A Study of Adaptive Perturbation Strategy for Iterated Local Search

Una Benlic and Jin-Kao Hao

LERIA, Université d'Angers, 2 Bd Lavoisier, 49045 Angers Cedex 01, France
{benlic,hao}@info.univ-angers.fr

Abstract. We investigate the contribution of a recently proposed adaptive diversification strategy (ADS) to the performance of an iterated local search (ILS) algorithm. ADS is used as a diversification mechanism by breakout local search (BLS), which is a new variant of the ILS metaheuristic. The proposed perturbation strategy adaptively selects between two types of perturbations (directed or random moves) of different intensities, depending on the current state of search. We experimentally evaluate the performance of ADS on the quadratic assignment problem (QAP) and the maximum clique problem (MAX-CLQ). Computational results accentuate the benefit of combining adaptively multiple perturbation types of different intensities. Moreover, we provide some guidance on when to introduce a weaker and when to introduce a stronger diversification into the search.

Keywords: adaptive perturbation strategy, iterated local search, breakout local search, quadratic assignment, maximum clique.

1 Introduction

To be successful, a heuristic approach needs to find a suitable balance between an intensified and a diversified search. Intensification is the ability of the method to examine in depth specific search areas while diversification is the capacity of the method to diversify the search in order to find promising new search areas. If the diversification is too weak, the search has a great chance to end up cycling between two or several previously encountered local optima. On the other hand, a too strong diversification is no better than a random restart and may reduce the chances of finding better solutions in the following iterations. Determining the right degree of diversification is not a straightforward task, since it greatly depends on structural characteristics of the given instance such as the distribution of local optima, the correlation between solutions, the number of global optima, etc. Additionally, the optimal diversification degree required at one stage of the search is not necessarily optimal at another stage. These facts constitute the motivation for our adaptive diversification mechanism.

In this paper, we investigate the contribution of a new adaptive diversification strategy (ADS) employed by a recent breakout local search (BLS) metaheuristic [3,4,5,6]. BLS is a variation of iterated local search (ILS) [9] since it

M. Middendorf and C. Blum (Eds.): EvoCOP 2013, LNCS 7832, pp. 61–72, 2013.

combines a descent-based local search with a perturbation mechanism. However, BLS has a particular focus on the importance of the perturbation phase. Based on some information on the search history, it dynamically determines the number of perturbation moves, and adaptively chooses between two or several types of perturbation moves of different intensities. In this work, we fix the number of perturbation moves, and evaluate the efficiency of this adaptive multi-type diversification on the quadratic assignment problem (QAP) and the maximum clique problem (MAX-CLQ). More precisely, we integrate ADS into a basic ILS algorithm and compare the performance of this adaptive diversification based ILS against two other ILS versions based respectively on random and directed perturbation moves. The obtained computational results accentuate the benefit of combining adaptively multiple perturbation types of different intensities. Furthermore, we analyze the distribution of local optima to provide some guidance on when to introduce a weaker or a stronger diversification into the search.

2 Iterated Local Search with a Adaptive Diversification Strategy

2.1 General Framework

The basic idea of iterated local search (ILS) is to alternate iteratively between a local search phase to attain local optima, and a perturbation phase (applied to the current or best found local optimum) to direct the search towards new unexplored regions of the search space.

Alg. 1 shows the general framework of our adaptive diversification based ILS (denoted by AD-ILS), which we later apply to two *NP*-hard problems considered in sections 3 and 4. Starting from an initial feasible solution, AD-ILS first initializes the best-found solution S_{best}, the tabu list H (see Section 2.2), the counter ω for consecutive non-improving local optima, and the global iteration counter *iter* (lines 1-5 of Alg. 1). While a stopping condition is not satisfied, AD-ILS applies a simple descent (ascent in case of maximization) local search to reach a local optimum S (lines 8-12 of Alg. 1). Each iteration of this descent-based procedure searches the given neighborhood for the best solution to replace the current solution, and stops if no improving neighbor exists (i.e., once local optimality is reached). After each solution transition, AD-ILS updates the tabu list H (see Section 2.2) and increments the global iteration counter *iter*.

If the quality of the local optimum S, reached in the last descent phase, is better than the quality of the best-found solution S_{best}, AD-ILS updates S_{best} and re-initializes the number of consecutive non-improving local optima ω (lines 13-15 of Alg. 1). Otherwise, ω is incremented by one (lines 16-17 of Alg. 1). If ω exceeds a certain threshold T, it is reset to zero (lines 19-21 of Alg. 1).

In order to escape from the current local optimum S, AD-ILS applies its perturbation mechanism ADS to S, and returns a perturbed solution which becomes a new starting point for the next phase of the descent/ascent procedure (line 22 of Alg. 1).

