Journal of the Operations Research
Society of Japan
Vol. 39, No. 4, December 1996

2次コスト多品種流問題に対する並列型主双対内点法[1]

山川栄樹　　松原康博　　福島雅夫
高松大学　　西日本旅客鉄道　　京都大学

(受理 1995 年 2 月 27 日；再受理 1995 年 10 月 2 日)

和文概要　本論文では，分離可能な2次のコスト関数をもつ多品種流問題に対して主双対内点法を適用し，それを制約条件のブロック構造を利用して並列化することを試みる．ここで提案するアルゴリズムにおいては，主双対内点法の各反復で解くべき連立方程式を，各品種に対応する複数の連立方程式とアークの総流量上限制約に対応する一つの連立方程式に分解し，それらを共役勾配法を用いて並列的に解く．また，共役勾配法の実行において，適切な前処理を行なうことにより探索点が非負領域の境界に近づいたときに生じる数値的悪条件を克服するとともに，連立方程式の係数の処理に工夫を加えて桁落ちによる計算誤差の発生を抑える．提案したアルゴリズムをコントロール並列型の計算モデルによって実現し，さらに並列計算機 CM-5 を用いて実際に数値実験を行なうことにより大規模な多品種流問題が効率よく解けることを確かめた．

1. はじめに

線形計画問題に対する多項式オーダの解法の一つとして考案された主双対内点法 [11] は，その後 2 次計画問題や非線形計画問題に対しても拡張され，多くの研究者によってその大域的収束性や局所的な収束率が研究されてきた [15, 16, 23, 24]．また，大規模問題に対する数値実験を通して，実用面での有効性も徐々に検証されつつある [3]．主双対内点法は，あるパラメータによって緩和された 1 次の最適性条件の方程式系をニュートン法 (Newton method) を用いて繰返し解き，主問題と双対問題の解の組に至るような点列を生成する方法であるが，生成された点列が属する領域によって大きく二つの方法に分類される．当初提案された主双対実行可能内点法は，主実行可能性および双対実行可能性を満たすような点列を生成するもので，解くべき方程式系の形が簡単である反面，人工問題を解くなどの手続きによって初期実行可能内点を生成する必要がある．そこで最近では，点列が非負制約の内部にとどまることのみを要求する主双対非実行可能内点法が注目を集め，その大域的収束性や計算量の解析が行われている [10, 14, 22]．主双対内点法の各反復で解くべきニュートン方程式とおなじような構造をもつ連立方程式は，内点法の一種である双対アフィン変換法 (dual affine scaling method) においても現れるが，文献 [1] では，コレスキー分解または共役勾配法を用いてこの連立方程式を解く場合において，大規模で疎な現実の問題を効率的に処理できるようなデータ構造を提案している．また，文献 [13] では，共役勾配法における効果的な前処理の方法が示されている．さらに，現実の大規模問題においてしばしば現れる特徴的なブロック構造が，主双対内点法のアルゴリズムにおいてどのように利用できるかについての検討も行われている [5, 9]．このような工夫は，単に計算効率の向上をもたらすだけでなく，最近次第に身近な

[1] 本研究は，第一の著者がエイ・ティ・アール人間情報通信研究所，第二・第三の著者が奈良先端科学技術大学院大学に所属していた時になされたものである．

ものとなってきた並列計算機の利用への道を開くものとなる．実際, 文献 [12] では, コストが線形関数で与えられる多品種流問題に対して, その構造を用いて分解したニュートン法の探索方向を求めるための連立方程式をコレスキー分解を用いて解く方法を提案し, 並列計算機を用いた数値実験により, 商用の内点法のコードを上回る性能が得られることを明らかにしている．

一方, 本論文で考察の対象とする多品種流問題に対しては, 交通流の分野で取扱われているものも含めると, すでに様々な解法が提案されている．特に, ネットワークの構造を利用した解法 [4] はコントロール並列型のアルゴリズムを, また, 双対理論から導かれる緩和法 [25] や交互方向乗数法を用いた解法 [7] は, 非常に効率的なデータ並列型のアルゴリズムを与える．

本論文では, コストが品種に関して分離可能な 2 次関数で与えられる多品種流問題に対する主双対実行可能内点法を並列化することを試みる．初期実行可能内点を生成する手間を考えると, 実用上は主双対非実行可能内点法を適用する方が有利であるが, ここでは並列計算機による内点法の実行に焦点を絞るため, 記述の簡単な主双対実行可能内点法を用いることにする．多品種流問題の制約条件は, 品種ごとのフロー保存を表す等式, 全品種のフローの和で定義されるアークの総流量の上限を表す不等式, 各品種のフローの上下限を表す不等式で構成される．主双対内点法の各反復において, その計算時間の大半は, 各品種のフローを表す変数および上で述べた各制約条件に対応する双対変数の探索方向を決定する連立方程式を解くために費やされる．ところが, フロー保存の式がもつブロック対角構造を利用すると, それに対応する双対変数の探索方向を求める連立方程式を, 品種ごとに独立な複数の小さな連立方程式に分解することができる．また, アークの総流量上限制約に対応する双対変数の探索方向を求める連立方程式の係数行列や定数ベクトルも, 品種ごとに定義される同様な連立方程式群を解くことによって得られる情報を利用して求めることができる．そこで, 品種ごとに独立なこれらの連立方程式群を複数の処理装置を用いて並列的に解くような, 並列型の主双対内点法を構築する．なお, これらの連立方程式の係数行列は, いずれもフロー保存の式の係数行列であるグラフの接続行列とその転置行列の積の形で定義される．したがって, 現実の大規模なネットワークに適用するためには, その接続行列がもつ疎な構造を維持しながらこれらの連立方程式を解き進めることが望ましい．文献 [12] にも見られるように, 主双対内点法では行列の分解による直接法で連立方程式を解くことが多い．このとき, 分解後の行列は一般に密な構造をもつが, 様々なオーダリングの処理を施すことによって疎な構造をある程度維持できることも報告されている [1] ．しかし本論文では, 疎な接続行列またはその転置行列とベクトルの積などの簡単な線形代数的計算のみによって構成できる共役勾配法を用いて連立方程式を解くことにする．各処理装置に解かせる連立方程式の次元数およびその係数行列を構成する接続行列の構造は全く同一であるが, 事実上反復解法的な性格をもつ共役勾配法を適用するためには, コントロール並列型の計算モデルを用いる必要がある．そこで, 十分な数の処理装置が利用できる場合には, 互いに独立な手続きを実行できる処理装置を各品種に対して一つづつ割当てる．ただし, アークの総流量上限制約に対応する連立方程式を解くためには, 品種ごとに並列的に求めた情報を集約しつつ共役勾配法の手続きを進める必要がある．そこで, もう一つの特別な処理装置を用意して, 品種ごとに割当てられた各処理装置において並列的に実行すべき作業の指示と, そこで得られた情報の収集を行わせる．さらにこの処理装置には, 主双対内点法の反復全体を制御する役割も与えることにする．また, 各処理装置においてデータ並列型のプログラムを実行することが可能ならば, 大規模で疎な問題に対しても, 共役勾配法の手続きに含まれる線形代数的な計算を高速に処理する

ことができる．実際，並列計算機 CM-5 においてはこのような形態のプログラミングが可能である．そこで，アルゴリズムの実際の性能を確かめるために，大規模で疎な多品種流問題を生成して CM-5 上で数値実験を行った．なお，一般に内点法は，探索点が変数の非負領域の境界に近づくにつれて，探索方向の計算や直線探索が数値的に不安定になるという問題を抱えている．そこで，各連立方程式の係数行列や各計算式の記述にあたっては，加減算の中に変数の逆数が極力現れないような工夫を施している．また，品種ごとに定義される連立方程式群を解くための共役勾配法には簡単な前処理を組込み，アークの総流量上限制約に対応する連立方程式に共役勾配法を適用する際も，各処理装置から容易に得ることができる情報を用いて，その前処理行列を近似的に生成することにする．特に，後者の連立方程式の係数行列は，前者の連立方程式群の係数行列に比べて本質的に悪条件となり易い構造となるため，ここで行うような数値計算上の工夫や前処理の利用は，共役勾配法を有効に動作させるうえで必要不可欠なものとなる．

本論文の構成は次の通りである．まず 2 章において，等式制約，不等式制約，変数の上下限制約をもつ一般の 2 次計画問題に対して，主双対内点法のアルゴリズムを記述する．3 章では，多品種流問題の制約条件がもつブロック構造を利用して，主双対内点法の探索方向を求めるための連立方程式を並列計算可能な複数の小さな連立方程式に分解するとともに，ステップサイズの上限値や相補条件の満足度を求める計算も並列的に実行できることを指摘する．次に 4 章において，分解された各連立方程式を解くための共役勾配法を記述し，同時に，収束を速めるための前処理行列の求め方を示す．さらに 5 章では，内点法のアルゴリズムを実行するコントロール並列型の計算モデルを述べる．さらに，グラフの構造を用いて共役勾配法の手続きをデータ並列型のプログラミング環境で実行する方法を説明する．実際に並列計算機 CM-5 を用いて行った数値実験とその結果は，6 章に掲げる．最後に，7 章で本論文のまとめを行う．

2. 主双対内点法

この章では，等式制約，不等式制約，変数の上下限制約をもつ一般の 2 次計画問題

$$
\begin{aligned}
&\text{目的関数：} && \frac{1}{2}x^{\top}Gx + h^{\top}x \;\rightarrow\; \text{最小} \\
&\text{制約条件：} && Ax = b, \\
& && Cx \leq d, \\
& && 0 \leq x \leq u
\end{aligned}
\tag{2.1}
$$

に対して，主双対内点法のアルゴリズムを述べる．ただし，G は $n \times n$ 半正定値対称行列，A は 階数 (rank) が m の $m \times n$ 行列，C は $\bar{n} \times n$ 行列，h, b, d, u は それぞれ R^n, R^m, $R^{\bar{n}}$, R^n のベクトルである．以下では，不等式制約 $Cx \leq d$, $x \leq u$ に対するスラック変数をそれぞれ $t \in R^{\bar{n}}$, $s \in R^n$, 制約条件 $Ax = b$, $Cx \leq d$, $x \leq u$, $x \geq 0$ に対する双対変数をそれぞれ $y \in R^m$, $v \in R^{\bar{n}}$, $w \in R^n$, $z \in R^n$ と書く．また，X, T, S, Z, V, W をそれぞれベクトル x, t, s, z, v, w の各要素を対角成分とする対角行列，$e_n \in R^n$, $e_{\bar{n}} \in R^{\bar{n}}$ を各要素が 1 のベクトルとする．主双対内点法は，あるパラメータ $\mu > 0$ に対して，$x > 0$, $t > 0$, $s > 0$, $z > 0$,

$v>0$, $w>0$ を満たす方程式系

$$
\begin{aligned}
&Gx+h-A^{\top}y+C^{\top}v+w-z=0,\\
&Ax=b,\\
&Cx+t=d,\\
&x+s=u,\\
&Zx=\mu e_n,\\
&Vt=\mu e_{\bar{n}},\\
&Ws=\mu e_n
\end{aligned}
\tag{2.2}
$$

の解 (x,t,s,y,z,v,w) を求め, 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を繰返して, 主問題 (2.1) およびその双対問題の解の組に収束する点列を生成する方法である.

方程式系 (2.2) をニュートン法に基づく反復解法を用いて解くことを考える. いま, (2.2) の各式のうち, 相補条件を除く最初の 4 式を満たし, $x>0$, $t>0$, $s>0$, $z>0$, $v>0$, $w>0$ であるような点 (x,t,s,y,z,v,w) を, 実行可能内点と呼ぶことにする. ある実行可能内点 (x,t,s,y,z,v,w) が与えられたとき, 方程式系 (2.2) に対するニュートン法の探索方向 $(\Delta x,\Delta t,\Delta s,\Delta y,\Delta z,\Delta v,\Delta w)$ は, 方程式系 (2.2) を線形化した連立方程式

$$
\begin{aligned}
&G\Delta x-A^{\top}\Delta y+C^{\top}\Delta v+\Delta w-\Delta z=0,\\
&A\Delta x=0,\\
&C\Delta x+\Delta t=0,\\
&\Delta x+\Delta s=0,\\
&Z\Delta x+X\Delta z=\mu e_n-Zx,\\
&V\Delta t+T\Delta v=\mu e_{\bar{n}}-Vt,\\
&W\Delta s+S\Delta w=\mu e_n-Ws
\end{aligned}
\tag{2.3}
$$

を解いて求められる. ステップサイズは, 問題 (2.1) に対する対数型の障壁関数 (logarithmic barrier function)

$$
f(x,t,s;\mu)=\frac{1}{2}x^{\top}Gx+h^{\top}x-\mu\sum_{i=1}^{n}\log x_i-\mu\sum_{j=1}^{\bar{n}}\log t_j-\mu\sum_{i=1}^{n}\log s_i
$$

を用いた直線探索により定めることもできるが, ここでは簡単に, 変数 x, t, s, z, v, w の値が正に保たれる範囲で可能な限り大きく選ぶことにする.

