輸送計画における並列計算

関西大学工学部管理工学科
山川 栄樹

1 はじめに

複数の供給地点から複数の需要地点へ物資を輸送する場合に，それぞれの供給地点から各需要地点にどれだけの量を運べば輸送費用の総和が最小になるかを求める問題は，輸送問題と呼ばれている．とくに，輸送する物資が1 種類である問題を単一品種の輸送問題，輸送する物資の種類が複数ある問題を多品種の輸送問題という．供給地点と需要地点を結ぶ輸送路を経由するある品種の物資の輸送量を変数とするとき，輸送問題は，いくつかの線形制約条件のもとで輸送費用を表す目的関数を最小化する最適化問題として定式化される．一般に，輸送問題の制約条件は，各供給地点および需要地点における流れ保存則と，各輸送路を経由する物資の量に関する上下限制約から成る．流れ保存則は，ある供給地点に接続する輸送路へ流出する物資の量の合計がその供給地点における物資の供給量に等しいという制約条件群と，ある需要地点に接続する輸送路から流入する物資の量の合計がその需要地点における物資の需要量に等しいという制約条件群で構成される．

単一品種の輸送問題は，供給地点に関する制約条件群と需要地点に関する制約条件群のそれぞれに各変数が係数1 で一つずつ表われるという特徴的な構造をもつ．このような構造を利用して，大規模な輸送問題を効率的に解くさまざまなアルゴリズムが提案されている．たとえば，輸送費用が各輸送路を経由する物資の輸送量の1 次関数で表現されるとき，輸送問題は線形計画問題として定式化されるが，飛び石法[8]と呼ばれる方法を用いると，通常の単体法の手続きよりもはるかに少ない計算量で解を求めることができる．一方，輸送費用が輸送量について分離可能な2 次または凸関数で与えられる場合には，Row-Action 法[13]と呼ばれる反復解法が有効である．この方法は，供給地点に関する制約条件のラグランジュ乗数群と需要地点に関する制約条件のラグランジュ乗数群をそれぞれ独立に更新できるため，大規模並列計算機を用いて効率的に実行できる．

多品種輸送問題の制約条件は，全体としてblock-angular と呼ばれる特別な構造をもっているため，本来逐次的なアルゴリズムを適用して解を求める場合でも，その主要な計算が各ブロックごとに並列的に実行できることがある．また，各ブロックに表われる流れ保存則を表す制約条件は，単一品種の輸送問題と同じ構造的特徴をもっているため，各ブロックの計算も並列的に実行できる場合が少なくない．このようなタイプの並列アルゴリズムは，VPP のような少数の高性能プロセッサを構成要素とするベクトル並列計算機を用いると，効率よく実行できる．

本稿では，輸送費用が分離可能な凸2 次関数で与えられる輸送問題に対する並列アルゴリズムについて述べ，ベクトル並列計算機における実行方法を示すとともに，VPP を用いた計算実験結果を紹介する．まず，第2 節において，単一品種の2 次輸送問題に対する並列アルゴリズムとして，並列型降下法と交互方向乗数法を取上げる．これらのアルゴリズムは，供給地点と需要地点および輸送路に関する情報の計算を並列的に実行できる粒度の細かい並列アルゴリズムである．つぎに，第3 節において，多品種の2 次輸送問題に対する並列アルゴリズムとして，並列型主双対内点法と予測子修正子近接乗数法を取上げる．これらのアルゴリズムは，各品種について独立な処理を並列的に実行できる粒度の粗い並列アルゴリズムである．最後に，第4 節において本稿をまとめる．

本稿では，つぎのような表記法を用いる．実数全体の集合を $\Re$，実数を成分とする n 次元ベクトル全体の集合を $\Re^n$ と書く．ベクトルは列ベクトルとし，行ベクトルを示すときは，転置記号 $\top$ を用いて $x^\top$ のように表す．また，二つのベクトル $x \in \Re^m$ と $y \in \Re^n$ をつなぎ合わせた $m+n$ 次元ベクトルは，表記の煩雑さを避けるため，$(x^\top, y^\top)^\top$ と書くかわりに $(x, y) \in \Re^{m+n}$ と表す．ベクトルの成分は下つき添え字を用いて $x_1, \ldots, x_m$ あるいは $x_i\,(i = 1, \ldots, m)$ と書き，ベクトルの列は括弧つきの上つき添え字を用いて $x^{(0)}, x^{(1)}, \ldots$ と書く．

2　単一品種の 2 次輸送問題に対する並列計算

つぎのように定式化される単一品種の 2 次輸送問題を考える．

$$
\begin{aligned}
\text{目的関数：}\quad & \sum_{(i,j)\in E} \left\{ \frac{\theta_{ij}}{2}(x_{ij})^2 + \pi_{ij}\, x_{ij} \right\} \rightarrow \text{最小} \\
\text{制約条件：}\quad & \sum_{j\in J_i} x_{ij} = \alpha_i \qquad (i = 1, \ldots, m) \\
& \sum_{i\in I_j} x_{ij} = \beta_j \qquad (j = 1, \ldots, n) \\
& x_{ij} \geqq 0 \qquad ((i,j) \in E)
\end{aligned}
\tag{1}
$$

ただし，$\{1, \ldots, m\}$ と $\{1, \ldots, n\}$ はそれぞれ供給地点と需要地点の集合，E はそれらを結ぶ輸送路の集合である．また，J_i は供給地点 i と輸送路で結ばれた需要地点の集合，I_j は需要地点 j と輸送路で結ばれた供給地点の集合であり，それぞれ

$$
\begin{aligned}
J_i &= \{ j \,|\, (i,j) \in E \} \qquad (i = 1, \ldots, m) \\
I_j &= \{ i \,|\, (i,j) \in E \} \qquad (j = 1, \ldots, n)
\end{aligned}
$$

と記述できる．変数 x_{ij} は，供給地点 i と需要地点 j を結ぶ輸送路を経由する物資の輸送量を表している．また，α_i は供給地点 i における物資の供給量，β_j は需要地点 j における物資の需要量を表しており，$\sum_{i=1}^m \alpha_i = \sum_{j=1}^n \beta_j > 0$ と仮定する．さらに，θ_{ij}, π_{ij} は供給地点 i から需要地点 j へ物資を輸送した場合の費用を表す 2 次関数の係数であり，$\theta_{ij} > 0\,((i,j) \in E)$ と仮定する．この節では，問題 (1) に対する並列アルゴリズムである並列型降下法と交互方向乗数法について述べ，ベクトル並列計算機による実行方法を示すとともに，VPP による計算実験結果を紹介する．

2.1　並列型降下法

並列型降下法 [5] は，凸計画問題の双対問題がもつ特殊な構造と，信頼領域法の考え方を利用した並列アルゴリズムである．表記を簡単にするために，$x_{ij}\,((i,j) \in E)$ を成分とするベクトルを x と書き，関数 f, g_i, h_j を

$$
f(x) = \begin{cases} \displaystyle\sum_{(i,j)\in E} \left\{ \frac{\theta_{ij}}{2}(x_{ij})^2 + \pi_{ij}\, x_{ij} \right\}, & x_{ij} \geqq 0\ ((i,j) \in E)\ \text{のとき} \\ +\infty, & \text{それ以外のとき} \end{cases}
$$

$$
g_i(x) = \begin{cases} 0, & \displaystyle\sum_{j\in J_i} x_{ij} = \alpha_i\ \text{のとき} \\ +\infty, & \text{それ以外のとき} \end{cases} \qquad (i = 1, \ldots, m)
$$

$$
h_j(x) = \begin{cases} 0, & \displaystyle\sum_{i\in I_j} x_{ij} = \beta_j\ \text{のとき} \\ +\infty, & \text{それ以外のとき} \end{cases} \qquad (j = 1, \ldots, n)
$$

で定義すると，単一品種の 2 次輸送問題 (1) は

$$\text{目的関数：}\quad f(x)+\sum_{i=1}^{m} g_i(x)+\sum_{j=1}^{n} h_j(x) \;\rightarrow\; \text{最小} \tag{2}$$

と書き直すことができる．問題 (2) に対する Fenchel の双対問題 [10] は

$$\text{目的関数：}\quad \Phi(\nu,\mu)\equiv f^*\left(\textstyle\sum_{i=1}^{m}\nu_i+\sum_{j=1}^{n}\mu_j\right)+\displaystyle\sum_{i=1}^{m} g_i^*(-\nu_i)+\sum_{j=1}^{n} h_j^*(-\mu_j) \;\rightarrow\; \text{最小} \tag{3}$$

で与えられる．ただし，$\nu_i\in\Re^{|E|}\,(i=1,\ldots,m)$, $\mu_j\in\Re^{|E|}\,(j=1,\ldots,n)$ に対して $\nu=(\nu_1,\ldots,\nu_m)$, $\mu=(\mu_1,\ldots,\mu_n)$ である．また，f^*, g_i^*, h_j^* はそれぞれ f, g_i, h_j の共役関数 [10] であり，

$$\begin{array}{rcll} f^*(\upsilon) & = & \sup\{x^\top\upsilon \;-\; f\,(x)\mid x\in\Re^{|E|}\} & \\ g_i^*(\nu_i) & = & \sup\{x^\top\nu_i \;-\; g_i(x)\mid x\in\Re^{|E|}\} & (i=1,\ldots,m) \\ h_j^*(\mu_j) & = & \sup\{x^\top\mu_j - h_j(x)\mid x\in\Re^{|E|}\} & (j=1,\ldots,n) \end{array}$$

で定義される．問題 (2) の最適解 x^* は，双対問題 (3) の最適解 (ν^*,μ^*) を用いて次式で計算できる．

$$x^*=\nabla f^*\left(\textstyle\sum_{i=1}^{m}\nu_i^*+\sum_{j=1}^{n}\mu_j^*\right)$$

問題 (3) の目的関数 Φ を現在の探索点 $(\nu^{(k)},\mu^{(k)})$ において近似したモデル関数

$$\begin{aligned}\Psi(\nu^{(k)},\mu^{(k)};d,e)\equiv f^*\left(\textstyle\sum_{i=1}^{m}\nu_i^{(k)}+\sum_{j=1}^{n}\mu_j^{(k)}\right)&+\sum_{i=1}^{m}\left\{(x^{(k)})^\top d_i+g_i^*(-\nu_i^{(k)}-d_i)\right\}\\ &+\sum_{j=1}^{n}\left\{(x^{(k)})^\top e_j+h_j^*(-\mu_j^{(k)}-e_j)\right\}\end{aligned}$$

を考えよう．ただし，$d_i\in\Re^{|E|}\,(i=1,\ldots,m)$, $e_j\in\Re^{|E|}\,(j=1,\ldots,n)$ に対して $d=(d_1,\ldots,d_m)$, $e=(e_1,\ldots,e_n)$ であり，$x^{(k)}$ は次式で定義される．

$$x^{(k)}=\nabla f^*\left(\textstyle\sum_{i=1}^{m}\nu_i^{(k)}+\sum_{j=1}^{n}\mu_j^{(k)}\right) \tag{4}$$

並列型降下法の各反復では，関数 Φ を直接最小化する代わりに，関数 Ψ をステップサイズを制限する凸 2 次項とともに最小化する部分問題

$$\text{目的関数：}\quad \frac{\lambda^{(k)}}{2}\left(\sum_{i=1}^{m}\|d_i\|^2+\sum_{j=1}^{n}\|e_j\|^2\right)+\Psi(\nu^{(k)},\mu^{(k)};d,e) \;\rightarrow\; \text{最小} \tag{5}$$

