CA-005

即時移動戦略に基づく k-opt 局所探索法
A k-opt Local Search based on First Improvement Move Strategy

岡野 傑士[1)]　片山 謙吾[1)]　金原 一歩[1)]　三宅 孝史[1)]　西原 典孝[1)]
Takeshi Okano　Kengo Katayama　Kazuho Kanahara　Takafumi Miyake　Noritaka Nishihara

1　まえがき

組合せ最適化問題に対する最も重要な基本的解法として局所探索法 (Local Search, LS) があげられる [16]．LS は予め定義された近傍操作によって生成される近傍解集合から，目的関数値を改善する良好な近傍解への移動を繰り返し，近傍内に良好な近傍解が存在しなくなった時に探索を終了する解改善法である．

代表的な LS として 1970 年代にグラフ分割問題 (Graph Partitioning Problem, GPP) [6] と巡回セールスマン問題 (Traveling Salesman Problem, TSP) [8] に対する Lin と Kernighan による LS が知られている．このアルゴリズムは単純な近傍操作を連鎖的に適用することで得られる解集合を改めて大きな近傍として捉える局所探索の一般化のアイデアであり，可変深度探索法 (Variable Depth Search) や可変 k-opt 局所探索法 (k-opt Local Search, KLS) などと呼ばれている．その後 1992 年に 2 次割当問題 (Quadratic Assignment Problem, QAP) に対して Murthy ら [11] によって応用され，1990 年半ばには Racer ら [13] や Yagiura ら [15] によって一般化割当問題に対して適用された．また 2000 年に TSP に対して Helsgaun によって変形法 [3] が提案され，2000 年前半にはバイナリー 2 次計画問題 (Binary Quadratic Programming, BQP) に対して Merz らによって示され [9]，片山らによって変形法が提案されている [5]．その後，最大クリーク問題 [4] や p-メディアン問題 [7] に応用され，近年でもナーススケジューリング問題 [1] や運搬車スケジューリング問題 [12] に適用されており，盛んに研究が行われている．

例えば GPP や QAP, BQP に対する KLS [6, 11, 9] は最良移動戦略の考えに基づき，単純な近傍操作を評価値が最も改善する解へ連鎖的に適用する．これらの最良移動戦略に基づく KLS は 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．これを緩和する一つの工夫として，BQP に対する片山らの変形 KLS [5] は，Merz らの最良移動戦略に基づく KLS [9] にランダム性を有する即時移動戦略に基づいた単純な近傍操作を一時的に導入している．探索の初めに即時移動戦略を取り入れることで，最良移動戦略に基づく KLS の探索の開始地点をランダムに方向付けて初期解への依存を軽減し，Merz らの KLS より平均的に良好な結果が得られることが示されている．

上述のように KLS は最良移動に基づく単純な近傍操作を連鎖的に適用することが主流であるが，与えられた初期解に依存する傾向が強いという問題点がある．本論文ではランダム性を有する即時移動戦略の考えに基づく KLS アルゴリズムとして，即時移動戦略に基づく k-opt 局所探索法 (First Improvement k-opt Local Search, FIKLS) を提案する．また提案法を QAP を対象として評価を行い，従来の局所探索アルゴリズムとの比較実験によりその有効性を示す．

1)　岡山理科大学 Okayama University of Science

```
 procedure Local Search (π)
 begin
1     repeat
2         g = 0;
3         π, g = Neighborhood Search (π, g);
4     until g > 0;
5     return π;
 end
```

図 1　QAP に対する局所探索法

2　2 次割当問題

2 次割当問題 (Quadratic Assignment Problem, QAP) の目的は，n 個の要素とこれらの要素を配置する n 個の配置場所が与えられたとき，以下の目的関数 (1) を最小化する要素配置 π を求めることである．

$$C(\pi) = \sum_{s=1}^{n}\sum_{t=1}^{n} w_{st} f_{\pi(s)\pi(t)} \tag{1}$$

式 (1) 中の s と $t(1 \le s,t \le n)$ は配置場所，$\pi(s)$ と $\pi(t)$ は配置場所 s と t にそれぞれ配置される各要素，w_{st} は配置場所 s,t 間のコスト，$f_{\pi(s)\pi(t)}$ は要素 $\pi(s),\pi(t)$ 間のコストを表している．要素 i 及びその要素 i が配置される場所 j が与えられたとき，$\pi(j) = i$ となる．

