岡山理科大学紀要　第55号 A pp. 25-34（2019）

反復局所探索法における探索の多様化に関する調査

岡野　傑士・金原　一歩・三宅　孝史・片山　謙吾

岡山理科大学大学院工学研究科情報工学専攻

(2019年10月31日受付、2019年12月9日受理)

1　はじめに

反復局所探索法 (Iterated Local Search, ILS) や memetic アルゴリズム，遺伝的アルゴリズムなどの多くのメタ戦略アルゴリズムでは，探索の集中化と多様化のバランスが重要である．局所探索を基本として探索の集中化を行い，局所探索で得た局所解の脱出や長期にわたる探索の停滞を防ぐため解に変化を与えて解を多様化する．例えばシンプルでありながら，高性能であることで知られている ILS は局所探索法と突然変異 (本研究では Kick と呼ぶ) を交互に繰り返す．局所探索法は近傍解集合から良好な近傍解への移動を繰り返し，改善解がなくなった場合，局所解に至ったとして探索を終了するアルゴリズムである．Kick は局所解から脱出し，再度局所探索を行えるよう解を変化させることが目的である．多様化が弱すぎると局所解から脱出できず探索が滞り，強すぎるとランダムな初期解からの探索と変わらなくなってしまう．そのため適度な多様化を行うことは ILS を設計するうえで重要な要素の 1 つといえる．

本論文では，2 次割当問題 (Quadratic Assignment Problem, QAP) を対象として，ILS における探索の多様化について調査する．LS や Kick，Kick の対象とする解による探索への影響を調査するため各 2 種類，計 8 種類の比較を行う．また，比較実験を通して，アルゴリズム改良の手がかりや特性に関する知見等の獲得を試みる．

2　2 次割当問題

本研究では，異なる 2 つのコスト行列から，複数存在する割当パターンのなかで，全体のコストが最小となる割当を求める問題を 2 次割当問題の定義とする．

n 個の点があり，各 2 点間の距離が $n \times n$ の行列 $\mathrm{W} = w_{i,j}$（$1 \leqq i, j \leqq n$）で与えられているとする．また，これとは別に n 個の各要素間の相互関係を表す $n \times n$ の行列 $\mathrm{F} = f_{i,j}$（$1 \leqq i, j \leqq n$）が与えられているとする．

2 次割当問題は式（1）で定義することができる．

$$C(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} f_{\pi(i)\pi(j)} \tag{1}$$

- n : 全頂点 (要素) の数
- π : n 個の要素 {1,. . . ,n} からなる順列で表される解
- w_{ij} : 頂点 i と頂点 j の間の距離
- $\pi(i)$: 頂点 i に割当られている要素
- $f_{\pi(i)\pi(j)}$: 頂点 i に割当られている要素から頂点 j に割当てられている要素間のコスト

QAP は式（1）を最小化する解 π を求める問題であり，問題サイズが n の場合, 解空間の大きさは $n!$である．

3　反復局所探索法

反復局所探索法 (Iterated Local Search, ILS) は，与えられた初期解に対し，局所探索法及び突然変異 (本研究では Kick と呼ぶ) を交互に繰り返し，探索を行うメタ戦略アルゴリズムである．

図 1 に ILS の処理手順を疑似コードで示す．まず Kick の強さ *kicksize* を設定し，ランダムな初期解を生成する (Line 1)．生成した初期解に対して局所探索法を実行し (Line 2)，得た局所解を π_{best} として保持する (Line 3)．その後 Kick により局所解を脱出し (Line 5)，脱出した解に対し局所探索法を実行する (Line 6)．局所探索法により得た局所解が π_{best} より改善していた場合，その解を π_{best} として保持する (Line 7)．この Kick 及び局所探索の処理を終了条件を満たすまで繰り返し (Line 8)，探索で得た最良解 π_{best} を出力し，探索を終了する (Line 9)．

図 1 では Kick に与えられる解 π は現在の解である．そのため π は探索空間上を動き続け，多様化しやすい．現在解の他にも Kick に与える解として探索中に得た最良解 π_{best} を与えることも考えられる．その場合，π_{best} の周辺に集中化した探索を行う．本研究では前者の図 1 のように Kick に現在解 π を与える ILS を ILS_R とし，後者のように Kick に

```
procedure Iterated Local Search
  begin
1   set the initial values of kicksize and generate a randomly solution π;
2   π := Local Search (π);
3   π_best := π;
4   repeat
5     π := Kick (π, kicksize);
6     π := Local Search (π);
7     if C(π) < C(π_best) then π_best := π; endif
8   until terminate = true;
9   return π_best;
  end;
```

図 1: 反復局所探索法

最良解 π_{best} を与える ILS を ILS_B とする．なお，図 1 の ILS は ILS_R を示しており，図 1 の Line 5 の前に現在解 π を最良解 π_{best} に変更する処理を入れることで ILS_B に変えることができる．

以下 Lines 2, 6 の局所探索法及び Line 5 の Kick について説明する．

3.1 局所探索法

局所探索法 (Local Search, LS) は，ある操作を行うことで生成可能な解の集合である近傍内に改善解が存在しなくなるまで近傍探索を繰り返す方法である [3].

本研究で ILS に導入する LS として，QAP で一般に利用される 2 つの配置場所の要素の交換を行う 2-opt 近傍操作により近傍を生成する 2-opt 局所探索法 (2-opt Local Search, 2LS) を使用する．問題サイズ n の問題例に対して，2-opt 近傍操作を用いて得られる近傍解の総数は $n(n-1)/2$ である．近傍操作によって生成された近傍解集合の中に，現在の解より良好な評価値を持つ近傍解は複数存在すると考えられる．その複数の改善解の中でどの解に移動するかの戦略として，最良移動戦略 (Best Improvement, BI) と即時移動戦略 (First Improvement, FI) が知られている [1]．本研究では，最良移動戦略に基づく 2-opt 局所探索法を BI2LS と示し，即時移動戦略に基づく 2-opt 局所探索法を FI2LS と示す [4].