の解 $(d^{(k)},e^{(k)})$ を求めて，つぎの探索点の候補 $(\overline{\nu}^{(k)},\overline{\mu}^{(k)})$ を

$$\begin{array}{ll} \overline{\nu}_i^{(k)}=\nu_i^{(k)}+d_i^{(k)} & (i=1,\ldots,m) \\ \overline{\mu}_j^{(k)}=\mu_j^{(k)}+e_j^{(k)} & (j=1,\ldots,n) \end{array}$$

により定める．ただし，$\lambda^{(k)}$ は正のパラメータであり，信頼領域法 [3] と同じような考え方により更新する．部分問題 (5) は，その目的関数から定数項 $f^*(\sum_{i=1}^{m}\nu_i^{(k)}+\sum_{j=1}^{n}\mu_j^{(k)})$ を除去することによって，$i=1,\ldots,m$ のそれぞれに対する独立な m 個の小さな部分問題

$$\text{目的関数：}\quad \frac{\lambda^{(k)}}{2}\|d_i\|^2+(x^{(k)})^\top d_i+g_i^*(-\nu_i^{(k)}-d_i) \;\rightarrow\; \text{最小} \tag{6}$$

と，$j=1,\ldots,n$ のそれぞれに対する独立な n 個の小さな部分問題

$$\text{目的関数：}\quad \frac{\lambda^{(k)}}{2}\|e_j\|^2+(x^{(k)})^\top e_j+h_j^*(-\mu_j^{(k)}-e_j) \;\rightarrow\; \text{最小} \tag{7}$$

に分割されるため，並列計算機を用いて効率的に解くことができる．

なお，式 (4) で定められる $x^{(k)}$ の値は，凸計画問題

$$\text{目的関数：}\quad f(x) - \left(\sum_{i=1}^{m} \nu_i^{(k)} + \sum_{j=1}^{n} \mu_j^{(k)}\right)^{\top} x \;\rightarrow\; \text{最小}$$

を解くことによって求められる．また，問題 (6) の解 $d_i^{(k)}$ と問題 (7) の解 $e_j^{(k)}$ は，それぞれの問題の Fenchel の双対問題

$$\text{目的関数：}\quad \frac{1}{2\lambda^{(k)}}\|y_i - x^{(k)}\|^2 + (\nu_i^{(k)})^{\top} y_i + g_i(y_i) \;\rightarrow\; \text{最小}$$

と

$$\text{目的関数：}\quad \frac{1}{2\lambda^{(k)}}\|z_j - x^{(k)}\|^2 + (\mu_j^{(k)})^{\top} z_j + h_j(z_j) \;\rightarrow\; \text{最小}$$

の解 $y_i^{(k)}, z_j^{(k)}$ を求め，つぎに

$$\begin{aligned} d_i^{(k)} &= \frac{1}{\lambda^{(k)}}(y_i^{(k)} - x^{(k)}) \qquad (i = 1, \ldots, m) \\ e_j^{(k)} &= \frac{1}{\lambda^{(k)}}(z_j^{(k)} - x^{(k)}) \qquad (j = 1, \ldots, n) \end{aligned}$$

とおくことによって求めることができる．

関数 f, g_i, h_j の定義を考慮すると，単一品種の 2 次輸送問題 (1) に対する並列型降下法は，つぎのように記述できる．

アルゴリズム 1.

ステップ 0: パラメータ $0 < \rho_0 < \rho_1 < \rho_2 < 1$ と $\gamma > 1$ を選ぶ．凸 2 次項の係数の初期値 $\lambda^{(0)} > 0$ を定め，初期探索点 $v_i^{(0)} \in \Re\,(i = 1, \ldots, m)$ と $w_j^{(0)} \in \Re\,(j = 1, \ldots, n)$ を選ぶ．初期探索点における最適解の推定値を

$$x_{ij}^{(0)} = \max\left\{ 0, -\frac{\pi_{ij} + v_i^{(0)} + w_j^{(0)}}{\theta_{ij}} \right\} \qquad ((i,j) \in E)$$

により計算し，$k = 0$ とおく．

ステップ 1: 等式制約条件の残差

$$\begin{aligned} r_i^{(k)} &= \sum_{j \in J_i} x_{ij}^{(k)} - \alpha_i \qquad (i = 1, \ldots, m) \\ s_j^{(k)} &= \sum_{i \in I_j} x_{ij}^{(k)} - \beta_j \qquad (j = 1, \ldots, n) \end{aligned}$$

を計算する．もし

$$\begin{aligned} r_i^{(k)} &= 0 \qquad (i = 1, \ldots, m) \\ s_j^{(k)} &= 0 \qquad (j = 1, \ldots, n) \end{aligned}$$

ならば， $x_{ij}^{(k)}\,((i,j) \in E)$ を問題 (1) の解として終了する．さもなければ，つぎの探索点の候補 $\overline{v}_i^{(k)}\,(i = 1, \ldots, m)$ と $\overline{w}_j^{(k)}\,(j = 1, \ldots, n)$ を

$$\begin{aligned} \overline{v}_i^{(k)} &= v_i^{(k)} + \frac{r_i^{(k)}}{\lambda^{(k)}|J_i|} \qquad (i = 1, \ldots, m) \\ \overline{w}_j^{(k)} &= w_j^{(k)} + \frac{s_j^{(k)}}{\lambda^{(k)}|I_j|} \qquad (j = 1, \ldots, n) \end{aligned}$$

により求める．さらに，

$$\overline{x}_{ij}^{(k)} = \max\left\{ 0, -\frac{\pi_{ij} + \overline{v}_i^{(k)} + \overline{w}_j^{(k)}}{\theta_{ij}} \right\} \qquad ((i,j) \in E)$$

とおく．

ステップ 2: モデル関数 Ψ の変化量と関数 Φ の実際の変化量を，それぞれ

$$\begin{aligned}
\Delta\Psi^{(k)} &= \sum_{i=1}^{m} (\overline{v}_i^{(k)} - v_i^{(k)}) r_i^{(k)} + \sum_{j=1}^{n} (\overline{w}_j^{(k)} - w_j^{(k)}) s_j^{(k)}, \\
\Delta\Phi^{(k)} &= \sum_{i=1}^{m} (v_i^{(k)} - \overline{v}_i^{(k)}) \alpha_i^{(k)} + \sum_{j=1}^{n} (w_j^{(k)} - \overline{w}_j^{(k)}) \beta_j^{(k)} \\
&\qquad + \sum_{(i,j)\in E} \frac{\theta_{ij}}{2} \left\{ \left(x_{ij}^{(k)}\right)^2 - \left(\overline{x}_{ij}^{(k)}\right)^2 \right\}
\end{aligned}$$

により計算する．

ステップ 3: 二つの変化量の比の値

$$q^{(k)} = \frac{\Delta\Phi^{(k)}}{\Delta\Psi^{(k)}}$$

を計算する．もし $q^{(k)} < \rho_0$ ならば，候補点 $\overline{v}_i^{(k)}\,(i = 1, \ldots, m)$ と $\overline{w}_j^{(k)}\,(j = 1, \ldots, n)$ を棄却し，

$$\begin{aligned}
v_i^{(k+1)} &= v_i^{(k)} \qquad (i = 1, \ldots, m) \\
w_j^{(k+1)} &= w_j^{(k)} \qquad (j = 1, \ldots, n) \\
x_{ij}^{(k+1)} &= x_{ij}^{(k)} \qquad ((i,j) \in E)
\end{aligned}$$

とおく．さもなければ，候補点 $\overline{v}_i^{(k)}\,(i = 1, \ldots, m)$ と $\overline{w}_j^{(k)}\,(j = 1, \ldots, n)$ を受理し，

$$\begin{aligned}
v_i^{(k+1)} &= \overline{v}_i^{(k)} \qquad (i = 1, \ldots, m) \\
w_j^{(k+1)} &= \overline{w}_j^{(k)} \qquad (j = 1, \ldots, n) \\
x_{ij}^{(k+1)} &= \overline{x}_{ij}^{(k)} \qquad ((i,j) \in E)
\end{aligned}$$

とおく．

ステップ 4: 2 次項の係数を

$$\lambda^{(k+1)} = \begin{cases} \gamma\,\lambda^{(k)}, & q^{(k)} \leqq \rho_1 \quad \text{のとき} \\ \lambda^{(k)}, & \rho_1 < q^{(k)} \leqq \rho_2 \quad \text{のとき} \\ \lambda^{(k)} / \gamma, & \rho_2 < q^{(k)} \quad \text{のとき} \end{cases}$$

により更新し，$k = k + 1$ としてステップ 1 に戻る． □

アルゴリズム 1 のステップ 1 は，三つの異なるタイプの処理から構成される．第 1 の処理では，輸送路に関する情報 $x_{ij}^{(k)}\,((i,j) \in E)$ を集約して，その輸送路が接続する供給地点に関する情報 $r_i^{(k)}\,(i = 1, \ldots, m)$ と需要地点に関する情報 $s_j^{(k)}\,(j = 1, \ldots, n)$ を計算する．第 2 の処理は，制約条件の残差 $r_i^{(k)}\,(i = 1, \ldots, m)$ と $s_j^{(k)}\,(j = 1, \ldots, n)$ をもとに，対応するラグランジュ乗数の候補値 $\overline{v}_i^{(k)}\,(i = 1, \ldots, m)$ と $\overline{w}_j^{(k)}\,(j = 1, \ldots, n)$ を求めるものであり，供給地点と需要地点のそれぞれに対して独立な処理である．第 3 の処理では，供給地点に関する情報 $\overline{v}_i^{(k)}\,(i = 1, \ldots, m)$ と需要地点に関する情報 $\overline{w}_j^{(k)}\,(j = 1, \ldots, n)$ をもとに，それらを結ぶ輸送路に関する情報 $\overline{x}_{ij}^{(k)}\,((i,j) \in E)$ を計算する．一方，ステップ 2 では，関

表 1: VPP500 による並列型降下法の計算実験結果

問題サイズ			反復回数	計算時間 [秒] (並列化効率 e_p)			
供給地点	需要地点	輸送路		$p=1$	$p=2$	$p=4$	$p=8$
2048	2048	16384	144	0.325	0.213 (0.763)	0.140 (0.580)	0.111 (0.366)
8192	8192	65536	171	1.695	0.969 (0.875)	0.533 (0.795)	0.336 (0.631)
32768	32768	262144	453	17.90	9.946 (0.900)	5.290 (0.846)	3.022 (0.740)
131072	131072	1048576	3852	527.5	293.6 (0.898)	158.1 (0.834)	90.03 (0.732)
1024	1024	16384	42	0.092	0.061 (0.754)	0.041 (0.561)	0.033 (0.348)
4096	4096	65536	47	0.420	0.236 (0.890)	0.132 (0.795)	0.082 (0.640)
16384	16384	262144	162	5.870	3.200 (0.917)	1.677 (0.875)	0.920 (0.798)
65536	65536	1048576	376	51.23	28.00 (0.915)	14.58 (0.878)	7.873 (0.813)

