容量制約をもつネットワーク設計の大規模問題に対する近似解法

Approximate Algorithms for Large Scale Instances of Capacitated Network Design Problem

片山 直登 流通経済大学 流通情報学部

平成 29 年 9 月 4 日

1 はじめに

アークとノードからなるネットワークと，ネットワーク上を流れる多品種の需要量が与えられたときに，アークに関するデザイン費用と品種に関するフロー費用の合計が最小化となるアークを選択し，各品種のフローの経路を求める問題をネットワーク設計問題とよぶ．アークに単位時間当たりの処理量である容量をもつ問題を容量制約をもつネットワーク設計問題 (capacitated network design problem,CND) とよぶ．

今日まで，CND に対する数多くの研究がなされており，ベンチマーク問題である C 問題および R 問題 (Crainic et al. 2001) がアルゴリズムの評価に用いられている．C 問題は，ノード数 20 から 30，アーク数 230 から 700，品種数 40 から 400 の問題と，ノード数 100，アーク数 400，品種数 10 から 30 の問題で構成されている．また，R 問題は，ノード数 10，アーク数 35 から 83，品種数 10 から 50 の問題と，ノード数 20，アーク数 120 から 315，品種数 40 から 200 の問題で構成されている．今日では，C 問題および R 問題は短時間で最適解または精度の高い近似解を求めることができるようになっている．

そのため最近では，CND に対するベンチマーク問題として，より大規模な問題である GT 問題が用いられるようになっている．GT 問題は，ノード数 500，アーク数 2000 から 3000，品種数 50 から 200 の問題で構成されており，(F,T) と (F,L) の 2 種類の問題群がある．(F,T) はアークにおけるデザイン費用がフロー費用に比べて相対的に高く，アーク容量がフロー量に比べて相対的に小さな問題である．また，(F,L) はアークにおけるデザイン費用がフロー費用に比べて相対的に高く，アーク容量がフロー量に比べて相対的に大きな問題である．C・R 問題と GT 問題を比較すると，最大規模の問題では，アーク数である 0-1 であるデザイン変数は 700 個から 3000 個に，連続変数であるアークフロー変数は 28 万個から 50 万個に増加している．また，アーク数の増加に加えてノード数も 100 個から 500 個に増加し，品種

の取りうるパス数は大幅に増加している．そのため，パスフローを用いた定式化を用いた場合には，パスフロー変数は大幅に増加することになる．

GT 問題に対して，Hewitt et al. (2010) は限定された広範囲の近傍を探索する厳密解法とヒューリスティクスを組合せた解法である IP 探索法を適用している．また，Munguí et al. (2017) はマルチ CPU を活用した並列局所探索法を適用し，IP 探索法により得られた解と比べて大幅な改善に成功している．

本研究では，CND に対する容量スケーリング法と MIP ソルバーによる近傍探索法を組み合わせた MIP 近傍探索法を GT 問題に適用可能なように改善し，提案した解法が GT 問題に対して有用な近似解を算出できることを示す.

2 CND の定式化

ノード集合 N，アーク候補集合 A，品種集合 K で定義されるネットワークを考える．変数として，アーク (i,j) を選択するか否かを表すデザイン変数 y_{ij}，アーク (i,j) 上の品種 k のフロー量を表すアークフロー変数 x_{ij}^k を用いる．費用として，アークを選択するときに発生するデザイン費用 f_{ij}，アーク (i,j) 上の品種 k の単位当たりのフロー費用 c_{ij}^k が発生する．これらの費用の和を最小化し，その最小値を Φ とおく．アーク (i,j) の容量 C_{ij}，品種 k の需要量 d^k が与えられ，品種 k は始点 O^k を出発し，終点 D^k に到着する．また，アーク A からなるネットワーク上でノード n から出るアーク集合 $N_n^+(A)$，アーク A からなるネットワーク上でノード n に入るアーク集合 $N_n^-(A)$ が与えられる．

このとき，アーク集合 A に対するアークフロー変数を用いた CND の定式化 $CNDA(A)$ は，次のように表される．

$CNDA(A)$

$$\Phi = \min \sum_{(i,j)\in A}\sum_{k\in K} c_{ij}^k x_{ij}^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{1}$$

subject to

$$\sum_{i\in N_n^+(A)} x_{in}^k - \sum_{j\in N_n^-(A)} x_{nj}^k = \begin{cases} -d^k & if\ n = O^k \\ d^k & if\ n = D^k \\ 0 & otherwise \end{cases} \qquad \forall n \in N,\ k \in K \tag{2}$$

