修　士　学　位　論　文

実務的なロジスティック・ネットワーク設計問題の求解高速化

2019年度
（2020年3月）

東京海洋大学大学院
海洋科学技術研究科
海運ロジスティクス専攻

佐 藤　良 亮

目 次

1　はじめに

1.1　研究背景

企業が製品を工場から流通センターを経由して卸まで配送する際には，新規の流通センターを建てるために，最適な施設配置の決定が必要であるが，求解時間が長く実用的でない．

本研究は，LION 株式会社との共同研究であり，求解の高速化を行うことを目的としている．

1.2　論文構成

本論文の構成は以下の通りである．

- 2 章では，問題設定について述べる
- 3 章では，定式化を示す
- 4 章では，実験について述べる
- 5 章では，まとめと今後の課題について述べる

2　問題設定

本章では，概要と条件について述べる．

2.1　概要

- 地点は工場，流通センター，卸があり，それぞれ複数の地点をもつ
- 製品は，工場，流通センター，卸の順に運ばれる
- 複数品目の製品を扱う

2.2　条件

- ある卸において，流通センターから製品が配送されてくる際は，全ての製品が 1 つの流通センターから運ばれること
- 都道府県や地方別に流通センター施設数の下限と上限を設定できるようにすること
- 既存の流通センターを固定した状態で，新しく流通センターの配置を決定できるようにすること

3　定式化

3.1　集合

PL: 工場の集合

D: 流通センターの集合

C: 卸の集合

PR: 製品の集合

TR: 輸送ルートの集合

DR: 配送ルートの集合

R_r: 地域 r に属する都道府県の集合

P_p: 都道府県 p に属する流通センターの集合

F: 固定する既存の流通センターの集合

3.2　パラメータ

tc_{ijp}: 工場 i から流通センター j への製品 p の輸送費用

dc_{jkp}: 流通センター j から卸 k への製品 p の配送費用

dl_j: 流通センター j における荷役費用

di_j: 流通センター j における在庫費用

Plu_{ip}: 工場 i における製品 p の生産容量

dcl_j: 流通センター j の取り扱い下限量

dcu_j: 流通センター j の取り扱い上限量

nl: 流通センター数の下限

nu: 流通センター数の上限

pl_p: 都道府県 p における流通センター数の下限

pu_p: 都道府県 p における流通センター数の上限

rl_r: 地域 r における流通センター数の下限

ru_r: 地域 r における流通センター数の上限

d_{kp}: 卸 k における製品 p の需要量

M: 非常に大きい値を表すパラメータ

3.3 変数

x_{ijp}: 工場 i から流通センター j への製品 p の輸送量

y_j: 流通センター j を建てるか否かのバイナリ変数

z_j: 流通センター j から卸 k への配送をするか否かのバイナリ変数

pls_{ip}: 工場 i の製品 p における余剰変数

dsu_j: 流通センター j における余剰変数

dsl_j: 流通センター j におけるスラック変数

3.4 モデル

min.

$$\sum_{(i,j,p)\in TR} tc_{ijp}x_{ijp} + \sum_{(j,k,p)\in DR} \sum_{(k,p)\in d_{kp}} \sum_{(j,k)\in z_{jk}} dc_{jkp}d_{kp}z_{jk}$$

$$+\sum_{j\in D}\sum_{p\in PR} di_{jp}\Big(\sum_{(i,j,p)\in TR} x_{ijp}\Big) + \sum_{j\in D} dl_j\Big(\sum_{(i,j,p)\in TR} x_{ijp}\Big)$$

$$+\sum_{i\in PL}\sum_{p\in PR} pls_{ip}M + \sum_{j\in D}(dsu_j + dsl_j)M \tag{1}$$

s.t.

$$\sum_{i \in PL} \sum_{(i,j,p) \in x} x_{ijp} = \sum_{k \in C} \sum_{(k,p) \in d_{kp}} \sum_{(j,k) \in z_{jk}} d_{kp} z_{jk} \qquad \forall j \in D, \forall p \in PR \qquad (2)$$

$$\sum_{j \in D} \sum_{(j,k) \in z} z_{jk} = 1 \qquad \forall k \in C \qquad (3)$$

$$z_{jk} \leq y_j \qquad \forall j, k \in z \qquad (4)$$

$$\sum_{j \in D} \sum_{(i,j,p) \in x} x_{ijp} \leq plu_{ip} + pls_{ip} \qquad \forall i, p \in plu \qquad (5)$$

$$dsu_j + dcu_j y_j \geq \sum_{i \in PL} \sum_{(i,j,p) \in x} x_{ijp} \qquad \forall j \in D \qquad (6)$$

$$dl_j y_j - dsl_j \leq \sum_{i \in PL} \sum_{(i,j,p) \in x} x_{ijp} \qquad \forall j \in D \qquad (7)$$

$$nl \leq \sum_{j \in D} y_j \leq nu \qquad (8)$$

$$pl_p \leq \sum_{j \in P_p} y_j \leq pu_p \qquad \forall p \in P \qquad (9)$$

$$rl_r \leq \sum_{p \in R_r} \sum_{j \in P_p} y_j \leq ru_r \qquad \forall r \in R \qquad (10)$$

$$y_j = 1 \qquad \forall j \in F \qquad (11)$$

目的関数と各制約式の意味を以下に示す.

