岡山理科大学紀要　第55号 A pp. 45–53（2019）

容量制約付き p-メディアン問題に対する局所探索法の探索性能の調査

三宅　孝史・金原　一歩・岡野　傑士・片山　謙吾 *

岡山理科大学大学院工学研究科情報工学専攻

* 岡山理科大学工学部情報工学科

（2019年10月31日受付、2019年12月9日受理）

1　まえがき

重要な応用を有する組合せ最適化問題の一つに容量制約付き p-メディアン問題（Capacitated p-Median Problem, CPMP）がある．CPMP は p-メディアン問題（p-Median Problem, PMP）を拡張した問題であり，容量が定められている施設を所定数開設し各顧客を開設施設に割当てるとき，その顧客から施設への距離の総和を最小化するように顧客を割当てる問題である．

CPMP をはじめとした多くの組合せ最適化問題を解く上で，最も基礎となる解探索手法として局所探索法が知られている [1]．局所探索法は，与えられた初期解を基に特定の操作によって新たな近傍解を生成するという処理を，解の改善がなされなくなるまで繰り返す解法である．遺伝的アルゴリズムをはじめとしたメタ戦略アルゴリズムの設計の際にも局所探索法を用いることが一般的であり，高性能な局所探索法の開発が必要となる．

CPMP は PMP の施設に容量制約条件を付け加えたものである．これまでに PMP に対する解法の研究 [2][3][4] は盛んに行われているが，CPMP に対する研究は PMP ほど多くはない．また，PMP に対する解法を容量制約条件を有する CPMP にそのまま適用することはできず，CPMP に対する容量制約条件を考慮した局所探索法を新たに設計しなければならない．

本論文では CPMP に対する局所探索法を数種設計し，基本的な解探索手法である多スタート局所探索法（Multi-start Local Search, MLS）の枠組みの下で各種局所探索法の探索性能を調査する．

2　容量制約付き p-メディアン問題

容量制約付き p-メディアン問題（Capacitated p-Median Problem, CPMP）は，避難計画や都市計画の分野にあらわれる実用上重要な最適化問題として知られている．CPMP の具体的な応用例としては，災害時における避難場所の策定などがあげられる．施設を避難所，顧客を住民へと置き換えることで避難問題としてとらえることができ，災害の前後にかかわらず、住民の避難所について，自宅からの移動距離および避難所の収容人数を考慮し計画することが可能になると考えられる．CPMP は p-メディアン問題 (p-Median Problem, PMP) を発展させた問題である．以下に PMP および CPMP の定義について述べる．

2.1　(容量制約なし)p-メディアン問題

p-メディアン問題 (p-Median Problem, PMP) は，グラフ $G = (V, E)$（ノード集合：V，枝集合：E）が与えられ，$|V| = n$ 個の施設候補のノード集合 $L(\subseteq V)$ から部分集合 $J(\subseteq L)$ を選択し，$|J| = p$ 個の施設ノードから最短距離となる全顧客ノードのコストの和を最小とする p 個の施設ノードを決定する問題である．各顧客 $i \in V$ は，p 個の施設の内，一つの施設 $j \in J$ に割当てられ，その割当コストは i から j までの距離となる．この割当コストを d_{ij}，解を x とすると，PMP の目的関数と制約条件は以下の式によって表すことができる．

$$\text{目的関数} \qquad \min f(x) = \sum_i \sum_j d_{ij} x_{ij} \tag{1}$$

$$\text{制約条件} \qquad \sum_j x_{ij} = 1, \quad \forall i \tag{2}$$

$$x_{ij} \leq y_j, \quad \forall i, j \tag{3}$$

$$\sum_j y_j = p \tag{4}$$

$$x_{ij}, y_j \in \{0, 1\} \tag{5}$$

制約条件として，式 (2) は各顧客が利用できる施設を一ヶ所のみとする制約であり，式 (3) は開設される施設のみを使用可能とする制約である．また，式 (4) は開設できる施設数の制約であり，式 (5) は要素を 0 もしくは 1 とすることを表す．