$$\sum_{k\in K} x_{ij}^k \le C_{ij} y_{ij} \quad \forall\ (i,j) \in A \tag{3}$$

$$x_{ij}^k \le d^k y_{ij} \quad \forall\ k \in K,\ (i,j) \in A \tag{4}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A \tag{5}$$

$$x_{ij}^k \geq 0 \quad \forall (i,j) \in A, k \in K \tag{6}$$

(1) 式は，フロー費用とデザイン費用の和であり，これを最小化することを表す．(2) 式は，ノード n における流入量と流出量の差が，ノード n が品種 k の始点 O^k であれば $-d^k$，終点 D^k であれば d^k，その他のノードであれば 0 となることを表すフロー保存式である．(3) 式の左辺はアーク (i,j) 上のフロー量の合計であり，右辺はアーク (i,j) が選択されるときに C_{ij}，選択されないとき 0 となる容量制約式である．(4) 式の左辺はアーク (i,j) 上の品種 k のパスフロー量の合計であり，右辺はアーク (i,j) が選択されるときに d^k，選択されないとき 0 となる強制制約式である．(5) 式はデザイン変数の 0-1 条件，(6) 式はアークフローの非負制約である．

品種 k の取り得るパス集合を P^k，全体のパス集合を P とし，品種 k のパス p のフロー量であるパスフロー変数を x_p^k とし，パス p がアーク (i,j) を含むとき 1，そうでないとき 0 である定数を δ_{ij}^p とする．このとき，アーク集合 A，パス集合 P，アーク容量 C に対して，パスフロー変数を用いた CND の定式化 $CNDP(A,P,C)$ は，次のように表される．

$CNDP(A,P,C)$

$$\min \quad \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k \sum_{p\in P^k} \delta_{ij}^p x_p^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{7}$$

subject to

$$\sum_{p\in P^k} x_p^k = d^k \quad \forall k \in K \tag{8}$$

$$\sum_{k\in K} \sum_{p\in P^k} \delta_{ij}^p x_p^k \leq C_{ij} y_{ij} \quad \forall (i,j) \in A \tag{9}$$

$$\sum_{p\in P^k} \delta_{ij}^p x_p^k \leq d^k y_{ij} \quad \forall (i,j) \in A, k \in K \tag{10}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A \tag{11}$$

$$x_p^k \geq 0 \quad \forall p \in P^k, k \in K \tag{12}$$

(7) 式は，フロー費用とデザイン費用の和であり，これを最小化することを表す．(8) 式は，品種 k のパスフロー量の和は需要量になることを表す需要保存式である．(9) 式の左辺はアーク (i,j) 上のフロー量の合計であり，右辺はアーク (i,j) が選択されるときに C_{ij}，選択されないとき 0 となる容量制約式である．(10) 式の左辺はアーク (i,j) 上の品種 k のパスフロー量の合計であり，右辺はアーク (i,j) が選択されるときに d^k，選択されないとき 0 となる強制制約式である．(11) 式はデザイン変数の 0-1 条件，(12) 式はパスフローの非負制約である．

$CNDP(A,P,C)$ において，デザイン変数の 0-1 条件を線形緩和し，(11) 式を次式で置き換えた問題を $CNDPL(A,P,C)$ とする．

$$0 \leq y_{ij} \leq 1 \quad \forall (i,j) \in A \tag{13}$$

3 近似解法

本研究で用いる解法は，容量スケーリング法と近傍探索を組み合わせた解法 (片山直登 2017) である．この解法は，容量スケーリング法により初期探索解を算出し，MIP ソルバー用いた近傍探索を行う MIP 近傍探索法である．直接，MIP 近傍探索法を GT 問題に適用すると膨大な計算時間が必要となる．そこで，大規模な問題に適用可能なように，MIP 近傍探索法に改良を加える．

3.1 列・行生成法

パスフローによる定式化の線形緩和問題 $CNDPL(A,P,C)$ を最適に解くことを考える．パスフロー変数は膨大な数となるため，あらかじめすべてを列挙しておくのは困難である．そこで，適当なパスの部分集合 $\bar{P}$ から始め，逐次，基底に入るであろうパスフロー変数を生成する．そのためには価格付け問題を解き，被約費用が負であるパスフロー変数を生成する．生成した変数を問題に加え，対応するパスを $\bar{P}$ に加えて $\bar{P}$ を更新し，再度 $CNDPL(A,\bar{P},C)$ を解き直す．この操作を被約費用が負である変数がなくなるまで繰り返す．被約費用が負であるパスフロー変数が存在しなければ，$CNDPL(A,P,C)$ の最適解が得られたことになる．

(8) 式に対する双対変数を π，(9)，(10) 式に対する非負の双対変数を u，w とする．これらの値は，$CNDPL(A,\bar{P},C)$ を最適に解くことにより求めることができる．