方程式系 (2.2) に含まれる相補条件 $Zx=\mu e_n$, $Vt=\mu e_{\bar{n}}$, $Ws=\mu e_n$ の達成度に対する評価関数として,

$$
\phi(x,t,s,z,v,w;\mu)=\max\left\{\frac{\|Zx-\mu e_n\|_1}{n},\ \frac{\|Vt-\mu e_{\bar{n}}\|_1}{\bar{n}},\ \frac{\|Ws-\mu e_n\|_1}{n}\right\}
$$

を考える. ただし, $\|\cdot\|_1$ は l_1 ノルムである. このとき, 十分小さな正の数 ε_μ に対して,

$$
\phi(x,t,s,z,v,w;\mu)\le\varepsilon_\mu \tag{2.4}
$$

が成立すればニュートン法の反復を終了し, 得られた点 (x,t,s,y,z,v,w) を方程式系 (2.2) の解と考える. さらに, 関係式 $0<\varepsilon_\mu\ll M_{tol}<M_\mu$ を満たすパラメータ M_{tol}, M_μ を用いて, 条件

$$
\phi(x,t,s,z,v,w;\mu)\le M_{tol}\mu \tag{2.5}
$$

が成立した段階で, 現在の μ に対するニュートン法の反復を打ち切る. そして, μ の値を

$$\mu := \frac{\phi(x,t,s,z,v,w;\mu)}{M_\mu}$$

により更新し, 停止条件 (2.4) が成り立つまで再びニュートン法の反復を繰返す [23] . このとき, 条件 (2.5) と $M_{tol} < M_\mu$ より, 次の不等式が成り立つことに注意しよう.

$$\frac{\phi(x,t,s,z,v,w;\mu)}{M_\mu} \leq \frac{M_{tol}}{M_\mu}\mu < \mu$$

以上をまとめると, 問題 (2.1) に対する主双対内点法のアルゴリズムは次のように書ける.

アルゴリズム **PDIP**

ステップ **1** 初期実行可能内点 $(x^{(1)}, t^{(1)}, s^{(1)}, y^{(1)}, z^{(1)}, v^{(1)}, w^{(1)})$ と, 十分大きな正の数 $\mu^{(1)}$ を選ぶ. パラメータ $0 < \varepsilon_\mu \ll M_{tol} < M_\mu$, $\tau \in (0,1)$ を定め, $k := 1$ とおく.

ステップ **2** 連立方程式

$$\begin{aligned}
&G\Delta x - A^\top \Delta y + C^\top \Delta v + \Delta w - \Delta z = 0,\\
&A\Delta x = 0,\\
&C\Delta x + \Delta t = 0,\\
&\Delta x + \Delta s = 0,\\
&Z^{(k)}\Delta x + X^{(k)}\Delta z = \mu^{(k)} e_n - Z^{(k)} x^{(k)},\\
&V^{(k)}\Delta t + T^{(k)}\Delta v = \mu^{(k)} e_{\overline{n}} - V^{(k)} t^{(k)},\\
&W^{(k)}\Delta s + S^{(k)}\Delta w = \mu^{(k)} e_n - W^{(k)} s^{(k)}
\end{aligned}$$

の解 $(\Delta x^{(k)}, \Delta t^{(k)}, \Delta s^{(k)}, \Delta y^{(k)}, \Delta z^{(k)}, \Delta v^{(k)}, \Delta w^{(k)})$ を求める.

ステップ **3** ステップサイズ $\theta^{(k)}$ を

$$\begin{aligned}
\eta^{(k)} &:= \max\Big\{\nu \;\Big|\; \big(x^{(k)}, t^{(k)}, s^{(k)}, z^{(k)}, v^{(k)}, w^{(k)}\big)\\
&\qquad + \nu\,\big(\Delta x^{(k)}, \Delta t^{(k)}, \Delta s^{(k)}, \Delta z^{(k)}, \Delta v^{(k)}, \Delta w^{(k)}\big) \geq 0\Big\},\\
\theta^{(k)} &:= \min\{\tau\eta^{(k)}, 1\},
\end{aligned}$$

により求め, 探索点を次のように更新する.

$$\begin{aligned}
&(x^{(k+1)}, t^{(k+1)}, s^{(k+1)}, y^{(k+1)}, z^{(k+1)}, v^{(k+1)}, w^{(k+1)})\\
&\quad := (x^{(k)}, t^{(k)}, s^{(k)}, y^{(k)}, z^{(k)}, v^{(k)}, w^{(k)})\\
&\qquad + \theta^{(k)}(\Delta x^{(k)}, \Delta t^{(k)}, \Delta s^{(k)}, \Delta y^{(k)}, \Delta z^{(k)}, \Delta v^{(k)}, \Delta w^{(k)})
\end{aligned}$$

ステップ **4** 停止基準

$$\phi(x^{(k+1)}, t^{(k+1)}, s^{(k+1)}, z^{(k+1)}, v^{(k+1)}, w^{(k+1)}; \mu^{(k)}) \leq \varepsilon_\mu$$

が成り立てば終了する.

ステップ **5** $\phi(x^{(k+1)}, t^{(k+1)}, s^{(k+1)}, z^{(k+1)}, v^{(k+1)}, w^{(k+1)}; \mu^{(k)}) \leq M_{tol}\,\mu^{(k)}$ ならば

$$\mu^{(k+1)} := \frac{\phi(x^{(k+1)}, t^{(k+1)}, s^{(k+1)}, z^{(k+1)}, v^{(k+1)}, w^{(k+1)}; \mu^{(k)})}{M_\mu},$$

さもなければ $\mu^{(k+1)} := \mu^{(k)}$ とする. $k := k+1$ としてステップ 2 へ戻る. □

3. 多品種流問題への適用

ここでは, ノード集合 $\mathcal{V}$ とアーク集合 $\mathcal{E}$ から成るグラフ $\mathcal{G}=(\mathcal{V},\mathcal{E})$ 上で定義され, コストが品種に関して分離可能な 2 次関数で与えられる多品種流問題にアルゴリズム PDIP を適用することを考える. グラフ $\mathcal{G}$ のノード／アーク接続行列を $\overline{A}$, 品種 ℓ に対するノードの需要／供給量ベクトルを b_ℓ, アークのフロー上限ベクトルを u_ℓ, またフロー全体に対するアークの容量ベクトルを d とすると, 品種数が L の問題は次のように記述できる.

$$
\begin{aligned}
&\text{目的関数：} && \sum_{\ell=1}^{L}\left\{\frac{1}{2}x_\ell^\top G_\ell x_\ell + h_\ell^\top x_\ell\right\} \ \rightarrow\ \text{最小} \\
&\text{制約条件：} && \overline{A}x_\ell = b_\ell, && \ell=1,\cdots,L, \\
& && \sum_{\ell=1}^{L} x_\ell \le d, && \\
& && 0\le x_\ell \le u_\ell, && \ell=1,\cdots,L.
\end{aligned}
\tag{3.1}
$$

ただし, $|\mathcal{V}|=\overline{m}$, $|\mathcal{E}|=\overline{n}$ で, G_ℓ は $\overline{n}\times\overline{n}$ 半正定値対称行列, $\overline{A}$ はその行が互いに一次独立なベクトル $\overline{a}_i^\top$, $i=1,\cdots,\overline{m}$, によって構成される $\overline{m}\times\overline{n}$ 行列, h_ℓ, b_ℓ, d, u_ℓ はそれぞれ $R^{\overline{n}}$, $R^{\overline{m}}$, $R^{\overline{n}}$, $R^{\overline{n}}$ のベクトルである. いま, $\overline{n}\times\overline{n}$ 単位行列を $I_{\overline{n}}$ と書き, $m=L\overline{m}$, $n=L\overline{n}$ として,

$$
A=\begin{pmatrix}\overline{A} & & \\ & \ddots & \\ & & \overline{A}\end{pmatrix},\quad
b=\begin{pmatrix}b_1\\ \vdots\\ b_L\end{pmatrix},\quad
C=(I_{\overline{n}}\cdots I_{\overline{n}}),
$$

$$
G=\begin{pmatrix}G_1 & & \\ & \ddots & \\ & & G_L\end{pmatrix},\quad
h=\begin{pmatrix}h_1\\ \vdots\\ h_L\end{pmatrix},\quad
x=\begin{pmatrix}x_1\\ \vdots\\ x_L\end{pmatrix},\quad
u=\begin{pmatrix}u_1\\ \vdots\\ u_L\end{pmatrix}
$$

とおけば, 問題 (3.1) は問題 (2.1) に帰着する. さらに, 問題 (3.1) の不等式制約 $x_\ell\le u_\ell$ に対するスラック変数を $s_\ell\in R^{\overline{n}}$, 制約条件 $\overline{A}x_\ell=b_\ell$, $x_\ell\le u_\ell$, $x_\ell\ge 0$ に対する双対変数を, それぞれ $y_\ell\in R^{\overline{m}}$, $w_\ell\in R^{\overline{n}}$, $z_\ell\in R^{\overline{n}}$ とし,

$$
s=\begin{pmatrix}s_1\\ \vdots\\ s_L\end{pmatrix},\quad
y=\begin{pmatrix}y_1\\ \vdots\\ y_L\end{pmatrix},\quad
w=\begin{pmatrix}w_1\\ \vdots\\ w_L\end{pmatrix},\quad
z=\begin{pmatrix}z_1\\ \vdots\\ z_L\end{pmatrix}
$$

とおく. このとき, アルゴリズム PDIP のステップ 2 で解くべき連立方程式 (2.3) は,

$$
\begin{aligned}
&G_\ell\Delta x_\ell-\overline{A}^\top\Delta y_\ell+\Delta v+\Delta w_\ell-\Delta z_\ell=0, && \ell=1,\cdots,L,\\
&\overline{A}\,\Delta x_\ell=0, && \ell=1,\cdots,L,\\
&\textstyle\sum_{\ell=1}^{L}\Delta x_\ell+\Delta t=0, && \\
&\Delta x_\ell+\Delta s_\ell=0, && \ell=1,\cdots,L,\\
&Z_\ell\,\Delta x_\ell+X_\ell\Delta z_\ell=\mu e_{\overline{n}}-Z_\ell x_\ell, && \ell=1,\cdots,L,\\
&V\ \Delta t\ +\ T\,\Delta v\ =\mu e_{\overline{n}}-V\,t, && \\
&W_\ell\Delta s_\ell+S_\ell\Delta w_\ell=\mu e_{\overline{n}}-W_\ell s_\ell, && \ell=1,\cdots,L,
\end{aligned}
\tag{3.2}
$$

と書き換えられる. そこで, 連立方程式 (3.2) を解く具体的方法について考えよう. まず, (3.2)

の第 4, 5, 7 式より

$$
\begin{aligned}
\Delta s_\ell &= -\Delta x_\ell, && \ell = 1, \cdots, L, \\
\Delta z_\ell &= X_\ell^{-1}\{\mu e_{\overline{n}} - Z_\ell(x_\ell + \Delta x_\ell)\}, && \ell = 1, \cdots, L, \\
\Delta w_\ell &= S_\ell^{-1}\{\mu e_{\overline{n}} - W_\ell(s_\ell + \Delta s_\ell)\}, && \ell = 1, \cdots, L,
\end{aligned} \tag{3.3}
$$

を, また, (3.2) の第 6 式より

$$
\Delta t = V^{-1}\{\mu e_{\overline{n}} - T(v + \Delta v)\} \tag{3.4}
$$

を得る. いま, G_ℓ が半正定値対称で $x_\ell > 0$, $s_\ell > 0$, $z_\ell > 0$, $w_\ell > 0$ であることに注意すると,

$$
\Xi_\ell = X_\ell S_\ell G_\ell + S_\ell Z_\ell + X_\ell W_\ell, \qquad \ell = 1, \cdots, L,
$$

は正定値対称行列となる. そこで,

$$
\begin{aligned}
\Psi_\ell &= \Xi_\ell^{-1} X_\ell S_\ell, && \ell = 1, \cdots, L, \\
\lambda_\ell &= \Xi_\ell^{-1}\{S_\ell(\mu e_{\overline{n}} - Z_\ell x_\ell) - X_\ell(\mu e_{\overline{n}} - W_\ell s_\ell)\}, && \ell = 1, \cdots, L,
\end{aligned} \tag{3.5}
$$