QAP は実用上重要な問題を多く有する組合せ最適化問題のひとつで，施設配置や電子回路上の部品の配置，集積回路上のトランジスタ数の最小化，タッチスクリーンデバイスのキーボードのキー配置など様々な応用例が挙げられる [10]．QAP は NP-困難 [2] であり，厳密解法では大規模な問題例の場合に膨大な時間を必要とする．したがって，実用的な時間内に優れた近似解を求めることができる近似解法が注目されている．

3　QAP に対する従来の局所探索法

局所探索法 (Local Search, LS) は予め定義された近傍操作を繰り返し，良好な近傍解が存在しなくなった時に局所解に至ったとして探索を終了する [16]．

図 1 に LS の処理手順の疑似コードを示す．Line 3 の近傍探索において得た解と近傍探索前の解との解の変化量を示す値であるゲイン値 g が 0 以下の場合，その局所解を出力して探索を終了する (Line 5)．

QAP に対する LS では一般的に 2-opt 局所探索法 (2-opt Local Search, 2LS) が利用され，2LS では Line 3 の近傍探索に 2-opt 近傍操作を適用する．2-opt 近傍操作は 2 つの配置場所の要素を入れ替える操作である．近傍操作によって生成される近傍内には良好な近傍解は複数存在すると考えられる．その複数の良好な近傍解の中でどの近傍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) の 2 種類の移動戦略が主流である．最良移動戦略は現在の解から生成可能な近傍の中で，評価値が最も良好である解に移動する方法である．即時移動戦略は近傍を生成する過程で，最初に見つかった評価値が良好な近傍解に移動する方法である．

本論文では，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．

```
procedure Best Improvement 2-opt Neighborhood Search (π, g)
begin
1    P := {1, ..., n};
2    find a pair of element i and j
                with max_{i,j∈P}, δ_{i,j} := SwapGain(i, j, π);
3    if δ_{i,j} > 0 then
4        π := SwapMove(i, j, π);
5        g = g + δ_{i,j};
6    endif
7    return π, g;
end
```

図 2　QAP に対する最良移動戦略に基づく 2-opt 近傍探索

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

図 3　QAP に対する即時移動戦略に基づく 2-opt 近傍探索

3.1　最良移動戦略に基づく 2-opt 局所探索法

BI2LS は，現在の解から生成可能な近傍全てを確認し，その中で最良の近傍解へ移動する最良移動戦略に基づく 2-opt 近傍探索を繰り返す LS である．

図 2 に最良移動戦略に基づく 2-opt 近傍探索の処理手順の疑似コードを示す．まず選択可能な解の要素集合 P を初期化する (Line 1)．2 つの要素 i, j $(i \neq j)$ を交換した場合の変化量 $\delta_{i,j}$ が最大となるように P から 2 つの要素 i, j を見つける (Line 2)．このとき P は n 個の要素で形成されているため，i, j の組み合わせ (近傍解の総数) は $n(n-1)/2$ 個になる．解が改善する場合，要素 i, j を交換 (Line 4) し，解のゲイン値 g を更新する (Line 5)．最後に解 π とゲイン値 g を出力し，終了する (Line 7)．