パスフロー変数 x_p に関する被約費用は，

$$\sum_{(i,j)\in A} \delta_{ij}^{p}(c_{ij}^{k}+u_{ij}+w_{ij}^{k})-\pi^{k} \quad \forall p\in P^{k}, k\in K \tag{14}$$

である．$\sum_{(i,j)\in A}\delta_{ij}^{p}(c_{ij}^{k}+u_{ij}+w_{ij}^{k})$ は，アーク (i,j) の長さを $c_{ij}^{k}+u_{ij}+w_{ij}^{k}$ としたとき，パス p の長さとなる．また，π^{k} は現在の品種 k のパス集合 $\bar{P}^{k}$ に含まれるパス上の始点・終点間の最短距離である．被約費用は「パス p の長さ $-$ 現在の最短距離」であるので，被約費用が負である変数を見つけることは現在の最短距離よりも短いパスを見つけることになる．

π^{k} は定数項として扱えるので，負のパスフロー変数を見つけるには，品種 k に対して，(14) 式の第一項を最小化するパス p を見つければ良い．したがって，品種 k に関する価格付け問題は，アーク (i,j) の長さを $c_{ij}^{k}+u_{ij}+w_{ij}^{k}$ とした最短経路問題に帰着される．

この最短経路問題において，始点・終点間の品種 k の最短距離を μ とする．$\mu-\pi^{k}<0$ であれば，品種 k における被約費用が負であるパスフロー変数が見つかったことになり，p が生成すべきパス，パスに対応するパスフロー変数 x_p^k が生成すべき変数となる．また，すべての品種 k について $\mu-\pi^{k}\geq 0$ であれば，被約費用が負である変数が存在しないため，$CNDPL(A,P,C)$ が最適に解けたことになる．

ここで，$CNDPL(A,\bar{P},C)$ には，左辺に変数が含まれない強制制約式 (10) は含まれないものとする．生成したパス p 上のアークに関して，品種 k に関する強制制約式がない場合は，品種 k の需要に関する強制制約式を問題に追加する．

3.2 容量スケーリングと列・行生成法

初期のアーク集合に対して容量スケーリング法を適用し，事前に対象となるアークを絞ることにする．容量スケーリング法は，線形緩和問題を解き，そのデザイン変数解の値にしたがってアーク容量を変化させ，0 また 1 のデザイン変数解を導出するものである．容量スケーリングでは，少ない繰り返し回数で多くのデザイン変数が 0 に収束することが知られている (Katayama et al. 2009)．そこで，アーク集合 A に対して容量スケーリング法を適用し，0 に収束しないデザイン変数のみを選定し，これらのアークのみを対象とし，その後の近傍探索を実施する．この処理により，少ない計算量で多くのアークを除外することができ，効率的に問題規模を縮小することが可能となる．もちろん，0 に収束したデザイン変数が最適解において 1 である可能があるため，限定されたアーク集合をもとに求めた解は最適解であることは保証できない．すべての変数が 0 または 1 に収束するためには多くの繰り返し回数が必要なため，0 に収束しないデザイン変数が一定数以下となる，または設定した繰り返し回数の上限に達した場合に容量スケーリングを終了する．

$CNDPL(A,P,C)$ において，l 回目の繰り返しにおけるアーク容量を C^l とし，(13) 式の上限である 1 を C/C^l に置き換え，限定されたパス集合 $\bar{P}$ を用いた線形緩和問題を $CNDPL(A,\bar{P},C^l)$ とする．$\bar{P}$ は，適当な初期集合から列生成で生成されたパスを加えた集合である．なお，少なくとも左辺のパスフロー変数が 1 つ以上生成されている強制制約式のみを考慮し，それら以外の強制制約式は含まないものとする．

$l-1$ 回目の繰り返しである $CNDPL(A,\bar{P},C^{l-1})$ のデザイン変数解を $\tilde{y}$ とすると，l 回目のアーク容量を次のように変更する．

$$C_{ij}^l := \lambda C_{ij}^{l-1}\tilde{y}_{ij} + (1-\lambda)C_{ij}^{l-1} \tag{15}$$