2.2 容量制約付き p-メディアン問題 CPMP

PMP に各施設が収容できる容量の制約を加えたものが容量制約付き p-メディアン問題 (Capacitated p-Median ProblemCPMP) である．CPMP では，式 (2)〜式 (5) の制約条件に新たに以下の制約が設けられる．

$$\sum_i u_i x_{ij} \ \leq \ C_j, \ \ \forall j \tag{6}$$

式 (6) において，C_j は施設 j の容量を，u_i は顧客 i の需要量を表す．この制約は，ある施設 j に割当てられた各顧客 i の需要量 u_i の和 $\sum_i u_i x_{ij}$ が，施設 j の容量 C_j を上回ってはならないことを表している．

3 CPMP に対する修正法 [5]

CPMP のような制約を有する組合せ最適化問題に対するアルゴリズムの設計の際は，実行可能領域の探索を維持することが困難である場合が生じる．そのような際の代表的なアルゴリズム設計方法として，実行可能解からの制約違反の程度を目的関数値に反映するペナルティ関数の導入や，実行不可能解が生成された時点で，強制的に実行可能解へと修正するアルゴリズムを設計することが一般的である．本研究では CPMP に対する修正法として標準的な修正法（standard Repair, sRepair）と距離優先の修正法（distance based Repair, dRepair）を設計した．2 つの修正法はどちらも nodeshift 操作によって解の修正を行う．nodeshift 操作はノードの割当て施設を変更する操作である．図 1 に nodeshift 操作の一例を示す．図 1 ではノード i が割当てられている施設が施設 A から施設 B に変更している様子を表している．各修正法はこの nodeshift 操作を容量制約条件を満たしていない施設に割当てられているノードに対して適用することで解の修正を行う．

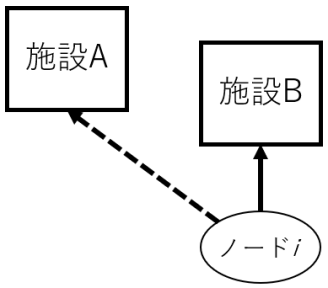

図 1: nodeshift 操作の一例

3.1 標準的な修正法 (standard Repair, sRepair)

sRepair は nodeshift 操作によって新たに割当てる施設に残許容量の一番大きい施設を選択する．図 2 に sRepair の疑似コードを示す．sRepair の処理手順として，容量制約条件を満たしていない開設施設の集団（OverMedians（M_o））からランダムに施設 f_i を 1 つ選択する（Line 4）．選ばれた施設に割当てられたノードの内，最も需要量（Demand）の小さいノード i を求める（Line 6）．ノード i の割当てが禁止されている施設と f_i を除く開設施設から残許容量が最も大きい施設 j を選択し，ノード i と施設 j に対して nodeshift 操作を行う（Lines 7–8）．残許容量は施設の空き容量であり，施設 f の残許容量は施設 f の最大容量（Capacity(f)）から同施設に割当てられているノードの需要量の総和（TotalDemand(f)）を引くことで求めることができる．以上の処理を施設 f_i が容量制約条件を満たすまで繰り返す．ここで修正法における解のサイクリングを防ぐため，施設 f_i から割当て施設を変更したノードの内，最も需要量が大きいものが再度施設 f_i に割当てられることを禁止する処理を行う（Line 14）．以上の処理をすべての施設で容量制約条件を満たすまで繰り返す．

```
    standard Repair(s)
    begin
 1   repeat
 2     max_demand := 0;
 3     initialize taboo_median;
 4     select one median f_i among them randomly with OverMedians(M_o);
 5     repeat
 6       i := argmin_{n∈AffiliationNodes(f_i)}{Demand(n)};
 7       j := argmax_{f∈M_o\{f_i}\{taboo_median[i]}} {Capacity(f)−TotalDemand(f)};
 8       s := NodeShiftMove(i, j, s);
 9       if max_demand < Demand(i) then
10         max_demand := Demand(i);
11         max_demand_node := i
12       endif
13     until f_i ∈ OverMedians(M_o);
14     taboo_median[max_demand_node] := f_i;
15   until OverMedains(M_o) = φ;
16   return s;
    end;
```