表 2: VPP800 による並列型降下法の計算実験結果

問題サイズ			計算時間 [秒] (並列化効率 e_p)					
供給地点	需要地点	輸送路	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$	$p=32$
2048	2048	16384	0.051	0.042 (0.607)	0.035 (0.364)	0.040 (0.159)	0.058 (0.055)	0.102 (0.016)
8192	8192	65536	0.288	0.178 (0.809)	0.107 (0.673)	0.085 (0.425)	0.094 (0.191)	0.137 (0.066)
32768	32768	262144	3.026	1.794 (0.843)	0.990 (0.764)	0.623 (0.607)	0.492 (0.384)	0.540 (0.175)
131072	131072	1048576	87.60	51.42 (0.852)	28.56 (0.767)	17.34 (0.631)	12.19 (0.449)	10.44 (0.262)
1024	1024	16384	0.015	0.013 (0.577)	0.011 (0.341)	0.013 (0.144)	0.018 (0.052)	0.031 (0.015)
4096	4096	65536	0.066	0.041 (0.805)	0.026 (0.635)	0.022 (0.375)	0.025 (0.165)	0.038 (0.054)
16384	16384	262144	0.965	0.564 (0.855)	0.306 (0.788)	0.186 (0.649)	0.151 (0.399)	0.171 (0.176)
65536	65536	1048576	9.630	5.198 (0.926)	2.760 (0.872)	1.567 (0.768)	1.018 (0.591)	0.829 (0.363)

数 Ψ と Φ の変化量を求めるために，供給地点と需要地点に関する情報の総和と，それらを結ぶ輸送路に関する情報の総和を計算している．

現実の大規模な輸送問題では，供給地点や需要地点の数は膨大なものの，各地点に接続する輸送路のなかで実際に物資を運搬する輸送路の数は非常に少ない．そこで，輸送路に関する情報を，第 1 次元が供給地点の番号で第 2 次元がそれぞれの供給地点から実際に物資が搬出される輸送路につけられた連番に対応するような 2 次元配列に格納して，供給地点に関する情報を格納する 1 次元配列とともに利用可能なプロセッサに分割して保持する．輸送路に関する情報は，第 1 次元が需要地点の番号で第 2 次元がそれぞれの需要地点へ実際に物資が搬入される輸送路につけられた連番に対応するようなもう一つの 2 次元配列にも重複して格納し，需要地点に関する情報を格納する 1 次元配列とともに利用可能なプロセッサに分割して保持する．これによって，ステップ 1 における $r_i^{(k)}$ と $s_j^{(k)}$ の計算は，各地点に接続する（少数の）輸送路に関する情報の総和演算を対応するプロセッサ上で逐次的に行い，独立に実行できる（多数の）供給地点と需要地点についての演算をベクトル的かつ並列的に処理することによって効率的に実現できる．なお，輸送路に関する情報を保持する二つの 2 次元配列は，対応する要素の値が常に一致していなければならない．しかし，ステップ 1 における $\overline{x}_{ij}^{(k)}$ $((i,j) \in E)$ の計算は非常に簡単であるため，一方のデータを他方に転送するのではなく，双方の配列要素の値を同時に更新することにする．また，各プロセッサに分配して格納された $\overline{v}_i^{(k)}$ と $\overline{w}_j^{(k)}$ の値を，`UNIFY` 命令 [4] を用いて他のすべてのプロセッサに事前に転送しておけば，$\overline{x}_{ij}^{(k)}$ の計算はそれぞれのプロセッサがもつデータだけを用いて実行できる．

計算実験は，ランダムに生成した単一品種の 2 次輸送問題 (1) に対して行った．VPP500 および VPP800 による計算実験の結果を，表 1 および表 2 に示す．表の各欄の数値は，それぞれのサイズの 5 題の問題例に対する実験結果の平均値である．また，表 の “計算時間” 欄に掲げた p の値は実際に使用したプロセッサの数であり，e_p の値は次式で定義される並列化効率の値である．

$$e_p = \frac{\text{1 個のプロセッサによる計算時間}}{p \text{ 個のプロセッサによる計算時間} \times p} \tag{8}$$

一般に $e_p \leqq 1$ となるが，$e_p = 1$ に近いほど並列処理が有効に行われていると判断できる．表 1 および表 2 より，使用するプロセッサ数が増えるにつれて並列処理の効率はやや低下するが，問題サイズが大きくなると，並列処理の効率はしだいに改善することが確かめられる．

2.2 交互方向乗数法

交互方向乗数法 [2] は，非線形計画問題に対する代表的なアルゴリズムである乗数法に，作用素分割法の考え方を組み込むことによって得られる並列アルゴリズムである．関数 f, g_i, h_j を

$$\begin{aligned} f(x) &= \sum_{(i,j)\in E} \left\{ \frac{\theta_{ij}}{2}(x_{ij})^2 + \pi_{ij}\, x_{ij} \right\} \\ g_i(x_{i\cdot}) &= \sum_{j\in J_i} x_{ij} - \alpha_i \qquad (i = 1, \ldots, m) \\ h_j(x_{\cdot j}) &= \sum_{i\in I_j} x_{ij} - \beta_j \qquad (j = 1, \ldots, n) \end{aligned}$$

で定義すると，単一品種の 2 次輸送問題 (1) は

$$\begin{array}{ll} \text{目的関数：} & f(x) \rightarrow \text{最小} \\ \text{制約条件：} & g_i(x_{i\cdot}) = 0 \qquad (i = 1, \ldots, m) \\ & h_j(x_{\cdot j}) = 0 \qquad (j = 1, \ldots, n) \\ & x \geqq 0 \end{array} \tag{9}$$

と書ける．ただし，$i=1,\ldots,m$ のそれぞれに対して $x_{i\cdot}\in\Re^{|J_i|}$ は $x_{ij}\,(j\in J_i)$ を成分とするベクトル，$j=1,\ldots,n$ のそれぞれに対して $x_{\cdot j}\in\Re^{|I_j|}$ は $x_{ij}\,(i\in I_j)$ を成分とするベクトルである．さらに，$y_i\in\Re^{|J_i|}\,(i=1,\ldots,m)$ と $z_j\in\Re^{|I_j|}\,(j=1,\ldots,n)$ に対して $y=(y_1,\ldots,y_m)$, $z=(z_1,\ldots,z_n)$ とおき，関数 F,G,H を

$$
\begin{aligned}
F(x) &= \begin{cases} f(x), & x_{ij}\geqq 0\ ((i,j)\in E)\ \text{のとき} \\ +\infty, & \text{それ以外のとき} \end{cases} \\
G(y) &= \begin{cases} 0, & g_i(y_i)=0\ (i=1,\ldots,m)\ \text{のとき} \\ +\infty, & \text{それ以外のとき} \end{cases} \\
H(z) &= \begin{cases} 0, & h_j(z_j)=0\ (j=1,\ldots,n)\ \text{のとき} \\ +\infty, & \text{それ以外のとき} \end{cases}
\end{aligned}
$$

で定義すると，問題 (9) は

$$
\begin{aligned}
&\text{目的関数：}\quad F(x)+G(y)+H(z)\ \rightarrow\ \text{最小} \\
&\text{制約条件：}\quad x_{i\cdot}=y_i \qquad (i=1,\ldots,m) \\
&\qquad\qquad\ \ x_{\cdot j}=z_j \qquad (j=1,\ldots,n)
\end{aligned}
\tag{10}
$$

と書き直すことができる．

問題 (10) に対する拡張ラグランジュ関数を次式で定義する．

$$
\begin{aligned}
L_\lambda(x,y,z;\nu,\mu) = F(x)+G(y)+H(z)+\sum_{i=1}^m \nu_i^\top(x_{i\cdot}-y_i)+\sum_{j=1}^n \mu_j^\top(x_{\cdot j}-z_j) \\
+\frac{\lambda}{2}\Big(\sum_{i=1}^m \|x_{i\cdot}-y_i\|^2+\sum_{j=1}^n \|x_{\cdot j}-z_j\|^2\Big)
\end{aligned}
$$

ただし，$\nu_i\in\Re^{|J_i|}\,(i=1,\ldots,m)$ と $\mu_j\in\Re^{|I_j|}\,(j=1,\ldots,n)$ に対して $\nu=(\nu_1,\ldots,\nu_m)$, $\mu=(\mu_1,\ldots,\mu_n)$ である．また，$\lambda>0$ は十分大きな値をとるパラメータである．通常の乗数法 [9] の各反復では，拡張ラグランジュ関数をもとの問題のすべての変数に関して同時に最小化する．一方，交互方向乗数法の各反復では，作用素分割法 [11] の考え方を組み込んで，もとの問題の一部の変数の値を固定して他の変数に関して拡張ラグランジュ関数を最小化する．具体的には，現在の反復点 $(x^{(k)},y^{(k)},z^{(k)};\nu^{(k)},\mu^{(k)})$ において，変数 y,z の値を固定した制約なし問題

$$
\text{目的関数：}\quad L_\lambda(x,y^{(k)},z^{(k)};\nu^{(k)},\mu^{(k)})\ \rightarrow\ \text{最小} \tag{11}
$$

の解 $x^{(k+1)}$ を求めたあと，変数 x の値を固定した制約なし問題

$$
\text{目的関数：}\quad L_\lambda(x^{(k+1)},y,z;\nu^{(k)},\mu^{(k)})\ \rightarrow\ \text{最小} \tag{12}
$$

を解いて，その解を $y^{(k+1)},z^{(k+1)}$ とおく．なお，ラグランジュ乗数は，乗数法と同様に式

$$
\begin{aligned}
\nu_i^{(k+1)} &= \nu_i^{(k)}+\lambda(x_{i\cdot}^{(k+1)}-y_i^{(k+1)}) \qquad (i=1,\ldots,m) \\
\mu_j^{(k+1)} &= \mu_j^{(k)}+\lambda(x_{\cdot j}^{(k+1)}-z_j^{(k+1)}) \qquad (j=1,\ldots,n)
\end{aligned}
$$

により更新する．問題 (11) と (12) を同時に解くことはできないが，関数 L_λ と F の定義より，問題 (11) は $(i,j)\in E$ のそれぞれに対する独立な $|E|$ 個の 1 変数の 2 次計画問題

$$
\begin{aligned}
&\text{目的関数：}\quad \left\{\frac{\theta_{ij}}{2}(x_{ij})^2+\pi_{ij}\,x_{ij}\right\}+\left(\nu_{ij}^{(k)}+\mu_{ij}^{(k)}\right)x_{ij} \\
&\qquad\qquad\qquad +\frac{\lambda}{2}\left\{(x_{ij}-y_{ij}^{(k)})^2+(x_{ij}-z_{ij}^{(k)})^2\right\}\ \rightarrow\ \text{最小} \\
&\text{制約条件：}\quad x_{ij}\geqq 0
\end{aligned}
\tag{13}
$$

に分解される．同様に，関数 L_λ と G, H の定義より，問題 (12) は $i = 1, \ldots, m$ のそれぞれに対する独立な m 個の小さな 2 次計画問題

$$\begin{array}{ll} \text{目的関数：} & \dfrac{\lambda}{2}\|x_{i\cdot}^{(k+1)} - y_i\|^2 - (\nu_i^{(k)})^\top y_i \ \rightarrow \text{最小} \\ \text{制約条件：} & \displaystyle\sum_{j \in J_i} y_{ij} = \alpha_i \end{array} \tag{14}$$

と，$j = 1, \ldots, n$ のそれぞれに対する独立な n 個の小さな 2 次計画問題

$$\begin{array}{ll} \text{目的関数：} & \dfrac{\lambda}{2}\|x_{\cdot j}^{(k+1)} - z_j\|^2 - (\mu_j^{(k)})^\top z_j \ \rightarrow \text{最小} \\ \text{制約条件：} & \displaystyle\sum_{i \in I_j} z_{ij} = \beta_j \end{array} \tag{15}$$

に分解されるため，並列計算機を用いて効率的に解くことができる．

問題 (13), (14), (15) はいずれも解析的に解けることに注意すると，単一品種の 2 次輸送問題 (1) に対する交互方向乗数法はつぎのように記述できる．

アルゴリズム 2.