3.1.1 最良移動戦略に基づく 2-opt 局所探索法

最良移動戦略 (Best Improvement, BI) は近傍解全てを生成し，評価値が最も良好な解への移動を行う．そのため deterministic な探索となり，与えられた初期解に強く依存する傾向が知られている．

図 2 に BI2LS の処理手順を疑似コードで示す．このアルゴリズムは近傍探索及び解の移動の処理からなる．π は現在の解，g は最良のゲイン値を保持するものである．まずゲイン値 g を初期化する (Line 2)．近傍探索の処理では，近傍解を生成し，現在の解から変化量 $\delta_{i,j}$ を求める (Line 5)．その結果，現在保持しているゲイン値 g より良好な変化量の解を発見した場合 (Line 6)，その変化量を保持し (Line 7)，その解に移動するために交換する要素 i, j を $swap\_i$, $swap\_j$ として保持する (Line 8)．この近傍探索の処理を $n(n-1)/2$ 個の近傍全てを探索するまで繰り返す (Lines 3〜11)．近傍探索の結果，保持しているゲイン値 g が改善していた場合，保持している要素 $swap\_i$, $swap\_j$ の交換により最良の近傍解へ移動する (Line 12)．以上の処理を近傍内に改善解がなくなる (近傍探索の結果，$g \leq 0$ を満たした) 場合，局所解に至ったとして現在の解 π を出力し，探索を終了する．

3.1.2 即時移動戦略に基づく 2-opt 局所探索法

即時移動戦略 (First Improvement, FI) は事前に定められた順番で近傍を生成する過程で，評価値が良好な近傍解を発見した場合にその解へすぐさま移動する戦略である．本研究ではランダムな順番で近傍を生成する．そのためランダムな探索となり，与えられた初期解にあまり依存しない．図 3 に FI2LS の処理手順を疑似コードで示す．このアルゴリズムは近傍探索及び解の移動の処理からなる．解の改善量 g，基準として選択可能な解の配置場所の集合 P_{out} を初期化する (Line 2)．まず，P_{out} からランダムに選ばれた配置場所 i を基準として選択し (Line 4)，P_{out} から選んだ配置場所 i を取り除く (Line 5)．解の要素集合 P_{in} を基準配置場所 i を取り除いた P_{ot} で初期化する (Line 6)．これにより，P_{in} は基準 i の対として選択できる配置場所の集合となる．次に基準とした配置場所 i とランダムな配置場所 j の交換操作に

```
procedure Best Improvement 2-opt Local Search (π)
    begin
 1    repeat
 2        g := 0;
 3        for i := 1 to n − 1
 4            for j := i + 1 to n
 5                δ_{i,j} := SwapGain(i, j, π);
 6                if δ_{i,j} > g then
 7                    g := δ_{i,j};
 8                    swap_i := i, swap_j := j;
 9                endif
10            endfor
11        endfor
12        if g > 0 then
              π := SwapMove(swap_i, swap_j, π);
          endif
13    until g ≤ 0
14    return π;
    end;
```

図 2: 最良移動戦略に基づく 2-opt 局所探索法

```
procedure First Improvement 2-opt Local Search (π)
    begin
 1    repeat
 2        g := 0, P_out := {1, . . . , n};
 3        repeat
 4            select i ∈ P_out randomly;
 5            P_out := P_out\{i};
 6            P_in := P_out;
 7            repeat
 8                select j ∈ P_in randomly;
 9                P_in := P_in\{j};
10                δ_{i,j} := SwapGain(i, j, π);
11                if δ_{i,j} > 0 then
12                    π := SwapMove(i, j, π);
13                    g := g + δ_{i,j};
14                endif
15            until P_in = ∅;
16        until P_out = ∅;
17    until g ≤ 0;
18    return π;
    end;
```

図 3: 即時移動戦略に基づく 2-opt 局所探索法

より現在解 π より改善するか確認する (Lines 8〜11)．改善される場合，要素値 $\pi(i), \pi(j)$ の交換を行い (Line 12)，その交換により得られる改善量を保持する (Line 13)．この処理を P_{in} が空集合になる (ほかの配置場所全てに伴う近傍探索を行う) まで繰り返す (Lines 7〜15)．同様に上述の一連の処理を P_{out} が空集合になるまで繰り返す (Lines 3〜16)．以上の処理を改善解がなくなるまで近傍を探索する (Lines 2〜17) ことで局所解に達する．

3.2 Kick

Kick は局所探索により得た局所解から脱出することを目的とする探索の多様化のための処理である．

本研究では，図 1 Line 5 の Kick 処理として 2 種類の Kick（BasePointKick と NonBasePointKick）[5] を使用する．本研究における Kick では，2 つの要素を交換する操作を *kicksize* 個の要素を交換するまで繰り返し，局所解を脱出する．