図 2: sRepair の疑似コード

3.2 距離優先の修正法（distance based Repair, dRepair）

dRepair は nodeshift 操作によって新たに割当てる施設を nodeshift 操作を適用するノードと他の施設との距離に基づいて選択する．図 3 に dRepair の疑似コードを示す．dRepair の処理手順として，sRepair と同様に容量制約条件を満たしていない開設施設の集団（OverMedians（M_o））からランダムに施設 f_i を 1 つ選択し（Line 4），選ばれた施設に割当てられたノードの内，最も需要量（Demand）の小さいノード i を求める（Line 6）．dRepair はノード i の割当てが禁止されている施設と f_i を除く開設施設から nodeshift 操作を適用しても容量制約条件を満たす施設があるならば，それらの施設の内最も距離が近い施設 f_j を選択し，ノード i と施設 f_j に対して nodeshift 操作を行う（Lines 8–10）．しかし nodeshift 操作を適用しても容量制約条件を満たす施設がない場合はノード i から最も近くにある施設 f_k を選択し，ノード i と施設 f_k に対して nodeshift 操作を行う（Lines 11–13）．以上の処理を sRepair と同様に施設 f_i が容量制約条件を満たすまで繰り返し，解のサイクリングを防ぐため，施設 f_i から割当て施設を変更したノードの内，最も需要量が大きいものが再度施設 f_i に割当てられることを禁止する処理を行う（Line 25）．以上の処理をすべての施設で容量制約条件を満たすまで繰り返す．

3.3 修正法の性能評価実験

2 種類の修正法の性能を調査するために実験を行った．実験では修正する実行不可能解を生成するためのアルゴリズムとしてランダム初期解生成方式と距離優先初期解生成方式の二つのアルゴリズムを設計した．ランダム初期解生成方式は p 個の施設をランダムに開設し，各地域の住民をそれぞれランダムな施設に割当てることで解を生成する．一方，距離優先初期解生成方式は p 個の施設をランダムに開設し，各地域の住民をそれぞれ最も近い施設に割当てることで解を生成する．これらの初期解生成方式によって得た実行不可能解に対して上記の修正法を適用し修正された実行可能解の精度を比較する．実験では各初期解生成方式と修正法の組合せによる以下の 4 種類の解の精度を比較する．

- ランダム初期解生成方式+sRepair
- ランダム初期解生成方式+dRepair
- 距離優先初期解生成方式+sRepair
- 距離優先初期解生成方式+dRepair

各組合せにおいて 10 試行解の修正を行い最良精度（Best）及び平均精度（Avg）を求めた．計算機は CPU: IntelCore i7 3.6GHz, RAM:15.6GiB を使用した．なお得られた解 x の目的関数値の精度（%）は $\frac{f(x)-f(x^*)}{f(x^*)} \times 100$ で求めた．ただし，x^* は既知の最適解である．

表 1 に各初期解生成方式ごとに各修正法によって修正した解の精度を示した．表は左から順に問題例名，ノード数 n，開設施設数 p，各修正法で修正した解の精度の Best と Avg を示している．各問題例において Best と Avg ごとに最

```
   distance based Repair(s)
   begin
 1  repeat
 2    max_demand := 0, d_min := infinity, capacitated_d_min := infinity;
 3    initialize taboo_median;
 4    select one median f_i among them randomly with OverMedians(M_o);
 5    repeat
 6      i := argmin_{n∈AffiliationNodes(f_i)}{Demand(n)};
 7      for all f ∈ M_o\{f_i}\{taboo_median[i]} do
 8        if TotalDemand(f) + Demand(i) ≤ Capacity(f)&Distance(i, f) < capacitated_d_min
 9          f_j := f, capacitated_d_min := Distance(i, f);
10        endif
11        if Distance(i, f) < d_min then
12          f_k := f, d_min := Distance(i, f);
13        endif
14      endfor
15      if capacitated_d_min ≠ infinity then
16        s := NodeShift(i, f_j, s);
17      else
18        s := NodeShift(i, f_k, s);
19      endif
20      if max_demand < Demand(i) then
21        max_demand := Demand(i);
22        max_demand_node := i
23      endif
24    until f_i ∈ OverMedians(M_o);
25    taboo_median[max_demand_node] := f_i;
26  until OverMedians(M_o) = φ;
27  return s;
   end;
```