ステップ 0: 拡張ラグランジュ関数のパラメータ $\lambda > 0$ を定める．初期探索点 $x_{ij}^{(0)} \geqq 0\,((i,j) \in E)$, $v_i^{(0)} \in \Re\,(i = 1, \ldots, m)$, $w_j^{(0)} \in \Re\,(j = 1, \ldots, n)$ を選ぶ．等式制約条件の残差

$$\begin{aligned} r_i^{(0)} &= \sum_{j \in J_i} x_{ij}^{(0)} - \alpha_i \qquad (i = 1, \ldots, m) \\ s_j^{(0)} &= \sum_{i \in I_j} x_{ij}^{(0)} - \beta_j \qquad (j = 1, \ldots, n) \end{aligned}$$

を計算し，$k = 0$ とおく．

ステップ 1: まず

$$\overline{x}_{ij}^{(k)} = 2\lambda x_{ij}^{(k)} - \pi_{ij} \qquad ((i,j) \in E)$$

とおき，

$$\begin{aligned} \overline{v}_i^{(k)} &= v_i^{(k)} + \frac{\lambda r_i^{(k)}}{|J_i|} \qquad (i = 1, \ldots, m) \\ \overline{w}_j^{(k)} &= w_j^{(k)} + \frac{\lambda s_j^{(k)}}{|I_j|} \qquad (j = 1, \ldots, n) \end{aligned}$$

を用いて輸送量を次式で更新する．

$$x_{ij}^{(k+1)} = \max\left\{0, \frac{\overline{x}_{ij}^{(k)} - \overline{v}_i^{(k)} - \overline{w}_j^{(k)}}{2\lambda + \theta_{ij}}\right\} \qquad ((i,j) \in E)$$

ステップ 2: 等式制約条件の残差

$$\begin{aligned} r_i^{(k+1)} &= \sum_{j \in J_i} x_{ij}^{(k+1)} - \alpha_i \qquad (i = 1, \ldots, m) \\ s_j^{(k+1)} &= \sum_{i \in I_j} x_{ij}^{(k+1)} - \beta_j \qquad (j = 1, \ldots, n) \end{aligned}$$

を計算する．もし

$$\begin{aligned} r_i^{(k+1)} &= 0 \qquad (i = 1, \ldots, m) \\ s_j^{(k+1)} &= 0 \qquad (j = 1, \ldots, n) \end{aligned}$$

ならば， $x_{ij}^{(k+1)}\,((i,j)\in E)$ を問題 (1) の解として終了する．さもなければ，ラグランジュ乗数を次式で更新する．

$$v_i^{(k+1)} = v_i^{(k)} + \frac{\lambda r_i^{(k+1)}}{|J_i|} \qquad (i=1,\ldots,m)$$
$$w_j^{(k+1)} = w_j^{(k)} + \frac{\lambda s_j^{(k+1)}}{|I_j|} \qquad (j=1,\ldots,n)$$

ステップ 3: $k=k+1$ としてステップ 1 に戻る． □

アルゴリズム 2 は，四つの異なるタイプの処理から構成される．まず，ステップ 1 における $\overline{x}_{ij}^{(k)}$ の計算は，輸送路 $(i,j)\in E$ について独立に実行できる．また，ステップ 1 における $\overline{v}_i^{(k)}, \overline{w}_j^{(k)}$ の計算とステップ 2 における $v_i^{(k+1)}, w_j^{(k+1)}$ の計算は，供給地点 $i=1,\ldots,m$ と需要地点 $j=1,\ldots,n$ のそれぞれに対して独立に実行できる．一方，ステップ 1 における輸送量の更新式は，本質的に供給地点に関する情報 $\overline{v}_i^{(k)}\,(i=1,\ldots,m)$ と需要地点に関する情報 $\overline{w}_j^{(k)}\,(j=1,\ldots,n)$ をもとに，それらを結ぶ輸送路に関する情報 $x_{ij}^{(k+1)}\,((i,j)\in E)$ を計算する処理である．さらに，ステップ 2 における制約条件の残差の計算は，輸送路に関する情報 $x_{ij}^{(k+1)}\,((i,j)\in E)$ を集約して，その輸送路に接続する供給地点に関する情報 $r_i^{(k+1)}\,(i=1,\ldots,m)$ と需要地点に関する情報 $s_j^{(k+1)}\,(j=1,\ldots,n)$ を求める処理である．すなわち，アルゴリズム 2 を構成する処理のタイプは，アルゴリズム 1 を構成する処理のタイプと同一である．したがって，データの分割と処理の分割に関しては，アルゴリズム 1 と全く同じ方法を採用することができる．

ランダムに生成した単一品種の 2 次輸送問題 (1) に対して，VPP500 および VPP800 を用いて行った計算実験の結果を，表 3 および表 4 に示す．表の各欄の意味は，並列型降下法の計算実験結果を示した表 1 および表 2 と同じである．また，並列型降下法と交互方向乗数法の性能比較ができるように，テスト問題は同じものを使用し，アルゴリズムを停止する際の制約条件の残差に関する許容誤差も同一にしている．表 3 および表 4 より，並列型降下法の場合と同様に，交互方向乗数法においても使用するプロセッサ数が増えるにつれて並列処理の効率はやや低下するが，問題サイズが大きくなると，並列処理の効率はしだいに改善することが確かめられる．また，適用する問題のサイズが大きくなるにつれて，交互方向乗数法は並列型降下法よりも優れた性能を示すことがわかる．

3 多品種の 2 次輸送問題に対する並列計算

つぎのように定式化される多品種の 2 次輸送問題を考える．

$$\begin{aligned}
\text{目的関数：}\quad & \sum_{\ell=1}^{L}\sum_{(i,j)\in E}\left\{\frac{\theta_{ij\ell}}{2}(x_{ij\ell})^2+\pi_{ij\ell}\,x_{ij\ell}\right\} \rightarrow \text{最小} \\
\text{制約条件：}\quad & \sum_{j\in J_i} x_{ij\ell}=\alpha_{i\ell} \qquad (i=1,\ldots,m;\quad \ell=1,\ldots,L) \\
& \sum_{i\in I_j} x_{ij\ell}=\beta_{j\ell} \qquad (j=1,\ldots,n;\quad \ell=1,\ldots,L) \\
& \sum_{\ell=1}^{L} x_{ij\ell}\leqq c_{ij} \qquad ((i,j)\in E) \\
& 0\leqq x_{ij\ell}\leqq u_{ij\ell} \qquad ((i,j)\in E;\qquad \ell=1,\ldots,L)
\end{aligned} \tag{16}$$

ただし，L は輸送する物資の品種数を表す 2 以上の整数である．集合 E, J_i, I_j はいずれも単一品種の 2 次輸送問題 (1) と同様の意味をもつ．変数 $x_{ij\ell}$ は，供給地点 i と需要地点 j を結ぶ輸送路を経由して輸送する物資のうち品種 ℓ の輸送量を表している．また，$\alpha_{i\ell}$ は供給地点 i において供給される物資の

表 3: VPP500 による交互方向乗数法の計算実験結果

問題サイズ			反復回数	計算時間 [秒] (並列化効率 e_p)			
供給地点	需要地点	輸送路		$p=1$	$p=2$	$p=4$	$p=8$
2048	2048	16384	82	0.202	0.138 (0.732)	0.088 (0.574)	0.064 (0.395)
8192	8192	65536	84	0.911	0.558 (0.816)	0.304 (0.749)	0.181 (0.629)
32768	32768	262144	123	5.479	3.302 (0.830)	1.760 (0.778)	0.986 (0.695)
131072	131072	1048576	970	143.8	85.67 (0.839)	45.60 (0.788)	25.57 (0.703)
1024	1024	16384	89	0.202	0.139 (0.727)	0.090 (0.561)	0.069 (0.366)
4096	4096	65536	91	0.846	0.512 (0.826)	0.277 (0.764)	0.170 (0.622)
16384	16384	262144	96	3.698	2.183 (0.847)	1.140 (0.811)	0.622 (0.743)
65536	65536	1048576	162	23.79	13.98 (0.851)	7.275 (0.818)	3.896 (0.763)

表 4: VPP800 による交互方向乗数法の計算実験結果

問題サイズ			計算時間 [秒] (並列化効率 e_p)					
供給地点	需要地点	輸送路	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$	$p=32$
2048	2048	16384	0.033	0.024 (0.688)	0.018 (0.458)	0.018 (0.229)	0.021 (0.098)	0.030 (0.034)
8192	8192	65536	0.189	0.101 (0.936)	0.058 (0.815)	0.039 (0.606)	0.035 (0.338)	0.040 (0.148)
32768	32768	262144	1.190	0.624 (0.954)	0.341 (0.872)	0.200 (0.744)	0.136 (0.547)	0.118 (0.315)
131072	131072	1048576	37.85	19.65 (0.963)	10.54 (0.898)	6.037 (0.784)	3.799 (0.623)	2.775 (0.426)
1024	1024	16384	0.033	0.026 (0.635)	0.021 (0.393)	0.021 (0.196)	0.0024 (0.086)	0.035 (0.029)
4096	4096	65536	0.165	0.087 (0.948)	0.051 (0.809)	0.038 (0.543)	0.036 (0.286)	0.043 (0.120)
16384	16384	262144	0.739	0.383 (0.965)	0.203 (0.910)	0.115 (0.803)	0.079 (0.585)	0.071 (0.325)
65536	65536	1048576	4.305	2.213 (0.973)	1.170 (0.920)	0.636 (0.846)	0.379 (0.710)	0.263 (0.512)

うち品種 ℓ の供給量，$\beta_{j\ell}$ は需要地点 j において必要とされる物資のうち品種 ℓ の需要量を表しており，$\sum_{i=1}^{m}\alpha_{i\ell}=\sum_{j=1}^{n}\beta_{j\ell}>0\,(\ell=1,\ldots,L)$ と仮定する．さらに，$\theta_{ij\ell},\pi_{ij\ell}$ は供給地点 i から需要地点 j へ品種 ℓ の物資を輸送した場合の費用を表す 2 次関数の係数であり，$\theta_{ij\ell}>0\,((i,j)\in E;\,\ell=1,\ldots,L)$ と仮定する．問題 (16) では，各輸送路に輸送量の上限値を設定している．すなわち，$u_{ij\ell}$ は輸送路 $(i,j)\in E$ を経由する物資のうち品種 ℓ の輸送量の上限値，c_{ij} は輸送路 $(i,j)\in E$ を経由する全品種の輸送量の合計に対する上限値であり，その輸送路の容量を表している．この節では，問題 (16) に対する並列アルゴリズムである並列型主双対内点法と予測子修正子近接乗数法について述べ，ベクトル並列計算機による実行方法を示すとともに，VPP を用いた計算実験結果を紹介する．

3.1 並列型主双対内点法

線形計画問題に対する多項式時間の解法の一つとして考案された主双対内点法 [7] は，現在では 2 次計画問題や非線形計画問題に対しても拡張され，大規模問題に対する有効な逐次アルゴリズムとしてさまざまな場面で適用されている．主双対内点法は，多品種輸送問題のような特徴的構造をもつ問題に適用すると，効率的な並列アルゴリズムになる [12]．多品種の 2 次輸送問題 (16) は，行列とベクトルを用いるとつぎのように書き換えられる．

$$
\begin{array}{lll}
\text{目的関数：} & \displaystyle\sum_{\ell=1}^{L}\left\{\frac{1}{2}x_\ell^\top\Theta_\ell\,x_\ell+\pi_\ell^\top x_\ell\right\}\ \rightarrow\ \text{最小} & \\
\text{制約条件：} & A\,x_\ell=b_\ell & (\ell=1,\ldots,L)\\
& \displaystyle\sum_{\ell=1}^{L}x_\ell\leqq c & \\
& 0\leqq x_\ell\leqq u_\ell & (\ell=1,\ldots,L)
\end{array}
\tag{17}
$$