(1) 目的関数である．総費用であり，総費用の最小化を目的とする．

以下は，輸送費用，配送費用，在庫費用，荷役費用を表す.

$$\sum_{(i,j,p)\in TR} tc_{ijp}x_{ijp} + \sum_{(j,k,p)\in DR}\sum_{(k,p)\in d_{kp}}\sum_{(j,k)\in z_{jk}} dc_{jkp}d_{kp}z_{jk}$$

$$+\sum_{j\in D}\sum_{p\in PR} di_{jp}(\sum_{(i,j,p)\in TR} x_{ijp}) + \sum_{j\in D} dl_j(\sum_{(i,j,p)\in TR} x_{ijp})$$

以下は，余剰変数やスラック変数に非常に大きな値を掛けることにより，余剰変数やスラック変数が選択されないようにしている.

$$\sum_{i\in PL}\sum_{p\in PR} Pps_{ip}M + \sum_{j\in D}(dsu_j + dsl_j)M$$

(2) 制約式である．工場から流通センターへの輸送量と，卸で発生する需要量は等しい.

$$\sum_{i\in PL}\sum_{(i,j,p)\in x} x_{ijp} = \sum_{k\in C}\sum_{(k,p)\in d_{kp}}\sum_{(j,k)\in z_{jk}} d_{kp}z_{jk} \qquad \forall j \in D, \forall p \in PR$$

(3) 制約式である．ある卸において，流通センターから製品が配送されてくる際は，全ての製品が 1 つの流通センターから運ばれる.

$$\sum_{j\in D}\sum_{(j,k)\in z} z_{jk} = 1 \qquad \forall k \in C$$

(4) 強化制約である．流通センター建てない場合は、その流通センターから卸に配送することはできない.

$$z_{jk} \leq y_j \qquad \forall j, k \in z$$

(5) 工場から流通センターへの輸送量は工場での生産容量以下である．ここでは，工場 i の製品 p における余剰変数を利用する.

$$\sum_{j\in D}\sum_{(i,j,p)\in x} x_{ijp} \leq plu_{ip} + pls_{ip} \qquad \forall i, p \in plu$$

(6) 工場から流通センターへの輸送量は流通センターの取り扱い上限量以下である．ここでは，流通センター j における余剰変数を利用する.

$$dsu_j + dcu_j y_j \geq \sum_{i\in PL}\sum_{(i,j,p)\in x} x_{ijp} \qquad \forall j \in D$$

(7) 工場から流通センターへの輸送量は流通センターの取り扱い下限量以上である．ここでは，流通センター j におけるスラック変数を利用する．

$$dl_j y_j - dsl_j \leq \sum_{i \in PL} \sum_{(i,j,p) \in x} x_{ijp} \qquad \forall j \in D \tag{12}$$

(8) 建てる流通センターの数は，流通センター数の下限以上であり，流通センター数の上限以下である．

$$nl \leq \sum_{j \in D} y_j \leq nu \tag{13}$$

(9) 都道府県 p で建てる流通センターの数は，都道府県 p における流通センター数の下限以上であり，都道府県 p における流通センター数の上限以下である．

$$pl_p \leq \sum_{j \in P_p} y_j \leq pu_p \qquad \forall p \in P \tag{14}$$

(10) 地域 r に属する都道府県 p における流通センターの数は，地域 r における流通センター数の下限以上であり，地域 r における流通センター数の上限以下である．

$$rl_r \leq \sum_{p \in R_r} \sum_{j \in P_p} y_j \leq ru_r \qquad \forall r \in R \tag{15}$$

(11) 固定する流通センターは建てる．

$$y_j = 1 \qquad \forall j \in F \tag{16}$$

4　実験

4.1　実験方法

本章では，高速化を考慮した実験の求解時間の比較を以下の方法で行う．

- 都道府県や地方別に流通センター施設数を固定せず，既存施設を固定していない状態

この状態の場合は，3 章で述べた制約式 (8)〜(11) は不必要となる．

4.1.1　求解高速化を考慮していない実験

求解高速化を考慮していない実験を行う．

4.1.2　求解高速化

高速化を考慮していない実験では求解時間が長く，実用的でない．桁数が大きいパラメータの桁数を少なくすること，小数を含むパラメータの整数化または小数点以下の桁数を四捨五入し少なくすることによって，求解の高速化をすることが期待できる．10 分や 1 時間程度で解を求められることを目指す．

以降は、以下の SCALE と ROUND を用いて実験を行う．

- SCALE … 桁数の大きいパラメータの桁数を少なくする
- ROUND … 小数を含むパラメータの整数化、または小数点以下の桁数を四捨五入することで少なくする

具体的には，値を SCALE で割り，四捨五入して ROUND の桁数までにすることとなる．以下に例を示す．

例）値を 12345.11111 とし，SCALE=100.0 , ROUND=1 とするとした場合

まず，100.0 で割ると 123.4511111 となる．そして小数第 2 位で四捨五入して 123.5 となる．

SCALE と ROUND は以下の方法で適用する．

- 制約式のそれぞれのパラメータに SCALE と ROUND を全て同じ値で適用
- 目的関数のそれぞれのパラメータに SCALE と ROUND を全て同じ値で適用