ここで，λ はスケーリングパラメータであり，0 から 1 の間の定数である．

容量スケーリング法により初期パス集合に生成されたパスを加えた集合を $\bar{P}$ とおき，0 に収束しないデザイン変数に対するアーク集合を $\bar{A}$ とする．このとき，アーク集合 $\bar{A}$ とパス集合 $\bar{P}$ で構成される問題は $CNDP(\bar{A},\bar{P},C)$ となる．$\bar{A}$ とパス集合 $\bar{P}$ は，それぞれ A と P の部分集合であるため，$CNDP(\bar{A},\bar{P},C)$ の最適解は $CNDP(A,P,C)$ の近似解，最適値は $CNDP(A,P,C)$ の上界値となる．$CNDP(\bar{A},\bar{P},C)$ は相対的に小さな問題となるため，MIP ソルバーで解き易い問題となる．得られた解 $\bar{y}$ は，次に示す MIP 近傍探索法の初期解に用いることができる．

3.3 MIP 近傍探索法

アーク集合を容量スケーリング法により得られたアーク集合 $\bar{A}$ に限定した問題 $CNDA(\bar{A})$ は $CNDA(A)$ と比較して小規模な問題とはなるが，容易に最適に解け

るとは限らない．そこで，定式化$CNDA(\bar{A})$において，近似解の近傍をMIPソルバー用いて探索するMIP近傍探索法を実施する．

このMIP近傍探索法では，次の2本の制約式を追加して，近傍を定義する．

$$\sum_{(i,j)\in A|\bar{y}_{ij}=1} y_{ij} \leq L-1 \tag{16}$$

$$\sum_{(i,j)\in A|\bar{y}_{ij}=1} y_{ij} \geq L-M \tag{17}$$

ここで，Lは現在の解において$\bar{y}_{ij}=1$である選択されたアーク数であり，Mは近傍の範囲である．(16)式は，現在の解で選択されたアークから，少なくとも1本のアークはネットワークから取り除くことを表す．(17)式は，現在の解で選択されたアークを高々M本のアークをネットワークから取り除くことを表す．

現在までの最良の上界値をUBとおく．このとき，$CNDA(\bar{A})$に次の式も追加する．

$$\Phi < UB \tag{18}$$

この式により，実行可能領域が制限され，現在までの最良値よりも良い上界値が存在しなければ$CNDA(\bar{A})$は実行不可能となる．また，探索済みの解の探索を除外することができる．

$CNDA(\bar{A})$に，(16)式，(17)式および(18)式を追加し，MIPソルバーを用いて，一定時間内で解を求める．M近傍において，現在の解より良い解が求められた場合には，現在の解を更新する．続いて，追加した3本の制約式を削除して，更新された解に対応した3本の制約式を追加し，近傍探索を繰り返す．このM近傍において，実行不可能，すなわち現在の解よりも良い解が無いと判断できた場合は，近傍探索を終了する．また，計算時間内に現在の解より良い解を算出することができない場合は，$M := \lfloor M/\alpha \rfloor (\alpha > 1)$として，$M$を減少させて探索範囲を縮小し，探索を繰り返す．なお，MIP近傍探索法の詳細は片山直登 (2017) を参照のこと．

3.4 大規模問題への適用

このMIP近傍探索法は，ベンチマーク問題であるC問題やR問題のような規模の問題に対しては，高速に適切な近似解を算出することができる．しかし，GT問題の規模の問題に直接適用すると大きな計算時間が必要となる．この原因は主に2つある．

1つは，容量スケーリングでは線形緩和解を繰り返し求めるが，大規模な問題では線形緩和解自体の最適解を求めることが容易ではないためである．容量スケーリングでは，繰り返しとともに，列生成と行生成によりパスフロー変数と強制制約式が生成され，その数が増加して問題の規模が大きくなる．一方，容量スケーリングでは緩和解をもとに，容量が現在のアークフロー量に近づくように調整す

る．このため，容量スケーリングの繰り返し回数とともに，非常にタイトな容量制約となり，大半の容量制約式と強制制約式のスラックが0となるような難しい問題となる．このことから，容量スケーリングの繰り返し回数とともに，大規模でよりタイトな線形緩和問題の最適解を求めることになり，必要な計算時間が大きく増加する．列・行生成法により，適時必要なパスフロー変数と強制制約式を生成して変数と制約式を制限してはいるが，それでも問題規模は大きくなる．加えて，列生成のために品種の始点・終点間の膨大な回数の最短経路問題を解く必要があり，大きな計算量が必要となる．以上のことから，GT問題のようなアーク数とノード数が多い問題においては，パスフロー変数が膨大なものとなることから，これらが付加されたタイトな問題をMIPソルバーで解くためには大きな計算時間が必要となる．

もう1つは，近傍探索をMIPソルバーに依存しているために，近傍探索における近傍を制限しても，大規模な混合整数計画問題を解く必要があるためである．このため，短時間で適切な近傍解を算出することには困難が伴う．そこで，混合整数計画問題を解くために，計算時間を十分にとる必要がある．