Algorithm 1. Adaptive_Diversification-based_Iterated_Local_Search

Require: The jump magnitude L, the threshold T and the tabu tenure γ.
Ensure: Solution S_{best}.
1: $S \leftarrow$ *Initial_Solution*
2: $S_{best} \leftarrow S$; */* Initialize the best-found solution S_{best} */*
3: $H \leftarrow 0$; */* Initialize the tabu list H */*
4: $\omega = 0$; */* Initialize the number of consecutive non-improving local opt. ω */*
5: $iter = 0$; */* Initialize the global iteration counter iter */*
6: **while** stopping condition not reached **do**
7: Let m be the best move eligible for S
8: **while** $f(S \oplus m)$ *is better than* $f(S)$ **do**
9: $S \leftarrow S \oplus m$ */* Perform the best-improving move */*
10: $H_m \leftarrow Iter + \gamma$ */* Update tabu list, γ is the tabu tenure */*
11: $Iter \leftarrow Iter + 1$
12: **end while**
13: **if** $f(S)$ *is better than* $f(S_{best})$ **then**
14: $S_{best} \leftarrow S$
15: $\omega = 0$
16: **else**
17: $\omega = \omega + 1$
18: **end if**
19: **if** $\omega > T$ **then**
20: $\omega = 0$
21: **end if**
22: $S \leftarrow$ *Adaptive_Diversification_Strategy(S, H, ω, L, iter, T, γ) /* Sect. 2.2*/*
23: **end while**

Since the local search phase is a simple decent/ascent procedure, it alone cannot escape from a local optimum. The performance of AD-ILS thus strongly depends on its perturbation mechanism ADS which is detailed in the next section.

2.2 Adaptive Diversification Strategy (ADS)

The AD-ILS algorithm, that we apply to QAP (Section 3) and MAX-CLQ (Section 4), employs a directed and a random perturbation to guide the search towards new regions of the search space. This adaptive perturbation mechanism is illustrated in Alg. 2.

The directed perturbation (DIRP) is based on the idea of tabu list from tabu search [7]. It uses a selection rule that favors the move that minimizes the degradation of the objective, under the constraint that this move is not prohibited by the tabu list. The information for move prohibition is maintained in a tabu list H, such that each element in H is the iteration number when the corresponding move was last performed, plus the tabu tenure γ (represented as a natural number). The tabu status of a move is neglected only if the move leads to a new solution better than the best solution found so far. The directed perturbation relies thus both on 1) history information which keeps track, for each move, of the last time (iteration) when it was performed and 2) on the quality of the moves to be applied for perturbation in order not to deteriorate too much the perturbed solution. History-based diversifications have previously been used in [2,8].

Algorithm 2. Adaptive_Diversification_Strategy($S, H, \omega, L, iter, T, \gamma$)

Require: Local optimum S, tabu list H, number of consecutive non-improving local optima visited ω, jump magnitude L, global iteration counter $Iter$, threshold T, tabu tenure γ.

```
1: P ← Determine_Probability_for_Directed_Perturbation(ω, T) /* see Eq. 1 */
2: if (P > random[0.0, 1.0]) then
3:     PERT = DIRP /* L moves of DIRP will be applied to S */
4: else
5:     PERT = RNDP /* L moves of RNDP will be applied to S */
6: end if
7: for i := 1 to L do
8:     S ← Perturbation_Move(S, PERT) /* Apply a move m of the predetermined perturb.*/
9:     H_m ← Iter + γ
10:    Iter ← Iter + 1
11:    if f(S) is better than f(S_best) then
12:        S_best ← S
13:        ω = 0
14:    end if
15: end for
```

The random perturbation (RNDP) is the most popular type of perturbation for ILS algorithms. It consists in performing randomly selected moves.

It is obvious that DIRP and RNDP introduce different balances between intensification and diversification. More precisely, DIRP is more oriented towards search intensification than RNDP since it considers the quality of moves in order not to degrade too much the resulting solution. The search thus has greater chances to end cycling between two or more local optima if DIRP is used alone. On the other hand, RNDP may prevent the search from cycling, but it may as well decrease the chances of finding a global optimum by passing too quickly to new regions while promising regions were not sufficiently exploited.

To insure the best balance as possible between an intensified and a diversified search, the adaptive diversification strategy ADS applies probabilistically DIRP and RNDP. The probability of applying a particular type of perturbation is determined dynamically depending on the search state, i.e., the current number ω of consecutive non-improving attractors visited (lines 13-21 of Alg. 1). The idea is to apply more often directed perturbations (with a higher probability) as the search progresses towards improved new local optima (the non-improving consecutive counter ω is small). With the increase of ω, the probability of using the directed perturbations progressively decreases while the probability of applying the random moves increases for the purpose of a stronger diversification.