2 種類の Kick は交換する要素の選択基準の違いがある．各 Kick を以下に簡潔に示す．

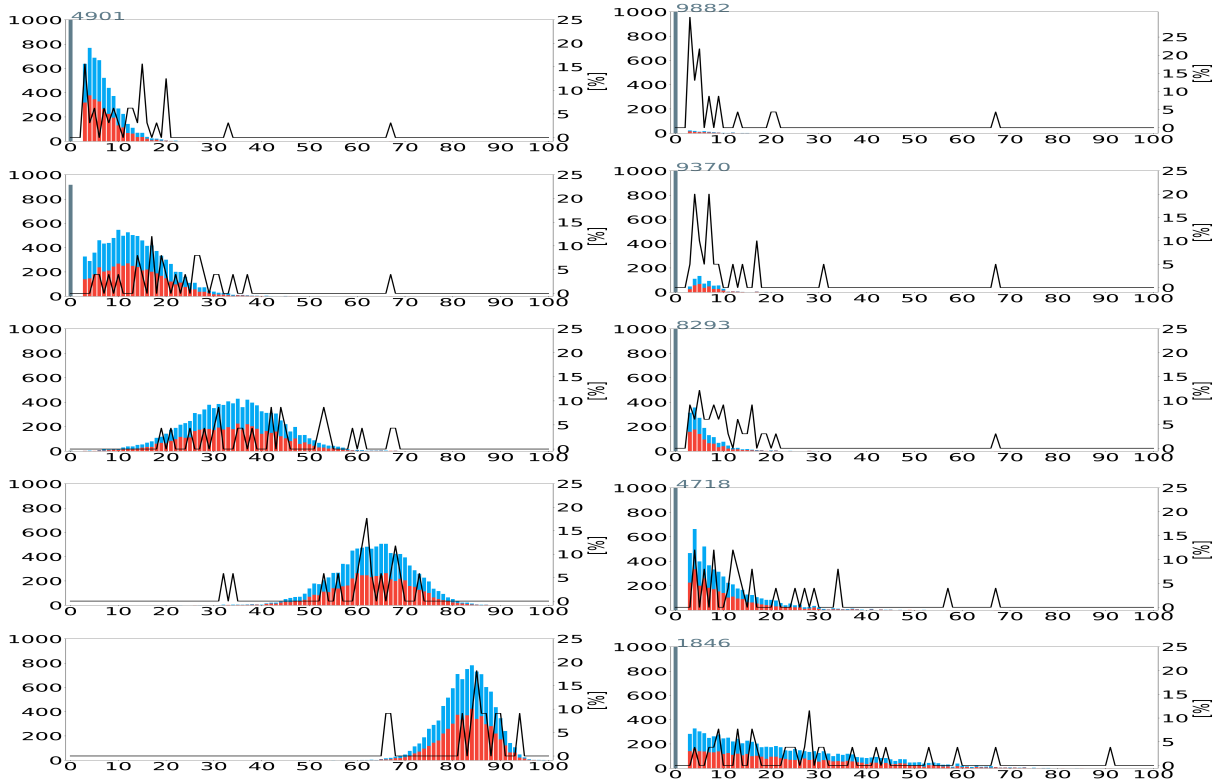
BasePointKick: 解 π に対しランダムな要素を基点とし，*kicksize*−1 回，重複しないランダムな他の要素と交換する．

NonBasePointKick: 解 π において重複しないランダムな 2 つの要素の交換を *kicksize*/2 回行う．

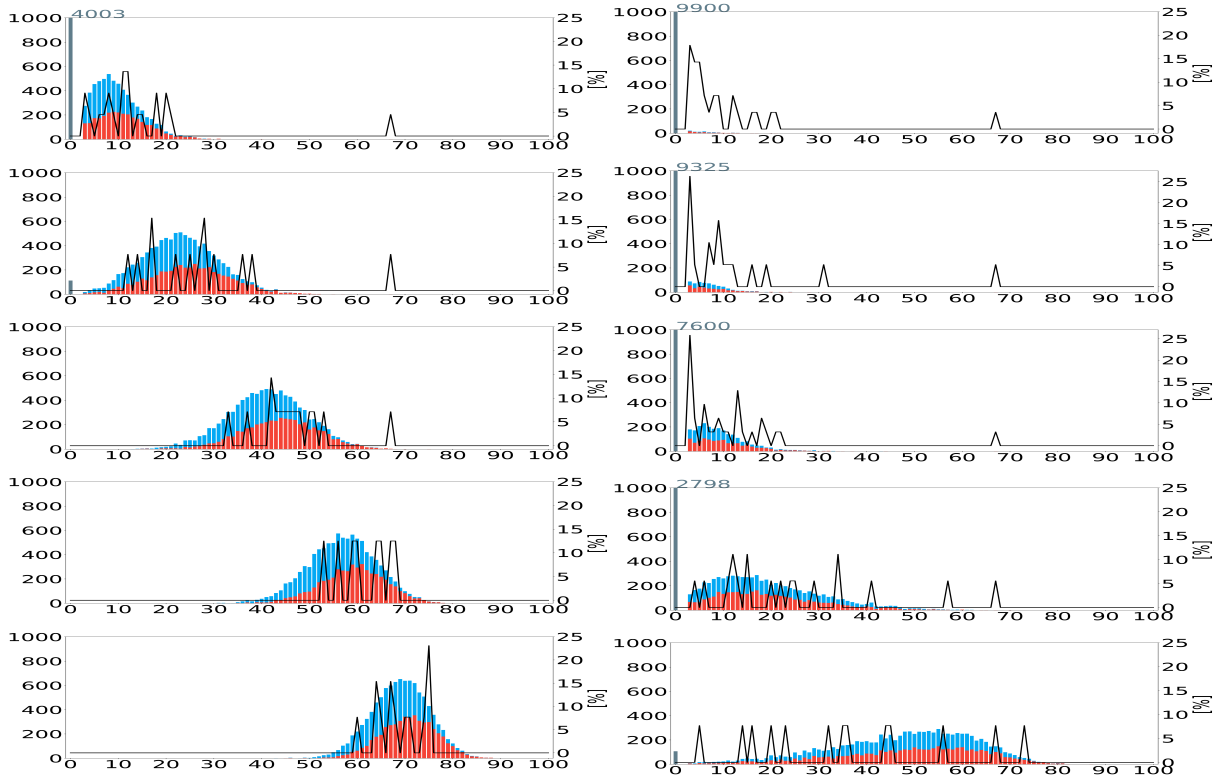
4 実験結果

上述した ILS の多様化に関する調査を行うために実験を行った．比較対象とするアルゴリズムは ILS_B, ILS_R に LS として BI2LS または FI2LS を，Kick として BasepointKick または NonBasepointKick をそれぞれ導入したアルゴリズム 8 種類である．各アルゴリズムはC++言語によりコード化し，コンパイラはオプション-O3 を付与した g++(Ver9.1.0) を使用した．計算機は CPU: Intel Xeon 3.60GHz，RAM: 8GB 上で実行した．対象とする問題例は QAP のベンチマーク問題例集である QAPLIB より取得した特徴 [2] の違う問題サイズが 100 の 3 例題 (tai100a, tai100b, wil100) とし，10000 回 Kick 及び LS を実行した結果を算出した．

