optimization: An Adaptive Algorithm for Determining Airline Seat Protection Levels". *Management Science*, 46 (6), 760-776.