Additionally, it has been observed from an experimental analysis that it is useful to guarantee a minimum of applications of DIRP. Therefore, we constrain the probability P of applying DIRP to take values no smaller than a threshold P_0:

$$P = \begin{cases} e^{-\omega/T} & \text{if } e^{-\omega/T} > P_0 \\ P_0 & \text{otherwise} \end{cases} \quad (1)$$

Once the type of perturbation is determined (lines 1-6 of Alg 2), AD-ILS applies L moves of the selected perturbation to the current local optimum S (lines 7-15 of Alg 2).

3 Case Study I: Quadratic Assignment Problem (QAP)

3.1 Problem Description

The *quadratic assignment problem* (QAP) is a well-known NP-hard problem which consists in assigning at minimal cost a set of n locations to a given set of n facilities, given a flow f_{ij} from facility i to facility j for all $i, j \in \{1, ..., n\}$ and a distance d_{ab} between locations a and b for all $a, b \in \{1, ..., n\}$. Let Π denote the set of the permutation functions $\pi : \{1, ..., n\} \rightarrow \{1, ..., n\}$, QAP can mathematically be formulated as follows:

$$min_{\pi \in \Pi} C(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} d_{\pi_i \pi_j} \tag{2}$$

where f and d are the flow and distance matrices respectively, and $\pi \in \Pi$ is a solution where π_i represents the location assigned to facility i. The problem objective is then to find a permutation π^* in Π that minimizes the sum of the products of the flow and distance matrices, i.e., $C(\pi^*) \leq C(\pi)$, $\forall \pi \in \Pi$.

3.2 Neighborhood Relation and Its Exploitation

A candidate solution for QAP can be encoded as a permutation π of $\{1, ..., n\}$, such that π_i denotes the location assigned to facility $i \in \{1, ..., n\}$. Like many existing local search methods for QAP, our ILS employs the swap move to π which consists in exchanging the locations of two facilities.

The neighborhood $N(\pi)$ of a solution π is then defined as the set of all the permutations that can be obtained by exchanging any two values π_u and π_v, i.e., $N(\pi) = \{\pi' : \pi'_u = \pi_v, \pi'_v = \pi_u, u \neq v \ and \ \pi'_i = \pi_i, \forall i \neq u, v\}$. The size of $N(\pi)$ is thus equal to $n(n-1)/2$.

The local search phase of ILS explores the whole neighborhood $N(\pi)$ to find the best swap move which is then applied to π to obtain a new solution. This process is repeated until a local optimum is reached. To evaluate the whole swap neighborhood $N(\pi)$ in $O(n^2)$ time, we use an effective strategy which incrementally updates the objective variation of each move [10].

3.3 Perturbation Types Combined with ADS

The *directed perturbation* (see Section 2.2) applies a swap move that minimizes the value of the objective function C, under the constraint that the move has not been applied during the last γ iterations (γ is the tabu tenure that takes a random value from a given range). The eligible moves for the directed perturbation are identified by the set A such that:

$$A = \{swap(u, v) : min\{\delta(\pi, u, v)\}, H_{uv} < Iter \ or \ (\delta(\pi, u, v) + c) < c_{best}, u \neq v\}$$

where H is the tabu list that keeps track of the iteration number when a move was last performed plus γ, $Iter$ the current iteration number, c the cost of the

current solution, and c_{best} the cost of the best solution discovered so far. A larger value of γ implies stronger diversification.

The *random perturbation* simply performs swap moves that are selected uniformly at random.

Our AD-ILS for QAP combines and applies these two types of perturbations as explained in Section 2.2.

3.4 Experimental Results and Comparisons

To evaluate the efficiency of the proposed AD-ILS algorithm for QAP, we carry out experiments on a set of 16 difficult QAPLIB instances[1] of three different types (unstructured instances, real-life like instances, grid-based instances).

We contrast the results of AD-ILS with those obtained with two other ILS versions which respectively employ perturbation strategies based on directed (DIR-ILS) and random moves (RND-ILS). For all the three ILS versions, the number of perturbation moves is $L = 0.15n$. For both AD-ILS and DIR-ILS, the tabu tenure γ takes a random value in the range $[0.9n, 1.1n]$. For AD-ILS, the setting of the parameters used for adaptive perturbation is $P_0 = 0.9$ and $T = 2500$. This setting of parameters is determined by a preliminary experiment and can be justified to some extent by the analysis provided in Section 5. We make 20 independent executions per instance, with the time limit per run set to 2 hours.