ただし，$\ell=1,\ldots,L$ のそれぞれに対して $x_\ell\in\Re^{|E|}$ は $x_{ij\ell}\,((i,j)\in E)$ を成分とするベクトル，$\pi_\ell\in\Re^{|E|},u_\ell\in\Re^{|E|}$ はそれぞれ $\pi_{ij\ell}\,((i,j)\in E),u_{ij\ell}\,((i,j)\in E)$ を成分とするベクトル，$b_\ell\in\Re^{m+n}$ は $\alpha_{i\ell}\,(i=1,\ldots,m)$ と $\beta_{j\ell}\,(j=1,\ldots,n)$ を成分とするベクトルである．また，$\ell=1,\ldots,L$ のそれぞれに対して $\Theta_\ell\in\Re^{|E|\times|E|}$ は $\theta_{ij\ell}\,((i,j)\in E)$ を対角要素とする対角行列，$A\in\Re^{(m+n)\times|E|}$ は各行が供給地点または需要地点に対応し各列がそれらを結ぶ輸送路に対応する接続行列である．さらに，$c\in\Re^{|E|}$ は $c_{ij}\,((i,j)\in E)$ を成分とするベクトルである．不等式制約条件 $\sum_{\ell=1}^{L}x_\ell\leqq c,\,x_\ell\leqq u_\ell$ に対するスラック変数をそれぞれ $s\in\Re^{|E|},\,t_\ell\in\Re^{|E|}$, 制約条件 $A\,x_\ell=b_\ell\,,\,\sum_{\ell=1}^{L}x_\ell\leqq c,\,x_\ell\leqq u_\ell\,,\,x_\ell\geqq 0$ に対するラグランジュ乗数をそれぞれ $v_\ell\in\Re^{m+n},\,w\in\Re^{|E|},\,y_\ell\in\Re^{|E|},\,z_\ell\in\Re^{|E|}$ と書く．また，$X_\ell\,,\,S,\,T_\ell\,,\,W,\,Y_\ell\,,\,Z_\ell$ をそれぞれベクトル $x_\ell\,,\,s,\,t_\ell\,,\,w,\,y_\ell\,,\,z_\ell$ の各要素を対角成分とする対角行列，$\mathbf{1}\in\Re^{|E|}$ を各成分が 1 のベクトルとする．そのとき，問題 (17) に対する主双対内点法の反復は，条件 $x_\ell>0,\,s>0,\,t_\ell>0,\,w>0,\,y_\ell>0,\,z_\ell>0$ を満たしながら，問題 (17) の 1 次の最適性条件をあるパラメータ $\mu>0$ によって近似した方程式系

$$
\begin{array}{ll}
\Theta_\ell\,x_\ell+\pi_\ell-A^\top v_\ell+w+y_\ell-z_\ell=0 & (\ell=1,\ldots,L)\\
A\,x_\ell=b_\ell & (\ell=1,\ldots,L)\\
\displaystyle\sum_{\ell=1}^{L}x_\ell+s=d & \\
x_\ell+t_\ell=u_\ell & (\ell=1,\ldots,L)\\
S\;w\;=\mu\,\mathbf{1} & \\
T_\ell\;y_\ell=\mu\,\mathbf{1} & (\ell=1,\ldots,L)\\
X_\ell\,z_\ell=\mu\,\mathbf{1} & (\ell=1,\ldots,L)
\end{array}
\tag{18}
$$

を解き, その結果をもとに μ の値をより小さな値に更新するという手続きから成る．

ここでは，方程式系 (18) の前半 4 組の式を満たす初期探索点が得られているものと仮定し，(18) をニュートン法に基づく反復解法で解くことにする．そのとき，問題 (17) に対する並列型主双対内点法はつぎのように記述できる．

アルゴリズム 3.

ステップ 0: パラメータ $0 < \varepsilon \ll M_1 < M_2$ と $\tau \in (0,1)$ を選び，近似パラメータの初期値 $\mu^{(0)}$ を十分大きな正の数に定める．条件

$$\begin{array}{ll}
\Theta_\ell\, x_\ell^{(0)} + \pi_\ell - A^\top v_\ell^{(0)} + w^{(0)} + y_\ell^{(0)} - z_\ell^{(0)} = 0 & (\ell = 1, \ldots, L) \\
A\, x_\ell^{(0)} = b_\ell & (\ell = 1, \ldots, L) \\
\displaystyle\sum_{\ell=1}^{L} x_\ell^{(0)} + s^{(0)} = d & \\
x_\ell^{(0)} + t_\ell^{(0)} = u_\ell & (\ell = 1, \ldots, L)
\end{array}$$

を満たすように初期探索点 $x_\ell^{(0)} > 0$, $s^{(0)} > 0$, $t_\ell^{(0)} > 0$, $v_\ell^{(0)} \in \Re^{m+n}$, $w^{(0)} > 0$, $y_\ell^{(0)} > 0$, $z_\ell^{(0)} > 0$ を選び，$k = 0$ とおく．

ステップ 1: 品種 $\ell = 1, \ldots, L$ のそれぞれに対して，以下の各値を計算する．

$$\begin{array}{rcl}
\Xi_\ell^{(k)} & = & \Theta_\ell X_\ell^{(k)} T_\ell^{(k)} + X_\ell^{(k)} Y_\ell^{(k)} + T_\ell^{(k)} Z_\ell^{(k)} \\
\Phi_\ell^{(k)} & = & \left(\Xi_\ell^{(k)}\right)^{-1} X_\ell^{(k)} T_\ell^{(k)} \\
\lambda_\ell^{(k)} & = & \left(\Xi_\ell^{(k)}\right)^{-1} \left\{ T_\ell^{(k)} \left(\mu^{(k)}\mathbf{1} - X_\ell^{(k)} z_\ell^{(k)}\right) - X_\ell^{(k)} \left(\mu^{(k)}\mathbf{1} - T_\ell^{(k)} y_\ell^{(k)}\right) \right\}
\end{array}$$

ステップ 2: 連立方程式

$$\begin{array}{rl}
& \left[W^{(k)} \displaystyle\sum_{\ell=1}^{L} \left\{ \Phi_\ell^{(k)} - \Phi_\ell^{(k)} A^\top \left(A\, \Phi_\ell^{(k)} A^\top \right)^{-1} A\, \Phi_\ell^{(k)} \right\} + S^{(k)} \right] \Delta w \\
= & W^{(k)} \displaystyle\sum_{\ell=1}^{L} \left\{ \lambda_\ell^{(k)} - \Phi_\ell^{(k)} A^\top \left(A\, \Phi_\ell^{(k)} A^\top \right)^{-1} A\, \lambda_\ell^{(k)} \right\} + \mu^{(k)}\mathbf{1} - S^{(k)} w^{(k)}
\end{array}$$

の解 $\Delta w^{(k)}$ を求め，

$$\Delta s^{(k)} = \left(W^{(k)}\right)^{-1} \left\{ \mu^{(k)}\mathbf{1} - S^{(k)} \left(w^{(k)} + \Delta w^{(k)} \right) \right\}$$

とおく．もし $\Delta s^{(k)} \geqq 0$, $\Delta w^{(k)} \geqq 0$ ならば，$\eta_0^{(k)} = 1$ とおく．さもなければ，

$$\eta_0^{(k)} = \min\Big\{ 1,\, \tau\, \max\{ \eta \mid s^{(k)} + \eta\, \Delta s^{(k)} \geqq 0,\, w^{(k)} + \eta\, \Delta w^{(k)} \geqq 0 \} \Big\}$$

とおく．

ステップ 3: 品種 $\ell = 1, \ldots, L$ のそれぞれに対して，連立方程式

$$A\, \Phi_\ell^{(k)} A^\top \Delta v_\ell = A \left(\Phi_\ell^{(k)} \Delta w^{(k)} - \lambda_\ell^{(k)} \right)$$

の解 $\Delta v_\ell^{(k)}$ を求め，

$$\begin{array}{rcl}
\Delta x_\ell^{(k)} & = & \Phi_\ell^{(k)} \left(A^\top \Delta v_\ell^{(k)} - \Delta w^{(k)} \right) + \lambda_\ell^{(k)} \\
\Delta t_\ell^{(k)} & = & -\Delta x_\ell^{(k)} \\
\Delta y_\ell^{(k)} & = & \left(T_\ell^{(k)}\right)^{-1} \left\{ \mu^{(k)}\, \mathbf{1} - Y_\ell^{(k)} \left(t_\ell^{(k)} + \Delta t_\ell^{(k)} \right) \right\} \\
\Delta z_\ell^{(k)} & = & \left(X_\ell^{(k)}\right)^{-1} \left\{ \mu^{(k)}\, \mathbf{1} - Z_\ell^{(k)} \left(x_\ell^{(k)} + \Delta x_\ell^{(k)} \right) \right\}
\end{array}$$

とおく．さらに，$\ell = 1, \ldots, L$ のそれぞれに対して，もし $\Delta x_\ell^{(k)} \geqq 0$, $\Delta t_\ell^{(k)} \geqq 0$, $\Delta y_\ell^{(k)} \geqq 0$, $\Delta z_\ell^{(k)} \geqq 0$ ならば，$\eta_\ell^{(k)} = 1$ とおく．さもなければ，

$$\eta_\ell^{(k)} = \min\Big\{ 1, \tau \max\{ \eta \mid x_\ell^{(k)} + \eta\, \Delta x_\ell^{(k)} \geqq 0,\, t_\ell^{(k)} + \eta\, \Delta t_\ell^{(k)} \geqq 0, \\ y_\ell^{(k)} + \eta\, \Delta y_\ell^{(k)} \geqq 0,\, z_\ell^{(k)} + \eta\, \Delta z_\ell^{(k)} \geqq 0, \} \Big\}$$

とおく．

ステップ 4: ステップサイズ $\eta^{(k)}$ を

$$\eta^{(k)} = \min\{ \eta_0^{(k)}, \eta_1^{(k)}, \ldots, \eta_L^{(k)} \}$$

により定め，つぎの探索点を

$$\begin{array}{lcll} x_\ell^{(k+1)} & = & x_\ell^{(k)} + \eta^{(k)} \Delta x_\ell^{(k)} & (\ell = 1, \ldots, L) \\ s^{(k+1)} & = & s^{(k)} + \eta^{(k)} \Delta s^{(k)} & \\ t_\ell^{(k+1)} & = & t_\ell^{(k)} + \eta^{(k)} \Delta t_\ell^{(k)} & (\ell = 1, \ldots, L) \\ v_\ell^{(k+1)} & = & v_\ell^{(k)} + \eta^{(k)} \Delta v_\ell^{(k)} & (\ell = 1, \ldots, L) \\ w^{(k+1)} & = & w^{(k)} + \eta^{(k)} \Delta w^{(k)} & \\ y_\ell^{(k+1)} & = & y_\ell^{(k)} + \eta^{(k)} \Delta y_\ell^{(k)} & (\ell = 1, \ldots, L) \\ z_\ell^{(k+1)} & = & z_\ell^{(k)} + \eta^{(k)} \Delta z_\ell^{(k)} & (\ell = 1, \ldots, L) \end{array}$$