とおくと, (3.3) の各式を用いて (3.2) の第 1 式を整理することにより, Δx_ℓ は次式で求められる.

$$
\Delta x_\ell = \Psi_\ell\left(\overline{A}^\top \Delta y_\ell - \Delta v\right) + \lambda_\ell, \qquad \ell = 1, \cdots, L. \tag{3.6}
$$

さらに, (3.6) を (3.2) の第 2 式に代入して整理すると,

$$
\overline{A}\Psi_\ell \overline{A}^\top \Delta y_\ell = \overline{A}(\Psi_\ell \Delta v - \lambda_\ell), \qquad \ell = 1, \cdots, L, \tag{3.7}
$$

を得る. 行列 $\overline{A}$ の各行は一次独立, Ψ_ℓ は正定値対称行列であることに注意すると, 連立方程式 (3.7) の係数行列も正定値対称である. そこで, (3.6) と (3.4) を (3.2) の第 3 式に代入し, (3.7) を考慮して整理すると

$$
\begin{aligned}
&\left[V\sum_{\ell=1}^{L}\left\{\Psi_\ell - \Psi_\ell \overline{A}^\top\left(\overline{A}\Psi_\ell\overline{A}^\top\right)^{-1}\overline{A}\Psi_\ell\right\} + T\right]\Delta v \\
&= V\sum_{\ell=1}^{L}\left\{\lambda_\ell - \Psi_\ell\overline{A}^\top\left(\overline{A}\Psi_\ell\overline{A}^\top\right)^{-1}\overline{A}\lambda_\ell\right\} + \mu e_{\overline{n}} - Vt
\end{aligned} \tag{3.8}
$$

を得る. 結局, 連立方程式 (3.8) を解いて Δv を求め, その結果をもとに (3.4) 式により Δt を計算する. 一方, 各品種 ℓ について独立に連立方程式 (3.7) を解いてΔy_ℓ を求めた上で, (3.6) および (3.3) の各式を用いてΔx_ℓ, Δs_ℓ, Δz_ℓ, Δw_ℓ を計算すればよいことがわかる.

なお, アルゴリズム PDIP の反復が進むにつれて, 変数 x_ℓ, s_ℓ, v の成分のいくつかが 0 に近づく可能性がある. 特に, 問題 (3.1) の解において狭義の相補条件が成り立っていない場合, すなわち

$$
t_j = 0, \quad v_j = 0
$$

であるような添字 $j \in \{1, \cdots, \overline{n}\}$ が存在するときには, 連立方程式 (3.8) の係数行列の第 j 行のすべての要素が 0 に近づくため, 数値的に悪条件となるおそれがある. そこで, 数値的安定性を確保するために実際の計算の場面においては両辺に左から対角行列 V^{-1} をかけて

式のスケーリングを行い, 連立方程式

$$
\begin{aligned}
& V^{-1}\left[V\sum_{\ell=1}^{L}\left\{\Psi_\ell-\Psi_\ell\overline{A}^{\top}\left(\overline{A}\Psi_\ell\overline{A}^{\top}\right)^{-1}\overline{A}\Psi_\ell\right\}+T\right]\Delta v \\
= \; & V^{-1}\left[V\sum_{\ell=1}^{L}\left\{\lambda_\ell-\Psi_\ell\overline{A}^{\top}\left(\overline{A}\Psi_\ell\overline{A}^{\top}\right)^{-1}\overline{A}\lambda_\ell\right\}+\mu e_{\overline{n}}-Vt\right]
\end{aligned}
\tag{3.9}
$$

を解いて Δv を求めることにする. また, (3.3) の第 2, 3 式および式 (3.4), さらに (3.5) の各式においては, 行列 X_ℓ^{-1}, S_ℓ^{-1}, V^{-1} および Ξ_ℓ^{-1} を括り出すことによって, 加減算を行う際の桁落ちによる計算誤差の発生を抑制している.

次に, アルゴリズム PDIP のステップ 3 について考える. いま,

$$
\begin{aligned}
\eta_0 &= \max\{\nu_0 \mid (t,v)+\nu_0(\Delta t,\Delta v)\geq 0\}, \\
\eta_\ell &= \max\{\nu_\ell \mid (x_\ell,s_\ell,z_\ell,w_\ell)+\nu_\ell(\Delta x_\ell,\Delta s_\ell,\Delta z_\ell,\Delta w_\ell)\geq 0\}, \quad \ell=1,\cdots,L,
\end{aligned}
\tag{3.10}
$$

とおくと, η_ℓ は各品種 ℓ について独立に計算でき, ステップサイズ θ の限界を与える値 η は,

$$
\eta=\min\{\eta_0,\eta_1,\cdots,\eta_L\} \tag{3.11}
$$

により求められる.

最後に, アルゴリズム PDIP のステップ 4 および 5 について考える. 多品種流問題 (3.1) に対して, 方程式系 (2.2) の相補条件に関する評価関数 ϕ は

$$
\phi(x,t,s,z,v,w;\mu)=\max\left\{\frac{1}{L}\sum_{\ell=1}^{L}\frac{\|Z_\ell x_\ell-\mu e_{\overline{n}}\|_1}{\overline{n}},\ \frac{\|Vt-\mu e_{\overline{n}}\|_1}{\overline{n}},\ \frac{1}{L}\sum_{\ell=1}^{L}\frac{\|W_\ell s_\ell-\mu e_{\overline{n}}\|_1}{\overline{n}}\right\}
$$

と書き換えられる. しかしながら, このままでは max の内部に ℓ に関する合算操作が 2 箇所存在するため, 品種ごとに独立に計算できる部分が小さくなってしまう. そこで,

$$
\begin{aligned}
\phi_0(t,v;\mu) &= \frac{\|Vt-\mu e_{\overline{n}}\|_1}{\overline{n}}, \\
\phi_\ell(x_\ell,s_\ell,z_\ell,w_\ell;\mu) &= \max\left\{\frac{\|Z_\ell x_\ell-\mu e_{\overline{n}}\|_1}{\overline{n}},\ \frac{\|W_\ell s_\ell-\mu e_{\overline{n}}\|_1}{\overline{n}}\right\}, \quad \ell=1,\cdots,L,
\end{aligned}
\tag{3.12}
$$

とおき, 各品種 ℓ について独立に ϕ_ℓ を計算した上で, 少し修正した評価関数

$$
\overline{\phi}(x,t,s,z,v,w;\mu)=\max\{\phi_0(t,v;\mu),\phi_1(x_1,s_1,z_1,w_1;\mu),\cdots,\phi_L(x_L,s_L,z_L,w_L;\mu)\} \tag{3.13}
$$

の値を求め, 収束判定と μ の値の更新を行うことにする.

4. 前処理つき共役勾配法による連立方程式の解法

多品種流問題 (3.1) にアルゴリズム PDIP を適用したとき, その各反復において, 全品種に関する情報を用いて定義される連立方程式 (3.9) と, 各品種ごとに並列的に処理可能な L 個の連立方程式 (3.7) を解かなければならない. ここでは, 問題 (3.1) を定義するグラフ $\mathcal{G}$ が大規模かつ疎である場合を想定して, 共役勾配法を用いてこれらの連立方程式を解くことにする.

まず, 連立方程式 (3.7) について考える. いま,

$$
\begin{aligned}
M_\ell &= \overline{A}\Psi_\ell\overline{A}^{\top}, \qquad \ell=1,\cdots,L, \\
q_\ell &= \overline{A}(\Psi_\ell\Delta v-\lambda_\ell), \quad \ell=1,\cdots,L,
\end{aligned}
$$

とおくと, $\overline{m} \times \overline{m}$ 行列 M_ℓ はいずれも正定値対称で, 各 Δy_ℓ はそれぞれ連立方程式

$$M_\ell u_\ell = q_\ell \tag{4.1}$$

の唯一の解 $u_\ell \in R^{\overline{m}}$ として求められる. 連立方程式 (4.1) に対する共役勾配法は, 理論的には係数行列 M_ℓ の相異なる固有値の数と等しい回数の反復で停止することが知られている. しかし実際には, 有限桁演算にともなう誤差等により, 無限点列を生成する反復解法として取扱わざるを得ない場合が多い. いま, 行列 Ψ_ℓ が正定値であることに注意すると, 行列 M_ℓ の対角成分はすべて正である. そこで, M_ℓ の対角成分のみを取り出した行列

$$D_\ell = \mathrm{diag}\left\{\overline{a}_i^\top \Psi_\ell \overline{a}_i\right\}_{i=1,\cdots,\overline{m}}$$

を用いた対角スケーリング (diagonal scaling) と呼ばれる最も簡単な前処理を行って, 共役勾配法の手続きを実行する際の数値的安定性を確保する.

連立方程式 (3.7) を解くための前処理つき共役勾配法の手続きは, 次のように記述される.

手続き $\mathbf{PCG}_\ell$

ステップ **1** 初期探索点 $u_\ell^{(1)} \in R^{\overline{m}}$ を選び, 許容誤差 $\varepsilon_\ell > 0$ と前処理行列

$$D_\ell := \mathrm{diag}\left\{\overline{a}_i^\top \Psi_\ell \overline{a}_i\right\}_{i=1,\cdots,\overline{m}}$$

を定める. また, 初期探索点における残差ベクトル

$$r_\ell^{(1)} := q_\ell - M_\ell u_\ell^{(1)}$$

を計算し, $j_\ell := 1$ とおく.

ステップ **2** 探索方向 $p_\ell^{(j_\ell)}$ を次のように定める.

$$p_\ell^{(j_\ell)} := D_\ell^{-1} r_\ell^{(j_\ell)}.$$

ステップ **3** 次式により, ステップサイズ $\alpha_\ell^{(j_\ell)}$ を計算する.

$$\begin{aligned}
\zeta_\ell^{(j_\ell)} &:= M_\ell p_\ell^{(j_\ell)}, \\
\omega_\ell^{(j_\ell)} &:= \left(p_\ell^{(j_\ell)}\right)^\top \zeta_\ell^{(j_\ell)}, \\
\xi_\ell^{(j_\ell)} &:= \left(p_\ell^{(j_\ell)}\right)^\top r_\ell^{(j_\ell)}, \\
\alpha_\ell^{(j_\ell)} &:= \xi_\ell^{(j_\ell)} / \omega_\ell^{(j_\ell)}.
\end{aligned}$$

ステップ **4** 次式により, 探索点と残差ベクトルの更新を行う.

$$\begin{aligned}
u_\ell^{(j_\ell+1)} &:= u_\ell^{(j_\ell)} + \alpha_\ell^{(j_\ell)} p_\ell^{(j_\ell)}, \\
r_\ell^{(j_\ell+1)} &:= r_\ell^{(j_\ell)} - \alpha_\ell^{(j_\ell)} \zeta_\ell^{(j_\ell)}.
\end{aligned}$$

ステップ **5** 停止基準

$$\| r_\ell^{(j_\ell+1)} \|_\infty \leq \varepsilon_\ell \| q_\ell \|_\infty$$

が成り立てば終了する．ただし，$\| \cdot \|_\infty$ は l_∞ ノルムである．

ステップ **6** $j_\ell = 0 \,(\mathrm{mod}\ \overline{m})$ ならば，$j_\ell := j_\ell + 1$ としてステップ 2 へ戻る．さもなければ，

$$\begin{aligned} \beta_\ell^{(j_\ell)} &:= \frac{\left(r_\ell^{(j_\ell+1)}\right)^\top D_\ell^{-1} r_\ell^{(j_\ell+1)}}{\xi_\ell^{(j_\ell)}}, \\ p_\ell^{(j_\ell+1)} &:= D_\ell^{-1} r_\ell^{(j_\ell+1)} + \beta_\ell^{(j_\ell)} p_\ell^{(j_\ell)} \end{aligned}$$

により共役方向の生成を行い，$j_\ell := j_\ell + 1$ としてステップ 3 へ戻る． □

なお，行列 M_ℓ は，グラフ $\mathcal{G}$ の接続行列 $\overline{A}$ を用いて $\overline{A}\Psi_\ell\overline{A}^\top$ と書けるので，Ψ_ℓ を構成する行列 G_ℓ が対角行列などの特殊な構造をもっている場合には，手続き PCG_ℓ のステップ 1 およびステップ 3 に現れる行列 M_ℓ とベクトルとのかけ算は，グラフの構造を利用して効率的に実行できる．