図 3: dRepair の疑似コード

も精度の良いものを太字で示した．結果から，各初期解生成方式ごとに比較すると，すべての問題例において dRepair が sRepair を上回る精度の解に修正していることがわかる．特に距離優先初期解生成方式によって生成した解に対して dRepair を適用した際の解の精度が良好であった．

表 1: 修正法の結果

	n	p	ランダム初期解生成方式				距離優先初期解生成方式			
			sRepair		dRepair		sRepair		dRepair	
			BEST	AVG	BEST	AVG	BEST	AVG	BEST	AVG
SJC1	100	10	237.84	245.10	158.44	170.95	13.20	16.62	**11.39**	**13.61**
SJC2	200	15	403.97	417.92	317.15	325.58	14.59	20.65	**10.69**	**15.37**
SJC3a	300	25	570.67	580.10	497.96	507.99	24.75	28.19	**18.89**	**21.24**
SJC3b	300	30	658.77	669.85	596.58	606.99	22.71	24.96	**17.24**	**19.03**
SJC4a	402	30	621.27	629.22	526.39	543.02	35.53	41.71	**24.95**	**26.51**
SJC4b	402	40	768.98	779.21	701.92	717.03	24.07	27.76	**20.74**	**22.42**
平均			543.58	553.57	466.41	478.59	22.47	26.65	**17.32**	**19.70**

4 CPMP に対する局所探索法

CPMP における解は各ノードをどの施設に割当てるかと，どの施設を開設するかという二つの要素によって構成されている．そこで本研究ではノードの割当て施設を変更する nodeshift 操作と開設施設を変更する medianshift 操作に基づく局所探索法を設計した．nodeshift 操作は第 3 章で説明したものと同様である．図 4 に medianshift 操作の一例を示す．図 4 では開設施設 A を閉鎖し代わりに施設 B を開設している様子を表している．medianshift 操作では閉鎖した施設に割当てられていたノードは全て新たに開設した施設に割当てなおす．本研究では各 shift 操作ごとに複数の移動戦略に基づく局所探索法を設計した．

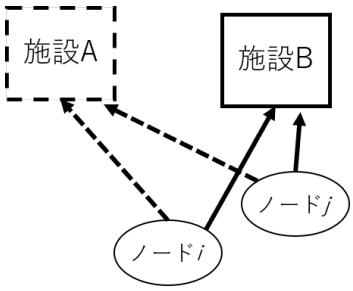

図 4: medianshift 操作の一例

4.1 nodeshift 操作及び medianshift 操作による局所探索法

nodeshift 操作による局所探索法として最良移動戦略に基づく NodeShift 局所探索法（Best Improvement Node Shift Local Search, BINSLS）及び即時移動戦略に基づく NodeShift 局所探索法（First Improvement Node Shift Local Search, FINSLS）を設計し，medianshift 操作による局所探索法として最良移動戦略に基づく MedianShift 局所探索法（Best Improvement Median Shift Local Search, BIMSLS）及び即時移動戦略に基づく MedianShift 局所探索法（First Improvement Median Shift Local Search, FIMSLS）を設計した．

以下に 2 種類の移動戦略及び 2 種類の shift 操作に基づく 4 種類の局所探索法の疑似コードを示す．図 5 で示す BINSLS はノード集合 N と開設施設集合 M_o から解の改善量 g が最も大きくなるようなノード i と施設 j の組合せを求め nodeshift 操作を行う．図 6 で示す FINSLS はノード集合 N と開設施設集合 M_o からランダムにノード i と施設 j を選択していき，解が改善される i と j の組合せの内，初めて見つかったものを基に nodeshift 操作を行う．図 7 で示す BIMSLS は開設施設集合 M_o と閉鎖施設集合 M_c から改善量 g が最も大きくなる開設施設 i と閉鎖施設 j の組合せを求め medianshift 操作を行う．図 8 で示す FIMSLS は開設施設集合 M_o と閉鎖施設集合 M_c からランダムに開設施設 i と閉鎖施設 j を選択していき，解が改善される i と j の組合せの内，初めて見つかったものを基に medianshift 操作を行う．各局所探索法は容量制約条件を満たす範囲で shift 操作を行い，以上の処理をそれぞれ解の改善がなされなくなるまで繰り返す．

```
    Best Improvement Node Shift Local Search(s)
    begin
 1      repeat
 2          g := 0, g_best := 0;
 3          for all i ∈ N do
 4              for all j ∈ M_o do
 5                  if TotalDemand(j) + Demand(i) ≤ Capacity(j) then
 6                      g := NodeShiftGain(i, j, s);
 7                      if g > g_best then
 8                          g_best := g, shift_i := i, shift_j := j;
 9                      endif
10                  endif
11              endfor
12          endfor
13          if g_best > 0 then s := NodeShiftMove(shift_i, shift_j, s); endif
14      until g_best ≤ 0
15      return s;
    end;
```