Table 1. Computational comparison of AD-ILS with DIR-ILS and RND-ILS on 16 hard QAP instances

Instance		AD-ILS			DIR-ILS			RND-ILS		
Name	BKR	$\%\rho_{best}$	$\%\rho_{avg}$	t(m)	$\%\rho_{best}$	$\%\rho_{avg}$	t(m)	$\%\rho_{best}$	$\%\rho_{avg}$	t(m)
tai40a	3139370	0.000(12)	0.030	30.2	**0.000(14)**	**0.022**	41.5	0.000(3)	0.116	49.4
tai50a	4938796	**0.000(4)**	**0.121**	62.3	0.000(3)	0.136	60.8	0.301(0)	0.576	58.0
tai60a	7205962	**0.000(1)**	**0.359**	65.9	0.191(0)	0.400	57.9	0.313(0)	0.837	47.9
tai80a	13499184	0.651(0)	0.764	67.8	**0.600(0)**	**0.755**	66.7	0.812(0)	1,179	43.9
tai100a	21052466	**0.626(0)**	0.804	59.7	0.648(0)	**0.788**	50.6	0.948(0)	1,218	68.2
tai80b	818415043	**0.000(9)**	0.423	34.2	0.000(1)	0.755	28,4	**0.000(9)**	**0.001**	65.4
tai100b	1185996137	**0.000(12)**	0.253	17,9	0.000(6)	0.382	37.9	0.000(10)	**0.001**	39.2
tai150b	498896643	**0.000(1)**	0.322	68.6	0.161(0)	0.429	80.2	0.023(0)	**0.138**	84.5
sko81	90998	**0.000(20)**	**0.000**	11.8	**0.000(20)**	**0.000**	2.7	0.011(0)	0.032	65.0
sko90	115534	**0.000(7)**	**0.045**	14,7	0.000(6)	0.063	24,6	0.011(0)	0.040	74.2
sko100a	152002	**0.000(12)**	**0.006**	11,7	0.000(8)	0.022	15,9	0.045(0)	0.069	64.1
sko100b	153890	**0.000(20)**	**0.000**	15,4	0.000(16)	0.019	16,9	0.016(0)	0.045	55.2
sko100c	147862	**0.000(20)**	**0.000**	9,8	0.000(19)	0.021	11,6	0.009(0)	0.046	69.4
sko100d	149576	0.000(10)	**0.002**	61,2	**0.000(11)**	0.066	29,6	0.044(0)	0.076	59.6
sko100e	149150	**0.000(18)**	**0.000**	40,3	0.000(16)	0.034	16,7	0.011(0)	0.030	69.8
sko100f	149036	**0.000(20)**	**0.000**	23,3	0.000(15)	0.013	23,1	0.023(0)	0.063	60.2
Average		**0.080**	**0.196**	38.9	0.100	0.244	**37.5**	0.160	0.279	60.9

Table 1 shows for each algorithm the best (column $\%\rho_{best}$) and average (column $\%\rho_{avg}$) percentage deviation from the best-known result (column BKR) obtained over 20 runs. The percentage deviation $\%\rho$ is computed as $\%\rho = 100(z - BKR)/z[\%]$, where z is the result obtained by a given approach and

[1] `http://www.seas.upenn.edu/qaplib/`

BKR the best-known objective value. Next to the percentage deviation ρ_{best}, we indicate in parentheses the number of times the best-known solution was found over 20 executions. Moreover, we provide the average times in minutes required to reach the returned solution after a trial. The best results are indicated in bold. The averaged results are provided in the last row.

From the results in Table 1, we can make the following conclusions. In most cases, the best performance is obtained with AD-ILS which reports an average $\%\rho_{avg}$ of 0.196 (vrs. 0.244 for DIR-ILS and 0.279 for RND-ILS) over the 16 QAP instances. Indeed, AD-ILS is unable to attain the best-known result from the literature only for two instances (tai80a and tai100a), while DIR-ILS and RND-ILS are unable to reach the best-known objective value for 4 and 13 instances respectively. The worst performance on the QAP instances is thus obtained with the RND-ILS algorithm, except for three real-life like instances (i.e., $tai80b$, $tai100b$ and $tai150b$) for which RND-ILS algorithm insures the best performance. The Posthoc test reveled that AD-ILS statistically outperforms RND-ILS with a p-value of 0.016. In Section 5, we provide an explanation for these performances based on a landscape analysis. In terms of computing times, AD-ILS and DIR-ILS show comparable performances with an average time of 38.7 and 37.5 minutes respectively for the 16 instances, while RND-ILS requires on average around 70 minutes. These results show the advantage of applying directed or adaptive perturbations over the classic random perturbations.