次に，連立方程式 (3.9) について考える．表記を簡単にするため，

$$M = V^{-1}\left[V\sum_{\ell=1}^{L}\left\{\Psi_\ell - \Psi_\ell\overline{A}^\top\left(\overline{A}\Psi_\ell\overline{A}^\top\right)^{-1}\overline{A}\Psi_\ell\right\} + T\right], \tag{4.2}$$

$$q = V^{-1}\left[V\sum_{\ell=1}^{L}\left\{\lambda_\ell - \Psi_\ell\overline{A}^\top\left(\overline{A}\Psi_\ell\overline{A}^\top\right)^{-1}\overline{A}\lambda_\ell\right\} + \mu e_{\overline{n}} - Vt\right] \tag{4.3}$$

とおく．Ψ_ℓ は正定値対称行列なので，$\Psi_\ell = \Phi_\ell^\top\Phi_\ell$ なる正則行列 Φ_ℓ が存在する．したがって，

$$\hat{A}_\ell = \overline{A}\Phi_\ell^\top, \qquad P_\ell = I_{\overline{n}} - \hat{A}_\ell^\top\left(\hat{A}_\ell\hat{A}_\ell^\top\right)^{-1}\hat{A}_\ell$$

とおくと，(4.2) の中括弧の内部は

$$\Psi_\ell - \Psi_\ell\overline{A}^\top\left(\overline{A}\Psi_\ell\overline{A}^\top\right)^{-1}\overline{A}\Psi_\ell = \Phi_\ell^\top P_\ell\Phi_\ell$$

と書き直すことができる．ここで，$P_\ell^\top P_\ell = P_\ell$ と $v > 0, t > 0$ に注意すると，行列 M は正定値対称であることがわかる．よって，Δv は連立方程式

$$Mu = q \tag{4.4}$$

の唯一の解 $u \in R^{\overline{n}}$ として求められる．

連立方程式 (4.4) を共役勾配法を用いて解く場合には，式 (4.2) および (4.3) の右辺に含まれる逆行列 $(\overline{A}\Psi_\ell\overline{A}^\top)^{-1}$ の取扱いが問題となる．ベクトル q を計算する際には，連立方程式

$$\left(\overline{A}\Psi_\ell\overline{A}^\top\right)u_\ell = \overline{A}\lambda_\ell \tag{4.5}$$

の解 $\hat{u}_\ell \in R^{\overline{m}}$ を各品種 ℓ について並列的に求め，

$$q = V^{-1}\left\{V\sum_{\ell=1}^{L}\left(\lambda_\ell - \Psi_\ell\overline{A}^\top\hat{u}_\ell\right) + \mu e_{\overline{n}} - Vt\right\}$$

とすればよい．連立方程式 (4.5) は，

$$\begin{aligned} M_\ell &= \overline{A}\Psi_\ell\overline{A}^\top, \\ q_\ell &= \overline{A}\lambda_\ell \end{aligned}$$

とおいて, 連立方程式 (4.1) に対する前処理つき共役勾配法の手続き PCG_ℓ を適用して解くことができる. 一方, 連立方程式 (4.4) に対する共役勾配法の第 j 反復においてステップサイズを計算する際には, 行列 M と探索方向 $p^{(j)}$ との積

$$\zeta^{(j)} = M\,p^{(j)}$$

を求める必要がある. この場合も, 連立方程式

$$\left(\overline{A}\,\Psi_\ell\,\overline{A}^{\top}\right) u_\ell = \overline{A}\,\Psi_\ell\,p^{(j)} \tag{4.6}$$

の解 $\tilde{u}_\ell \in R^{\overline{m}}$ を各品種 ℓ について並列的に求め,

$$\zeta^{(j)} = V^{-1}\left\{ V\sum_{\ell=1}^{L}\left(\Psi_\ell\,p^{(j)} - \Psi_\ell\,\overline{A}^{\top}\tilde{u}_\ell\right) + T\,p^{(j)} \right\}$$

とする. また, 連立方程式 (4.6) を解くには,

$$\begin{aligned} M_\ell &= \overline{A}\,\Psi_\ell\,\overline{A}^{\top}, \\ q_\ell &= \overline{A}\,\Psi_\ell\,p^{(j)} \end{aligned}$$

とおいて, 手続き PCG_ℓ を適用する. なお, 初期探索方向の計算に必要な行列 M と初期探索点 $u^{(1)}$ の積も, 各品種 ℓ について連立方程式

$$\left(\overline{A}\,\Psi_\ell\,\overline{A}^{\top}\right) u_\ell = \overline{A}\,\Psi_\ell\,u^{(1)} \tag{4.7}$$

を解くことにより, 同様に求められる.

ところで, 連立方程式 (4.5) に対して手続き PCG_ℓ を適用した際に生成される互いに共役な探索方向ベクトルを $p_\ell^{(1)}, \cdots, p_\ell^{(\widehat{m}_\ell)}$, $\widehat{m}_\ell \le \overline{m}$, とし,

$$\begin{aligned} P_\ell &= \left[p_\ell^{(1)} \cdots p_\ell^{(\widehat{m}_\ell)} \right], \\ \Omega_\ell &= \mathrm{diag}\left\{ \omega_\ell^{(j_\ell)} \right\}_{j_\ell = 1, \cdots, \widehat{m}_\ell} \end{aligned}$$

とおくと, 連立方程式 (4.5) の係数行列 $M_\ell = \overline{A}\,\Psi_\ell\,\overline{A}^{\top}$ の逆行列は,

$$\left(\overline{A}\,\Psi_\ell\,\overline{A}^{\top}\right)^{-1} \cong P_\ell\,\Omega_\ell^{-1}\,P_\ell^{\top}$$

のように評価できることが知られている [21]. したがって,

$$v_\ell^{(j_\ell)} = \overline{A}^{\top} p_\ell^{(j_\ell)}, \qquad j_\ell = 1, \cdots, \widehat{m}_\ell,$$

とおくと,

$$\begin{aligned} \overline{A}^{\top}\left(\overline{A}\,\Psi_\ell\,\overline{A}^{\top}\right)^{-1}\overline{A} &\cong \overline{A}^{\top} P_\ell\,\Omega_\ell^{-1}\,P_\ell^{\top}\,\overline{A} \\ &= \left[v_\ell^{(1)} \cdots v_\ell^{(\widehat{m}_\ell)} \right] \mathrm{diag}\left\{ \frac{1}{\omega_\ell^{(j_\ell)}} \right\}_{j_\ell = 1, \cdots, \widehat{m}_\ell} \left[v_\ell^{(1)} \cdots v_\ell^{(\widehat{m}_\ell)} \right]^{\top} \\ &= \sum_{j_\ell = 1}^{\widehat{m}_\ell} \frac{v_\ell^{(j_\ell)}\left(v_\ell^{(j_\ell)}\right)^{\top}}{\omega_\ell^{(j_\ell)}} \end{aligned}$$

を得る．よって，$\overline{A}^{\top}(\overline{A}\Psi_\ell\overline{A}^{\top})^{-1}\overline{A}$ の対角部分は$\sum_{j_\ell=1}^{\widehat{m}_\ell}(\Upsilon_\ell^{(j_\ell)})^2/\omega_\ell^{(j_\ell)}$ と近似できることがわかる．ただし，$\Upsilon_\ell^{(j_\ell)}$ はベクトル $v_\ell^{(j_\ell)}$ の各要素を対角成分とする対角行列である．いま，正定値対称行列 Ψ_ℓ の対角部分のみから成る行列を $\overline{\Psi}_\ell$ と書くことにすると，式(4.2) で定義される行列 M の対角部分の近似行列として，対角行列

$$D = V^{-1}\left[V\sum_{\ell=1}^{L}\left\{\overline{\Psi}_\ell - \overline{\Psi}_\ell\left(\sum_{j_\ell=1}^{\widehat{m}_\ell}\frac{\left(\Upsilon_\ell^{(j_\ell)}\right)^2}{\omega_\ell^{(j_\ell)}}\right)\overline{\Psi}_\ell\right\} + T\right]$$

を考えることができる．しかも，ベクトル $v_\ell^{(j_\ell)}$ は，手続き PCG_ℓ のステップ 3 における $\zeta_\ell^{(j_\ell)}$ の計算の中で求められるため，行列 D の計算そのものはきわめて容易である．

連立方程式 (4.5) は，連立方程式 (4.4) に対して共役勾配法を適用する際に，ベクトル q を計算する目的で最初に一度だけ解かれる．その際に得られた行列 D のすべての対角成分が正であれば，連立方程式 (4.4) に対する共役勾配法の反復において，行列 D を用いた対角スケーリングに基づく前処理を組込むことができる．

連立方程式 (3.9) を解くための前処理つき共役勾配法の手続きは，次のように記述される．

手続き **PCG**

ステップ **1** 初期探索点 $u^{(1)} \in R^{\overline{n}}$ を選び，許容誤差 $\varepsilon > 0$ を定めて $j := 1$ とおく．

ステップ **2** M_ℓ, q_ℓ を次のようにおく．

$$\begin{aligned} M_\ell &:= \overline{A}\Psi_\ell\overline{A}^{\top}, \quad \ell = 1,\cdots,L, \\ q_\ell &:= \overline{A}\lambda_\ell, \qquad \ell = 1,\cdots,L. \end{aligned}$$

各 ℓ に対して並列的に手続き PCG_ℓ を実行し，連立方程式

$$M_\ell u_\ell = q_\ell, \qquad \ell = 1,\cdots,L,$$

の解 $\hat{u}_\ell$, $\ell = 1,\cdots,L$, を求め，

$$q := V^{-1}\left\{V\sum_{\ell=1}^{L}\left(\lambda_\ell - \Psi_\ell\overline{A}^{\top}\hat{u}_\ell\right) + \mu e_{\overline{n}} - Vt\right\}$$

とおく．さらに，前処理行列

$$D := V^{-1}\left[V\sum_{\ell=1}^{L}\left\{\overline{\Psi}_\ell - \overline{\Psi}_\ell\left(\sum_{j_\ell=1}^{\widehat{m}_\ell}\frac{\left(\Upsilon_\ell^{(j_\ell)}\right)^2}{\omega_\ell^{(j_\ell)}}\right)\overline{\Psi}_\ell\right\} + T\right]$$

を求める．

ステップ **3** M_ℓ, q_ℓ を次のようにおく．

$$\begin{aligned} M_\ell &:= \overline{A}\Psi_\ell\overline{A}^{\top}, \quad \ell = 1,\cdots,L, \\ q_\ell &:= \overline{A}\Psi_\ell u^{(1)}, \quad \ell = 1,\cdots,L. \end{aligned}$$

各 ℓ に対して並列的に手続き PCG_ℓ を実行し, 連立方程式

$$M_\ell u_\ell = q_\ell, \qquad \ell = 1, \cdots, L,$$

の解 $\bar{u}_\ell$, $\ell = 1, \cdots, L$, を求め,

$$r^{(1)} := q - V^{-1}\left\{ V \sum_{\ell=1}^{L} \left(\Psi_\ell u^{(1)} - \Psi_\ell \overline{A}^{\top} \bar{u}_\ell \right) + T u^{(1)} \right\}$$

とおく.

ステップ **4** 探索方向 $p^{(j)}$ を, 次のように定める.

$$p^{(j)} := D^{-1} r^{(j)}.$$

ステップ **5** M_ℓ, q_ℓ を次のようにおく.

$$\begin{aligned} M_\ell &:= \overline{A} \Psi_\ell \overline{A}^{\top}, \quad \ell = 1, \cdots, L, \\ q_\ell &:= \overline{A} \Psi_\ell p^{(j)}, \quad \ell = 1, \cdots, L. \end{aligned}$$

各 ℓ に対して並列的に手続き PCG_ℓ を実行し, 連立方程式

$$M_\ell u_\ell = q_\ell, \qquad \ell = 1, \cdots, L,$$

の解 $\tilde{u}_\ell$, $\ell = 1, \cdots, L$, を求め,

$$\zeta^{(j)} := V^{-1}\left\{ V \sum_{\ell=1}^{L} \left(\Psi_\ell p^{(j)} - \Psi_\ell \overline{A}^{\top} \tilde{u}_\ell \right) + T p^{(j)} \right\}$$

とおく.

ステップ **6** 次式により, ステップサイズ $\alpha^{(j)}$ を計算する.

$$\begin{aligned} \omega^{(j)} &:= \left(p^{(j)} \right)^{\top} \zeta^{(j)}, \\ \xi^{(j)} &:= \left(p^{(j)} \right)^{\top} r^{(j)}, \\ \alpha^{(j)} &:= \xi^{(j)} / \omega^{(j)}. \end{aligned}$$

ステップ **7** 次式により, 探索点と残差ベクトルの更新を行う.

$$\begin{aligned} u^{(j+1)} &:= u^{(j)} + \alpha^{(j)} p^{(j)}, \\ r^{(j+1)} &:= r^{(j)} - \alpha^{(j)} \zeta^{(j)}. \end{aligned}$$

ステップ **8** 停止基準

$$\| r^{(j+1)} \|_\infty \leq \varepsilon \| q \|_\infty$$

が成り立てば終了する.