図 5: BINSLS の擬似コード

```
   First Improvement Node Shift Local Search(s)
   begin
1    repeat
2      g := 0;
3      for all i ∈ M_o do
4        for all j ∈ M_c do
5          if TotalDemand(j) + Demand(i) ≤ Capacity(j) then
6            g := NodeShiftGain(i, j, s);
7            if g > 0 then s := NodeShiftMove(i, j, s) break ; endif
8          endif
9        endfor
10       if g > 0 then break ; endif
11     endfor
12   until g ≤ 0
13   return s;
   end;
```

図 6: FINSLS の擬似コード

```
   Best Improvement Median Shift Local Search(s)
   begin
1    repeat
2      g := 0, g_best := 0;
3      for all i ∈ M_o do
4        for all j ∈ M_c do
5          if TotalDemand(i) ≤ Capacity(j) then
6            g := MedianShiftGain(i, j, s);
7            if g_best < g then
8              g_best := g, shift_i := i, shift_j := j;
9            endif
10         endif
11       endfor
12     endfor
13     if g_best > 0 then s := MedianShiftMove(shift_i, shift_j, s); endif
14   until g_best ≤ 0
15   return s;
   end;
```

図 7: BIMSLS の擬似コード

```
   First Improvement Median Shift Local Search(s)
   begin
1    repeat
2      g := 0;
3      for all i ∈ M_o do
4        for all j ∈ M_c do
5          if TotalDemand(i) ≤ Capacity(j) then
6            g := MedianShiftGain(i, j, s);
7            if g > 0 then s := MedianShiftMove(i, j, s) break ; endif
8          endif
9        endfor
10       if g > 0 then break ; endif
11     endfor
12   until g ≤ 0
13   return s;
   end;
```

図 8: FIMSLS の擬似コード

図 9 に CPMP に対する局所探索法の疑似コードを示す．本研究における局所探索法は nodeshift 操作による局所探索法（Line 4）と medianshift 操作による局所探索法（Line 7）を交互に切り替えながら（Line 9）繰り返すことで探索を行う．そして 2 種類の局所探索法の両方で解の改善がなされなくなったとき処理を終了する（Line 10）．本研究では Line 4 の局所探索法として nodeshift 操作による BINSLS または FINSLS を，Line 7 の局所探索法として medianshift 操作による BIMSLS または FIMSLS を用いる．

```
    Local Search For CPMP(s)
    begin
 1    nodeshift := true, medianshift := false;
 2    repeat
 3      if nodeshift = true then
 4        NodeShiftLocalSearch(s);
 5      endif
 6      if medianshift = true then
 7        MedianShiftLocalSearch(s);
 8      endif
 9      nodeshift :=!nodeshift, medianshift :=!medianshift;
10    until not improved by both nodeshift and medianshift
11    return s;
    end;
```

図 9: CPMP に対する局所探索法の擬似コード

上述した通り本研究における局所探索法は NodeShift 局所探索法と MedianShift 局所探索法の 2 つの局所探索法からなるため，以下の 4 種類の組合せが考えられる．

- BINSLS と BIMSLS を行う局所探索法（BINSLS_BIMSLS）
- BINSLS と FIMSLS を行う局所探索法（BINSLS_FIMSLS）
- FINSLS と BIMSLS を行う局所探索法（FINSLS_BIMSLS）
- FINSLS と FIMSLS を行う局所探索法（FINSLS_FIMSLS）

5 実験結果

各局所探索法の探索性能を評価するために基本的な解探索手法である多スタート局所探索法（Multi-start LocalSearch, MLS）の枠組みの下で実験を行った．各局所探索法で探索する初期解として第 3 章と同様の初期解生成方式及び修正法の以下の 4 種類の組合せによる解を用いた．