4 Case Study II: Maximum Clique Problem (MAX-CLQ)

4.1 Problem Description

Given an undirected graph $G = (V, E)$ where V is the set of vertices and E the set of edges, a clique C of G is a subset of V such that all the vertices in C are pairwise adjacent, i.e., $\forall v, u \in C, \{v, u\} \in E$. The *maximum clique problem* (MAX-CLQ) is to find a clique C of the maximal cardinality. It is one of the first problems shown to be *NP*-complete.

4.2 Neighborhood Relations and Their Exploitation

For solution transformations, ILS employs four distinct move operators (moves for short) whose basic idea is to generate a new clique from the current clique C by adding vertices $v \in V \setminus C$ to C, swapping vertices u and v such that $u \in C$ and $v \in V \setminus C$, or removing vertices $v \in C$ from C.

Three sets PA, OM and OC are involved in the definition of these moves. The vertex set PA consists of nodes excluded from the clique C that are connected to all the vertices in C, i.e., $PA = \{v : v \notin C, \forall u \in C, \{v, u\} \in E\}$.

The OM set consists of vertex pairs (v, u) such that v is excluded from C and is connected to all vertices in C except to vertex $u \in C$, i.e.,
$OM = \{(v, u) : v \notin C \text{ and } u \in C, |N(v) \cap C| = |C| - 1, \{v, u\} \notin E\}$, where $N(v) = \{i : i \in V, \{i, v\} \in E\}$.

The OC set consists of all the vertices excluded from the clique C, i.e., $OC = \{v : V \setminus C\}$.

The four moves M_1 to M_4 can then be defined as follows:

M_1: Select a vertex $v \in PA$ and insert it into C. After this move, the change in the objective function is given by the following expression: $\Delta = w_v$.

M_2: Select a vertex pair $(v, u) \in OM$. Insert v into C and remove u from C. The change in the objective function can be computed as: $\Delta = w_v - w_u$.

M_3: Select a vertex $v \in C$ and remove it from C. The change in the objective function is given as: $\Delta = -w_v$.

M_4: Select a vertex $v \in OC$ such that $(w_v + \sum_{\{v,u\} \in E, u \in C} w_u) \geq \alpha * f(C)$, where $f(C)$ is the current solution cost and $0 < \alpha < 1$. Add v to C. Repair the resulting clique C by removing from C all vertices x such that $\{v, x\} \notin E$.

The descent-based local search phase of ILS consists in identifying the best move m from the union $M_1 \cup M_2$ and applying it to C to obtain a new solution. This procedure is repeated until a local optimum is reached. The directed perturbation (see Section 4.3) applies a move m from $M_1 \cup M_2 \cup M_3$. For random perturbation (see Section 4.3), m is selected from M_4.

4.3 Perturbation Types Combined with ADS

As previously explained in Section 2.2, the *directed perturbation* is based on the tabu search principles and favors non-tabu moves that minimize the cost degradation. Move prohibition is determined in the following way. Each time a vertex v is added into the clique C, it can be removed from C without restrictions. However, each time v is dropped from C, it is forbidden to place it back to C for γ iterations. The value of γ is determined by the following relation:

$$\gamma = \phi + random(|OM|),$$

where ϕ is a coefficient and *random* is a function which returns at random a value ranging from 1 to $|OM|$ (the number of elements in the OM set, see Section 4.2).

The eligible moves for the directed perturbation are identified by the set A such that:

$$A = \{m : m \in \{M_1 \cup M_2 \cup M_3\}, max\{\Delta_m\}, prohibited(m) = false \text{ or } (\Delta_m + f(C)) > f_{best}\}$$

where Δ_m is the change in the objective function after performing move m (see Section 4.2). Note that the directed perturbation considers all the eligible moves from the union of three types of moves M_1, M_2 and M_3 (see Section 4.2).

The *random perturbation*, which is significantly stronger than the directed perturbation, consists in performing moves randomly selected from the set of moves M_4 (see Section 4.2). The degree of random perturbation can be adjusted by changing the value of parameter α ($0 < \alpha < 1$). If $\alpha \approx 0$, the random perturbation is very strong and can be compared to a random restart. If $\alpha \approx 1$, the strength of the random perturbation is insignificant.