ステップ **9** $j = 0 \,(\mathrm{mod}\, \bar{n})$ ならば, $j := j + 1$ としてステップ 4 へ戻る. さもなければ,

$$\begin{aligned} \beta^{(j)} &:= \frac{\left(r^{(j+1)} \right)^{\top} D^{-1} r^{(j+1)}}{\xi^{(j)}}, \\ p^{(j+1)} &:= D^{-1} r^{(j+1)} + \beta^{(j)} p^{(j)} \end{aligned}$$

により共役方向の生成を行い, $j := j + 1$ としてステップ 5 へ戻る. □

なお, 実際の数値計算の観点から考えると, アルゴリズム PDIP の初期の段階においては, 探索点が問題 (3.1) とその双対問題の解の組から十分に遠いと考えられるので, 連立方程式 (3.9) を必ずしも厳密に解く必要はない. そこで, 実際の計算においては, 手続き PCG の許容誤差 ε をアルゴリズム PDIP の反復回数 k を用いて

$$\varepsilon = \max\left\{\varepsilon_0\,,\, 10^{-k-1}\right\} \tag{4.8}$$

のように緩和することにする [2] .

5. 並列計算機での実行方法

ここでは, 前章までに述べたアルゴリズムを並列計算機上で実行する方法について考える. なお以下では, コスト関数が各枝に対しても分離可能で, グラフ $\mathcal{G}$ が供給節点の集合 $\mathcal{V}_1$ と需要節点の集合 $\mathcal{V}_2$ で構成される 2 部グラフである場合に限って議論を進める. 実際, 2 部グラフ上で定義され, 目的関数が分離可能な凸 2 次関数であるようなネットワーク問題は, さまざまな並列アルゴリズムの数値実験においてしばしば取り上げられている [6, 7, 8, 25, 27]. このとき, 行列 G_ℓ, $\ell = 1, \cdots, L$, は正定値対角行列で, フロー保存を表す式は

$$\begin{aligned} \sum_{j:(i,j)\in\mathcal{E}} (x_\ell)_{ij} &= \left(b_\ell^S\right)_i, \quad i \in \mathcal{V}_1, \quad \ell = 1, \cdots, L, \\ \sum_{i:(i,j)\in\mathcal{E}} (x_\ell)_{ij} &= \left(b_\ell^D\right)_j, \quad j \in \mathcal{V}_2, \quad \ell = 1, \cdots, L, \end{aligned} \tag{5.1}$$

のように記述される. ただし, b_ℓ^S, b_ℓ^D はそれぞれ品種 ℓ の供給量ベクトル, 需要量ベクトルで,

$$\sum_{i\in\mathcal{V}_1} \left(b_\ell^S\right)_i = \sum_{j\in\mathcal{V}_2} \left(b_\ell^D\right)_j, \qquad \ell = 1, \cdots, L,$$

とする. さらに, 2 部グラフ $\mathcal{G}$ の接続行列の行のうち, 供給節点の集合 $\mathcal{V}_1$ に対応する部分のみから構成される行列を A^S, 需要節点 $\mathcal{V}_2$ に対応する部分のみから構成される行列を A^D と書く. このとき, 式 (5.1) は

$$\begin{aligned} \left(A^S x_\ell\right)_i &= \left(b_\ell^S\right)_i, \quad i \in \mathcal{V}_1, \quad \ell = 1, \cdots, L, \\ \left(A^D x_\ell\right)_j &= \left(b_\ell^D\right)_j, \quad j \in \mathcal{V}_2, \quad \ell = 1, \cdots, L, \end{aligned}$$

と書き直せる. ところが, 各要素が 1 のベクトル $e_{|\mathcal{V}_1|} \in R^{|\mathcal{V}_1|}$, $e_{|\mathcal{V}_2|} \in R^{|\mathcal{V}_2|}$ を用いると

$$\left(A^S\right)^\top e_{|\mathcal{V}_1|} - \left(A^D\right)^\top e_{|\mathcal{V}_2|} = 0$$

となるから, グラフ $\mathcal{G}$ の接続行列の行は 1 次独立ではない. そこで, A^D からその最後の 1 行を削除した行列を $\tilde{A}^D$, b_ℓ^D からその最後の 1 要素を削除したベクトルを $\tilde{b}_\ell^D$ とし, 問題 (3.1) の $\overline{A}$, b_ℓ をそれぞれ

$$\overline{A} = \begin{bmatrix} A^S \\ \tilde{A}^D \end{bmatrix}, \qquad b_\ell = \begin{pmatrix} b_\ell^S \\ \tilde{b}_\ell^D \end{pmatrix}, \quad \ell = 1, \cdots, L, \tag{5.2}$$

と定める.

アルゴリズム PDIP のステップ 2 において連立方程式 (3.9) を手続き PCG を用いて解く際に, そのステップ 2, 3 および 5 で各品種 ℓ について手続き PCG_ℓ を並列的に実行できること, また, 連立方程式 (3.7) も手続き PCG_ℓ を用いて各品種 ℓ について独立に解けることから, ここで考えている主双対内点法はコントロール並列型のアルゴリズムとみなすこと

ができる．一方，大規模で疎な問題を取扱う場面を想定すると，手続き PCG_ℓ に含まれる線形代数的な計算をデータ並列型のプログラミング環境のもとで実行することによって，より高速な処理を実現することができるものと期待される．そこで以下では，並列型の主双対内点法のアルゴリズムを並列計算機上で実行するためのコントロール並列型の計算モデルと，共役勾配法の手続きをデータ並列型のプログラミング環境のもとで実行するための具体的方策について述べる．

5.1. コントロール並列型のプログラミング環境での主双対内点法の実行

互いに独立な手続きを実行できる複数の処理装置をもつ並列計算機上で十分な数の処理装置を利用できるという理想的な状況を想定すると，各品種 ℓ に対して一つづつの処理装置を割当て，それぞれにおいて手続き PCG_ℓ を並列的に実行させるという方式が，並列型の主双対内点法を実行する最も自然なコントロール並列型の計算モデルである．ただし，手続き PCG においては，これら L 個の処理装置がそれぞれ手続き PCG_ℓ を実行することによって得た情報を集約した上で若干の線形代数的計算を行う必要がある．また，3 章でも指摘したように，アルゴリズム PDIP のステップ 3, 4 において必要となる η, ϕ の値を求めるためには，品種ごとに独立に計算できる η_ℓ, ϕ_ℓ の他に，アークの総流量上限制約のスラック変数 t と対応する双対変数 v の値によって定義される η_0, ϕ_0 の計算が必要となる．そこで，もう一つの特別な処理装置を用意して，品種単位に分割できないこれらの情報にかかる処理を行わせると共に，手続き PCG およびアルゴリズム PDIP 全体の動きを制御する役割をもたせることにする．以下では，全品種にわたる情報を調整する役割を果たす後者の処理装置をマスター・ノード，品種ごとに独立な処理をつかさどる前者の処理装置をワーカー・ノードと呼ぶことにする．この並列計算モデルにおいては，ワーカー・ノードの行うデータ通信の相手先がマスター・ノードのみに限定されるため，各ワーカー・ノードで実行すべきプログラムはまったく同一のものとなる．したがって，コントロール並列型のプログラミング環境を提供する多くの並列計算機で採用されている SPMD (Single-Program Multiple-Data) 型の処理機構の上で，比較的容易に実現することができる．また，マスター・ノードを中心としたデータ通信を利用することによって，各ワーカー・ノードの同期をとることも可能である．ただし，マスター・ノードとワーカー・ノードがデータ通信を介して交互に動作する形態となるため，マスター・ノードの負荷が大きくなって並列化効率の低下を招くことのないように注意を払う必要がある．

次に，実際の計算において必要となる様々な情報を，マスター・ノードとワーカー・ノードのいずれが保持すべきかについて考える．多品種流問題 (3.1) を定義する情報のうちアルゴリズムを実行する際に必要となるのは，各品種 ℓ に対して定められる連立方程式 (3.7) に現れる行列 G_ℓ および $\overline{A}$ である．したがって，G_ℓ は対応する品種 ℓ にかかる処理を行うワーカー・ノードが，$\overline{A}$ はすべてのワーカー・ノードが保持する必要がある．連立方程式 (3.9) にも G_ℓ と $\overline{A}$ は現れているが，手続き PCG のステップ 2, 3 および 5 において各品種 ℓ について独立に実行される手続き PCG_ℓ の中でのみ用いられるため，マスター・ノードがその値を保持する必要はない．アルゴリズム PDIP のパラメータ $\varepsilon_\mu, M_{tol}, M_\mu, \tau$ と，式 (4.8) によって手続き PCG の許容誤差を与える ε_0 についてはマスター・ノードのみが，また，手続き PCG_ℓ の許容誤差を与える ε_ℓ は対応する品種 ℓ にかかる処理を行うワーカー・ノードのみがその値を保持していれば十分である．ただし，主双対内点法のパラメータ μ の値は，マスター・ノードとすべてのワーカー・ノードがおなじ値を保持している必要がある．

