
Combination of Guided Local Search and Estimation of Distribution Algorithm for Quadratic Assignment Problems

Qingfu Zhang, Jianyong Sun, Edward Tsang and John Ford

Department of Computer Science

University of Essex

Wivenhoe Park, Colchester CO4 3SQ, U.K.

E-mail: qzhang@essex.ac.uk

Abstract

Guided Local Search (GLS) is a general meta-heuristic that sits on top of local search algorithms and help them escape from local minima. However, GLS is often trapped in a small local area. Estimation of Distribution Algorithm(EDA) is a new evolutionary algorithm in which statistical information are explicitly extracted from the current population and then used for generating new solutions. This paper combines EDA with GLS and proposes a hybrid algorithm, called EDA/GLS for Quadratic Assignment Problems (QAP). In EDA/GLS, a novel EDA-like operator, called guided mutation is used with GLS for exploitation. A restart strategy based on statistical information is employed for exploration. Experimental results show that EDA/GLS outperforms GLS.

1 Introduction

Guided Local Search (GLS) [1] is a general meta-heuristic that sits on top of local search procedures and help them escape from local minima. When the given local search procedure is trapped in a local minimum, GLS changes the objective function, by increasing penalties present in an augmented objective function, associated with features contained in this local minimum. The local search then continues to search using the augmented objective function, which will hopefully guide the search to escape from the local minimum. GLS has been successfully applied to a number of optimization and search problems such as the SAT problem [2], the vehicle routing problem [3], BT's workforce scheduling problem [4], the radio link frequency assignment problem [5], function optimiza-

tion [6], the travelling salesman problem [7] and the quadratic assignment problem [8].

GLS can easily escape from the first several local minima it met in the early stage of its search procedure. However, after visiting a number of local minima, GLS might be re-trapped in some of these local minima it has visited. In other words, there is a high probability that GLS is trapped in a small region in the search space. A very natural way to overcome this shortcoming is to hybridize GLS with other population-based algorithms. In fact, combinations of GLS with genetic algorithms have been proposed recently in [9][10] for solving TSP and other combinatorial optimization problems.

Estimation of Distribution Algorithms [11] are a new class of population-based algorithms. EDAs maintain and successively improve a population of candidate solutions and a probability model for promising solutions until some stopping condition is met. EDAs select the best solutions from the current population and explicitly extract global statistical information from the selected solutions. Based on these global information, the probability model is updated and then new solutions are generated from the probability model to replace fully or partially the old population. EDAs are good at locating promising areas in the search space but they lack in the ability of refining a single solution.

This paper combines GLS with EDAs and proposes a new hybrid algorithm, which is called Estimation Distribution Algorithm with Guided Local Search (EDA/GLS) for the quadratic assignment problem. EDA/GLS maintains a population of points and a probability matrix in the search space. The probability matrix models the distribution of the promising points in the search space. The mechanism for creating new solutions in EDA/GLS is different from other existing EDAs. We introduce a new operator called the guided mutation to generate new solutions. Guided by

the probability matrix in EDA/GLS, the guided mutation generates a new solution by moving a solution in a random direction. At each generation, the statistical information is extracted from some selected points in the current population and used to update the probability matrix. Then, the guided mutation is employed to generate new points and GLS applied to these new points to create new members for the next generation. We test GLS/EDA on the QAP problems. The experimental results show that EDA/GLS outperforms GLS.

2 Quadratic Assignment Problem

The Quadratic Assignment Problem (QAP) is a combinatorial optimization problem and arises in many applications such as facility location, scheduling, manufacturing and statistical data analysis. In the QAP, n departments have to be assigned to n locations such that the cost of the assignment, depending on the distances between the locations and the flows of materials between the departments, is minimized. Mathematically, it can be described as follows:

$$\min c(\pi) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} b_{\pi(i)\pi(j)} \quad (1)$$

where a_{ij} is the flow of materials from department i to department j , b_{kl} is the distance between location k to location l and π is a permutation of $\{1, 2, \dots, n\}$. $\pi(i) = k$ if department i is assigned to location k . The QAP is one of the most difficult NP-hard combinatorial problem. Solving QAP instances with $n > 30$ is computationally impractical for exact algorithms such as branch-and-bound methods. Therefore, a variety of heuristics for dealing with large QAP instances have been developed, including Tabu Search [12], Ant Colony algorithms [13], Evolution Strategies [14], Genetic Algorithms [15], Neural Networks [16] and Memetic Algorithms [17].