とおく．

ステップ 5: まず

$$\begin{array}{lcll} \delta_S^{(k+1)} & = & \dfrac{\| S^{(k+1)} w^{(k+1)} - \mu^{(k)}\, \mathbf{1} \|_1}{|E|} & \\ \delta_{T_\ell}^{(k+1)} & = & \dfrac{\| T_\ell^{(k+1)} y_\ell^{(k+1)} - \mu^{(k)}\, \mathbf{1} \|_1}{|E|} & (\ell = 1, \ldots, L) \\ \delta_{X_\ell}^{(k+1)} & = & \dfrac{\| X_\ell^{(k+1)} z_\ell^{(k+1)} - \mu^{(k)}\, \mathbf{1} \|_1}{|E|} & (\ell = 1, \ldots, L) \end{array}$$

とおき，方程式系 (18) の後半 3 組の式の達成度を次式で評価する．

$$\delta^{(k+1)} = \max \left\{ \delta_S^{(k+1)}, \frac{1}{L} \sum_{\ell=1}^{L} \delta_{T_\ell}^{(k+1)}, \frac{1}{L} \sum_{\ell=1}^{L} \delta_{X_\ell}^{(k+1)} \right\}$$

もし $\delta^{(k+1)} \leqq \varepsilon$ ならば，$x_\ell^{(k+1)}$ $(\ell = 1, \ldots, L)$ を問題 (16) の解として終了する．さもなければ，$\mu^{(k)}$ を

$$\mu^{(k+1)} = \begin{cases} \delta^{(k+1)} / M_2\,, & \delta^{(k+1)} \leq M_1\, \mu^{(k)} \text{のとき}, \\ \mu^{(k)}, & \text{それ以外のとき}, \end{cases}$$

により更新し，$k = k + 1$ とおいてステップ 1 に戻る． □

アルゴリズム 3 のステップ 1, 3, 4, 5 は，一部を除いて品種 $\ell = 1, \ldots, L$ のそれぞれについて独立に実行できる．ステップ 2 の連立方程式は，係数行列および右辺ベクトルに $A\, \Phi_\ell^{(k)} A^\top$ の逆行列を含む複雑な構造をもっている．ところが，$\ell = 1, \ldots, L$ のそれぞれに対する独立な L 個の連立方程式

$$A\, \Phi_\ell^{(k)} A^\top \nu_\ell = A\, \lambda_\ell^{(k)}$$

の解を $\widetilde{\nu}_\ell^{(k)} \in \Re^{m+n}$ とおくと，右辺ベクトルは

$$W^{(k)} \sum_{\ell=1}^{L} \left(\lambda_\ell^{(k)} - \Phi_\ell^{(k)} A^\top \widetilde{\nu}_\ell^{(k)} \right) + \mu^{(k)} \mathbf{1} - S^{(k)} w^{(k)}$$

と書ける．また，d を適当な $|E|$ 次元ベクトルとして，$\ell = 1, \ldots, L$ のそれぞれに対する独立な L 個の連立方程式

$$A \, \Phi_\ell^{(k)} A^\top \nu_\ell = A \, \Phi_\ell^{(k)} d$$

の解を $\widehat{\nu}_\ell^{(k)} \in \Re^{m+n}$ とおくと，係数行列とベクトル d の積は

$$W^{(k)} \sum_{\ell=1}^{L} \left(\Phi_\ell^{(k)} d - \Phi_\ell^{(k)} A^\top \widehat{\nu}_\ell^{(k)} \right) + S^{(k)} d$$

と計算できる．これは，$A\Phi_\ell^{(k)} A^\top$ を係数行列とする連立方程式を繰り返し解くことによって，ステップ 2 の連立方程式が共役勾配法を用いて比較的容易に解けることを意味している．なお，ステップ 3 で解く連立方程式の係数行列も $A\Phi_\ell^{(k)} A^\top$ である．したがって，$h_\ell \, (\ell = 1, \ldots, L)$ を適当な $|E|$ 次元ベクトルとするとき，アルゴリズム 3 の各反復における主要な計算は，$\ell = 1, \ldots, L$ のそれぞれに対する独立な L 個の連立方程式

$$A \, \Phi_\ell^{(k)} A^\top \nu_\ell = A \, h_\ell \tag{19}$$

を並列的に解くことである．それゆえ，アルゴリズム 3 は，並列計算機を用いて効率的に実行できる．

連立方程式 (19) も，共役勾配法を用いて解くことができる．行列 $\Phi_\ell^{(k)}$ は対角行列であるから，連立方程式 (19) に対する共役勾配法の反復における主要な計算は，行列 A とあるベクトル $p_\ell \in \Re^{|E|}$ および $q_\ell \in \Re^{m+n}$ の積 $A \, p_\ell$ および $A^\top q_\ell$ である．行列 A はこの輸送問題が定義されるネットワークの接続行列であるから，ベクトル p_ℓ の成分を $p_{ij\ell} \, ((i,j) \in E)$ とすると，ベクトル $A \, p_\ell$ は

$$A \, p_\ell = \left(\sum_{j \in J_1} p_{1j\ell}, \ldots, \sum_{j \in J_m} p_{mj\ell}, \sum_{i \in I_1} p_{i1\ell}, \ldots, \sum_{i \in I_n} p_{in\ell} \right)^\top$$

となる．したがって，$A \, p_\ell$ の計算は，輸送路に関する情報を集約してその輸送路が接続する供給地点に関する情報と需要地点に関する情報を生成する処理になる．一方，$q_\ell = (\xi_{1\ell}, \ldots, \xi_{m\ell}, \zeta_{1\ell}, \ldots, \zeta_{n\ell})^\top$ とおくと，$A^\top q_\ell$ の (i,j) 成分は

$$\left(A^\top q_\ell \right)_{ij} = \xi_{i\ell} + \zeta_{j\ell} \qquad ((i,j) \in E)$$

となる．これは，供給地点に関する情報と需要地点に関する情報をもとに，それらを結ぶ輸送路に関する情報を計算する処理である．連立方程式 (19) に対する共役勾配法の反復には，供給地点と需要地点について独立に実行できる処理や，輸送路ごとに独立に実行できる処理も含まれている．

アルゴリズム 3 を実行する際には，利用可能なプロセッサのそれぞれが担当する品種を決めて，品種ごとに独立な計算を並列的に処理させる．そのために，添字 ℓ をもつデータは対応する品種の処理を担当するプロセッサに分割して保持する．一方，添字 ℓ をもたないデータは，すべてのプロセッサに同じ値を重複して持たせる．これに伴って，ステップ 2 の $\Delta s^{(k)}$ と $\eta_0^{(k)}$ のように添字 ℓ をもたないデータの計算は，すべてのプロセッサ上で同時に実行する．また，ステップ 4 の $\eta^{(k)}$ やステップ 5 の $\delta^{(k+1)}$ の値は，グローバル関数 [4] を用いることによって容易に計算できる[1]．なお，ステップ 2 の連立方程式を解く際には，まず，各プロセッサが担当する品種 ℓ に対する連立方程式 (19) を解き，その解を `UNIFY` 命令を用いて他のすべてのプロセッサに転送する．これによって，各プロセッサは同時に共役勾配法の反復を進めることができ，すべてのプロセッサにおいて同時に $\Delta w^{(k)}$ の値が計算される．

[1] 一般に，グローバル関数を用いるとデータ転送の負荷が大きくなるが，$\eta^{(k)}$ や $\delta^{(k+1)}$ の計算において各プロセッサから転送されるのはスカラー値であり，データ転送の負荷は無視できるほど小さいと考えられる．

表 5: VPP500 における並列型主双対内点法の計算実験結果

問題サイズ				反復回数	計算時間 [秒] (並列化効率 e_p または e'_p)†			
供給地点	需要地点	輸送路	品種		$p=1$	$p=2$	$p=4$	$p=8$
8192	8192	65536	2	31	196.7	107.1 (0.918)	–	–
			4	34	456.2	241.6 (0.944)	134.0 (0.851)	–
			8	39	1196.	619.6 (0.965)	331.4 (0.902)	194.5 (0.769)
16384	16384	131072	2	34	483.3	277.3 (0.871)	–	–
			4	37	1218.	663.4 (0.918)	373.8 (0.815)	–
			8	42	**	1751.	948.7 (0.923)	543.7 (0.805)
4096	4096	65536	2	29	87.39	47.80 (0.914)	–	–
			4	33	215.2	114.6 (0.939)	64.81 (0.830)	–
			8	39	586.2	295.9 (0.991)	159.9 (0.917)	98.92 (0.741)
8192	8192	131072	2	36	235.4	131.0 (0.898)	–	–
			4	41	564.4	308.2 (0.916)	173.4 (0.814)	–
			8	47	**	752.5	411.4 (0.915)	247.0 (0.761)

† 輸送路数が 131072 で品種数が 8 の場合のみ，並列化効率を e'_p で評価している．

表 6: VPP800 における並列型主双対内点法の計算実験結果

問題サイズ				反復回数	計算時間 [秒] (並列化効率 e_p)			
供給地点	需要地点	輸送路	品種		$p=1$	$p=2$	$p=4$	$p=8$
8192	8192	65536	2	31	33.25	18.86 (0.881)	–	–
			4	34	77.32	41.86 (0.924)	25.42 (0.760)	–
			8	39	202.6	106.0 (0.955)	59.47 (0.852)	41.26 (0.614)
16384	16384	131072	2	34	81.87	48.64 (0.842)	–	–
			4	37	207.0	115.5 (0.896)	71.57 (0.723)	–
			8	42	571.2	302.9 (0.943)	172.4 (0.828)	119.0 (0.600)
4096	4096	65536	2	29	13.50	7.561 (0.893)	–	–
			4	33	33.18	17.89 (0.928)	10.66 (0.778)	–
			8	39	87.53	45.78 (0.956)	25.48 (0.859)	17.27 (0.634)
8192	8192	131072	2	36	36.73	21.14 (0.869)	–	–
			4	41	88.78	49.17 (0.903)	29.44 (0.754)	–
			8	47	224.6	119.2 (0.942)	67.62 (0.830)	45.59 (0.616)

計算実験は，ランダムに生成した多品種の 2 次輸送問題 (16) に対して行った．VPP500 および VPP800 による計算実験の結果を，表 5 および表 6 に示す．表の各欄の数値は，それぞれのサイズの 5 題の問題例に対する実験結果の平均値である．また，表 の “計算時間” 欄に掲げた p の値は，実際に使用したプロセッサの数である．計算実験では，使用するプロセッサの数を，解くべき問題に含まれる品種の数の約数に設定してアルゴリズム 3 を実行した．その際，品種と同数のプロセッサを使用した場合には，各プロセッサにそれぞれ別な品種の処理を担当させたが，使用するプロセッサ数が品種の数より少ない場合には，各プロセッサにそれぞれ複数の品種に関する処理を逐次的に実行させた．なお，品種の数より多くのプロセッサを利用し，各品種に関する処理を複数のプロセッサで並列的に実行させるような実験は行っていない．実験を行わなかったケースについては，計算時間の欄に “–” で示している．一方，“**” の欄は，メモリ不足で実行できなかったケースを示す．それぞれの表には，式 (8) で定義される並列化効率 e_p の値も示している．ただし，表 5 において，輸送路の数が 131072 で品種の数が 8 の問題は，VPP500 の 1 個のプロセッサで解けなかったため，$p = 4, 8$ の場合については

$$e'_p = \frac{\text{2 個のプロセッサによる計算時間} \times 2}{p \text{ 個のプロセッサによる計算時間} \times p}$$