4.1.3 MST ファイルの使用

MST ファイル [1] を使用することでさらに求解高速化が期待できる．MST ファイルは混合整数線形計画問題の初期解を指定するために使用する．指定された初期解から初期可能解を構築しようとし，求解高速化が期待できる．

MST ファイルは連続変数を保存することができないため，本研究の実験ではバイナリ変数のみ保存することになる．

例えば、

- 制約式のパラメータ SCALE = 1000.0、ROUND = 0
- 目的関数のパラメータで SCALE と ROUND は無し

で求める場合に

- 制約式のパラメータ SCALE = 1000.0、ROUND = 0
- 目的関数のパラメータで SCALE = 1.0、ROUND = 0

を求めた MST ファイルを使用してから，求めた方が求解高速化が期待できる．

4.1.4 目的関数値の比較

SCALE と ROUND を様々に行うことによって求められた解を，SCALE と ROUND を行っていない目的関数に代入することで目的関数値がどのように変わるかを比較する．

以下のことに注意が必要である．

- 制約式のパラメータで SCALE と ROUND を行っているため，求まった輸送量は本来より SCALE した分だけ小さくなっている．比較する前に輸送量に SCALE をかける．
- 需要は制約式のパラメータで SCALE と ROUND を行っているため，目的関数にある需要にも比較する前に SCALE をかける．

比較する中で最も元の問題に近いと思われる目的関数の値と，その他の実験での目的関数値で比較する．

ここでは，目的関数値の比較として，以下の式で求めた値を用いる．

- （その他の実験での目的関数値）/ （最も元の問題に近いと思われる目的関数の値）

そのため，最も元の問題に近いと思われる目的関数の比較の値は 1.0 とする．

4.1.5　制約式の逸脱量

SCALE と ROUND を様々に行うことによって求められた解を，SCALE と ROUND を行っていない制約式の変数に代入することで，それぞれの制約式でどれだけ逸脱しているかを確認する．

以下のことに注意が必要である．

- 制約式のパラメータで SCALE と ROUND を行っているため，求まった輸送量は本来より SCALE した分だけ小さくなっている．比較する前に輸送量に SCALE をかける．

逸脱量を求める制約式は，以下の 4 つである．3 章の定式化を参照されたい．

(2) 工場から流通センターへの輸送量と，卸で発生する需要量は等しい

(5) 工場から流通センターへの輸送量は工場での生産容量以下である（工場の製品における余剰変数を利用する）

(6) 工場から流通センターへの輸送量は流通センターの取り扱い上限量以下である（流通センターにおける余剰変数を利用する）

(7) 工場から流通センターへの輸送量は流通センターの取り扱い下限量以上である（流通センターにおけるスラック変数を利用する）

以下に，逸脱となる場合について述べる．

- (2) で逸脱となるのは，
  - 工場から流通センターへの輸送量が卸で発生する需要量より多い場合 … a
  - 工場から流通センターへの輸送量が卸で発生する需要量より少ない場合 … b
- (5) で逸脱となるのは，工場から流通センターへの輸送量が，工場での生産容量と工場の製品における余剰変数の和より多い場合
- (6) で逸脱となるのは，工場から流通センターへの輸送量と流通センターにおける余剰変数の和が，流通センターの取り扱い上限量より多い場合
- (7) で逸脱となるのは，工場から流通センターへの輸送量から流通センターにおけるスラック変数を引いたものが，流通センターの取り扱い下限量より少ない場合

4.1.6　各モデルの要約統計量の出力

各モデルの統計を確認する [1]．求解の必要がなく，モデルを作成するだけで出力することができる．

4.2 実験環境

以下に，実験環境を示す．

OS 名	:	Windows 10 Pro
プロセッサ	:	Intel(R) Core(TM) i7-5960X CPU @ 3.00GHz 3.00GHz
実装メモリ (RAM)	:	64.0 GB
使用ソフト	:	Python ver.3.7.4
		Gurobi Optimizer ver.9.0.0