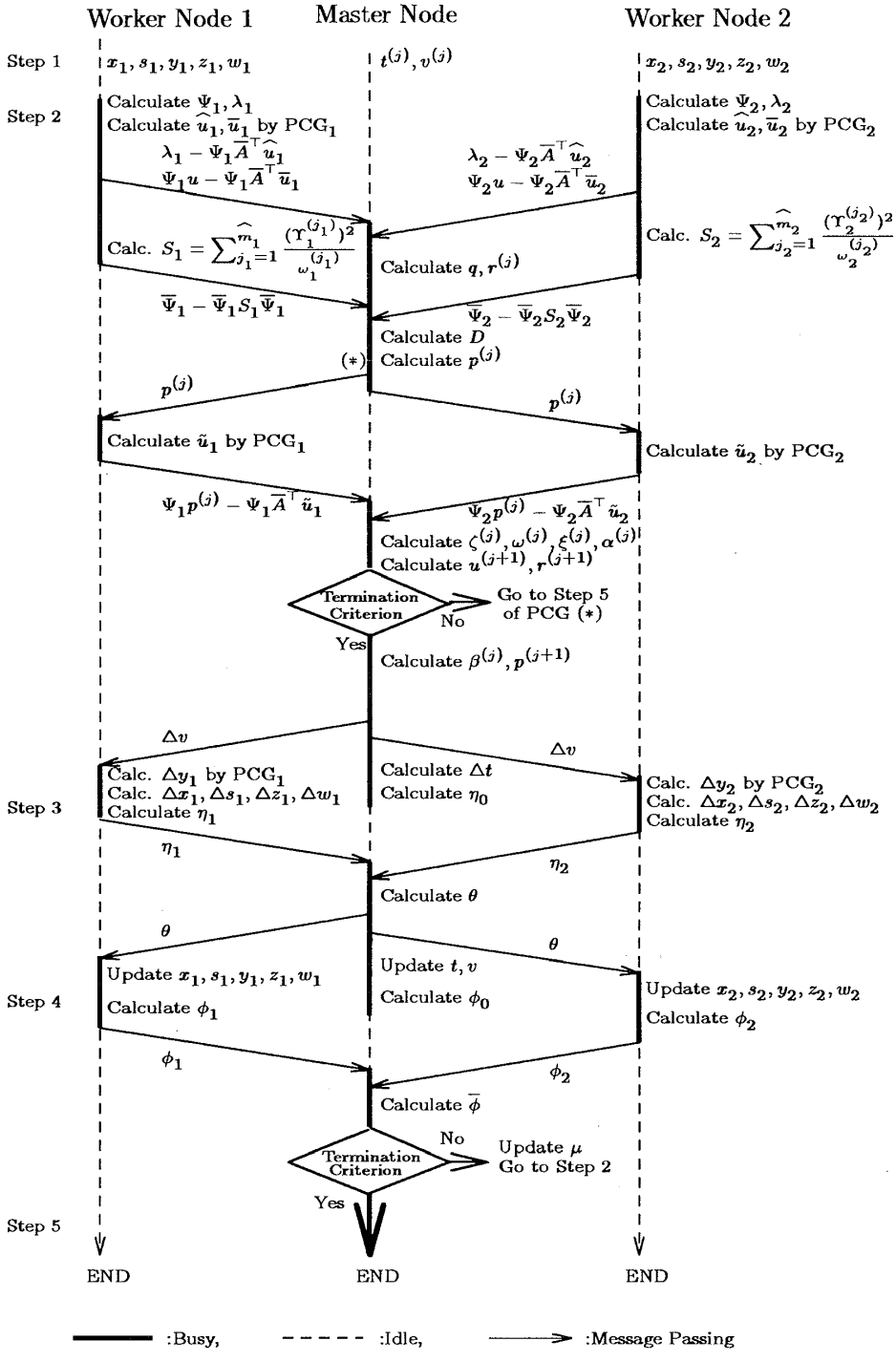

図 1: アルゴリズム PDIP を実行する並列計算モデル：$L = 2$ の場合

最後に, この並列計算モデルを用いて並列型の主双対内点法のアルゴリズムが実際にどのように実行されるかについて説明する. 品種数が 2 の場合の例を, 図 1 に掲げる. 初期実行可能内点を構成する各変数のうち, t, v の値がマスター・ノードに, また, $x_\ell, s_\ell, y_\ell, z_\ell, w_\ell$ の値が品種 ℓ にかかる処理を行うワーカー・ノードにそれぞれ保持されているものとする. 各ワーカー・ノードは, あらかじめ式 (3.5) にしたがって Ψ_ℓ, λ_ℓ を並列的に計算する. アルゴリズム PDIP のステップ 2 では, まず手続き PCG を用いて連立方程式 (3.9) を解く. 手続き PCG のステップ 2 および 3 では, 各ワーカー・ノードが独立にそれぞれ連立方程式 (4.5), (4.7) の解 $\hat{u}_\ell$ および $\bar{u}_\ell$ をいずれも手続き PCG_ℓ を用いて求めた後, それぞれ $\lambda_\ell - \Psi_\ell \overline{A}^\top \hat{u}_\ell$ および $\Psi_\ell u^{(1)} - \Psi_\ell \overline{A}^\top \bar{u}_\ell$ を並列的に計算してマスター・ノードに伝達する. マスター・ノードは, これらの値と変数 v, t の値などを用いてそれぞれ q および $r^{(1)}$ を計算する. ただしステップ 2 においては, 手続き PCG_ℓ の実行中に各ワーカー・ノードにおいて $(\Upsilon_\ell^{(j_\ell)})^2 / \omega_\ell^{(j_\ell)}$ の値を累積しておき, $\overline{\Psi}_\ell - \overline{\Psi}_\ell (\sum_{j_\ell=1}^{\widehat{m}_\ell} (\Upsilon_\ell^{(j_\ell)})^2 / \omega_\ell^{(j_\ell)}) \overline{\Psi}_\ell$ の値を並列的に計算してマスター・ノードに伝達する. マスター・ノードは, これらの値と変数 v, t の値を用いて前処理行列 D を求め, ステップ 4 にしたがって探索方向 $p^{(j)}$ を計算し, 各ワーカー・ノードに伝達する. ステップ 5 では, 各ワーカー・ノードが独立に連立方程式 (4.6) の解 $\tilde{u}_\ell$ を手続き PCG_ℓ を用いて求めた後, $\Psi_\ell p^{(j)} - \Psi_\ell \overline{A}^\top \tilde{u}_\ell$ を並列的に計算してマスター・ノードに伝達する. マスター・ノードは, これらの値と変数 v, t の値などを用いて $\varsigma^{(j)}$ を計算する. 以下, 手続き PCG のステップ 6 から 9 までは, マスター・ノードが実行する.

なお, 手続き PCG を適用して得られた連立方程式 (3.9) の解 Δv は, 次に解くべき連立方程式 (3.7) の定数ベクトルを計算するために各ワーカー・ノードに伝達されるので, アルゴリズム PDIP の 2 反復目以降で手続き PCG を実行する際の初期探索点としてこの値を用いるならば, マスター・ノードから各ワーカー・ノードへ初期探索点の値を改めて伝達する必要はない. また, アルゴリズム PDIP の最初の反復で手続き PCG を実行する際には, 0 などの自明な値を初期探索点として用いればよい. 各ワーカー・ノードは, マスター・ノードから伝達された Δv の値をもとに, 手続き PCG_ℓ を用いて連立方程式 (3.7) を並列的に解くことによって Δy_ℓ の値を計算する. さらに, $\Delta x_\ell, \Delta s_\ell, \Delta z_\ell, \Delta w_\ell$ の値も, (3.6) と (3.3) の各式を用いて並列的に計算する. 一方, マスター・ノードは, 式 (3.4) を用いて Δt の値を計算する.

アルゴリズム PDIP のステップ 3 では, 式 (3.10) にしたがってマスター・ノードが η_0 を, 各ワーカー・ノードが η_ℓ をそれぞれ並列的に計算する. 各ワーカー・ノードから η_ℓ の値を伝達されたマスター・ノードは, 式 (3.11) にしたがって η の値を計算し, さらにそれを用いて決定したステップサイズ θ を各ワーカー・ノードに伝達する. これをもとに, 変数 t, v の値はマスター・ノードで, また, $x_\ell, s_\ell, y_\ell, z_\ell, w_\ell$ の値は各ワーカー・ノードで並列的に更新される. アルゴリズム PDIP のステップ 4 についても同様に, 式 (3.12) にしたがってマスター・ノードが ϕ_0 を, 各ワーカー・ノードが ϕ_ℓ をそれぞれ並列的に計算する. 各ワーカー・ノードから ϕ_ℓ の値を伝達されたマスター・ノードは, 式 (3.13) にしたがって $\overline{\phi}$ の値を計算し, 収束判定を行う. まだ収束していないと判定された場合には, アルゴリズム PDIP のステップ 5 にしたがってマスター・ノードがパラメータ μ の値を更新し, 各ワーカー・ノードにその値を伝達する.

5.2. データ並列型のプログラミング環境での共役勾配法の実行

データ並列型のプログラムを実行できる並列計算機の最も基礎的な機能として, いくつかの部分 (segment) に分けられた配列に対して, 部分配列ごとにその要素の値を累積 (segmented-add-scan) したり, 各部分の片端の要素の値をその部分配列全体に複写 (segmented-copy-scan) する処理がある [26]. これらの機能を組合せると, その非零要素を行または列を単位とした部分に分けて 1 次元配列に格納した大規模行列に対して, ベクトルとの積などの線形代数的な計算を効率的に実行することができる. いま, 行列 G_ℓ は対角行列と仮定しているため, 大規模な多品種流問題に対して並列型の主双対内点法を実行する場合には, 前処理つき共役勾配法の手続き PCG および PCG$_\ell$ のいくつかのステップにおいて現れる行列 $\overline{A}$ またはその転置行列 $\overline{A}^{\mathrm{T}}$ とベクトルの積を計算する方法が問題になる. 一般に, ある行列とその転置行列の双方についてベクトルとの積を上の二つの機能を用いて実現しようとする場合には, 行の単位を表す情報と列の単位を表す情報を用意した上で, 行を単位とした場合の 1 次元配列への格納順と列を単位とした 1 次元配列への格納順との間の対応関係を示すポインタ情報を作成して, 二つの順序で格納されたデータの整合性をとる必要がある. ところが, ネットワークのノード／アーク接続行列は各列に非零要素が 2 箇所だけ存在するという特殊な構造をもっているため, 行を単位とした 1 次元配列への格納順のみを考え, おなじアークに対応する二つの非零要素間の対応関係を示すポインタ情報を保持しておけば, 所期の目的を果たすことができる. 実際, 行列とベクトルの積は上述の機能を組合せることによって得られ, 転置行列とベクトルとの積も, 各配列要素においてポインタ情報を用いて転送されてきたデータとの和をとることによって実現することができる. ただし, ポインタを用いたデータ転送は配列要素間の 1 対 1 通信となるため, 処理系によっては他の演算に比べて負荷がかかる場合がある. なお, 行列 $\overline{A}$ の非零要素の値はすべて 1 であるため, 積をとるべきベクトルをおなじ長さの 1 次元配列に格納し, その配列上でこれらの操作を行うだけでよい.

接続行列の非零要素数は, グラフ $\mathcal{G}$ のアーク数 $|\mathcal{E}|$ の2 倍である. そこで, アルゴリズムの実行にあたっては, 必要となる各変数をその長さが $2|\mathcal{E}|$ の 1 次元配列に接続行列の行, すなわちグラフのノードを単位とした順序で格納する. 例えば, x_ℓ などのアークに関する変数の各要素はそのアークの供給節点と需要節点に対応する 2 箇所に, また, y_ℓ などのノードに関する変数の各要素はそのノードに対応する部分配列全体におなじ値を保持する. なお, すでに述べたように, 2 部グラフの接続行列はその行が一次独立とならないため, 需要節点の最後の一つに対応する行を削除して $\overline{A}$ としている. そこで, あらかじめ各 1 次元配列の最後尾にある部分配列の各要素を当該演算の単位元に設定し直すことによって, 行を削除した場合とおなじ結果を得るようにしている.

簡単な例を用いて, 具体的に説明しよう. $\mathcal{G}$ を $|\mathcal{V}_1| = 4$, $|\mathcal{V}_2| = 3$, $|\mathcal{E}| = 10$ で, 供給節点 $\mathcal{V}_1$ に関する接続行列 A^S と需要節点 $\mathcal{V}_2$ に関する接続行列 A^D がそれぞれ

$$A^S = \begin{bmatrix} 1 & 1 & 1 & & & & & & & \\ & & & 1 & 1 & & & & & \\ & & & & & 1 & 1 & 1 & & \\ & & & & & & & & 1 & 1 \end{bmatrix},$$

$$A^D = \begin{bmatrix} 1 & & & 1 & & 1 & & & & \\ & 1 & & & 1 & & 1 & & 1 & \\ & & 1 & & & & & 1 & & 1 \end{bmatrix}$$

で与えられる 2 部グラフとする．このとき，アークに関する変数を $x \in R^{10}$，ノードに関する変数を $y \in R^6$ として，2 種類の積

$$\overline{A}x = \begin{pmatrix} x_1+x_2+x_3 \\ x_4+x_5 \\ x_6+x_7+x_8 \\ x_9+x_{10} \\ x_1+x_4+x_6 \\ x_2+x_5+x_7+x_9 \end{pmatrix}, \qquad \overline{A}^{\top}y = \begin{pmatrix} y_1+y_5 \\ y_1+y_6 \\ y_1 \\ y_2+y_5 \\ y_2+y_6 \\ y_3+y_5 \\ y_3+y_6 \\ y_3 \\ y_4+y_6 \\ y_4 \end{pmatrix}$$

を求めることを考える．接続行列の行の単位を表す情報を格納する論理配列を SEGM，接続行列の列の二つの非零要素間の対応関係を示すポインタ情報を格納する整数配列を SEND，変数 x の値を格納する実数配列を XVAL，変数 y の値を格納する実数配列を YVAL とするとき，長さ 20 の各 1 次元配列にはそれぞれ次のようなデータを保持する．

SEGM = (	T	F	F	T	F	T	F	F	T	F	T	F	F	T	F	F	F	T	F	F),
SEND = (	11	14	18	12	15	13	16	19	17	20	1	4	6	2	5	7	9	3	8	10),
XVAL = (	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_1	x_4	x_6	x_2	x_5	x_7	x_9	0	0	0),
YVAL = (	y_1	y_1	y_1	y_2	y_2	y_3	y_3	y_3	y_4	y_4	y_5	y_5	y_5	y_6	y_6	y_6	y_6	0	0	0).