の値を掲げている．問題サイズの増大にともなってアルゴリズム 3 の反復回数は増加する傾向にあるが，その増加率はごくわずかであり，内点法の特徴をよく表していると言える．また，同じサイズの問題では，使用するプロセッサ数が増えるにつれて並列処理の効率はやや低下する傾向にあることがわかる．しかし，同じ数のプロセッサを使用した場合，供給地点や需要地点の数および輸送路の数が同じならば，品種の数の増加に伴って並列処理の効率はしだいに改善していくことが確められる．なお，品種の数を一定に保ちながら輸送路の数を増加させると，並列処理の効率はわずかながら低下してしまう．これは，アルゴリズム 3 のステップ 2 において連立方程式を共役勾配法を用いて解く場合，その各反復において輸送路の数と同じサイズのベクトルである連立方程式 (19) の解を各プロセッサ間で転送しなければならないため，輸送路の数が増えるとその負荷がかなり増大するためと考えられる．

3.2 予測子修正子近接乗数法

予測子修正子近接乗数法 [1] は，乗数法に近接点法の考え方を組み込むことによって得られる並列アルゴリズムである．表記を簡単にするために，$x_{ij\ell}\,((i,j) \in E;\, \ell = 1, \ldots, L)$ を成分とするベクトルを x，$y_{ij}\,((i,j) \in E)$ を成分とするベクトルを y と書く．そのとき，集合 X, Y を

$$\begin{aligned} X = \Big\{ x \;\Big|\; & \sum_{j \in J_i} x_{ij\ell} = \alpha_{i\ell} \;\; (i = 1, \ldots, m;\, \ell = 1, \ldots, L), \\ & \sum_{i \in I_j} x_{ij\ell} = \beta_{j\ell} \;\; (j = 1, \ldots, n;\;\; \ell = 1, \ldots, L), \\ & 0 \leqq x_{ij\ell} \leqq u_{ij\ell} \;\; ((i,j) \in E; \quad \ell = 1, \ldots, L) \Big\} \\ Y = \Big\{ y \mid & \; y_{ij} \leqq c_{ij} \;\; ((i,j) \in E) \Big\} \end{aligned}$$

で定義し，関数 F, G を

$$F(x) = \begin{cases} \displaystyle\sum_{\ell=1}^{L} \sum_{(i,j) \in E} \left\{ \frac{\theta_{ij\ell}}{2} (x_{ij\ell})^2 + \pi_{ij\ell}\, x_{ij\ell} \right\}, & x \in X \text{ のとき} \\ +\infty, & x \notin X \text{ のとき} \end{cases}$$

$$G(y) = \begin{cases} 0, & y \in Y \text{ のとき} \\ +\infty, & y \notin Y \text{ のとき} \end{cases}$$

とおくと，多品種の 2 次輸送問題 (16) はつぎのように書きかえられる．

$$\begin{aligned} &\text{目的関数：}\quad F(x)+G(y)\ \rightarrow\ \text{最小}\\ &\text{制約条件：}\quad \sum_{\ell=1}^{L} x_{ij\ell}=y_{ij}\qquad ((i,j)\in E)\end{aligned} \tag{20}$$

問題 (20) に対する拡張ラグランジュ関数を次式で定義する．

$$L_\lambda(x,y;z)=F(x)+G(y)+\sum_{(i,j)\in E} z_{ij}\Big(\sum_{\ell=1}^{L} x_{ij\ell}-y_{ij}\Big)+\frac{\lambda}{2}\sum_{(i,j)\in E}\Big(\sum_{\ell=1}^{L} x_{ij\ell}-y_{ij}\Big)^2$$

ただし，z は問題 (20) の各等式制約条件に対するラグランジュ乗数 $z_{ij}\,((i,j)\in E)$ を成分とするベクトルであり，$\lambda>0$ は十分大きな値をとるパラメータである．問題 (20) に対する通常の乗数法の各反復では，拡張ラグランジュ関数の制約なし最小化問題を解き，その解をもとにラグランジュ乗数の値を更新する．乗数法に近接点法 [6] の考え方を組み込むと，各反復で解く部分問題は，拡張ラグランジュ関数にステップサイズを制限する凸 2 次項を加えた関数を最小化する問題

$$\text{目的関数：}\quad L_\lambda(x,y;z^{(k)})+\frac{1}{2\lambda}\sum_{\ell=1}^{L}\sum_{(i,j)\in E}(x_{ij\ell}-x_{ij\ell}^{(k)})^2+\frac{1}{2\lambda}\sum_{(i,j)\in E}(y_{ij}-y_{ij}^{(k)})^2\ \rightarrow\ \text{最小} \tag{21}$$

となる．予測子修正子近接乗数法の各反復では，問題 (21) を解く代わりに，二つの独立な問題

$$\text{目的関数：}\quad F(x)+\sum_{(i,j)\in E}\tilde{z}_{ij}^{(k+1)}\sum_{\ell=1}^{L}x_{ij\ell}+\frac{1}{2\lambda}\sum_{\ell=1}^{L}\sum_{(i,j)\in E}(x_{ij\ell}-x_{ij\ell}^{(k)})^2\ \rightarrow\ \text{最小} \tag{22}$$

と

$$\text{目的関数：}\quad G(y)-\sum_{(i,j)\in E}\tilde{z}_{ij}^{(k+1)}y_{ij}+\frac{1}{2\lambda}\sum_{(i,j)\in E}(y_{ij}-y_{ij}^{(k)})^2\ \rightarrow\ \text{最小} \tag{23}$$

を並列的に解いて，それらの解を $x^{(k+1)}, y^{(k+1)}$ とおく．ただし，

$$\tilde{z}_{ij}^{(k+1)}=z_{ij}^{(k)}+\lambda\Big(\sum_{\ell=1}^{L}x_{ij\ell}^{(k)}-y_{ij}^{(k)}\Big)\qquad((i,j)\in E)$$

である．予測子修正子近接乗数法の妥当性は，これらの問題の最適性条件を比較することによって直観的に理解できる．実際，問題 (21) の最適性条件が

$$\begin{aligned} 0\in\partial F(x)+\Big\{\,p\in\Re^{|E|\times L}\ \Big|\ p_{ij\ell}=\ z_{ij}^{(k)}+\lambda\Big(\sum_{\ell'=1}^{L}x_{ij\ell'}-y_{ij}\Big)+\frac{1}{\lambda}(x_{ij\ell}-x_{ij\ell}^{(k)})\\ ((i,j)\in E;\,\ell=1,\ldots,L)\Big\}\\ 0\in\partial G(y)+\Big\{\,q\in\Re^{|E|}\ \Big|\ q_{ij}\ =-z_{ij}^{(k)}-\lambda\Big(\sum_{\ell=1}^{L}x_{ij\ell}-y_{ij}\Big)+\frac{1}{\lambda}(y_{ij}\ -y_{ij}^{(k)}\,)\ ((i,j)\in E)\Big\}\end{aligned} \tag{24}$$

と書けるのに対して，問題 (22) と (23) の最適性条件は

$$\begin{aligned} 0\in\partial F(x)+\Big\{\,p\in\Re^{|E|\times L}\ \Big|\ p_{ij\ell}=\ \tilde{z}_{ij}^{(k+1)}+\frac{1}{\lambda}(x_{ij\ell}-x_{ij\ell}^{(k)})\ ((i,j)\in E;\,\ell=1,\ldots,L)\Big\}\\ 0\in\partial G(y)+\Big\{\,q\in\Re^{|E|}\ \Big|\ q_{ij}\ =-\tilde{z}_{ij}^{(k+1)}+\frac{1}{\lambda}(y_{ij}\ -y_{ij}^{(k)}\,)\ ((i,j)\in E)\Big\}\end{aligned} \tag{25}$$

となる．近接点法の性質より，条件 (24) を満たす x,y は $x^{(k)},y^{(k)}$ の十分近くにあると考えられるので，条件 (25) は条件 (24) のよい近似となっている．

関数 F の定義より，問題 (22) は $\ell = 1, \ldots, L$ のそれぞれに対する独立な L 個の部分問題

$$\begin{aligned}
\text{目的関数：}\quad & \sum_{(i,j)\in E}\left\{\frac{\theta_{ij\ell}}{2}(x_{ij\ell})^2 + \pi_{ij\ell}\, x_{ij\ell}\right\} + \sum_{(i,j)\in E}\left\{\frac{1}{2\lambda}(x_{ij\ell} - x_{ij\ell}^{(k)})^2 + \tilde{z}_{ij}^{(k+1)} x_{ij\ell}\right\} \to \text{最小} \\
\text{制約条件：}\quad & \sum_{j\in J_i} x_{ij\ell} = \alpha_{i\ell} && (i = 1, \ldots, m) \\
& \sum_{i\in I_j} x_{ij\ell} = \beta_{j\ell} && (j = 1, \ldots, n) \\
& 0 \leqq x_{ij\ell} \leqq u_{ij\ell} && ((i,j) \in E)
\end{aligned} \tag{26}$$

に分解できる．問題 (26) は単一品種の 2 次輸送問題であるから，2 節で述べたいずれかのアルゴリズムを用いて並列的に解くことができる[2]．一方，関数 G の定義より，問題 (23) は $(i,j) \in E$ のそれぞれに対する独立な $|E|$ 個の 1 変数の問題

$$\begin{aligned}
\text{目的関数：}\quad & \frac{1}{2\lambda}(y_{ij} - y_{ij}^{(k)})^2 - \tilde{z}_{ij}^{(k+1)} y_{ij} \to \text{最小} \\
\text{制約条件：}\quad & 0 \leqq y_{ij} \leqq c_{ij}
\end{aligned} \tag{27}$$

に分解できる[3]．問題 (27) は解析的に解けて，その解 $y_{ij}^{(k+1)}$ は

$$y_{ij}^{(k+1)} = \text{mid}\left\{0, y_{ij}^{(k)} + \lambda \tilde{z}_{ij}^{(k+1)}, c_{ij}\right\}$$

となる．ただし，mid は括弧内の数の中央値を表す．なお，ラグランジュ乗数は，乗数法と同様に式

$$z_{ij}^{(k+1)} = z_{ij}^{(k)} + \lambda\left(\sum_{\ell=1}^{L} x_{ij\ell}^{(k+1)} - y_{ij}^{(k+1)}\right) \qquad ((i,j) \in E)$$

により更新する[4]．

結局，多品種の 2 次輸送問題 (16) に対する予測子修正子近接乗数法は，つぎのように記述できる．

アルゴリズム 4.