図 4〜7 に探索アルゴリズムの違いによる多様化の度合いを示す．各図は各アルゴリズムを tai100a 及び tai100b に対して実行した結果の局所解同士の距離のグラフであり，上から *kicksize* が 10, 20, 30, 40, 50 の結果である．各グラフの横軸は Kick 前の局所解と Kick 及び LS 後の局所解の距離を表しており，左の縦軸が積み上げ棒グラフの軸で，算出回数を示し，右の縦軸が折れ線グラフの軸で，解の改善が起こった割合を示している．また棒グラフの赤色は直前の局所解より改善した回数を示し，青色は改悪した回数を示しており，灰色は同じコストとなった回数を示している．また目視で比較しやすいように棒グラフの上限を 1000 としており，出現回数が 1000 回を大きく超えた場合は，同色で回数を表記している．

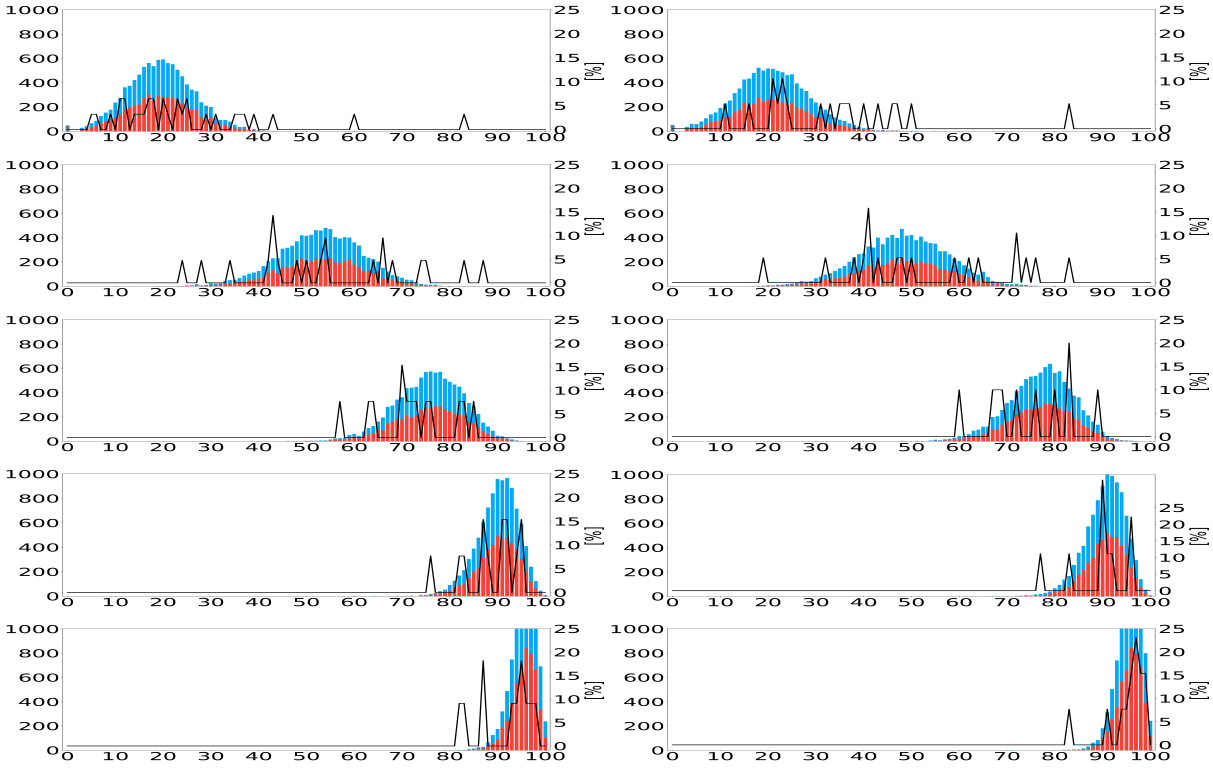


ILS_B に BasepointKick を使用した結果　　ILS_B に NonBasepointKick を使用した結果

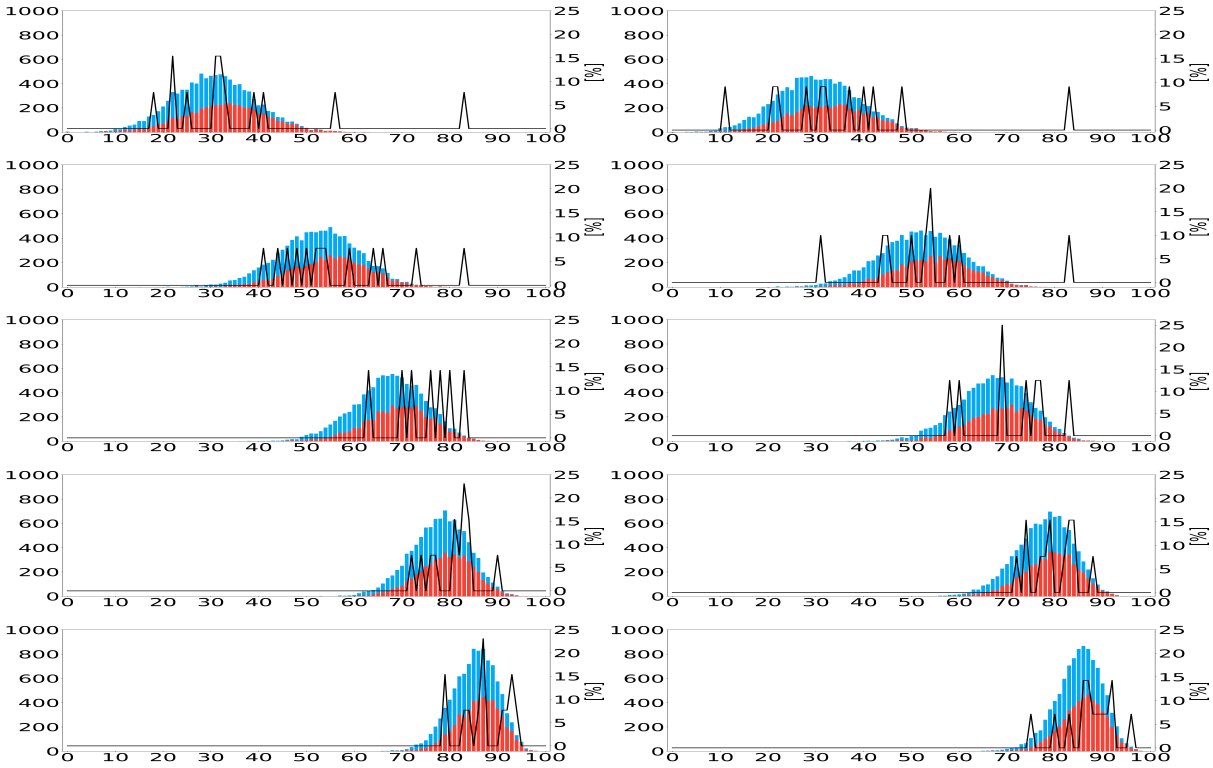


ILS_R に BasepointKick を使用した結果　　ILS_R に NonBasepointKick を使用した結果

図 4: BI2LS を使用した各アルゴリズムの tai100a に対する結果

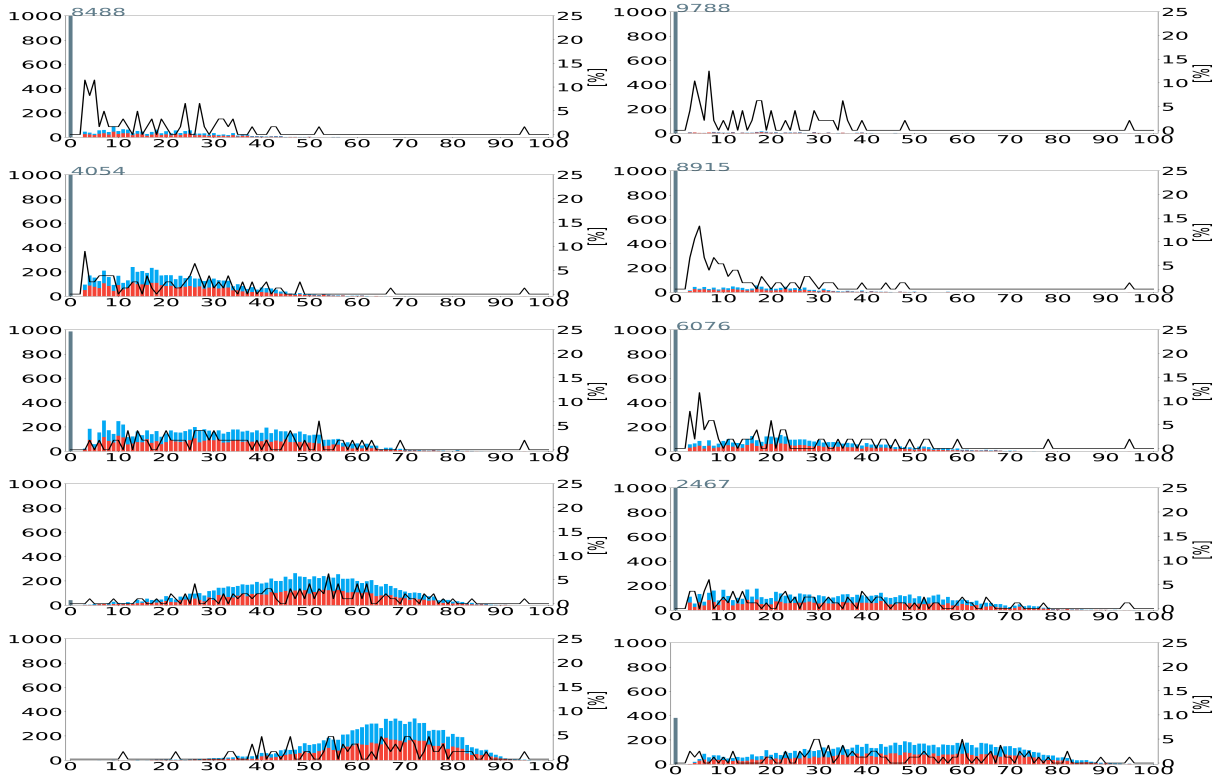


ILS_B に BasepointKick を使用した結果　　ILS_B に NonBasepointKick を使用した結果

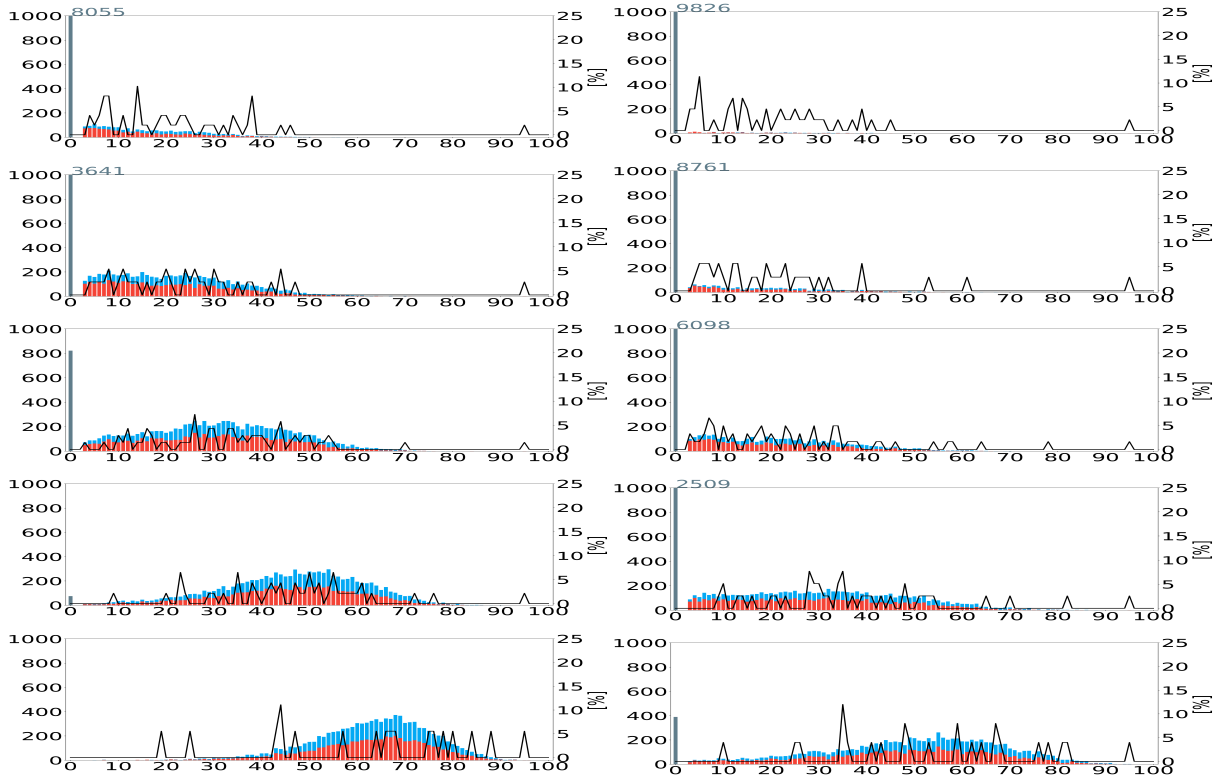


ILS_R に BasepointKick を使用した結果　　ILS_R に NonBasepointKick を使用した結果

図 5: FI2LS を使用した各アルゴリズムの tai100a に対する結果