BI2LS は常に最良の解へ移動するため，deterministic な探索になり，初期解に強く依存する傾向がある．

3.2　即時移動戦略に基づく 2-opt 局所探索法

FI2LS は，近傍を生成する過程で見つかった改善解へ即時に移動する即時移動戦略に基づく 2-opt 近傍探索を繰り返す LS である．

図 3 に即時移動戦略に基づく 2-opt 近傍探索の処理手順の疑似コードを示す．まず基準として選択可能な要素の集合 P_{out} を初期化する (Line 1)．P_{out} からランダムに要素 i を基準として選択し，P_{out} から選んだ要素 i を取り除く (Line 3)．また対として選択可能な要素集合 P_{in} を P_{out} で初期化する (Line 4)．これにより，集合 P_{in} は集合 P_{out} から選択した基準要素 i の対として選択できる要素の集合となる．次に基準 i と P_{in} から選んだランダムな要素 j の交換操作により，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 i, j の交換を行い (Line 9)，ゲイン値 g を更新する (Line 10)．この処理を P_{in} 及び P_{out} が空集合になるまで繰り返す (Lines 2–13)．P_{in} 及び P_{out} が空になった場合，2 つの要素を入れ替えて生成される近傍解 $n(n-1)/2$ 個全てを確認したことになる．最後に解 π とゲイン値 g を出力し，終了する (Line 14)．このアルゴリズムは 1 度の移動による改善量は BI2LS と同等かそれより少なくなる．

FI2LS は見つかった改善解への移動を繰り返すため，ランダムな探索になり，初期解への依存は少ない．

3.3　k-opt 局所探索法

可変深度探索法 (Variable Depth Search) は単純な近傍操作を連鎖的に適用することで得られる解集合を改めて大きな近傍 (k-opt 近傍) として捉える局所探索の一般化のアイデアであり，可変 k-opt 局所探索法 (variable k-opt Local Search, KLS) とも呼ばれる．QAP に対しては，グラフ分割問題に対する KLS のアイデア [6] に基づいて 1992 年に Murthy らによって応用されている [11]．このアルゴリズムは単純な近傍操作として最良移動戦略に基づく 2-opt 近傍探索を連鎖的に適用しており，本論文では最良移動戦略に基づく k-opt 局所探索法 (Best Improvement k-opt Local Search, BIKLS) と呼ぶ．

図 4 に最良移動戦略に基づく k-opt 近傍探索の処理手順の疑似コード [1] を示す．π は現在解，π_{best} は探索中に見つかった最良解であり，g_{best} は π_{best} に対応したゲイン値である．まず，現在解 π を π_{prev} として保持し，要素集合 P を初期化する (Line 1)．P は k-opt 近傍探索において解のサイクリング [16] を防ぐためのものである．ループ処理 (Lines 2–8) によって単純な近傍操作を繰り返すことで連鎖的に近傍を生成し，k-opt 近傍を探索する．k-opt 近傍探索では毎回のループ処理において，2 つの要素が要素集合 P から選ばれるため，ループ処理は $n/2$ 回繰り返される．それと共に，2 つの要素を P から削除することで 1 度交換された要素は k-opt 近傍探索が終了するまで交換の対象にならない．ループ処理は単純な近傍操作として以下の 3 つの処理で構成される．

1) 制限付き 2-opt 近傍探索
2) 解の移動
3) 最良解の保持

以下，3 つの処理について説明する．**1) 制限付き 2-opt 近傍探索** (Line 3) として，BI2LS の疑似コードである図 2 の Line 2 と同じ処理を行う．しかし要素集合 P は図 4 Line 7 において交換した要素 i, j の削除が行われ，i, j の組み合わせはループ回数の増加に伴い最大 $n(n-1)/2$ 個から P の要素数に従い減っていく．そのため BI2LS と異なり，ループ回数が増えるにつれて対象となる要素は制限されることになる．**2) 解の移動**の処理 (Lines 4, 5) として，選ばれた要素 i, j の交換によって解を移動し (Line 4)，ゲイン値 g を更新する (Line 5)．**1)** と **2)** の処理を繰り返すことにより連鎖的な処理を実現している．連鎖的な処理における最良解を保持するために **3) 最良解の保持**の処理 (Line 6) を行う．**3)** の処理において，交換処理により得た解 π が現在までの k-opt 近傍探索において見つかった最良解 π_{best} よりも良好な解であった場合，その解を最良解 π_{best} として保持する．以上の近傍探索処理を制限付き 2-opt 近傍探索を行うための要素のペアが要素集合 P に存在しなくなるまで行うことで k-opt 近傍操作を行う (Lines 2–8)．k-opt 近傍探索において改善解を得た場合 (Line 9)，最良解 π_{best} と π_{best} のゲイン値 g_{best} を返す (Line 10)．改善解を得られなかっ