論理配列 SEGM は，左から順に見ていったときに，値 T をもつある要素から値 T をもつ次の要素の直前の要素までの部分に，接続行列のおなじ行，すなわちグラフのおなじノードに対応する要素が格納されていることを表している．実際，実数配列 YVAL のおなじ部分配列に含まれる要素には，変数 y のおなじ成分の値が保持されている．ただし，A^D の最終行に対応する各配列の最後の部分は実際の計算に用いる行列 $\overline{A}$ では削除されているため，実数配列 XVAL および YVAL の該当部分には0 を格納している．実数配列 XVAL に対して，論理配列 SEGM が示す部分配列ごとに和を計算し，その結果を実数配列 YVAL とおなじような形式で格納すれば，$\overline{A}x$ が得られることは明らかであろう．一方，整数配列 SEND は，当該要素に格納されているデータとおなじアークに関する情報が，他にどの番号の要素に格納されているかを示している．例えば，整数配列 SEND の第 4 要素は 12 であるが，実数配列 XVAL の第 4 要素 x_4 とおなじデータが XVAL の第12 要素にも格納されていることを確かめることができる．さらにこのことから，実数配列 YVAL の第 4 要素 y_2 と第 12 要素 y_5 との和は，$\overline{A}^{\top}y$ の第 4 要素となっていることもわかる．したがって，実数配列 YVAL の各要素を整数配列 SEND の対応する要素が示す位置に転送し，その位置にある YVAL の要素との和をとれば，実数配列 XVAL と同様な形式で格納された $\overline{A}^{\top}y$ の計算結果を得ることができる．

グラフ $\mathcal{G}$ が 2 部グラフとは限らない一般のネットワークである場合は，接続行列 $\overline{A}$ の非零要素のうち，流出側の節点に対応する要素の値が -1 となっている．そこで，上に示した 4 つの配列のほかに長さ 20 の論理配列 NODE を用意し，各アークの流出側の節点に対応する要素には T，流入側の節点に対応する要素には F を格納する．そして，XVAL の部分配列の和または YVAL の要素の転送を行う前に，論理配列 NODE の対応する要素の値が T である要素の符号を反転しておけばよい．

6. 数値実験結果

ここでは, エイ・ティ・アール人間情報通信研究所が保有する並列計算機 CM-5 上で行った数値実験の結果について述べる. CM-5 は 2 のべき乗個の処理ノード (processing node) から構成される大規模並列計算機である. 各処理ノードは, 標準的な SPARC プロセッサと合計 4 個の浮動小数点演算用のベクトル・ユニットを搭載しており, そのピーク性能は 128 MFLOPS と評価されている [19] . また, 各処理ノードには 8 Mbyte のローカル・メモリが 4 個装備されており, 大規模なデータの取扱いが可能であるとともに, データ・ネットワークを介して各処理ノード間で 1 対 1 のデータ通信を行うことができる. したがって, 処理ノード全体に分散された大規模なデータに対して一斉に処理を行うデータ並列型のプログラムだけでなく, メッセージ受渡し (message passing) などの機能を提供するライブラリーを利用することによって, 各処理ノードがそれぞれ独立に処理を行うコントロール並列型のプログラムを実行することもできる. しかも, 後者の並列計算モデルにおいては, 各処理ノードごとにデータ並列型のプログラムを実行することも可能であり, 前章で述べた並列計算機上でのアルゴリズム実行の各方策を容易に実現することができる.

数値実験には 32 個の処理ノードをもつ CM-5 を用い, 一つの処理ノードをマスター・ノードに, そして, 残りの処理ノード群をワーカー・ノードに割当てた. したがって, 最大 31 品種までの多品種流問題を解くことができる. 各処理ノードにおいて実行するプログラムのコーディングには, データ並列型のプログラミング言語である CM Fortran [18] を用いた. CM Fortran は, 配列要素同士の演算を並列的に実行したり, 配列要素の移動や部分和の計算を高速に実行できるように開発された FORTRAN 系統の言語であり, 各処理ノードにおいて実行しなければならない大規模で疎な連立方程式に対する共役勾配法の手続きを, 効率的に処理できるものと期待される. また, ソフトウェア・ライブラリー CMMD [20] を用いて各処理ノード間でのメッセージ受渡しや同期を取るための処理を行い, コントロール並列型の計算モデルを実現した.

数値実験を通じて, アルゴリズム PDIP のパラメータは

$$\varepsilon_\mu = 10^{-6}, \quad M_{tol} = 300, \quad M_\mu = 500, \qquad \tau = 0.995$$

とし, μ の初期値として $\mu^{(1)} = 1000$ を選んだ. 一方, 手続き PCG_ℓ における許容誤差は,

$$\varepsilon_\ell = 10^{-6}, \qquad \ell = 1, \cdots, L,$$

とした. また, 式 (4.8) によって手続き PCG の許容誤差を与える ε_0 の値は

$$\varepsilon_0 = 10^{-6}$$

とし, アルゴリズム PDIP の 5 反復目以降では許容誤差 ε が常に 10^{-6} となるようにしている. さらに, 手続き PCG_ℓ および PCG の初期探索点は, いずれもアルゴリズム PDIP の 1 反復目においては 0, 以後はアルゴリズム PDIP の前反復で当該手続きを適用した際に得られた解を用いている. また, 線形代数的な計算の実行にあたっては, 5 章で述べた 2 部グラフに特化したデータ構造と手法を適用した. なお, 実験はすべて倍精度計算により行った.

テスト問題としては, 2 部グラフ $\mathcal{G}$ 上で定義されるランダムに生成された多品種流問題 (3.1) を用いた. ただし, 行列 G_ℓ は正定値対角行列とし, $\overline{A}$ および b_ℓ は式 (5.2) のように定めた. また, 大規模なテスト問題を効率的に生成するために, 文献 [7, 8] に述べられているのと同様な方法を用いて, テスト問題そのものも CM-5 上で並列的に生成している. すなわち,

まずグラフ $\mathcal{G}$ が連結となるように $2\max\{|\mathcal{V}_1|, |\mathcal{V}_2|\}$ 本のアークを規則的に張った後, アーク数が $|\mathcal{E}|$ となるまで, ランダムにアークを生成する. 一方, アルゴリズム PDIP のステップ 1 で必要となる初期実行可能内点を得るための特別な手続きが要らないように, 次のような方法で問題の生成を行った. まず, 各品種の平均的なフロー x_ℓ とスラック変数 s_ℓ, t およびこれらに対応する双対変数 z_ℓ と w_ℓ, v を, いずれも区間 $[1, 100]$ の一様乱数で生成する. また, フロー保存の等式制約条件に対応する双対変数 y_ℓ を, 区間 $[-50, 50]$ の一様乱数で生成する. そして, これらの値をそのまま初期実行可能内点 $(x^{(1)}, t^{(1)}, s^{(1)}, y^{(1)}, z^{(1)}, v^{(1)}, w^{(1)})$ とする. さらに, 各品種のコスト関数に含まれる 2 次の係数行列 G_ℓ の対角要素を, 区間 $[1, 10]$ の一様乱数で生成する. このとき, 方程式系 (2.2) の最初の 4 式に対応する関係式

$$
\begin{aligned}
&\overline{A}^\top y_\ell - v - w_\ell + z_\ell - G_\ell x_\ell = h_\ell, && \ell = 1, \cdots, L,\\
&\overline{A}\, x_\ell = b_\ell, && \ell = 1, \cdots, L,\\
&\sum_{\ell=1}^{L} x_\ell + t = d,\\
&x_\ell + s_\ell = u_\ell, && \ell = 1, \cdots, L,
\end{aligned}
$$

を用いれば, 問題 (3.1) の係数ベクトル h_ℓ, b_ℓ, u_ℓ, $\ell = 1, \cdots, L$, と d を求めることができる. 様々な大きさの問題に対するアルゴリズムの性能を確かめるために, 各ノードの平均次数が 16 および 8 であるような問題について, グラフのサイズと品種数を変化させながら数値実験を行った. 具体的には, 供給節点と需要節点の数は同一とし, それぞれ 512 から 4096 まで, アーク数は 8192 から 32768 まで, また, 品種数は 4 から 28 まで変化させた. したがって, 問題 (3.1) の変数および (変数の上下限制約を除く) 制約条件の数にして, それぞれ最大 917504 $(= 32768 \times 28)$ および 262144 $(= 4096 \times 2 \times 28 + 32768)$ の規模をもつ問題まで解いたことになる.

数値実験の結果を表 1 に掲げる. 表の各欄の数値は, 各サイズの問題に対して乱数の種類を変えて生成した 5 つの問題例に対する結果を平均したものである. この表をもとに, まず 2 部グラフ $\mathcal{G}$ のサイズを固定した場合の品種数と計算時間の関係について考える. 一例として, 供給節点と需要節点の数がそれぞれ 4096, アーク数が 32768 の場合を図 2 に示す. 多品種流問題 (3.1) において, 変数の総数およびアークの総流量上限制約を除く制約条件の総数は, いずれも品種数 L に比例して増加する. 一般に逐次計算機を用いて最適化問題を解く場合には, 変数および制約条件の数で規定される問題サイズが大きくなるにつれて, 比例関係以上の割合で計算時間が増加する. したがって, ブロック構造を利用せずに問題 (3.1) を直接逐次計算機で解こうとすると, 品種数の増加に伴って急激に計算時間が増加すると予想される. ところが, 図 2 からもわかるように, 提案したアルゴリズムの並列計算機 CM-5 による実行時間の増加は品種数に対してほぼ直線的で, その増加率も比較的低く抑えられている. よって, ここで提案した並列型の主双対内点法は, 各品種ごとに一つづつの処理装置を割り当てることができるという条件のもとで, 多品種流問題 (3.1) を効率的に解くことができ, その傾向は品種数が多くなるほど顕著になるものと考えられる. 内点法の反復回数は, グラフのサイズと品種数の増加にともなって徐々に増大しているが, いずれも 30 回以内に抑えられており, 良好な振舞いを示していることがわかる. また, Δv に関する連立方程式 (3.9) の反復回数は, その変数の数であるグラフ $\mathcal{G}$ のアーク数や品種数が増えるにしたがって若干増加している. さらに, 次数 8 の問題の方が次数 16 の問題に比べて多少反復回数が多くなる傾向がみられる. しかし, いずれの場合も 8 回から 25 回程度の反復で解けており, 前処理

表 1: 2 部グラフ上の多品種流問題に対する数値実験結果 †

2 部グラフのサイズ ($\|\mathcal{V}_1\|, \|\mathcal{V}_2\|, \|\mathcal{E}\|$)	品種数 L	反復回数 内点法 (共役勾配法 ††)	計算時間 (秒)
(512, 512, 8192)	4	19.2 (141.0)	483.86
	8	20.8 (189.2)	631.82
	12	22.2 (233.0)	774.62
	16	22.8 (269.4)	895.06
	20	23.2 (306.2)	1022.67
	24	23.4 (328.8)	1112.54
	28	23.2 (344.6)	1189.32
(1024, 1024, 16384)	4	20.0 (157.0)	1223.56
	8	22.0 (211.4)	1614.08
	12	23.6 (262.6)	1988.46
	16	24.2 (300.6)	2286.73
	20	25.4 (341.4)	2612.25
	24	26.0 (372.4)	2885.42
	28	26.4 (403.2)	3156.09
(2048, 2048, 32768)	4	21.8 (173.6)	3328.04
	8	25.2 (247.8)	4609.58
	12	26.0 (301.0)	5505.11
	16	26.6 (344.0)	6261.19
	20	27.4 (386.6)	7058.96
	24	28.6 (437.6)	8013.86
	28	28.0 (455.0)	8406.38
(1024, 1024, 8192)	4	18.0 (183.8)	790.80
	8	20.0 (256.4)	1080.97
	12	21.2 (321.4)	1341.68
	16	22.4 (377.0)	1580.88
	20	23.0 (425.4)	1801.50
	24	23.6 (480.6)	2059.37
	28	23.6 (523.2)	2255.40
(2048, 2048, 16384)	4	19.2 (203.0)	2081.05
	8	21.4 (279.4)	2811.49
	12	23.2 (357.8)	3558.55
	16	24.0 (415.4)	4126.40
	20	25.2 (477.6)	4751.79
	24	25.6 (527.6)	5291.17
	28	26.4 (582.0)	5893.77
(4096, 4096, 32768)	4	20.8 (231.6)	5888.91
	8	24.2 (326.8)	8197.79
	12	25.4 (402.4)	9930.85
	16	26.0 (472.0)	11552.58
	20	27.2 (549.0)	13450.14
	24	27.6 (610.4)	14996.18
	28	28.8 (706.4)	17336.92

† 表の各欄の数値は, 各サイズのテスト問題の 5 つの問題例に対する結果の平均である.
†† 共役勾配法の反復回数は, Δv に関する連立方程式を解くための反復回数の累積である.