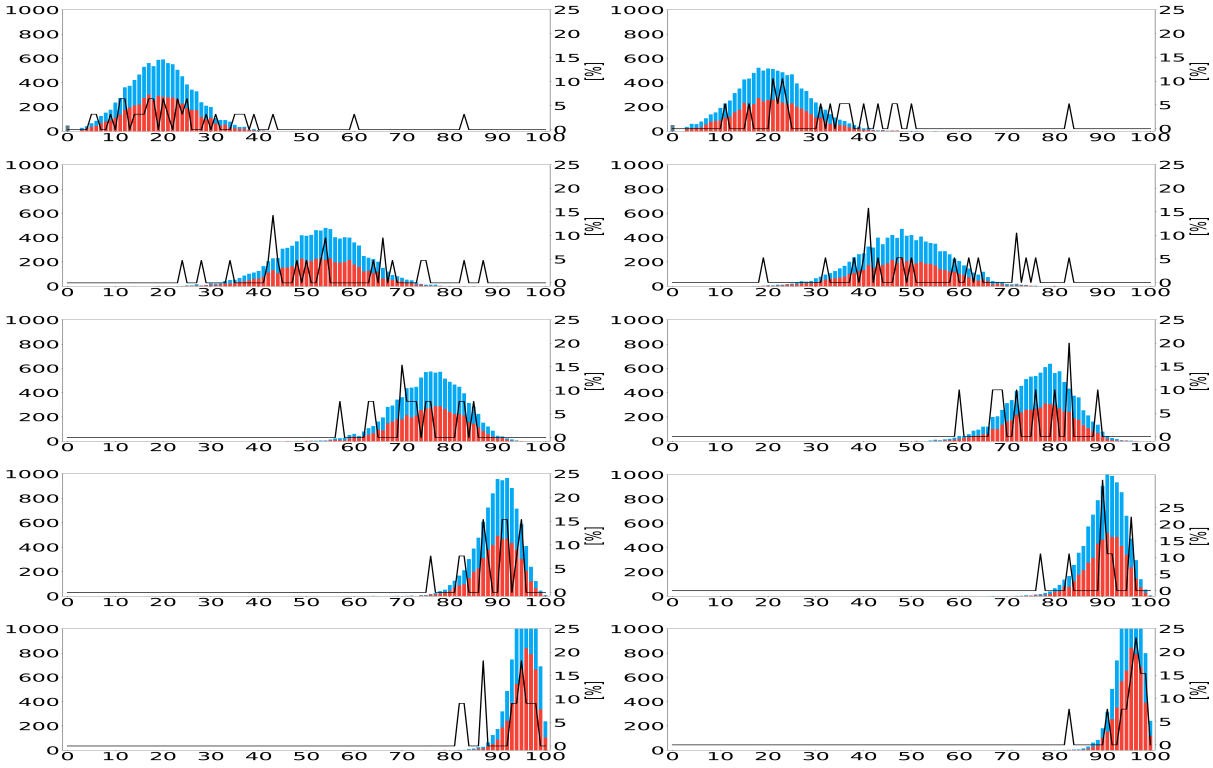


ILS_B に BasepointKick を使用した結果　　ILS_B に NonBasepointKick を使用した結果

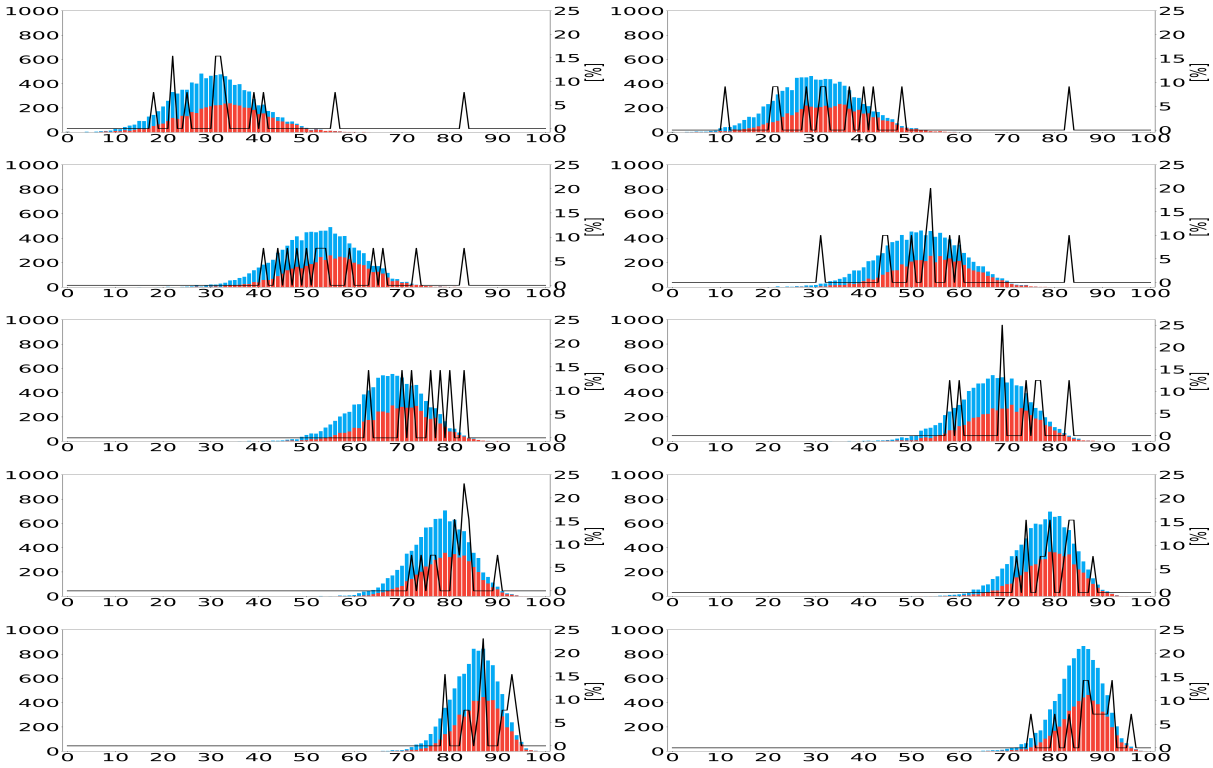


ILS_R に BasepointKick を使用した結果　　ILS_R に NonBasepointKick を使用した結果

図 6: BI2LS を使用した各アルゴリズムの tai100b に対する結果



ILS_B に BasepointKick を使用した結果　　ILS_B に NonBasepointKick を使用した結果



ILS_R に BasepointKick を使用した結果　　ILS_R に NonBasepointKick を使用した結果

図 7: FI2LS を使用した各アルゴリズムの tai100b に対する結果

表 1: 各アルゴリズムの最良，平均及び *kicksize*10〜50 の結果

Instance	ILS_B BI2LS BasepointKick								ILS_B BI2LS NonBasepointKick							
	best_size	best [%]	avg [%]	10	20	30	40	50	best_size	best	avg	10	20	30	40	50
tai100a	25	1.095	1.737	1.768	1.592	1.437	1.631	2.076	42	1.061	1.924	2.424	2.048	1.753	1.977	1.603
tai100b	46	0.000	1.966	2.061	1.742	2.787	0.342	0.382	48	0.081	2.075	2.526	2.080	2.613	0.383	0.360
wil100	42	0.005	0.196	0.344	0.504	0.181	0.103	0.131	48	0.064	0.378	0.535	0.207	0.463	0.087	0.196