4.3 実験結果

4.3.1 求解高速化を考慮していない実験

丸 1 日，つまり 86400 秒たっても解を得ることができなかった．

4.3.2 求解高速化

以下の方法で実験を行う．

- 1. 制約式のパラメータ SCALE=100.0, ROUND=0 とし，目的関数のパラメータは以下のように変える．

(1) SCALE=100.0, ROUND=2

(2) SCALE=10.0, ROUND=1

(3) SCALE=1.0, ROUND=0

(4) SCALE, ROUND は無し

- 2. 制約式のパラメータ SCALE=1000.0, ROUND=0 とし，目的関数のパラメータは以下のように変える．

(1) SCALE=100.0, ROUND=2

(2) SCALE=10.0, ROUND=1

(3) SCALE=1.0, ROUND=0

(4) SCALE, ROUND は無し

- 3. 制約式のパラメータ SCALE=10000.0, ROUND=0 とし，目的関数のパラメータは以下のように変える．

(1) SCALE=100.0, ROUND=2

(2) SCALE=10.0, ROUND=1

(3) SCALE=1.0, ROUND=0

(4) SCALE, ROUND は無し

以上の通りに，実験を行う．

目的関数のパラメータで SCALE と ROUND したことにより，本来存在するものが 0 になってしまう場合は全て 0，または全て 1 とする．

- 1. 制約式のパラメータ SCALE=100.0, ROUND=0 の場合

  求解時間は以下の通りである．

表 1: 制約式のパラメータ SCALE=100.0, ROUND=0

目的関数のパラメータ		求解時間	線形緩和時間
SCALE	ROUND	(秒)	(秒)
100.0	2	21912.66	2790.59
	1(全て 0)	14220.55	6619.87
	1(全て 1)	12128.71	7748.60
	0(全て 0)	15483.27	7107.05
	0(全て 1)	27927.99	9101.48
10.0	1	11760.20	5646.95
	0(全て 0)	12070.36	8188.86
	0(全て 1)	9840.38	6925.81
無し	0	10409.73	6406.34
無し	無し	11756.13	6165.13

どれも 1 時間程度では解が得られず，求解に 3 時間程度かかる結果となった．求解時間に大きな差はあまり無いこともわかる．

- 2. 制約式のパラメータ SCALE=1000.0, ROUND=0 の場合

  求解時間は以下の通りである．

表 2: 制約式のパラメータ SCALE=1000.0, ROUND=0 の場合

目的関数のパラメータ		求解時間	線形緩和時間
SCALE	ROUND	(秒)	(秒)
100.0	2	2841.37	1386.54
	1(全て 0)	1649.24	1081.82
	1(全て 1)	1974.36	1207.96
	0(全て 0)	2024.45	771.53
	0(全て 1)	2903.17	1741.56
10.0	1	2973.72	1301.58
	0(全て 0)	1618.99	768.91
	0(全て 1)	1803.60	1235.25
無し	0	1668.79	1019.22
無し	無し	4723.32	3146.94

  以上より，目的関数のパラメータで SCALE と ROUND を適用すれば，1 時間以内で解を得ることができる．

- 3. 制約式のパラメータ SCALE=10000.0, ROUND=0 の場合求解時間は以下の通りである．

表 3: 制約式のパラメータ SCALE=10000.0, ROUND=0 の場合

目的関数のパラメータ		求解時間	線形緩和時間
SCALE	ROUND	(秒)	(秒)
100.0	2	1049.45	905.17
	1(全て 0)	744.22	528.13
	1(全て 1)	805.43	489.58
	0(全て 0)	560.76	159.10
	0(全て 1)	1358.56	1047.05
10.0	1	897.92	708.80
	0(全て 0)	2491.49	2087.88
	0(全て 1)	932.50	706.73
無し	0	987.76	760.85
無し	無し	892.58	569.94

以上より，1 時間以内で解を得ることができ，求解時間に大きな差はなかった．

ここまでの実験から，求解時間は制約式のパラメータでの SCALE の値で大きく変わることがわかり，制約式のパラメータで SCALE=1000.0 とすると，求解時間が急激に短くなることがわかる．

また，制約式のパラメータが同じ場合では，目的関数のパラメータで SCALE と ROUND を何かしら適用すると，求解時間が短くなりやすいことがわかる．

4.3.3 MST ファイルの使用

- 制約式のパラメータ
  SCALE = 1000.0, ROUND = 0
- 目的関数のパラメータ
  SCALE は無し, ROUND = 0

を求解する実験で考える．

MST ファイルを用いる場合は，

- 制約式のパラメータ
  SCALE = 1000.0, ROUND = 0
- 目的関数のパラメータ
  SCALE も ROUND も無し

を求解した MST ファイルを用いてから，求解する．

求解時間を比較する際，MST ファイルを用いた実験は 2 段階の求解時間の合計で比較する必要がある．以下の表で，MST ファイルを用意する段階を MST 前，MST ファイルを使用した段階を MST 後ということとする．

表 4: MST ファイルの使用と未使用で求解時間を比較

MST ファイル	求解時間 (秒)
未使用	4723.32
使用	3791.55
使用 (MST 前)	1668.79
使用 (MST 後)	2122.76

MST ファイルを用いることで求解の高速化をすることができた．

MST 前と MST 後のそれぞれの制約式のパラメータでの SCALE が同じ値の場合は求解高速化ができることがわかった．MST 後の制約式のパラメータでの SCALE を MST 前よりも小さいものとしたときにも，求解の高速化が期待できる．

制約式のパラメータでSCALE=1000.0, ROUND=0 の場合に，目的関数のパラメータでの各SCALE で最も求解時間の速い実験を，MST ファイルを用いて実験したときと比較する．

つまり，制約式のパラメータでSCALE = 1000.0, ROUND = 0 とし，目的関数のパラメータは以下のように変える．

(1) 目的関数のパラメータでSCALE = 100.0, ROUND = 1(全て 0)

(2) 目的関数のパラメータでSCALE = 10.0, ROUND = 0(全て 0)

(3) 目的関数のパラメータでSCALE とROUND は無し

を求解する実験で考える．

(1) の場合

- 制約式のパラメータ
  SCALE = 10000.0, ROUND = 0
- 目的関数のパラメータ
  SCALE = 100.0, ROUND = 1(全て 0)