そこで，解の精度をなるべく落とさずに，計算時間を短縮する2つの方法を提案する．

(1) 列生成回数の制限

容量スケーリング法では，容量スケーリングのたびに，容量を変更した緩和問題を作成し，列生成法を用いて緩和問題に必要なパスフロー変数を生成し，緩和問題の最適解を求める．線形緩和問題を最適に解くためには，パスフロー変数が生成されなくなるまで列生成を繰り返す必要があるが，これには大きな計算時間を必要とする．

しかしながら，緩和問題の最適解ではなく近似解であっても，解の精度は落ちるであろうが，容量スケーリング法自体は機能する．そこで，列生成の回数の上限を設定し，容量スケーリングの繰り返しごとに列生成の回数を制限する．一定数以上のパスフロー変数は生成しないため，被約費用が負が存在することになるが，得られる解は線形緩和問題の近似解とはなる．しかし，容量スケーリングにかかる計算時間を大幅に減少できる可能性がある．なお，上限回数により生成されなかったパスは，次回の容量スケーリングにおいて生成される可能性がある．列生成回数を制限した列・行生成のアルゴリズムをAlgorithm1に示す．

(2) 変数・制約式の限定

容量スケーリングの繰り返しごとに，多くのデザイン変数の値が0に収束する．デザイン変数値0に収束した場合には，アーク容量も0に収束する．デザイン変数が0に収束した後，デザイン変数の単位当たりのデザイン費用は非常に大きくなり，デザイン変数が再び正の値をとるためには大きな費用が必要となるため，それ以降の繰り返しにおいて正となる可能性は極めて低い．そこで，容量スケーリングの繰り返しごとに，0に収束したデザイン変数を定式化から削除する．続いて，これらのデザイン変数に対応するアーク上を流れるパスフロー変数も不要となるため，これらを削除する．さらに，削除したアークに対応する容量制約式

と強制制約式も削除する．これにより，容量スケーリングの繰り返しごとに，問題が大幅に縮小され，容量スケーリングにかかる計算時間を減少させることができる．

0 に収束したデザイン変数と関係する制約式を定式化から削除することは，アーク集合 $\bar{A}$ から 0 に収束したアークを削除することと等価である．これらに限定されたアーク集合 $\bar{A}$ を対象とした MIP 近傍探索法を実施する．変数・制約式の限定を考慮したアルゴリズムを Algorithm2 に示す．

4 数値実験

本研究で提案した MIP 近傍探索法の有効性を検討するために，大規模なベンチマーク問題である GT 問題の 24 問 (Hewitt 2010 Online;accessed 2017-05-01) に対して，数値実験を行った．本研究として，容量スケーリング時に列生成を制限した MIP 近傍探索法，および容量スケーリング時に列生成を制限し，かつアークを削除した MIP 近傍探索法を比較した．

数値実験で使用した設定した機器等は以下の通りである．

- 使用 OS および言語：UBUNTU 17.04，C++
- 最適化ソルバー：Gurobi 7.02
- CPU AMD Ryzen7 1800X 3.6GHz 8Cores，RAM 32GByte
- 使用コア数：容量スケーリング 1 コア，近傍探索 8 コア

また，数値実験で使用した設定したパラメータは以下の通りである．

- スケーリングパラメータ λ：0.05〜0.50
- スケーリングにおける列生成回数：100(列生成制限時)，50(アーク削除時)
- スケーリング法の終了判定アーク数 ArcNum：500
- 近傍 M：5
- 近傍 M の変更基準 α：5
- 近傍探索における MIP ソルバー計算時間の上限 T：750 秒

結果を比較するために，IP 探索法 (Hewitt et al. 2010) と並列局所探索法 (Munguí et al. 2017) の結果を併記した．なお，近似解の誤差を算出するために下界値を使用した．下界値は，最適化ソルバー Gurobi を用いて 30 時間計算して得られた値と Munguí et al. (2017) が示した値の大きい方を採用した．

表 1 に GT 問題に対する上界値の平均誤差を示す．誤差 (Gaps) は「(各解法の上界値－下界値)/下界値」であり，平均誤差は全問の平均値である．

従来の研究では，IP 探索法は IPSE，並列局所探索法は PALO とする．MIPR は列生成を制限した MIP 近傍探索法，MIPD は列生成制限とアーク削除の MIP 近傍探索法である．

従来の研究では，IP 探索法は平均誤差 24.58%，並列局所探索法は平均誤差 16.13%である．一方，本研究の列生成制限した MIP 近傍探索法は平均誤差 14.25%，列生成制限とアーク削除の MIP 近傍探索法は平均誤差 14.37%であった．従来解法である並列局所探索法と比べると，列生成を制限した MIP 近傍探索法では 1.88%，列生成制限とアーク削除の MIP 近傍探索法は 1.76%，平均誤差が小さい解を算出することができている．