Average	37.667	0.367	1.300	1.391	1.279	1.468	0.692	0.863	42.042	0.403	1.413	1.737	1.387	1.543	0.822	0.764

Instance	ILS_B FI2LS BasepointKick								ILS_B FI2LS NonBasepointKick							
	best_size	best [%]	avg [%]	10	20	30	40	50	best_size	best	avg	10	20	30	40	50
tai100a	15	1.116	1.785	1.276	1.709	1.885	2.127	2.335	12	1.271	1.833	1.665	1.525	1.839	2.251	2.350
tai100b	48	0.058	1.884	3.088	1.786	1.810	0.375	0.191	46	0.130	1.901	2.520	2.520	1.962	1.786	0.270
wil100	20	0.078	0.136	0.105	0.078	0.210	0.165	0.196	14	0.067	0.133	0.153	0.100	0.121	0.142	0.193
Average	30.167	0.405	1.277	1.465	1.213	1.343	0.840	0.896	28.510	0.468	1.320	1.519	1.383	1.366	1.250	0.894

Instance	ILS_R BI2LS BasepointKick								ILS_R BI2LS NonBasepointKick							
	best_size	best [%]	avg [%]	10	20	30	40	50	best_size	best	avg	10	20	30	40	50
tai100a	16	1.280	2.035	1.453	1.973	2.260	2.436	2.409	5	1.242	2.079	1.823	2.085	2.249	2.199	2.359