を求解したMST ファイルを用いてから，求解する．

表 5: MST ファイルの使用と未使用で求解時間を比較

MST ファイル	求解時間 (秒)
未使用	1649.24
使用	2432.35
使用 (MST 前)	744.22
使用 (MST 後)	1618.13

MST ファイルを用いないほうが速い結果となってしまった．

(2) の場合

- 制約式のパラメータ
  SCALE = 10000.0, ROUND = 0
- 目的関数のパラメータ
  SCALE = 10.0, ROUND = 0(全て 0)

を求解した MST ファイルを用いてから，求解する．

表 6: MST ファイルの使用と未使用で求解時間を比較

MST ファイル	求解時間 (秒)
未使用	1618.99
使用	4178.91
使用 (MST 前)	2491.49
使用 (MST 後)	1687.42

MST ファイルを用いないほうが速い結果となってしまった．

(3) の場合

- 制約式のパラメータ
  SCALE = 10000.0, ROUND = 0
- 目的関数のパラメータ
  SCALE と ROUND は無し

を求解した MST ファイルを用いてから，求解する．

表 7: MST ファイルの使用と未使用で求解時間を比較

MST ファイル	求解時間 (秒)
未使用	1668.79
使用	3861.57
使用 (MST 前)	987.76
使用 (MST 後)	2873.81

MST ファイルを用いないほうが速い結果となってしまった．

これより，

- 制約式のパラメータで SCALE と ROUND が同じ
- 目的関数のパラメータで SCALE と ROUND を変える

の場合は，MST ファイルにより求解高速化ができ，

- 制約式のパラメータで SCALE と ROUND を変える
- 目的関数のパラメータで SCALE と ROUND が同じ

の場合は，MST ファイルにより求解高速化ができないことがわかる．

4.3.4 目的関数値の比較

4.3.2 と同じ方法で比較を行う．

- 1. 制約式のパラメータ SCALE=100.0, ROUND=0 の場合

表 8: 制約式のパラメータ SCALE=100.0, ROUND=0

目的関数のパラメータ		目的関数値	求解時間
SCALE	ROUND	の比較	(秒)
100.0	2	0.9999996938935971	21912.66
	1(全て 0)	1.0033758732951827	14220.55
	1(全て 1)	1.0033707066885338	12128.71
	0(全て 0)	1.0135028683350225	15483.27
	0(全て 1)	1.0553768715937826	27927.99
10.0	1	1.0000044274624815	11760.20
	0(全て 0)	1.0033391013481192	12070.36
	0(全て 1)	1.003946788870128	9840.38
無し	0	1.0000137261254243	10409.73
無し	無し	1.0	11756.13

- 2. 制約式のパラメータ SCALE=1000.0, ROUND=0 の場合

表 9: 制約式のパラメータ SCALE=1000.0, ROUND=0

目的関数のパラメータ		目的関数値	求解時間
SCALE	ROUND	の比較	(秒)
100.0	2	1.0000417871771095	2841.37
	1(全て 0)	1.0035053469194903	1649.24
	1(全て 1)	1.003502144679684	1974.36
	0(全て 0)	1.0092381456540105	2024.45
	0(全て 1)	1.071612122023213	2903.17
10.0	1	1.0000004300603451	2973.72
	0(全て 0)	1.0035116071268333	1618.99
	0(全て 1)	1.0035055184719344	1803.60
無し	0	1.0000004300603451	1668.79
無し	無し	1.0	4723.32

- 3. 制約式のパラメータ SCALE=10000.0, ROUND=0 の場合

表 10: 制約式のパラメータ SCALE=10000.0, ROUND=0

目的関数のパラメータ		目的関数値	求解時間
SCALE	ROUND	の比較	(秒)
100.0	2	0.9986490629984158	1049.45
	1(全て 0)	1.0051341309935455	744.22
	1(全て 1)	1.0041534654046085	805.43
	0(全て 0)	1.0149315014925007	560.76
	0(全て 1)	1.055043475313331	1358.56
10.0	1	0.9988317489911104	897.92
	0(全て 0)	1.0078228977329595	2491.49
	0(全て 1)	1.003225891971717	932.50
無し	0	1.003991517500285	987.76
無し	無し	1.0	892.58