- ランダム初期解生成方式+sRepair
- ランダム初期解生成方式+dRepair
- 距離優先初期解生成方式+sRepair
- 距離優先初期解生成方式+dRepair

実験ではベンチマーク問題例として，取得可能な CPMP の問題例より，ノード数 100〜402 の 6 例題を使用した．各問題例ごとに 10 試行 MLS による探索を行い，それぞれの最良精度（Best）及び平均精度（Avg）を求めた．本実験では比較実験の公平性を保つため，全アルゴリズムの各試行において開設施設数 p 秒という計算打ち切り時間を設けた．なお得られた解 x の目的関数値の精度（%）は $\frac{f(x)-f(x^*)}{f(x^*)} \times 100$ で求めた．ただし，x^* は既知の最良解である．表 2〜5 に実験の結果を初期解ごとに示す．各表は左から順に問題例名，ノード数 n，開設施設数 p，それぞれの局所探索法ごとに得た解の精度の Best と Avg を示している．各問題例において Best と Avg ごとに最も精度の良いものを太字で示した．結果から多くの問題例で dRepair によって修正した解を初期解とした FINSLS_FIMSLS によって得た解の精度が良好であった．特に距離優先初期解生成方式に dRepair を適用した初期解に対して FINSLS_FIMSLS による探索を行い得た解の精度が平均的に良好であることを観測した．以上の結果から，施設とノードの距離を考慮した初期解に対してより多様な解の探索を行える FINSLS_FIMSLS による探索が有効であったと考えられる．

6 むすび

本論文では，CPMP に対する局所探索法を設計し，その探索性能の調査を行った．MLS の枠組みの下で各局所探索法の探索性能を比較した．結果から距離優先初期解生成方式によって生成した解を dRepair によって修正し得た初期解に対して即時移動戦略に基づく NodeShift 局所探索法及び即時移動戦略に基づく MedianShift 局所探索法を交互に行う FINSLS_FIMSLS によって得た解の精度が良好であることを観測した．これはノードと施設間の距離を考慮した初期解に対して即時移動戦略による幅の広い探索を行ったことが有効に働いたと考えられる．

表 2: MLS の結果：ランダム初期解生成方式+sRepair

	n	p	BINSLS_BIMSLS		BINSLS_FIMSLS		FINSLS_BIMSLS		FINSLS_FIMSLS	
			BEST	AVG	BEST	AVG	BEST	AVG	BEST	AVG
SJC1	100	10	0.29	1.28	0.79	1.63	**0.00**	0.93	**0.00**	0.85
SJC2	200	15	0.73	1.09	0.73	1.53	0.80	1.04	0.57	1.00
SJC3a	300	25	3.08	3.97	3.57	4.16	2.25	3.08	**1.49**	2.79
SJC3b	300	30	1.93	3.89	1.93	4.10	1.94	3.42	2.28	3.23
SJC4a	402	30	3.50	5.35	3.50	5.35	3.91	4.97	3.52	4.44
SJC4b	402	40	4.46	5.17	4.46	5.12	3.84	4.35	3.48	4.20
平均			2.33	3.46	2.50	3.65	2.13	2.97	1.89	2.75

表 3: MLS の結果：ランダム初期解生成方式+dRepair

	n	p	BINSLS_BIMSLS		BINSLS_FIMSLS		FINSLS_BIMSLS		FINSLS_FIMSLS	
			BEST	AVG	BEST	AVG	BEST	AVG	BEST	AVG
SJC1	100	10	0.79	1.15	0.79	1.12	0.32	0.98	0.32	**0.82**
SJC2	200	15	0.74	1.16	0.74	1.13	0.66	1.12	**0.43**	1.03
SJC3a	300	25	3.47	4.09	2.89	3.95	2.39	3.30	2.13	2.78
SJC3b	300	30	1.78	4.02	1.78	3.88	2.11	3.43	**1.73**	**3.02**
SJC4a	402	30	3.12	5.21	3.12	5.21	3.23	4.61	3.27	4.49
SJC4b	402	40	4.46	5.12	4.16	5.03	4.02	4.35	3.07	4.29
平均			2.39	3.46	2.25	3.39	2.12	2.96	**1.83**	2.74