3 Guided Local Search for QAP

In GLS, solution features are defined to distinguish between solutions with different characteristics, so that bad characteristics can be penalized by GLS and hopefully removed by the local search algorithm. The features depend on problems and to a certain extent on the local search algorithm. In the QAP, a very natural choice for the feature set is the department-location assignments. Each feature (u, v) (department u is assigned to location v) should have the following components:

- An indicator function, $I_{uv}(\pi)$ indicating whether the feature is present in a solution π or not

$$I_{uv}(\pi) = \begin{cases} 1 & \text{if } \pi(u) = v \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

- An cost function $c_{uv}(\pi)$ which is defined as:

$$c_{uv}(\pi) = \begin{cases} \sum_{j=1}^n a_{uj} b_{v\pi(j)} & \text{if } I_{uv}(\pi) = 1 \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

- A penalty $\phi(u, v)$, initially set to 0, used to penalize occurrences of the feature in local minima.

3.1 Selective Penalty Modifications

When the local search algorithm returns a local minimum $\tilde{\pi}$, GLS penalizes (increments the penalty of the feature) all the features present in π which have maximum utility, $U(\tilde{\pi}; u, v)$, which defined as

$$U(\tilde{\pi}; u, v) = \frac{c_{uv}(\tilde{\pi})}{1 + \phi(u, v)}. \quad (4)$$

The idea is to penalize features, which have high costs first, although the utility of doing so decreases as the feature is penalized more and more times.

3.2 Augmented Cost Function

GLS uses an augmented cost function to guide the local search algorithm out of the local minimum. The idea behind it is to make the local minimum most costly than the surrounding search space, where the feature are not present. The augmented cost function in the GLS for QAP can be defined as

$$h(\pi) = c(\pi) + \lambda \sum_{u=1}^n \sum_{v=1}^n I_{uv}(\pi) \phi(u, v) \quad (5)$$

The parameter λ is used to alter the intensification of the search for solutions. A higher value for λ will result in a more diverse search, where plateaus and basins in the search space are searched more coarsely; a low value of λ will result in a more intensive search for the solution.

3.3 Local Search

The local search used in this paper is the 2-opt heuristic. Let π be a solution, its 2-opt neighborhood $N(\pi)$ is defined as the set of all possible solutions resulting from π by swapping two of the elements. The 2-opt heuristic searches the neighborhood of the current solution for a lower cost solution. If such a solution is

found, it replaces the current solution and the search continues. Otherwise, the algorithm returns a local minimum. There are several different ways in searching the neighborhood. In our experiments, the entire neighborhood is searched and a solution with the lowest cost is selected. The Local Search can be described as follows:

LocalSearch(π, f)

Input: an initial solution π and an objective function f .

Output: $\tilde{\pi}$: a local minimum of f .

Step 1 Set $\tilde{\pi} = \pi$

Step 2 Find a solution π' with the smallest f value in $N(\tilde{\pi})$.

Step 3 If $f(\tilde{\pi}) - f(\pi') \leq 0$, return $\tilde{\pi}$; otherwise, set $\tilde{\pi} = \pi'$ and go to Step 2.

3.4 Efficient Computation of Cost Functions

In the local search for the QAP, the changes of the original cost function and the augmented cost function after a swap can be calculated efficiently. Let π' be a resultant solution from swapping elements r and s from π . Denote

$$\begin{aligned}\Delta c(\pi, r, s) &= c(\pi) - c(\pi'), \\ \Delta h(\pi, r, s) &= h(\pi) - h(\pi').\end{aligned}$$