以上の実験から，目的関数のパラメータの小数を少なくすると，目的関数値が悪くなりやすいことがわかる．

4.3.5　制約式の逸脱量

例として，

- 目的関数のパラメータで SCALE は無し，ROUND=0
- 制約式のパラメータで

1. SCALE = 100.0 , ROUND = 0
2. SCALE = 1000.0 , ROUND = 0
3. SCALE = 10000.0 , ROUND = 0

の場合で，逸脱量を確認する．各制約式の逸脱量の分布を箱ひげ図を用いて示す．横軸は制約式の SCALE の値，縦軸は逸脱量を表す．

- 制約式 (2) の a
  - 工場から流通センターへの輸送量が卸で発生する需要量より多い場合

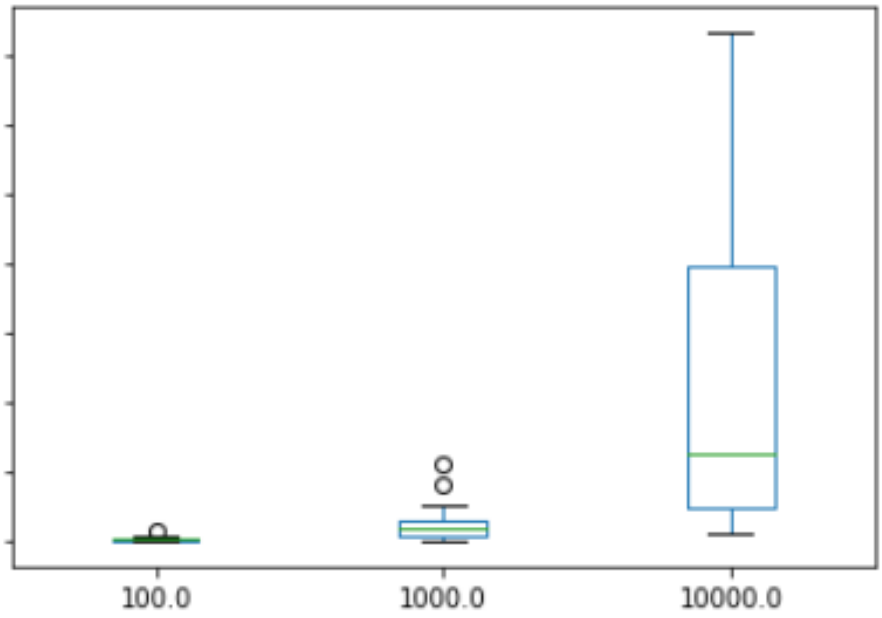

図 1: 制約式 (2) の a

- 制約式 (2) の b
  - 工場から流通センターへの輸送量が卸で発生する需要量より少ない場合

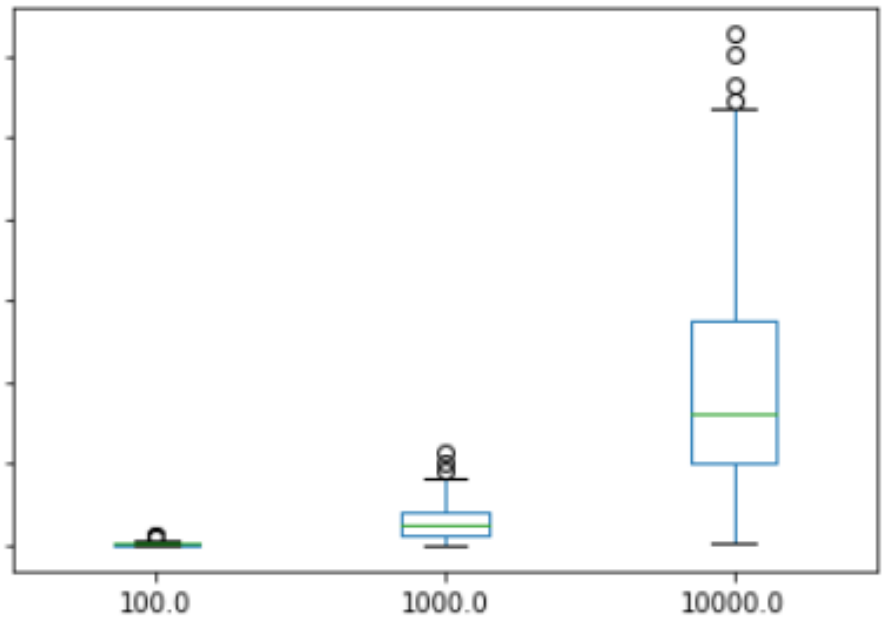

図 2: 制約式 (2) の b

制約式 (2) の a と b では，共に制約式の SCALE が大きいほど逸脱量が多いことがわかる．

- 制約式 (5)
  - 工場から流通センターへの輸送量が、工場での生産容量と工場の製品における余剰変数の和より多い場合

  少し，逸脱している制約式が見られる程度であった．

- 制約式 (6)
  - 工場から流通センターへの輸送量と流通センターにおける余剰変数の和が、流通センターの取り扱い上限量より多い場合

  一切逸脱していなかった．

- 制約式 (7)
  - 工場から流通センターへの輸送量から流通センターにおけるスラック変数を引いたものが、流通センターの取り扱い下限量より少ない場合

  少し，逸脱している制約式が見られる程度であった．

制約式 (2) で多くの逸脱量が見られる結果となった．需要量は，小さい値から大きい値まで存在しており，SCALE することで，本来 0 でない需要量が 0 になってしまう．制約式 (2) では需要量が含まれており，逸脱量が多くなったと考えられる．

4.3.6　各モデルの要約統計量の出力

モデルの統計を確認することができる．以下は確認できる情報の説明である．

- Linear constraint matrix ⋯ 線形制約行列
- Variable types ⋯ 変数の種類
- Matrix coefficient range ⋯ 行列係数の範囲
- Objective coefficient range ⋯ 目的関数の係数の範囲
- Variable bound range ⋯ 束縛変数の範囲
- RHS coeffcient range ⋯ 右辺係数の範囲