tai100b	14	0.000	0.825	1.745	1.769	0.062	0.092	0.115	24	0.027	0.331	0.044	0.115	0.223	0.313	0.326
wil100	9	0.053	0.108	0.162	0.073	0.081	0.122	0.172	7	0.021	0.151	0.104	0.145	0.138	0.215	0.231
Average	16.878	0.450	1.072	1.220	1.300	0.942	0.975	0.898	13.219	0.435	0.908	0.798	0.911	0.888	0.926	0.954

Instance	ILS_R FI2LS BasepointKick								ILS_R FI2LS NonBasepointKick							
	best_size	best [%]	avg [%]	10	20	30	40	50	best_size	best	avg	10	20	30	40	50
tai100a	30	1.221	1.727	1.581	1.768	1.221	1.759	2.182	6	1.349	2.086	1.872	2.155	2.368	2.175	2.343
tai100b	34	0.008	1.176	2.236	2.862	0.035	0.073	0.110	42	0.073	0.289	0.114	0.091	0.101	0.130	0.265
wil100	28	0.019	0.183	0.230	0.116	0.167	0.086	0.108	8	0.053	0.135	0.071	0.109	0.146	0.185	0.241
Average	26.305	0.421	0.998	1.211	1.414	0.578	0.711	0.838	20.576	0.474	0.877	0.817	0.942	0.798	0.800	0.922