Then we have

$$\begin{aligned}\Delta c(\pi, r, s) &= \\ & (a_{rr} - a_{ss})(b_{\pi(s)\pi(s)} - b_{\pi(r)\pi(r)}) + \\ & (a_{rs} - a_{sr})(b_{\pi(s)\pi(r)} - b_{\pi(r)\pi(s)}) + \\ & \sum_{k=1, k \neq r, s}^n ((a_{kr} - a_{ks})(b_{\pi(k)\pi(s)} - b_{\pi(k)\pi(r)}) \\ & + (a_{rk} - a_{sk})(b_{\pi(s)\pi(k)} - b_{\pi(r)\pi(k)})),\end{aligned}$$

and

$$\begin{aligned}\Delta h(\pi, r, s) &= \Delta c(\pi, r, s) + \lambda\{\phi(r, \pi(s)) + \phi(s, \pi(r))\} \\ & - \{\phi(r, \pi(r)) + \phi(s, \pi(s))\}.\end{aligned}$$

When A and B are symmetric,

$$\Delta c(\pi, r, s) = 2 \sum_{k=1, k \neq r, s}^n (a_{rk} - a_{sk})(b_{\pi(s)\pi(k)} - b_{\pi(r)\pi(k)}).$$

3.5 Guided Local Search

The pseudocode of the GLS for the QAP is given in the following.

GuidedLocalSearch($\tilde{\pi}, c, \lambda$)

Input: an initial solution $\tilde{\pi}$, the objective function c for the QAP, the parameter λ .

Output: a solution π^* .

Step 1 Set $\phi(i, j) = 0$ for all $1 \leq i, j \leq n$. Set $\pi^* = \tilde{\pi}$ and $\bar{\pi} = \tilde{\pi}$.

Step 2 $\bar{\pi} =$

$$\mathbf{LocalSearch}(\bar{\pi}, c(\pi) + \lambda \sum_{u=1}^n \sum_{v=1}^n I_{uv}(\pi)\phi(u, v))$$

Step 3 If $c(\pi^*) - c(\bar{\pi}) > 0$, set $\pi^* = \bar{\pi}$.

Step 4 If the stopping condition is met, return π^* .

Step 5 For each i in $\{1, 2, \dots, n\}$ such that $U(\bar{\pi}; i, \bar{\pi}(i))$ (see (4)) is maximized, set

$$\phi(i, \bar{\pi}(i)) := \phi(i, \bar{\pi}(i)) + 1.$$

Go to Step 2.

In step 1, the penalty for each feature is set to zero, the best solution found so far π^* and the current solution $\bar{\pi}$ is set to be the initial solution $\tilde{\pi}$, Step 2 call the local search on the current solution with respect to the augmented objective function. In Step 3, the best solution found so far π^* is set to $\bar{\pi}$ if $\bar{\pi}$ has the lowest original cost function value found so far. If the stopping condition is met in Step 4, GLS returns π^* , the solution with the lowest original cost found during the search. Step 5 selects features with maximum utility in the current solution and increase the penalty for these selected features.

4 EDA/GLS for the QAP

At each generation t , EDA/GLS maintains a population of N solutions $Pop(t) = \{\pi^1(t), \pi^2(t), \dots, \pi^N(t)\}$ and a probability matrix

$$P(t) = \begin{pmatrix} p_{11}(t) & \cdots & p_{1n}(t) \\ \vdots & & \vdots \\ p_{n1}(t) & \cdots & p_{nn}(t) \end{pmatrix}.$$

$P(t)$ models the distribution of the solutions in $Pop(t)$.

4.1 Initialization

EDA/GLS starts with N random solutions and then applies GLS to each of them. The N resultant solutions $\{\pi^1(0), \pi^2(0), \dots, \pi^N(0)\}$ of GLS constitute the initial population $Pop(0)$.

4.2 Computation of Probability Matrix

Assume that the population at generation t is $Pop(t) = \{\pi^1(t), \pi^2(t), \dots, \pi^N(t)\}$, the probability matrix $P(t)$ can be obtained as follows:

$$p_{ij}(t) = \frac{1}{N} \sum_{k=1}^N I_{ij}(\pi^k(t)), \quad (1 \leq i, j \leq n). \quad (6)$$

Therefore, $p_{ij}(t)$ is the percentage of the solutions in $Pop(t)$ that uses feature (i, j) .

4.3 Generation of New Solutions: Guided Mutation

The guided mutation mutates an existing solution guided by a probability matrix $P = (p_{ij})_{n \times n}$ and a positive parameter $\delta < 1$. The guided mutation works as follows:

GuidedMutation(π, P, δ)

Input: a solution $\pi = (\pi(1), \dots, \pi(n))$, a probability matrix and a positive parameter $\delta < 1$.