4.4 Experimental Results and Comparisons

We report computational results using 6 instances from the BHOSLIB benchmark[2] and 11 instances from the more popular DIMACS benchmark[3]. To evaluate the significance of ADS, we compare the performances of AD-ILS, DIR-ILS and RND-ILS. For AD-ILS, the setting of the parameters used for adaptive perturbation is $P_0 = 0.9$ and $T = 2000$. $\alpha = 0.8$ for both AD-ILS and RND-ILS. For both AD-ILS and DIR-ILS, the coefficient ϕ for tabu tenure is set to 7. The number of perturbation moves L is set to $L = 0.05|V|$ for AD-ILS and DIR-ILS, while $L = 0.01|V|$ for RND-ILS. This setting of parameters is determined by a preliminary experiment. Each ILS version is independently executed 50 times, with the time-limit per run set to 90 minutes.

Table 2 reports the computational results. Column BR indicates the best-known or optimal (indicated with an asterisk) result. For each ILS, we report the best (column $|C|_{best}$) and the average result (column $|C|_{avg}$) obtained over 50 independent runs, as well as the average computing time in minutes required to reach the best reported result from column $|C|_{best}$. Next to the best-found clique value $|C|_{best}$, we indicate in parentheses the number of times the best-known solution was found over 50 executions.

Table 2. Computational comparison of AD-ILS with DIR-ILS and RND-ILS on 11 hard DIMACS instances and 6 large BHOSLIB instances

Instance		AD-ILS			DIR-ILS			RND-ILS		
Name	BR	$\|C\|_{best}$	$\|C\|_{avg}$	t(m)	$\|C\|_{best}$	$\|C\|_{avg}$	t(m)	$\|C\|_{best}$	$\|C\|_{avg}$	t(m)
brock800_1	23⋆	**23(9)**	**21.36**	43.8	23(4)	21.16	28.9	21(0)	20.98	4.0
brock800_2	24⋆	**24(27)**	**22.62**	34.6	24(4)	21.24	36.7	24(4)	21.24	25.6
brock800_3	25⋆	**25(41)**	**24.46**	41.6	25(15)	22.9	28.3	25(5)	22.3	47.5
brock800_4	26⋆	**26(45)**	**25.5**	22.5	26(37)	24.7	45.3	26(21)	23.1	43.7
C1000.9	68	**68(50)**	**68.0**	0.5	**68(50)**	**68.0**	0.1	59(0)	58.5	31.9
C2000.9	80	79(0)	77.66	55.0	79(0)	**78.36**	43.4	64(0)	62.88	44.3
keller6	59	**59(50)**	**59.0**	3.4	**59(50)**	**59.0**	0.5	50(0)	47.66	11.9
san1000	15⋆	15(12)	11.2	27.4	**15(33)**	**13.34**	30.9	15(1)	9.66	0.1
san400_0.7_1	40⋆	**40(50)**	**40.0**	18.5	**40(50)**	**40.0**	2.5	23(0)	21.84	43.5
san400_0.7_3	22⋆	**22(50)**	**22.0**	0.0	**22(50)**	**22.0**	0.0	**22(50)**	**22.0**	16.5
hamming10-4	40	**40(50)**	**40.0**	0.0	**40(50)**	**40.0**	0.0	38(0)	35.92	53.6
frb53-24-1	53⋆	53(2)	52.04	33.6	**53(4)**	**52.08**	36.0	46(0)	44.76	36.9
frb53-24-3	53⋆	53(22)	52.44	33.4	**53(50)**	**53.0**	20.7	46(0)	44.92	62.2
frb53-24-5	53⋆	53(37)	52.74	35.2	**53(50)**	**53.0**	17.8	46(0)	44.72	50.6
frb56-25-1	56⋆	56(1)	54.88	6.3	**56(15)**	**55.3**	33.2	49(0)	46.88	82.9
frb56-25-3	56⋆	56(2)	55.0	56.1	**56(13)**	**55.26**	56.7	48(0)	47.06	22.2
frb56-25-5	56⋆	56(33)	55.62	43.9	**56(49)**	**55.98**	10.9	48(0)	47.04	28.5

Like for QAP, AD-ILS significantly and statistically outperforms RND-ILS with a p-value = 3.751173e-04 according to the Posthoc test, which once again highlights the drawback of the classic random perturbation often used within the general ILS framework. However, the contribution of ADS is less significant in comparison with the directed perturbation strategy. Although both AD-ILS and DIR-ILS can attain the best-known result for all the used instances except for instance $C2000.9$, DIR-ILS outperforms AD-ILS in terms of average results