各結果より *kicksize* よりも Kick 前の局所解と Kick 及び LS 後の局所解の距離が大きく離れる場合があることがわかる．例えば図 6 における *kicksize*10 のとき，局所解同士の距離が 95 離れている．このように *kicksize* が小さいときも大きく離れることが多い．実験結果より 1 回目の Kick を行ったときに大きく離れていることを確認した．また *kicksize* が大きくなるにつれて局所解同士の距離は大きくなるものの同じ傾向を示すことを確認した．

両方の Kick において BI2LS を使用した場合，*kicksize* が小さい場合に高い確率で同じ局所解に戻ってしまうことがわかる．また NonBasepointKick を使用した場合 BasepointKick を使用した場合より同じ局所解に戻ってしまうこともわかる．これは deterministic な探索となる BI2LS では複雑な変化を与えたほうが違う局所解に至りやすく交換回数が少ない NonBasepointKick が同じ局所解に戻ってしまったと考えられる．また現在解に対して Kick を続ける ILS_R より最良解に対して Kick を続ける ILS_B が同じ局所解から脱出しきれないことが多いことを観測した．しかし違う局所解に至った場合，ILS_B がより離れること結果が得られた．ILS_R は多様化しやすいため深い局所解に嵌りにくく，逆に ILS_B は集中化しやすいため深い局所解に嵌り，局所解から脱出しきれないことが多く，脱出できた場合は大きく離れることになったと考えられる．これに対しランダム性のある探索を行う FI2LS では，Kick の違いによる影響が少ないことが確認できる．ただし BI2LS を使用した場合と同様に違う局所解に至った場合，ILS_B がより離れること傾向は同じである．しかしより集中化しやすい ILS_B の場合は *kicksize* が 10 のとき，直前の局所解へ戻ることがあるが，より多様化しやすい ILS_R の場合は *kicksize* が 10 の場合でも直前の局所解に戻ることは確認されなかった．問題例の違いによる影響は，BI2LS を使用した場合に大きく表れ，tai100b が tai100a より満遍なく探索が行えている．また wil100 でも tai100b と同様の傾向を観測した．

表 1 に各アルゴリズム実験結果を示す．“best [%]”及び“avg [%]”は，既知の最良解のコストからの最良誤差率及び *kicksize* が 4〜50 までの各実行結果の平均誤差率である．best_size は best の解を算出した *kicksize* であり，10, 20, 30, 40, 50 は各 *kicksize* における算出した解の誤差を表している．なお，解の評価値の誤差率は式 (2) によって算出した．式 (2) 中の $C(\pi)$ は得られた解 π の評価値，$C(\pi^*)$ は最適解 (既知の最良解) の評価値を表している．

$$\text{解の評価値の誤差率 } (\%) = \frac{C(\pi) - C(\pi^*)}{C(\pi^*)} \times 100 \tag{2}$$

表 1 より，問題例ごとに最良の結果を算出した *kicksize* が違うことがわかる．また LS や Kick，Kick の対象によっても差異があることが確認できる．例えば ILS_B に BI2LS 及び BasepointKick を使用した tai100a や wil100 の場合，best 解を算出した *kicksize* に近い場合，良好な解を算出しやすいが，tai100b の場合は一概には言えない．この傾向は問題例や探索方法に依存しているとは言えない．このことから適切な *kicksize* は問題例や探索方法により一意に決定せず，探索の状況なども鑑みなければならないと考えられる．

5 むすび

組合せ最適化問題に対する多くのメタ戦略アルゴリズムでは探索の集中化と多様化のバランスが重要である．本論文では，代表的な組合せ最適化問題である 2 次割当問題を対象として，反復局所探索法を用い探索アルゴリズムの違いによる多様化の度合及び精度を調査した．比較実験の結果，適切な多様化の度合いは問題例や探索アルゴリズムによって一意に決定せず，探索の状況などに応じて変化すると考えられる．

6 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，実施したものである．

参考文献

[1] P. Hansen and N. Mladenović. First vs. best improvement: An empirical study. *Discrete Applied Mathematics*, Vol. 154, No. 5, pp. 802 – 817, 2006.

[2] É. D. Taillard. Comparison of iterative searches for the quadratic assignment problem. *Location Science*, Vol. 3, No. 2, pp. 87–105, 1995.

[3] 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.

[4] 岡野傑士, 金原一歩, 片山謙吾. 2 次割当問題に対する局所探索法の結合アルゴリズムの特性. 岡山理科大学紀要. A, Vol. 54, pp. 27–36, 2018.

[5] 岡野傑士, 片山謙吾, 金原一歩, 三宅孝史. 2 次割当問題に対する事前選択式可変近傍探索法. 令和元年度（第 70 回）電気・情報関連 学会中国支部連合大会論文集, pp. R19–23–01–01, 2019.

Performance Study of Diversification in Iterated Local Search for the Quadratic Assignment Problem

Takeshi OKANO, Kazuho KANAHARA and Takafumi MIYAKE and Kengo KATAYAMA
Graduate School of Engineering,
Okayama University of Science,
1-1 Ridai-cho, Kita-ku, Okayama, 700-0005, Japan

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

Balancing search between intensification and diversification is important for metaheuristic algorithms such as iterated local search (ILS), memetic algorithm and genetic algorithm. Intensification means to focus on the search in a local region by exploiting the information that a current good solution is found in this region. Diversification means to generate diverse solutions so as to explore the search space on the global scale. For example, ILS, which is known for its simple but high performance, alternates between local search and mutation (called Kick in this paper). Local search method repeats to select a better neighboring solution within a predefined neighborhood. The role of the local search method is to obtain a local optimal solution. The purpose of Kick is to change the solution so that it can escape from the local optimal solution. If the diversification is too weak, the search cannot be escaped from the current local region. If it is too strong, it is too far from the current local region. In this paper, we investigate degree of diversification in ILS for the quadratic assignment problem.

Keywords: combinatorial optimization; quadratic assignment problem; iterated local search.