1）Murthy らのアルゴリズムでは各 k-opt 近傍処理において初めの単純な近傍操作処理 (Line 3) の結果，改善解がなければ探索を終了する処理が行われている．この処理は図 4 のアルゴリズムより若干高速になるが若干の精度低下を起こすことを我々は確認している．

```
   procedure Best Improvement k-opt Neighborhood Search (π, g)
   begin
 1     π_prev := π, g_best := -∞, P := {1, ..., n};
 2     repeat
 3     find a pair of element i and j
                 with max_{i,j∈P} δ_{i,j} := SwapGain(i, j, π);
 4         π := SwapMove(i, j, π);
 5         g = g + δ_{i,j};
 6         if g > g_best then g_best := g, π_best := π; endif
 7         P := P\{i, j};
 8     until |P| < 2;
 9     if g_best > 0 then
10         return π_best, g_best;
11     else
12         return π_prev, 0;
13     endif
   end
```

図 4　QAP に対する最良移動戦略に基づく k-opt 近傍探索

た ($g_{best} \leq 0$) 場合 (Line 11)，k-opt 局所解に到達したとみなし，最良の解である与えられた解 π_{prev} とゲイン値として 0 を返す (Line 12)．

4　QAP に対する新しい k-opt 局所探索法

上述した BIKLS では，単純な近傍操作として最良移動戦略に基づく 2-opt 近傍探索を連鎖的に適用する．最良移動戦略は deterministic な探索となるため，与えられた初期解に依存する傾向が強い．この問題を解決するために提案法では，即時移動戦略に基づく解の移動を連鎖的に行い，ランダムな方向への移動を繰り返す．これにより初期解の影響を大きく軽減できると考えられる．

提案法である即時移動戦略に基づく k-opt 局所探索法 (First Improvement k-opt Local Search, FIKLS) は，単純な近傍操作として解が改善する場合には即時移動戦略に基づき初めに見つかった改善解へ移動し，もし改善解がない場合には最良移動戦略に基づき最も改悪しない解への移動を行うことで，連鎖的な近傍操作を実現する．