ステップ 0: 拡張ラグランジュ乗数のパラメータ $\lambda > 0$ を定める．適当な初期探索点 $x^{(0)} \in \Re^{|E|\times L}$, $y^{(0)} \in \Re^{|E|}$, $z^{(0)} \in \Re^{|E|}$ を選び，$k = 0$ とおく．

ステップ 1: 予測子 $\tilde{z}^{(k+1)} \in \Re^{|E|}$ を次式により定める．

$$\tilde{z}_{ij}^{(k+1)} = z_{ij}^{(k)} + \lambda\left(\sum_{\ell=1}^{L} x_{ij\ell}^{(k)} - y_{ij}^{(k)}\right) \qquad ((i,j) \in E)$$

ステップ 2: 品種 $\ell = 1, \ldots, L$ のそれぞれに対して，単一品種の 2 次輸送問題

$$\begin{aligned}
\text{目的関数：}\quad & \sum_{(i,j)\in E}\left\{\frac{1}{2}\left(\theta_{ij\ell} + \frac{1}{\lambda}\right)(x_{ij\ell})^2 + \left(\pi_{ij\ell} - \frac{x_{ij\ell}^{(k)}}{\lambda} + \tilde{z}_{ij}^{(k+1)}\right)x_{ij\ell}\right\} \to \text{最小} \\
\text{制約条件：}\quad & \sum_{j\in J_i} x_{ij\ell} = \alpha_{i\ell} && (i = 1, \ldots, m) \\
& \sum_{i\in I_j} x_{ij\ell} = \beta_{j\ell} && (j = 1, \ldots, n) \\
& 0 \leqq x_{ij\ell} \leqq u_{ij\ell} && ((i,j) \in E)
\end{aligned}$$

を解き，その解を $x^{(k+1)}$ とおく．

[2] 問題 (26) は輸送量の上限制約条件をもつため，2 節で述べたアルゴリズムを少し修正して適用する必要がある．

[3] 問題 (27) の制約条件は本来 $y_{ij} \leqq c_{ij}$ であるが，問題 (16), (20) の制約条件を考慮すると，実質上 $0 \leqq y_{ij} \leqq c_{ij}$ という区間制約と考えてよい．

[4] $\tilde{z}_{ij}^{(k+1)}$ の右辺括弧内に表われる $x_{ij\ell}^{(k)}, y_{ij}^{(k)}$ をそれぞれ $x_{ij\ell}^{(k+1)}, y_{ij}^{(k+1)}$ に改めたものが $z_{ij}^{(k+1)}$ である．その意味で，前者を予測子，後者を修正子とよぶ．

表 7: VPP800 における予測子修正子近接乗数法の計算実験結果

問題サイズ				反復回数	計算時間 [秒] (並列化効率 e_p)					
地点	地点	輸送路	品種		$p=1$	$p=2$	$p=4$	$p=8$	$p=16$	$p=32$
8192	8192	65536	2	415	420.1	238.0 (0.883)	151.8 (0.692)	91.77 (0.572)	70.88 (0.370)	82.00 (0.161)
			4	129	246.7	131.7 (0.937)	72.94 (0.846)	47.53 (0.649)	30.78 (0.501)	27.38 (0.282)
			8	154	602.9	314.2 (0.959)	167.8 (0.898)	92.26 (0.817)	62.26 (0.605)	44.59 (0.423)
16384	16384	131072	2	324	575.2	301.6 (0.954)	190.3 (0.756)	114.7 (0.627)	88.62 (0.406)	99.67 (0.180)
			4	176	734.4	397.2 (0.924)	226.9 (0.809)	144.9 (0.634)	90.35 (0.508)	75.25 (0.305)
			8	232	2038.	1056. (0.965)	578.4 (0.889)	333.8 (0.763)	219.9 (0.579)	149.6 (0.426)
4096	4096	65536	2	159	84.07	43.22 (0.973)	27.46 (0.765)	16.78 (0.626)	14.60 (0.360)	17.28 (0.152)
			4	108	118.5	60.40 (0.981)	31.23 (0.949)	20.20 (0.733)	13.12 (0.565)	12.75 (0.290)
			8	147	347.4	177.4 (0.979)	90.99 (0.955)	48.01 (0.904)	31.99 (0.679)	22.73 (0.478)
8192	8192	131072	2	289	370.6	197.6 (0.938)	122.3 (0.758)	70.89 (0.653)	51.40 (0.451)	56.44 (0.205)
			4	130	354.0	181.8 (0.974)	98.21 (0.901)	61.68 (0.717)	37.49 (0.590)	30.49 (0.363)
			8	149	838.1	438.4 (0.956)	233.7 (0.897)	131.5 (0.797)	84.43 (0.620)	55.13 (0.467)

ステップ 3: 点 $y^{(k)}$ を次式で更新する．

$$y_{ij}^{(k+1)} = \mathrm{mid}\left\{0, y_{ij}^{(k)} + \lambda \tilde{z}_{ij}^{(k+1)}, c_{ij}\right\} \qquad ((i,j) \in E)$$

ステップ 4: 修正子 $z^{(k+1)}$ を次式により定める．

$$z_{ij}^{(k+1)} = z_{ij}^{(k)} + \lambda \left(\sum_{\ell=1}^{L} x_{ij\ell}^{(k+1)} - y_{ij}^{(k+1)} \right) \qquad ((i,j) \in E)$$

ステップ 5: 点 $x^{(k+1)}, y^{(k+1)}, z^{(k+1)}$ が適当な収束判定条件を満たせば終了する．さもなければ，$k = k+1$ とおいてステップ 1 に戻る． □

アルゴリズム 4 のステップ 2 における $x^{(k+1)}$ の計算とステップ 3 における $y^{(k+1)}$ の計算は並列的に実行できるが，前者の計算量に比べて後者の計算量は著しく小さいので，アルゴリズム 4 ではこれらの計算を逐次的に実行することにしている．アルゴリズム 4 の主要な計算は，ステップ 2 において独立な L 個の単一品種の 2 次輸送問題を解くことである．そこで，アルゴリズム 4 を実行する際には，利用可能なプロセッサのそれぞれが担当する品種を決めて，対応する単一品種問題を並列的に解く．単一品種の 2 次輸送問題は，2.2 節で述べた交互方向乗数法を用いて解くことにする．そのとき，利用可能なプロセッサの数が品種数 L より少ない場合は，一部または全部のプロセッサが複数個の単一品種問題を逐次的に解かざるをえない．しかし，利用可能なプロセッサの数が品種数 L より多い場合には，

一部または全部の単一品種問題を複数個のプロセッサを用いて並列的に解くことができる．交互方向乗数法のアルゴリズムをサブルーチン化し，SUBPROCESSOR 文 [4] を用いて利用するプロセッサ数を指定しておけば，SPREAD REGION 内でこのサブルーチンを品種の数だけ呼び出すことによってこのような並列計算を比較的容易に実現することができる．交互方向乗数法のサブルーチン内におけるデータの分割と処理の分割は，2.2 節で述べた通常の交互方向乗数法の場合と同様の方法で行う．ステップ 2 で求めた $x^{(k+1)}$ の値は，UNIFY 命令を用いて他のすべてのプロセッサに転送する．これによって，ステップ 4 とつぎの反復のステップ 1 において各輸送路を経由する物資の輸送量をすべての品種について合計する処理は，各プロセッサで同時に実行できる．

ランダムに生成した多品種の 2 次輸送問題 (16) に対して，VPP800 を用いて行った計算実験の結果を表 7 に示す．表の各欄の意味は，並列型主双対内点法の計算実験結果を示した表 5 および表 6 と同じである．また，並列型主双対内点法と予測子修正子近接乗数法の性能比較ができるように，テスト問題は同じものを使用している．並列型主双対内点法の場合と同様に，予測子修正子近接乗数法においても，同じサイズの問題では使用するプロセッサ数が増えるにつれて並列処理の効率はやや低下するが，供給地点や需要地点の数および輸送路の数が同じならば，品種の数の増加に伴って並列処理の効率はしだいに改善していくことが確かめられる．なお，同じサイズの問題を同じ数のプロセッサで解いたときの性能は，予測子修正子近接乗数法は並列型主双対内点法に遠くおよばない．しかし，単一品種の部分問題を複数プロセッサを用いて解くことにすれば，並列型主双対内点法に匹敵する計算時間で同じ問題を解ける可能性もあると考えられる．

4 おわりに

本稿では，輸送費用が分離可能な凸 2 次関数で与えられる単一品種の輸送問題と多品種の輸送問題に対する並列アルゴリズムを二つずつ紹介し，ベクトル並列計算機で実行する方法を示すとともに，VPP を用いた計算実験を通してその性能を検証した．とくに，単一品種の輸送問題に対する並列アルゴリズムにおいては，その主要な計算が供給地点と需要地点に関する情報を並列的に計算する処理，輸送路に関する情報を並列的に計算する処理，輸送路に関する情報を集約して対応する供給地点に関する情報と需要地点に関する情報を並列的に求める処理，そして，供給地点に関する情報と需要地点に関する情報をもとにそれらを接続する輸送路に関する情報を並列的に求める処理の四つのタイプで構成されることを明らかにした．そして，これら四つのタイプの処理を効率よく実行するために，輸送路に関する情報を第 1 次元が供給地点の番号である 2 次元配列に格納して，供給地点に関する情報を格納する 1 次元配列とともに利用可能なプロセッサに分割して保持するとともに，輸送路に関する情報を第 1 次元が需要地点の番号であるもう一つの 2 次元配列にも重複して格納し，需要地点に関する情報を格納する 1 次元配列とともに利用可能なプロセッサに分割して保持することを提案した．このような実行方法を用いることにより，供給地点と需要地点の数がそれぞれ 10 万，輸送路の数が 100 万前後の問題を，VPP800 を用いて数秒程度で解くことができた．一方，多品種の輸送問題に対する並列アルゴリズムにおいては，品種ごとに独立な処理を複数のプロセッサで並列的に処理することによって，供給地点と需要地点の数がそれぞれ 1 万，輸送路の数が 10 万前後で，数品種の物資の輸送方法を決定する問題を，VPP800 を用いて 2 分以内で解くことができた．

輸送費用が分離可能な凸 2 次関数で与えられる場合には，アルゴリズムを構成するすべての計算が線形計算となるため，ベクトル並列計算機の性能を十分に引き出せるような並列アルゴリズムを構成することができる．しかし，輸送費用が一般の凸関数で与えられる場合には，並列的に処理可能なそれぞれの部分問題を反復法を用いて解かなければならない．そのとき，各部分問題の解が得られるまでの時間にばらつきが生じるため，プロセッサへの処理の割り当てにさまざまな工夫が必要になる．また，多品種の輸送問題において，輸送費用が輸送路と品種の双方について分離可能であることは少なく，各輸送路を経由する全品種の物資の輸送量の総和に比例した費用がかかる場合も少なくない．このような，より一般的な目的関数をもつ輸送問題に対する効率的な並列アルゴリズムの開発は，今後の

課題である.

謝辞

本研究の一部は，京都大学大型計算機センター開発計画によるものである.

参考文献

[1] G. Chen and M. Teboulle, “A Proximal-Based Decomposition Method for Convex Minimization Problems,” *Mathematical Programming*, Vol. 64 (1994), pp. 81–101.

[2] J. Eckstein, “The Alternating Step Method for Monotropic Programming on the Connection Machine CM-2,” *ORSA Journal on Computing*, Vol. 5 (1993), pp. 84–96.

[3] R. Fletcher *Practical Methods of Optimization, Second Edition*, John Wiley, 1987.

[4] 富士通株式会社, UXP/V VPP プログラミングハンドブック, 1997.

[5] M. Fukushima, M. Haddou, V.H. Nguyen, J.-J. Strodiot, T. Sugimoto and E. Yamakawa, “A Parallel Descent Algorithm for Convex Programming, ” *Computational Optimization and Applications*, Vol. 5 (1996), pp. 5–37.

[6] 茨木俊秀, 福島雅夫, 最適化の手法, 共立出版, 1993.

[7] S. Wright, *Primal-Dual Interior Point Methods*, SIAM, 1997

[8] 古林隆, 線形計画法, 産業図書, 1980.

[9] 今野浩, 山下浩, 非線形計画法, 日科技連, 1978.

[10] R.T. Rockafellar, *Convex Analysis*, Princeton University Press, 1970.

[11] 山川栄樹, 福島雅夫, 数理計画における並列計算, 朝倉書店, 2001.

[12] 山川栄樹, 松原康博, 福島雅夫, “2 次コスト多品種流問題に対する並列型主双対内点法,” *Journal of the Operations Research Society of Japan*, Vol. 39 (1996), pp. 566–591.

[13] S.A. Zenios, “Data Parallel Computing for Network-Structured Optimization Problems,” *Computational Optimization and Applications*, Vol. 4 (1994), pp. 199–242.