表 4 に GT 問題に対する個別の目的関数値を示す．LB は下界値，斜体文字は最良値である．また，表 5 に GT 問題に対する個別の誤差を示す．

24 問題の内，並列局所探索法が 3 問，列生成を制限した MIP 近傍探索法では 14 問，列生成制限とアーク削除の MIP 近傍探索法では 7 問において，誤差が最も小さな最良値を算出している．また，並列局所探索法と比較すると，列生成を制限した MIP 近傍探索法では 20 問，列生成制限とアーク削除の MIP 近傍探索法では 19 問で優れた解を算出している．

並列局所探索法が最良の上界値を算出している問題は，F/T 問題の品種数 50 または 100 と少ない問題である．容量スケーリング法では，ネットワークの規模に比べて品種数が少ない場合，初期の繰り返しにおいて，線形緩和問題においてフローが 0 となるアークが多くなり，これらのアークは 0 に収束する可能性が高くなる．そのため，これらのアークの多くが最適解に含まれる場合は，容量スケーリング法によって限定されたアーク集合と最適解を含むアーク集合との差が大きくなる．その結果，容量スケーリング法と組み合わせた解法の解の誤差が大きくなる場合が生じる傾向がある．一方，列生成を制限した MIP 近傍探索法よりも，列生成制限とアーク削除の MIP 近傍探索法の方が良い上界値を算出している場合もある．

表 5 に GT 問題に対する平均計算時間を示す．IP 探索法と並列局所探索法はメタヒューリスティクスのため，3600 秒で計算を打ち切っている．なお，別途，IP 探索法では最良解の探索までの時間も示しており，大半の問題は 700 秒以下である．使用しているコンピュータは以下の通りである．

- IP 探索法：Intel Xeon×8CPUs，2.66GHz，RAM 8GByte
- 並列局所探索法：Intel Xeon X5650-6Cores×2CPUs×8Nodes，2.66 GHz，RAM 24 GB

なお，使用しているコンピュータが異なっているため，計算時間を直接比較することはできない．ただし，IP 探索法では 8 個の CPU をもつコンピュータを使用し，並列局所探索法では 2 個の CPU をもつコンピュータを 8 台連結した 96 コアをもつ高度な計算能力を備えたクラスタコンピュータを使用している．

列生成を制限した MIP 近傍探索法では平均 15384 秒となり，約 4.3 時間を要しており，列生成を制限して計算時間の短縮を図ってはいるが，それでも大きな計算

時間を要している．一方，列生成制限とアーク削除のMIP近傍探索法では平均9152秒となり，約2.6時間を要している．計算時間の短縮には成功しているが，それでも大きな計算時間を要しており，並列局所探索法の3600秒と比較して，約2.6倍となっている．使用したコンピュータは8コアのデスクトップコンピュータであり，96コアをもつクラスタコンピュータに比べると大きく計算能力は劣ると考えられるため，実質的には提案した解法は計算時間の面からも優れているものと考えられる．

5　おわりに

本研究では，大規模問題に適用可能とするために，CNDに対する容量スケーリング法とMIPソルバーによる局所探索法を組み合わせた近似解法に改良を加えた容量スケーリング後に列生成を制限したMIP近傍探索法，および容量スケーリング時に列生成を制限し，かつアークを削除したMIP近傍探索法を提案した．従来の解法であるIP探索法と並列局所探索法と比較するために，GT問題を用いた数値実験を行い，提案した解法が大規模な問題に対して有用な近似解を算出できることを示した．大半のGT問題において，従来の2つの研究による最良解よりも良い近似解を求めることができた．計算時間の短縮化を行ってはいるが，計算時間は従来の解法と比べて大きくなった．今後は，各種パラメータのチューニングに加え，アルゴリズムの改良によりさらなる計算時間の短縮化を図ることが必要である．

本研究は科学研究費基盤研究C(課題番号17K01268)による成果の一部である．

参考文献

T. G. Crainic, A. Frangioni, and B. Gendron. Bundle-based relaxation methods for multicommodity capacitated fixed charge network design problems. *Discrete Applied Mathematics*, 112:73–99, 2001.

M. Hewitt. GT instances. `https://www.researchgate.net/publication/304825234`, 2010 Online;accessed 2017-05-01.

M. Hewitt, G. L. Nemhauser, and M. Savelsbergh. Combining exact and heuristics approaches for the capacitated fixed charge network flow problem. *Journal on Computing*, 22:314–325, 2010.