図 5 に提案法の k-opt 近傍探索の疑似コードを示す．提案法のループ処理 (Lines 2–22) は BIKLS と同様に **1), 2), 3)** の処理によって構成される．提案法ではループ処理の **1) 制限付き 2-opt 近傍探索** (Lines 3–17) として，即時移動戦略と最良移動戦略を巧みに組み合わせた処理を行う．その処理として，FI2LS の処理に基づき近傍内に改善解がないときのみ最良移動する処理を行う．まず **1) 制限付き 2-opt 近傍探索**における FI2LS の処理について説明する．FI2LS と同様に P_{out} と P_{in} を初期化する (Lines 3–6)．FI2LS との違いとして BIKLS と同様に Line 21 により，要素集合 P はループ回数が増えるにつれて徐々に減っていき，対象となる要素は FI2LS と異なり制限される．またランダムに選んだ要素 i, j を交換して改善する場合 (Line 10)，要素 i, j を $swap\_i, swap\_j$ として保持し (Line 11)，P_{out} 及び P_{in} を空集合にする (Line 12) ことで制限付き 2-opt 近傍探索を終了し，**2) 解の移動**の処理へ移る (Line 18 に移動する)．次に **1) 制限付き 2-opt 近傍探索**における近傍内に改善解がないときのみ最良移動する処理について説明する．まず制限付き 2-opt 近傍操作中に見つけた最良の変化量として δ_{best} を初期化する (Line 3)．ランダムに選んだ要素 i, j を交換しても改善せず，δ_{best} より良好な i, j を交換した場合の改善量が見つかった ($0 \geq \delta_{i,j} > \delta_{best}$) 場合 (Line 13)，$\delta_{best}$ を更新し，i, j を $swap\_i, swap\_j$ として保持する (Line 14)．以上の 2 つの処理が組み合わさり提案法における **1) 制限付き 2-opt 近傍探索**となる．提案法 FIKLS と BIKLS との

```
   procedure First Improvement k-opt Neighborhood Search (π, g)
   begin
 1     π_prev := π, g_best := -∞, P := {1, ..., n};
 2     repeat
 3         P_out := P, δ_best := −∞;
 4         repeat
 5             select i ∈ P_out randomly, P_out := P_out\{i};
 6             P_in := P_out;
 7             repeat
 8                 select j ∈ P_in randomly, P_in := P_in\{j};
 9                 δ_{i,j} := SwapGain(i, j, π);
10                 if δ_{i,j} > 0 then
11                     swap_i := i, swap_j := j;
12                     P_out := ∅, P_in := ∅;
13                 else if δ_{i,j} > δ_best then
14                     δ_best := δ_{i,j}, swap_i := i, swap_j := j;
15                 endif
16             until P_in = ∅;
17         until P_out = ∅;
18         π := SwapMove(swap_i, swap_j, π);
19         g := g + δ_{swap_i,swap_j};
20         if g > g_best then g_best := g, π_best := π; endif
21         P := P\{swap_i, swap_j};
22     until |P| < 2;
23     if g_best > 0 then
24         return π_best, g_best;
25     else
26         return π_prev, 0;
27     endif
   end
```

図 5　QAP に対する即時移動戦略に基づく k-opt 近傍探索

違いは **1)** の処理だけであり，提案法の **1)** の処理 (Lines 3–17) は BIKLS の図 4 の Line 3 に相当する．

以上説明した提案法は連鎖的に適用する単純な近傍処理として，即時移動戦略に最良移動戦略を巧みに導入した制限付き 2-opt 近傍探索を繰り返し適用することで，k-opt 近傍解の探索を行うアルゴリズムである．

5　実験結果

上述した提案法 FIKLS の性能を評価するため，従来法の BI2LS, FI2LS, BIKLS と比較実験を行った．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++(Ver5.4.0) 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 45 例題とし，特徴により 4 つ (“type1”～“type4”) に分類した [14]．また，各アルゴリズムに対する終了条件として計算許容時間を 600sec と設定した．各アルゴリズムは問題例データの読み込み時間は含まず，ランダムな初期解の生成時間を含み，各問題例に対して各アルゴリズムを制限時間内繰り返す．

表 1 は各問題例に対して各解法を実行した結果のうち，各 type で問題サイズの大きな 5 例題 (type2 のみ 8 例題) の結果の抜粋及び 45 例題の平均結果である．“#LS”の欄は，制限時間内での LS を繰り返した回数である．“best [%]”及び“avg [%]”は，算出した最良及び平均（対応する“#LS”）の精度である．また各 type の下の Avg は抜粋した例題の平均を示しており，一番下の All_Avg は全 45 例題の平均の結果を示し，太字の箇所は各解法の比較のもとで最良の箇所を示している．

なお，解の評価値の精度は式 (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．

$$\text{解の評価値の精度 (\%)} = \frac{C(\pi) - C(\pi^*)}{C(\pi^*)} \times 100 \qquad (2)$$