4.3.2 と同じ方法で結果を示す．
Linear constraint matrix の結果を示す．

表 11: 制約式のパラメータ SCALE=100.0, ROUND=0

目的関数のパラメータ		Linear constraint matrix			求解時間
SCALE	ROUND	constrs	Vars	NZs	(秒)
100.0	2	142092	185487	1494065	21912.66
	1(全て 0)	142092	185487	1494065	14220.55
	1(全て 1)	142092	185487	1494065	12128.71
	0(全て 0)	142092	185487	1494065	15483.27
	0(全て 1)	142092	185487	1494065	27927.99
10.0	1	142092	185487	1494065	11760.20
	0(全て 0)	142092	185487	1494065	12070.36
	0(全て 1)	142092	185487	1494065	9840.38
無し	0	142092	185487	1494065	10409.73
無し	無し	142092	185487	1494065	11756.13

表 12: 制約式のパラメータ SCALE=1000.0, ROUND=0

目的関数のパラメータ		Linear constraint matrix			求解時間
SCALE	ROUND	constrs	Vars	NZs	(秒)
100.0	2	142092	185487	1101766	2841.37
	1(全て 0)	142092	185487	1101766	1649.24
	1(全て 1)	142092	185487	1101766	1974.36
	0(全て 0)	142092	185487	1101766	2024.45
	0(全て 1)	142092	185487	1101766	2903.17
10.0	1	142092	185487	1101766	2973.72
	0(全て 0)	142092	185487	1101766	1618.99
	0(全て 1)	142092	185487	1101766	1803.60
無し	0	142092	185487	1101766	1668.79
無し	無し	142092	185487	1101766	4723.32

表 13: 制約式のパラメータ SCALE=10000.0, ROUND=0

目的関数のパラメータ		Linear constraint matrix			求解時間
SCALE	ROUND	constrs	Vars	NZs	(秒)
100.0	2	142092	185487	814630	1049.45
	1(全て 0)	142092	185487	814630	744.22
	1(全て 1)	142092	185487	814630	805.43
	0(全て 0)	142092	185487	814630	560.76
	0(全て 1)	142092	185487	814630	1358.56
10.0	1	142092	185487	814630	897.92
	0(全て 0)	142092	185487	814630	2491.49
	0(全て 1)	142092	185487	814630	932.50
無し	0	142092	185487	814630	987.76
無し	無し	142092	185487	814630	892.58

Linear constraint matrix は制約式のパラメータの SCALE と ROUND が同じであれば，目的関数のパラメータでの SCALE と ROUND に関係なく，等しいことがわかる．制約式のパラメータでの SCALE が変わると大きくなると，制約式の係数が 0 でないものを表す NZs は小さくなることがわかる．

次に Variable types であるが，SCALE と ROUND をしても結果は変わらないものである．

次に，Matrix coefficient range の結果を示す．

表 14: 制約式のパラメータ SCALE=100.0, ROUND=0

目的関数のパラメータ		Matrix coefficient range		求解時間
SCALE	ROUND	min	max	(秒)
100.0	2	1	68000	21912.66
	1(全て 0)	1	68000	14220.55
	1(全て 1)	1	68000	12128.71
	0(全て 0)	1	68000	15483.27
	0(全て 1)	1	68000	27927.99
10.0	1	1	68000	11760.20
	0(全て 0)	1	68000	12070.36
	0(全て 1)	1	68000	9840.38
無し	0	1	68000	10409.73
無し	無し	1	68000	11756.13

表 15: 制約式のパラメータ SCALE=1000.0, ROUND=0

目的関数のパラメータ		Matrix coefficient range		求解時間
SCALE	ROUND	min	max	(秒)
100.0	2	1	6800	2841.37
	1(全て 0)	1	6800	1649.24
	1(全て 1)	1	6800	1974.36
	0(全て 0)	1	6800	2024.45
	0(全て 1)	1	6800	2903.17
10.0	1	1	6800	2973.72
	0(全て 0)	1	6800	1618.99
	0(全て 1)	1	6800	1803.60
無し	0	1	6800	1668.79
無し	無し	1	6800	4723.32

表 16: 制約式のパラメータ SCALE=10000.0, ROUND=0

目的関数のパラメータ		Matrix coefficient range		求解時間
SCALE	ROUND	min	max	(秒)
100.0	2	1	680	1049.45
	1(全て 0)	1	680	744.22
	1(全て 1)	1	680	805.43
	0(全て 0)	1	680	560.76
	0(全て 1)	1	680	1358.56
10.0	1	1	680	897.92
	0(全て 0)	1	680	2491.49
	0(全て 1)	1	680	932.50
無し	0	1	680	987.76
無し	無し	1	680	892.58

Matrix coefficient range は制約式のパラメータの SCALE と ROUND が同じであれば，目的関数のパラメータでの SCALE と ROUND に関係なく，等しいことがわかる．制約式のパラメータでの SCALE が変わると大きくなると，小さくなることがわかる．

次に，Objective coefficient range の結果を示す．

表 17: 制約式のパラメータ SCALE=100.0, ROUND=0

目的関数のパラメータ		Objective coefficient range		求解時間
SCALE	ROUND	min	max	(秒)
100.0	2	0.14	1e+09	21912.66
	1(全て 0)	0.1	1e+09	14220.55
	1(全て 1)	0.1	1e+09	12128.71
	0(全て 0)	1	1e+09	15483.27
	0(全て 1)	1	1e+09	27927.99
10.0	1	1.4	1e+09	11760.20
	0(全て 0)	1	1e+09	12070.36
	0(全て 1)	1	1e+09	9840.38
無し	0	14	1e+09	10409.73
無し	無し	14.4321	1e+09	11756.13