N. Katayama, M. Z. Chen, and M. Kubo. A capacity scaling procedure for the multi-commodity capacitated network design problem. *Journal of Computational and Applied Mathematics*, 232(2):90–101, 2009.

L. Munguí, S. Ahmed, D. A. Bader, G. L. Nemhauser, V. Goel, and Y. Shao. A parallel local search frame work for the fixed-charge multicommodity network flow problem. *Computers & Operations Research*, 77:44–57, 2017.

片山直登. 容量制約をもつネットワーク設計問題に対するMIP近傍探索法. **流通経済大学流通情報学部紀要**, 22(1):1–18, 2017.

Algorithm 1: Solving $CNDPL(\bar{A}, \bar{P}, C)$

Set Col_{max};
for $ite \leftarrow 1$ **to** Col_{max} **do**
  Solve $CNDPL(\bar{A}, \bar{P}, C)$;
  Get the optimal solution $\tilde{y}$ of $CNDPL(\bar{A}, \bar{P}, C)$;
  Get the optimal dual solution of $CNDPL(\bar{A}, \bar{P}, C)$;
  for $k \in K$ **do**
    Solve the shortest path problem with reduced arc costs;
    Get the shortest path p and the shortest path length μ for commodity k;
    if $\mu <$ *the current path length* π^k **then**
      Add p to $\bar{P}$;
      Generate path flow variable x_p^k, and generate associated forcing constraints for each (i, j);
      Add x_p^k and associated forcing constraints;
    end
    if *no new path flow variable have generated* **then**
      break;
    end
  end
end
Return $\bar{P}$,$\tilde{y}$;

Algorithm 2: MIP Neighborhood Search

Set A, $\bar{P}$, λ, ϵ, ITE_{min}, ITE_{max}, $ArcNum$, M, α, T;
$\bar{P}$,$\tilde{y}$ = Solving $CNDPL(A, \bar{P}, C)$;
$\bar{A} \leftarrow A$; $C^0 \leftarrow C$; $l \leftarrow 1$;
repeat
  for $(i, j) \in \bar{A}$ **do**
    $C_{ij}^l \leftarrow \lambda C_{ij}^{l-1} \tilde{y}_{ij} + (1 - \lambda) C_{ij}^{l-1}$;
  end
  $\bar{P}$,$\tilde{y}$ = Solving $CNDPL(\bar{A}, \bar{P}, C^l)$;
  $n \leftarrow 0$;
  for $(i, j) \in \bar{A}$ **do**
    if $\tilde{y}_{ij} > \epsilon$ **then**
      $n \leftarrow n + 1$;
    else
      $\bar{A} \leftarrow \bar{A} \backslash \{(i, j)\}$;
    end
  end
until $l \geq ITE_{min}$ *and* $n \leq ArcNum$, *or* $l \geq ITE_{max}$;
Solve $CNDP(\bar{A}, \bar{P}, C)$ within time T;
Get the solution $\hat{y}$ and the upper bound UB;
repeat
  Add equations (15), (16) and (17) to $CNDA(\bar{A})$ for the current solution $\hat{y}$;
  Solve $CNDA(\bar{A})$ within time T;
  Delete equations (15), (16) and (17) from $CNDA(\bar{A})$;
  if $CNDA(\bar{A})$ *has no feasible solution* **then**
    break;
  else
    if *the solution* $\tilde{y}$ *of* $CNDA(\bar{A})$ *is found* **then**
      $\hat{y} \leftarrow \tilde{y}$;
      $UB \leftarrow UB_{local}$
    else
      $M \leftarrow \lfloor M/\alpha \rfloor$;
    end
  end
until $M = 0$;
Return $\hat{y}, UB$;

表 1: Average Gap for GT-Category Problems(%)