表 1 より，提案法 FIKLS は BIKLS と比べ，平均的に良好な結果を示している．ランダム性を有するために

deterministic ではない探索が KLS の広い近傍に対して行われた結果，ランダム性を有している FI2LS や提案法と同じ広い近傍を探索する BIKLS より良好な結果を得たと考えられる．また#LS の欄より，FIKLS が BIKLS よりも#LS が少なく実行速度が遅いことは，3.2 節に示した通り，1 度の移動における改善量が少なく交換回数が多くるためであり，BIKLS より近傍生成の回数や交換回数が増えることを追加実験により確認した．また 45 例題に対して各解法を 10000 回ずつ実行した結果における平均の標準偏差は BI2LS が 2.638, FI2LS が 2.618, BIKLS が 2.518, FIKLS が 2.373 であり，FIKLS が最も低く，比較解法より安定して平均的に良好な結果を得られることが明らかになった．

6 むすび

本論文では，一般的な最良移動に基づく KLS の初期解に強く依存する問題を解消するため，ランダム性を有する即時移動戦略に基づく k-opt 局所探索法を提案し，2 次割当問題に対して適用した．また比較実験の結果から従来法より安定して平均的に良好な結果を得られることを示した．今後の課題として，2 次割当問題以外の組合せ最適化問題に適用することなどがあげられる．

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] E. K. Burke, T. Curtois, R. Qu, and G. V. Berghe. A time predefined variable depth search for nurse rostering. *INFORMS Journal on Computing*, Vol. 25, No. 3, pp. 411–419, 2012.

[2] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[3] K. Helsgaun. An effective implementation of the Lin-Kernighan traveling salesman heuristic. *European Journal of Operational Research*, Vol. 126, No. 1, pp. 106–130, 2000.

[4] K. Katayama, A. Hamamoto, and H. Narihisa. An effective local search for the maximum clique problem. *Information Processing Letters*, Vol. 95, No. 5, pp. 503 – 511, 2005.

[5] 片山謙吾, 成久洋之. バイナリー 2 次計画問題に対する変形 k-opt 局所探索法. 電子情報通信学会論文誌 (A), Vol. J84-A, No. 3, pp. 430–435, 2001.

[6] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, Vol. 49, No. 2, pp. 291–307, 1970.

[7] 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.

[8] S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research.*, Vol. 21, No. 2, pp. 498–516, 1973.

[9] P. Merz and B. Freisleben. Greedy and local search heuristics for unconstrained binary quadratic programming. *Journal of Heuristics*, Vol. 8, No. 2, pp. 197–213, 2002.

[10] A. B. Mohamed, M. Gunasekaran, R. Heba, and Z. A. Nasser. A comprehensive review of quadratic assignment problem: variants, hybrids and applications. *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–24, 2018.

[11] K. A. Murthy, Y. Li, and P. M. Pardalos. A local search algorithm for the quadratic assignment problem. *Informatica*, Vol. 3, pp. 524–538, 1992.

[12] T. Otsuki and K. Aihara. New variable depth local search for multiple depot vehicle scheduling problems. *Journal of Heuristics*, Vol. 22, No. 4, p. 567–585, 2016.

[13] M. Racer and M. M. Amini. A robust heuristic for the generalized assignment problem. *Annals of Operations Research*, Vol. 50, pp. 487–503, 1994.

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

[15] M. Yagiura, T. Yamaguchi, and T. Ibaraki. A variable depth search algorithm with branching search for the generalized assignment problem. *Optimization Methods and Software*, Vol. 10, No. 2, pp. 419–441, 1998.

[16] 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.