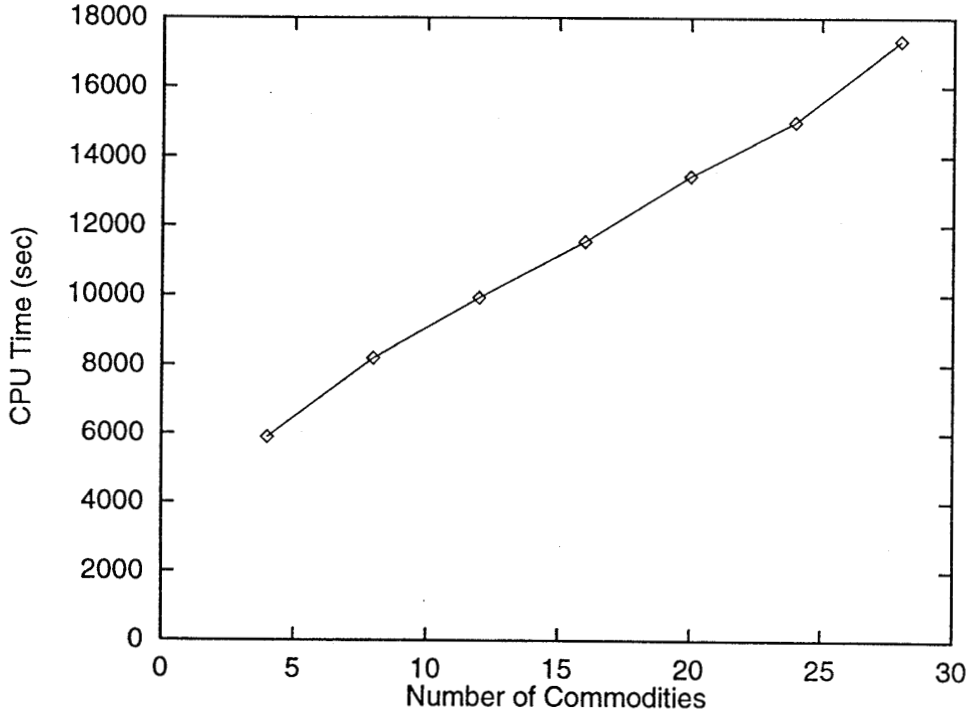

図 2: 品種数と計算時間：供給節点と需要節点の数が 4096, アーク数が 32768 の場合

がきわめて有効に働いているものと考えられる．実際, 4 章で述べた連立方程式 (3.9) に対する近似的な前処理を行わない場合には, 共役勾配法の反復回数および全体の計算時間がいずれも約 2 倍となった．なお, 3 章の最後でも指摘したように, 探索方向 Δv を連立方程式 (3.8) により計算すると, 問題(3.1) の解において狭義の相補条件が成り立っていない場合には数値的悪条件に陥る可能性がある．実際, (3.8) に対して手続き PCG と同様な前処理つき共役勾配法を適用した場合には, 内点法のアルゴリズム PDIP の探索点が (3.1) の解に近づくにつれて, 共役勾配法の反復回数は急激に増加し, 最終的に 100 倍以上の計算時間を要する例が多く観測された．例えば, 供給節点および需要節点の数を 512, アーク数を 8192, 品種数を 4 として生成したあるテスト問題において, 連立方程式 (3.9) に対する手続き PCG の反復回数は内点法の全反復を通じて 10 回以下にとどまり, 計算時間は 472 秒であった．これに対して, 連立方程式 (3.8) を前処理つき共役勾配法により解いた場合は, その反復回数が内点法の第 15 反復において 133 回, 第 17 反復以降は 8200 回余りに達し, 56637 秒もの計算時間を要した．この例からも, 連立方程式 (3.9) において, 対角行列 V^{-1} による式のスケーリングが有効に動作していることがわかる．

なお, 連立方程式 (3.7) および (3.9) を係数行列の分解に基づく直接法で解くためには, 各ワーカー・ノードおよびマスター・ノードにおいて非常に多くの記憶領域が必要となる．そのため, 各処理ノードが保有するローカル・メモリの容量が 32 Mbyte である CM-5 において, おなじ問題を直接法を用いて解くことは不可能であった．また, 本アルゴリズムを逐次計算機上で実行した場合との比較も行わなかったので, 今回の実験で得られた計算時間そのものについて評価することはできない．文献 [8] には, 双対理論と近接点法の考え方を組合せることによって得られる並列アルゴリズムをデータ並列型の計算モデルを用いて実現し, 単一品種の輸送問題を解いた数値実験結果が示されている．それによると, 例えば 4096 個の供

給節点および需要節点と 32768 本のアークをもつ問題が, 32 個の処理ノードから構成される CM-5 上で 5.68 秒で解けている．並列化効率が 100% であると仮定すると, 1 個の処理ノードでは約 182 秒かかる．また, 品種数が 28 の場合に計算時間も単純に 28 倍になると考えれば約 5090 秒で解けることになる．一方, 交互方向乗数法という並列アルゴリズムを用いると, 64 個の処理ノードをもつ CM-5 上でおなじ規模をもつ単一品種の輸送問題が 5 秒足らずで解けることも報告されている [7] ．しかし, これらの方法はいずれもそれ自身が高度なデータ並列型のアルゴリズムであり, その計算時間は CM-5 の超並列計算機としての構造的特徴を活かした実行方法によって得られたものである．これに対して, 本論文で提案したアルゴリズムはコントロール並列型の処理を骨格としており, CM-5 の処理機構における制約により, マスター・ノードが複数のワーカー・ノードとの間でやり取りするメッセージを逐次的に処理せざるを得ない．したがって, 処理ノード間でのメッセージ受渡しの方法を改善することができれば, 本数値実験で要した17337 秒という処理時間は大幅に短縮され, 他の並列アルゴリズムと同等な性能を発揮できるものと期待される．

7. おわりに

本論文では, 分離可能な 2 次のコスト関数をもつ多品種流問題を解くための並列型の主双対内点法を示した．アルゴリズムは, 品種に関して独立な連立方程式を解くことに帰着し, コントロール並列型の計算モデルを用いることによって並列計算機上に実現できる．また, 問題のグラフ構造を利用すると, 各連立方程式は, データ並列型のプログラミング環境のもとで共役勾配法を用いて効率的に解くことができる．さらに, 内点法の反復において探索点が非負領域の境界に近づいた場合に陥る数値的悪条件や桁落ちにともなう計算誤差の発生を解消するための方策と, 複雑な係数行列をもつ連立方程式に対する共役勾配法の前処理行列を近似的に計算する仕組みを組込むことによって, 大規模で疎な問題を実用的な反復回数と計算時間で解けることが確かめられた．

数値実験では, 探索点が実行可能内点を逸脱しない範囲でステップサイズをできるだけ大きくとるという方針を採用した．しかし, 主双対内点法や対数型の障壁関数に基づく内点法アルゴリズムに対しては, 現在も様々な直線探索の方法が検討されており [17], その実現方法を工夫すれば, 並列型主双対内点法においてもその反復回数をさらに削減できる可能性がある．また, 相補条件の緩和度を示すパラメータ μ の更新についても, 問題のサイズや構造に応じた適切な方法を選択することによって内点法の収束を速めることができるものと考えられる．このような実用上の工夫とその検証が今後の課題である．

謝辞　多くの有意義なコメントを頂いたレフェリーの方々に感謝致します．なお, 本研究は一部文部省科学研究費補助金 (一般研究 (C) 06650443) によるものである．

参考文献

[1] Adler, I., Karmarkar, N., Resende, M.G.C. and Veiga, G.: Data Structures and Programming Techniques for the Implementation of Karmarkar's Algorithm. *ORSA Journal on Computing*, Vol. 1 (1989), 84–106.

[2] Breitfeld, M.G. and Shanno, D.F.: Computational Experience with Penalty-Barrier Methods for Nonlinear Programming. Rutcor Research Report 17-93. Rutgers Center for Operations Research, Rutgers University, New Brunswick, N.J., 1994.

[3] Carpenter, T.J., Lustig, I.J., Mulvey, J.M. and Shanno, D.F.: Separable Quadratic Programming via a Primal-Dual Interior Point Method and Its Use in a Sequential Procedure. *ORSA Journal on Computing*, Vol. 5 (1993), 182–191.

[4] Chen, R.J. and Meyer, R.R.: Parallel Optimization for Traffic Assignment. *Mathematical Programming*, Vol. 42 (1988), 327–345.

[5] Choi, I.C. and Goldfarb, D.: Exploiting Special Structure in a Primal-Dual Path-Following Algorithm. *Mathematical Programming*, Vol. 58 (1993), 33–52.

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

[7] Eckstein, J. and Fukushima, M.: Some Reformulations and Applications of the Alternating Direction Method of Multipliers. *Large Scale Optimization: State of the Art* (eds. W.W. Hager, D.W. Hearn and P.M. Pardalos). Kluwer Academic Publishers, Dordrecht, The Netherlands, 1994, 115–134.

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

[9] Hurd, J.K. and Murphy, F.H.: Exploiting Special Structure in Primal Dual Interior Point Methods. *ORSA Journal on Computing*, Vol. 4 (1992), 38–44.

[10] Kojima, M., Megiddo, N. and Mizuno, S.: A Primal-Dual Infeasible-Interior-Point Algorithm for Linear Programming. *Mathematical Programming*, Vol. 61 (1993), 263–280.

[11] Kojima, M., Mizuno, S. and Yoshise, A.: A Polynomial-Time Algorithm for a Class of Linear Complementarity Problems. *Mathematical Programming*, Vol. 44 (1989), 1–26.

[12] Lustig, I.J. and Li, G.: An Implementation of a Parallel Primal-Dual Interior Point Method for Block-Structured Linear Programs. *Computational Optimization and Applications*, Vol. 1 (1992), 141–161.

[13] Mehrotra, S.: Implementations of Affine Scaling Methods: Approximate Solutions of Systems of Linear Equations Using Preconditioned Conjugate Gradient Methods. *ORSA Journal on Computing*, Vol. 4 (1992), 103–118.

[14] Mizuno, S.: Polynomiality of Infeasible Interior Point Algorithm for Linear Programming. *Mathematical Programming*, Vol. 67 (1994), 109–119.

[15] Monteiro, R.D.C. and Adler, I.: Interior Path Following Primal-Dual Algorithms. Part I: Linear Programming. *Mathematical Programming*, Vol. 44 (1989), 27–41.

[16] Monteiro, R.D.C. and Adler, I.: Interior Path Following Primal-Dual Algorithms. Part II: Convex Quadratic Programming. *Mathematical Programming*, Vol. 44 (1989), 43–66.

[17] Murray, W. and Wright, M.H.: Line Search Procedures for the Logarithmic Barrier Function. *SIAM Journal on Optimization*, Vol. 4 (1989), 229–246.

[18] Thinking Machines Corporation: *CM Fortran Programming Guide, Version 2.1.* Cambridge, Massachusetts, 1994.

[19] Thinking Machines Corporation: *CM-5 CM Fortran Performance Guide, Version 2.1.* Cambridge, Massachusetts, 1994.

[20] Thinking Machines Corporation: *CMMD Reference Manual, Version 3.0.* Cambridge, Massachusetts, 1993.

[21] 戸川 隼人：共役勾配法. 教育出版, 1977.
[22] Wright, S.J.: An Infeasible-Interior-Point Algorithm for Linear Complementarity Problems. *Mathematical Programming*, Vol. 67 (1994), 29–51.
[23] Yamashita, H.: A Globally Convergent Primal-Dual Interior Point Method for Constrained Optimization. *Optimization: Modeling and Algorithms – 3.* The Institute of Statistical Mathematics, Tokyo, 1993, 272–297.
[24] Yamashita, H. and Yabe, H.: Superlinear and Quadratic Convergence of Primal-Dual Interior Point Methods for Constrained Optimization. *Optimization: Modeling and Algorithms – 4.* The Institute of Statistical Mathematics, Tokyo, 1994, 388–416.
[25] Zenios, S.A.: On the Fine-Grain Decomposition of Multicommodity Transportation Problems. *SIAM Journal on Optimization*, Vol. 1 (1991), 643–669.
[26] Zenios, S.A.: Data Parallel Computing for Network-Structured Optimization Problems. *Computational Optimization and Applications*, Vol. 3 (1994), 199–242.
[27] Zenios, S.A. and Censor, Y.: Massively Parallel Row-Action Algorithms for Some Nonlinear Transportation Problems. *SIAM Journal on Optimization*, Vol. 1 (1991), 373–400.

山川 栄樹：
高松大学 経営学部 産業経営学科,
〒761-01 香川県高松市春日町960番地.
E-mail: yamakawa@takamatsu-u.ac.jp

ABSTRACT

A PARALLEL PRIMAL-DUAL INTERIOR POINT METHOD FOR MULTICOMMODITY FLOW PROBLEMS WITH QUADRATIC COSTS

Eiki Yamakawa — *Takamatsu University*
Yasuhiro Matsubara — *West Japan Railway Co.*
Masao Fukushima — *Kyoto University*

We present a parallel implementation of a primal-dual interior point method for multicommodity flow problems with separable quadratic costs. Exploiting the block structure of the problem, the system of linear equations to be solved at each iteration is decomposed into independent systems of linear equations associated with respective commodities and another system of linear equations corresponding to the coupling constraint. We solve them in parallel by applying the conjugate gradient (CG) method with an appropriate preconditioner, which significantly improves the speed of convergence of the CG method particularly when the generated feasible interior points approach the boundary of the non-negative orthant. Some devices are also introduced to reduce numerical errors caused by cancellation in executing the CG iterations. We implement the algorithm on the Connection Machine Model CM-5 under the control parallel programming environment. The computational results indicate that the proposed approach is very efficient for large-scale multicommodity flow problems.