IPS	PAL	MIPR	MIPD
24.58	16.13	14.25	14.37

表 2: Results for GT-Category Problems

N/A/K/F/C	LB	IPS	PAL	MIPR	MIPD
500/2000/050/F/L	3477086	3823610	3722839	*3684997*	3685000
500/2000/100/F/L	5611485	6453880	6005177	*5998164*	6021802
500/2000/150/F/L	6863228	8081600	7510651	*7460985*	7508950
500/2000/200/F/L	8165040	9828350	9338097	9006397	*8991436*
500/2500/050/F/L	3211475	3612030	3491664	*3442960*	3484280
500/2500/100/F/L	5144270	6400140	5909401	*5725155*	5740127
500/2500/150/F/L	6818030	9089920	8138918	*7940573*	7947338
500/2500/200/F/L	8057922	10099200	9788913	*9245826*	9275190
500/3000/050/F/L	2989771	3457280	3369303	3331662	*3321234*
500/3000/100/F/L	5012129	6015950	5773133	*5550014*	5564884
500/3000/150/F/L	6449785	8919720	7741294	7612219	*7570176*
500/3000/200/F/L	7439015	10040000	9195115	8834674	*8807533*
500/2000/050/F/T	4326550	4949780	4892012	*4885949*	4903703
500/2000/100/F/T	6368730	7619670	*7273916*	7275823	7330582
500/2000/150/F/T	7256504	8807650	8014986	*7908672*	7924290
500/2000/200/F/T	8845440	11893100	10617796	10534572	*10526348*
500/2500/050/F/T	3927990	4600200	4406080	4418094	*4375932*
500/2500/100/F/T	5330490	6953660	*6365848*	6385223	6414804
500/2500/150/F/T	6120540	7571640	7037860	*6889150*	6913510
500/2500/200/F/T	8582837	11452900	10727261	*10456499*	10476206
500/3000/050/F/T	3529370	4262350	*4035362*	4098767	4068199
500/3000/100/F/T	5442880	7186810	6634387	*6516032*	6544023
500/3000/150/F/T	6256551	8709390	7517445	*7434358*	7477130
500/3000/200/F/T	7650794	10390700	9751002	9557845	*9492740*

表 3: Gaps for GT-Category Problems(%)

N/A/K/F/C	IPS	PAL	MIPR	MIPD
500/2000/050/F/L	9.97	7.07	5.98	5.98
500/2000/100/F/L	15.01	7.02	6.89	7.31
500/2000/150/F/L	17.75	9.43	8.71	9.41
500/2000/200/F/L	20.37	14.37	10.30	10.12
500/2500/050/F/L	12.47	8.72	7.21	8.49
500/2500/100/F/L	24.41	14.87	11.29	11.58
500/2500/150/F/L	33.32	19.37	16.46	16.56
500/2500/200/F/L	25.33	21.48	14.74	15.11
500/3000/050/F/L	15.64	12.69	11.44	11.09
500/3000/100/F/L	20.03	15.18	10.73	11.03
500/3000/150/F/L	38.29	20.02	18.02	17.37
500/3000/200/F/L	34.96	23.61	18.76	18.40
500/2000/050/F/T	14.40	13.07	12.93	13.34
500/2000/100/F/T	19.64	14.21	14.24	15.10
500/2000/150/F/T	21.38	10.45	8.99	9.20
500/2000/200/F/T	34.45	20.04	19.10	19.00
500/2500/050/F/T	17.11	12.17	12.48	11.40
500/2500/100/F/T	30.45	19.42	19.79	20.34
500/2500/150/F/T	23.71	14.99	12.56	12.96
500/2500/200/F/T	33.44	24.99	21.83	22.06
500/3000/050/F/T	20.77	14.34	16.13	15.27
500/3000/100/F/T	32.04	21.89	19.72	20.23
500/3000/150/F/T	39.20	20.15	18.83	19.51
500/3000/200/F/T	35.81	27.45	24.93	24.08

表 4: Average Computation Time for GT-Category Problems(Seconds)

IPS	PAL	MIPR	MIPD
3600	3600	15384	9152

表 5: Computation Time for GT-Category Problems(Seconds)

N/A/K/F/C	IPS	PAL	MIPR	MIPD
500/2000/050/F/L	3600	3600	2219	2584
500/2000/100/F/L	3600	3600	9671	8813
500/2000/150/F/L	3600	3600	13778	8546
500/2000/200/F/L	3600	3600	14134	9360
500/2500/050/F/L	3600	3600	5507	3547
500/2500/100/F/L	3600	3600	9492	7942
500/2500/150/F/L	3600	3600	22851	12372
500/2500/200/F/L	3600	3600	31637	15568
500/3000/050/F/L	3600	3600	6546	2338
500/3000/100/F/L	3600	3600	15366	9135
500/3000/150/F/L	3600	3600	27619	18709
500/3000/200/F/L	3600	3600	33924	22302
500/2000/050/F/T	3600	3600	4300	2325
500/2000/100/F/T	3600	3600	4517	5398
500/2000/150/F/T	3600	3600	13619	8404
500/2000/200/F/T	3600	3600	12706	4645
500/2500/050/F/T	3600	3600	3874	1913
500/2500/100/F/T	3600	3600	10010	4589
500/2500/150/F/T	3600	3600	25099	28430
500/2500/200/F/T	3600	3600	17734	11757
500/3000/050/F/T	3600	3600	5624	2555
500/3000/100/F/T	3600	3600	18851	3287
500/3000/150/F/T	3600	3600	21687	9269
500/3000/200/F/T	3600	3600	38460	15858