表 4: MLS の結果：距離優先初期解生成方式+sRepair

	n	p	BINSLS_BIMSLS		BINSLS_FIMSLS		FINSLS_BIMSLS		FINSLS_FIMSLS	
			BEST	AVG	BEST	AVG	BEST	AVG	BEST	AVG
SJC1	100	10	0.58	1.06	0.58	0.99	0.44	0.88	0.44	0.88
SJC2	200	15	0.52	1.04	0.52	0.98	0.75	1.06	0.74	1.12
SJC3a	300	25	2.52	3.05	2.52	3.01	1.96	2.74	1.96	2.86
SJC3b	300	30	2.86	3.48	2.29	3.27	2.85	3.47	2.65	3.41
SJC4a	402	30	**2.72**	4.56	**2.72**	4.47	4.16	4.77	3.46	4.65
SJC4b	402	40	3.61	4.61	3.61	4.34	3.66	4.50	4.38	4.93
平均			2.13	2.97	2.04	2.84	2.30	2.90	2.27	2.98

表 5: MLS の結果：距離優先初期解生成方式+dRepair

	n	p	BINSLS_BIMSLS		BINSLS_FIMSLS		FINSLS_BIMSLS		FINSLS_FIMSLS	
			BEST	AVG	BEST	AVG	BEST	AVG	BEST	AVG
SJC1	100	10	0.77	1.06	0.77	0.95	0.79	1.02	0.29	0.84
SJC2	200	15	0.30	1.01	0.30	0.91	0.56	**0.90**	0.48	1.05
SJC3a	300	25	2.15	2.89	2.15	2.88	2.52	2.85	1.66	**2.28**
SJC3b	300	30	2.29	3.30	2.29	3.26	2.22	3.35	2.29	3.15
SJC4a	402	30	2.91	4.46	2.91	4.43	2.91	4.72	3.33	**4.31**
SJC4b	402	40	3.12	4.56	3.12	4.37	3.11	4.36	**2.93**	**4.12**
平均			1.92	2.88	1.92	2.80	2.02	2.87	**1.83**	**2.62**

7 謝辞

本研究の一部は JSPS 科研費（基盤研究（C）19K12166）の助成を受けたものである．

参考文献

[1] 柳浦睦憲, 茨木俊秀. 組合せ最適化—メタ戦略を中心として—. 朝倉書店, 2001.

[2] R. Whitaker. A fast algorithm for the greedy interchange of large-scale clustering and median location prolems. *INFOR*, Vol. 21, p. 95–108, 1983.

[3] Y. Kochetov, T. Levanova, E. Alekseeva, and M. Loresh. Large neighborhood local search for the p-median problem. *Yugoslav Journal of Operations Research*, Vol. 15, No. 1, pp. 53–63, 2005.

[4] M. G. C. Resende and R. F. Werneck. A fast swap-based local search procedure for location problems. *Annals of Operations Research*, Vol. 150, No. 1, pp. 205–230, 2007.

[5] 三宅孝史, 金原一歩, 岡野傑士, 片山謙吾. 容量制約付き p-メディアン問題に対する修正アルゴリズムの検討. 平成 30 年度（第 69 回）電気・情報関連 学会中国支部連合大会論文集, 2018.

Search Performance of Local Search Methods for the Capacitated p-Median Problem

Takafumi MIYAKE, Kazuho KANAHARA, Takeshi OKANO and Kengo KATAYAMA*

Graduate School of Engineering,
Okayama University of Science,
**Department of Information and Computer Engineering,*
Faculty of Engineering,
1-1 Ridai-cho, Kita-ku, Okayama, 700-0005, Japan

(Received October 31, 2019; accepted December 9, 2019)

Capacitated p-Median Problem (CPMP) is one of the important combinatorial optimization problems. Local Search methods are widely used to solve combinatorial optimization problems including CPMP. Currently, there are many researches on algorithms for uncapacitated p-Median Problem (PMP). However, research for CPMP is less than those of PMP. The algorithms for PMP cannot be applied directly for CPMP in performing the effective search. Therefore, in this paper, we design some Local Search methods for CPMP, and investigate its search performance.

Keywords: combinatorial optimization; capacitated p-median problem; local search.