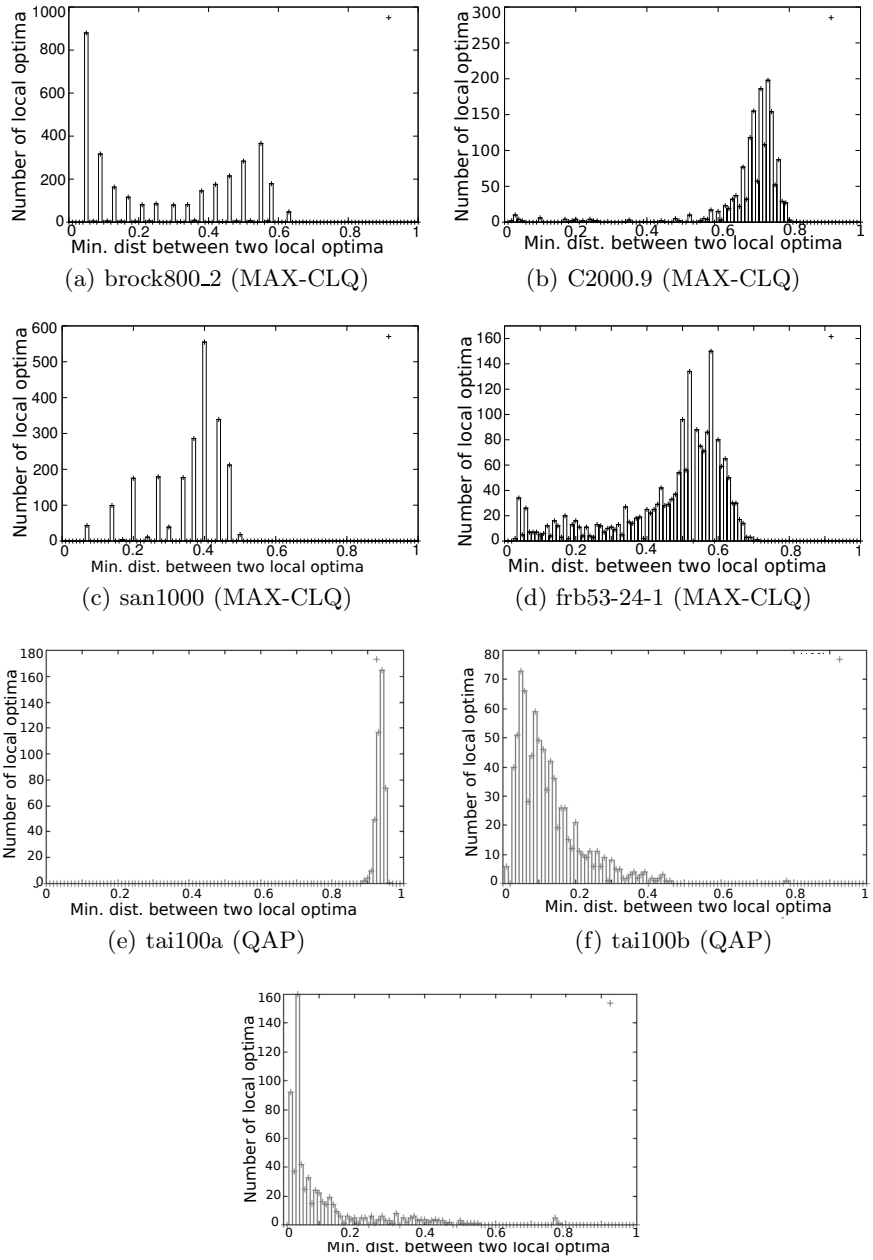
[2] `http://iridia.ulb.ac.be/~fmascia/maximum_clique/BHOSLIB-benchmark`
[3] `http://cs.hbg.psu.edu/txn131/clique.html`

on all the BHOSLIB instances and 2 DIMACS instances ($C2000.9$ and $san1000$). However, AD-ILS shows better performance than DIR-ILS on the four hard *brock* instances. In Section 5, we justify these results with an analysis of distribution of high quality local optima. In terms of average computing times, the difference between AD-ILS and DIR-ILS is not very obvious.

5 Analysis

We observed from the computational comparisons (see sections 3.4 and 4.4) that the best performance with ILS is often obtained when directed (weaker) and random (stronger) perturbations are adaptively combined. On the other hand, the results also showed that for some instances (e.g., BHOSLIB instances from the MAX-CLQ benchmark) it is more useful to apply only the weak (directed) perturbation, while for several other instances (i.e., QAP real-life like instances) the best performance is achieved with random perturbation. In this section, we try to justify such results by investigating the minimal distances between pairs of medium or high quality local optima. To measure distance between solutions for QAP and MAX-CLQ, we use the well-known hamming distance. Medium and high quality local optima may be viewed as 'strong' attractors since it is more likely that they are visited during the search than a low quality local optimum. More precisely, given a set of medium or high quality local optima S, for all $lo_i \in S$ we determine the distance d_{min} between lo_i and some other solution $lo_j \in S$ which is the closest to lo_i, i.e., $d_{min} = min_{lo_j \in S, lo_j \neq lo_i} d(lo_i, lo_j)$. For each possible distance $d_i \in [0, Max]$ (Max is the maximal distance), we then count the number of time that d_i is the distance between $lo \in S$ and another solution in S which is the closest to lo.