表 18: 制約式のパラメータ SCALE=1000.0, ROUND=0

目的関数のパラメータ		Objective coefficient range		求解時間
SCALE	ROUND	min	max	(秒)
100.0	2	0.14	1e+09	2841.37
	1(全て 0)	0.1	1e+09	1649.24
	1(全て 1)	0.1	1e+09	1974.36
	0(全て 0)	1	1e+09	2024.45
	0(全て 1)	1	1e+09	2903.17
10.0	1	1.4	1e+09	2973.72
	0(全て 0)	1	1e+09	1618.99
	0(全て 1)	1	1e+09	1803.60
無し	0	14	1e+09	1668.79
無し	無し	14.4321	1e+09	4723.32

表 19: 制約式のパラメータ SCALE=10000.0, ROUND=0

目的関数のパラメータ		Objective coefficient range		求解時間
SCALE	ROUND	min	max	(秒)
100.0	2	0.34	1e+09	1049.45
	1(全て 0)	0.3	1e+09	744.22
	1(全て 1)	0.3	1e+09	805.43
	0(全て 0)	1	1e+09	560.76
	0(全て 1)	1	1e+09	1358.56
10.0	1	3.4	1e+09	897.92
	0(全て 0)	3	1e+09	2491.49
	0(全て 1)	3	1e+09	932.50
無し	0	34	1e+09	987.76
無し	無し	33.6918	1e+09	892.58

Objective coefficient range は制約式のパラメータの SCALE と ROUND が同じ場合，目的関数のパラメータでの SCALE が大きいほど小さくなりやすい．また，制約式のパラメータでの SCALE が変わると大きくなると，小さくなることがわかる．

次に Variable bound range であるが，SCALE と ROUND をしても結果は変わらないものである．

最後に，RHS coefficient range の結果を示す．

表 20: 制約式のパラメータ SCALE=100.0, ROUND=0

目的関数のパラメータ		RHS coefficient range		求解時間
SCALE	ROUND	min	max	(秒)
100.0	2	1	34691	21912.66
	1(全て 0)	1	34691	14220.55
	1(全て 1)	1	34691	12128.71
	0(全て 0)	1	34691	15483.27
	0(全て 1)	1	34691	27927.99
10.0	1	1	34691	11760.20
	0(全て 0)	1	34691	12070.36
	0(全て 1)	1	34691	9840.38
無し	0	1	34691	10409.73
無し	無し	1	34691	11756.13

表 21: 制約式のパラメータ SCALE=1000.0, ROUND=0

目的関数のパラメータ		RHS coefficient range		求解時間
SCALE	ROUND	min	max	(秒)
100.0	2	1	3469	2841.37
	1(全て 0)	1	3469	1649.24
	1(全て 1)	1	3469	1974.36
	0(全て 0)	1	3469	2024.45
	0(全て 1)	1	3469	2903.17
10.0	1	1	3469	2973.72
	0(全て 0)	1	3469	1618.99
	0(全て 1)	1	3469	1803.60
無し	0	1	3469	1668.79
無し	無し	1	3469	4723.32

表 22: 制約式のパラメータ SCALE=10000.0, ROUND=0

目的関数のパラメータ		RHS coefficient range		求解時間
SCALE	ROUND	min	max	(秒)
100.0	2	1	347	1049.45
	1(全て 0)	1	347	744.22
	1(全て 1)	1	347	805.43
	0(全て 0)	1	347	560.76
	0(全て 1)	1	347	1358.56
10.0	1	1	347	897.92
	0(全て 0)	1	347	2491.49
	0(全て 1)	1	347	932.50
無し	0	1	347	987.76
無し	無し	1	347	892.58

RHS は制約式のパラメータの SCALE と ROUND が同じであれば，目的関数のパラメータでの SCALE と ROUND に関係なく，等しいことがわかる．制約式のパラメータでの SCALE が変わると大きくなると，小さくなることがわかる．

5　おわりに

5.1　まとめ

第 1 章
　第 1 章では，本研究の研究背景と目的について述べた．本研究の目的は，最適な施設配置の決定の際に，求解の高速化を行うことである．

第 2 章
　第 2 章では，本研究の問題設定について述べた．

第 3 章
　第 3 章では，定式化について述べた．

第 4 章
　第 4 章では，実験について述べた．求解の高速化を考慮していない実験と，考慮した実験，そして MST ファイルを用いた求解の高速化，目的関数値の比較，制約式の逸脱量，モデルの要約統計の出力について述べた．

5.2　今後の課題

　今後の課題として，小さい需要を集約することがあげられる．需要は小さい値から大きい値まで存在し，SCALE をすると本来存在する需要が 0 となってしまう．集約することで本来存在する需要が 0 となってしまうということが起こりにくくなる．

謝辞

本研究を行うにあたり，熱心なご指導をいただいた指導教員である久保幹雄教授，共同研究という形でご協力いただいた LION 株式会社の皆様に深く感謝いたします．また，日頃の議論を通じ，多くの知識や示唆をいただいた研究室の皆様に感謝いたします. 誠にありがとうございました．

参考文献

[1] Documentation. GUROBI OPTIMIZATION.
URL:https://www.gurobi.com/documentation/