表 1 従来の 2 種類の 2LS と KLS 及び提案法の実験結果

type	instance	BI2LS best[%]	BI2LS avg[%]	BI2LS #LS	FI2LS best[%]	FI2LS avg[%]	FI2LS #LS	BIKLS best[%]	BIKLS avg[%]	BIKLS #LS	FIKLS best[%]	FIKLS avg[%]	FIKLS #LS
type1	tai40a	1.438	5.046	1,878,070	1.586	4.826	2,257,330	1.184	4.149	213,514	**0.779**	**4.042**	112,002
	tai50a	1.907	5.025	963,670	1.941	4.781	1,080,540	1.596	4.172	117,804	**1.532**	**4.058**	61,881
	tai60a	2.350	4.756	547,298	2.140	4.503	603,977	1.792	3.954	70,611	**1.712**	**3.833**	36,469
	tai80a	2.423	4.207	224,674	2.181	3.964	228,708	**2.000**	3.520	29,186	2.165	**3.399**	14,836
	tai100a	2.167	3.778	112,681	2.283	3.536	107,130	**1.973**	3.147	14,263	2.032	**3.026**	7,180
	Avg	2.057	4.563	745,279	2.026	4.322	855,537	1.709	3.788	89,076	**1.644**	**3.671**	46,474
type2	sko100a	0.491	2.083	61,739	0.382	1.943	109,167	0.467	1.736	12,160	**0.362**	**1.530**	6,409
	sko100b	0.535	2.017	62,327	0.502	1.885	110,905	0.342	1.676	12,097	**0.277**	**1.494**	6,459
	sko100c	0.565	2.301	60,897	**0.341**	2.159	109,956	0.383	1.939	11,939	0.403	**1.732**	6,394
	sko100d	0.620	2.064	62,956	0.509	1.931	111,419	**0.448**	1.725	12,247	0.505	**1.545**	6,508
	sko100e	0.636	2.320	62,449	0.476	2.168	108,746	0.488	1.946	12,054	**0.417**	**1.737**	6,400
	sko100f	0.652	2.026	63,920	0.513	1.903	111,951	**0.450**	1.703	12,473	0.518	**1.542**	6,564
	wil100	0.262	1.098	62,655	0.309	1.045	112,138	0.259	0.939	12,432	**0.247**	**0.843**	6,526
	tho150	0.702	2.424	15,832	**0.619**	2.258	27,316	0.684	2.084	3,043	0.676	**1.835**	1,658
	Avg	0.558	2.042	56,597	0.456	1.911	100,200	0.440	1.718	11,056	**0.426**	**1.532**	5,865
type3	ste36a	0.294	12.009	1,800,820	0.252	11.631	3,238,150	**0.231**	10.571	242,765	**0.231**	**9.752**	110,882
	ste36b	**0.000**	21.423	1,597,650	**0.000**	21.335	2,990,720	**0.000**	19.259	222,877	**0.000**	**17.891**	98,751
	ste36c	0.132	9.455	1,714,120	**0.000**	9.124	3,115,510	**0.000**	8.468	232,215	**0.000**	**7.726**	104,437
	esc64a	**0.000**	0.862	1,088,890	**0.000**	1.724	1,281,590	**0.000**	**0.000**	169,210	**0.000**	**0.000**	91,915
	esc128	**0.000**	12.500	151,792	**0.000**	14.063	153,801	**0.000**	9.375	21,233	**0.000**	**7.813**	10,750
	Avg	0.085	11.250	1,270,654	0.050	11.575	2,155,954	**0.046**	9.535	177,660	**0.046**	**8.636**	83,347
type4	tai50b	**0.000**	7.252	313,324	0.052	6.752	368,547	0.014	6.959	69,130	0.013	**5.880**	26,980
	tai60b	0.097	7.985	161,960	0.075	7.203	181,532	0.110	7.676	37,723	**0.050**	**6.195**	14,654
	tai80b	0.231	6.097	62,483	0.319	5.613	64,742	0.385	5.839	15,130	**0.043**	**4.990**	6,188
	tai100b	0.567	5.304	28,137	0.479	4.922	28,797	**0.301**	5.099	6,949	0.449	**4.280**	2,946
	tai150b	1.140	3.530	6,983	**1.113**	3.201	5,704	1.369	3.329	1,554	1.153	**2.810**	697
	Avg	0.407	6.034	114,577	0.407	5.538	129,864	0.436	5.781	26,097	**0.342**	**4.831**	10,293
All_Avg		0.462	6.092	2,303,746	0.417	5.964	2,973,508	0.356	5.342	253,651	**0.336**	**4.901**	133,137