The results of this study for 4 MAX-CLQ instances and 3 QAP instances are given in Figure 1. The x-axis shows the normalized minimal distance between two 'strong' attractors, while the y-axis shows the number of pairs of 'strong' attractors separated by the given distance. Figure 1 indicates that there exists a significant difference in the distribution of medium and high quality local optima for QAP and MAX-CLQ instances. For *brock800-2,* $tai100b$ and $sko100a$ the minimal distances between two strong attractors are generally small, compared to instances $C2000.9$, *frb53-24-1* and $tai100a$. Intuitively, a weaker diversification introduced into the search for such instances may cause the search to cycle between 'strong' attractors that are not globally optimal solutions. For an effective solving of these instances, strong diversifications are required. On the other hand, for instances $C2000.9$, *frb53-24-1* and $tai100a$, the distribution of local optima prevents the search from cycling even with weak diversification. For this reason it may be worthwhile to perform a more intensive search. These observations justify to some extent why DIR-ILS provides the best performance on $C2000.9$ and frb instances, while RND-ILS seems to be the best for real-life like instances (i.e., $tai80b$, $tai100b$ and $tai150b$).



(a) brock800_2 (MAX-CLQ) (b) C2000.9 (MAX-CLQ)

(c) san1000 (MAX-CLQ) (d) frb53-24-1 (MAX-CLQ)

(e) tai100a (QAP) (f) tai100b (QAP)

(g) sko100a (QAP)

Fig. 1. Distribution of medium and high quality local optima (i.e., 'strong' attractors) for 4 MAX-CLQ and 3 QAP instances of different types and structures

6 Conclusion

The purpose of this paper is to investigate the performance of the adaptive diversification strategy (ADS) which constitutes an essential component of the recently proposed breakout local search (BLS). ADS adaptively applies a directed (weaker) and a random (stronger) perturbation according to the current search progress. We integrated ADS into the basic iterated local search (ILS) framework and evaluated its performance on the quadratic assignment problem (QAP) and the maximum clique problem (MAX-CLQ). Numerical results showed that the AD-ILS outperforms the standard ILS based on random moves on almost all the tested instances, which highlights the drawback of this classic perturbation strategy. Moreover, AD-ILS outperforms on most QAP instances and on several hard MAX-CLQ instances the ILS version which applies solely directed perturbation moves. We performed an analysis of the distribution of local optima to provide some guidance on when to introduce a weaker and when to introduce a stronger diversification into the search.

Acknowledgment. We are grateful to the referees for their comments. The work is partially supported by the Pays de la Loire Region (France) within the RaDaPop (2009-2013) and LigeRO (2010-2013) projects.

References

1. Aarts, E.H.L., Lenstra, J.K.: Simulated Annealing. In: Local Search in Combinatorial Optimization, ch. 1, pp. 1–17. Wiley (1997)
2. Battiti, R., Protasi, M.: Reactive Search, a history-based heuristic for MAX-SAT. ACM Journal of Experimental Algorithmics 2 (1996)
3. Benlic, U., Hao, J.-K.: A Study of Breakout Local Search for the Minimum Sum Coloring Problem. In: Bui, L.T., Ong, Y.S., Hoai, N.X., Ishibuchi, H., Suganthan, P.N. (eds.) SEAL 2012. LNCS, vol. 7673, pp. 128–137. Springer, Heidelberg (2012)
4. Benlic, U., Hao, J.K.: Breakout local search for maximum clique problems. Computers & Operations Research 40(1), 192–206 (2013)
5. Benlic, U., Hao, J.K.: Breakout local search for the max-cut problem. Engineering Applications of Artificial Intelligence (in press, 2013)
6. Benlic, U., Hao, J.K.: Breakout local search for the quadratic assignment problem. Applied Mathematics and Computation 219(9), 4800–4815 (2013)
7. Glover, F., Laguna, M.: Tabu Search, 408 p. Kluwer Academic Publishers, Boston (1998) ISBN: 0-7923-8187-4
8. Kelly, J.P., Laguna, M., Glover, F.: A study of diversification strategies for the quadratic assignment problem. Computers & Operations Research 21(8), 885–893 (1994)
9. Lourenco, H.R., Martin, O., Stützle, T.: Iterated local search. In: Handbook of Meta-heuristics. Springer, Heidelberg (2003)
10. Taillard, E.: Robust taboo search for the quadratic assignment problem. Parallel Computing 17, 443–455 (1991)