Output: $\sigma = (\sigma(1), \dots, \sigma(n))$, a solution.

Step 1 Randomly pick up $[\delta n]$ integers uniformly from $\{1, 2, \dots, n\}$ and let these integers constitute a set $K \subset I$. Set $V = I \setminus K$.

Step 2 For each $i \in K$, set $\sigma(i) = \pi(i)$ and $U = U \setminus \{\pi(i)\}$.

Step 3 While($U \neq \Phi$) do:

Select a i from V , then randomly pick up a $k \in U$ with probability

$$\frac{p_{ik}}{\sum_{j \in U} p_{ij}}.$$

Set $\sigma(i) = k$, $U = U \setminus \{k\}$ and $V = V \setminus \{i\}$.

Step 4 Return σ .

The goal of the guided mutation is to generate a solution σ (assignment). Step 1 randomly divide the departments into two groups. The first group has $[\delta n]$

departments and the second group has $n - [\delta n]$ departments. In Step 2, department i in the first group is assigned to site $\pi(i)$, which is the site for this department in solution π . Step 3 arranges the departments in the second group sequentially according to the probability matrix. To apply the above guided mutation at generation t in EDA/GLS, we take a solution π from $Pop(t)$ to be an input solution and let $P(t)$ to be the input probability matrix. Since the current population $Pop(t)$ are the best solutions EDA/GLS have visited so far, the current probability matrix $P(t)$ contains the global¹ statistical information extracted from $Pop(t)$, the output of the guided mutation could lie in a promising area. We would like to make the following remarks on the guided mutation:

Remark 1: The commonly-used mutation for permutation representation randomly swaps some-selected pairs of elements in a solution π . The probability of an arrangement σ being the resultant solution is determined by the distance between π and σ . By contrast, the mutation direction in some mutations for real value vectors can be controlled by a matrix. In the guided mutation, the mutation direction is randomly determined by the probability matrix. Therefore, the guided mutation can be regarded as a generalization of these mutations in the case of permutation representation.

Remark 2: Crossover operators often apply to two solution and generate one or two offspring solutions. An offspring takes pieces from both parents. However, crossovers do not use the global information. In guided mutation, part of σ is from the input solution π and the other part of π is determined by the probability matrix P . Therefore, the guided mutation can be regarded as a crossover applied to π and P . The guided mutation combines the information from the individual solution (π) and the global information about the probability distribution of the best solutions found so far (P).

Remark 3: In ant colony algorithms and other EDAs such as compact genetic algorithms, new trial points are sampled a probability model. These algorithms ignore the locations of the best individual solutions found so far. In comparison, the guided mutation directly make use of solutions in the current population.

4.4 Restarting Strategy

In EDA/GLS, if the average cost of the population does not decrease for successive L generations, EDA/GLS will re-initialize its population. Since

¹by ‘‘global’’, we mean that all the members in $Pop(t)$ have made contributions towards $P(t)$.

EDA/GLS has intensively exploited the current area, new solutions should be from the current population as far as possible. Let $P = (p_{ij})$ be the current probability matrix, EDA/GLS generates a new solution as follows.

REstart(P)

Input: $P = (p_{ij})$: a probability matrix.

Output: $\pi = (\pi(1), \dots, \pi(n))$, a solution.

Step 1 Set $U = \{1, 2, \dots, n\}$

Step 2 For $i = 1, 2, \dots, n$

Randomly pick up a $k \in U$ with probability

$$\frac{[1 - p_{ik}]}{\sum_{j \in U} [1 - p_{ij}]}$$

Set $\pi(i) = k$ and $U = U \setminus \{k\}$.

Step 3 Guided Local Search

$$\pi = \mathbf{GuidedLocalSearch}(\pi, c, \lambda).$$

Step 4 Return π .

Obviously, the larger p_{ij} is, the smaller the probability of π having feature (i, j) in the above procedure. Therefore, π could be far from the current population.

4.5 Structure of EDA/GLS

The flow of EDA/GLS is described as follows:

Step 0 Parameter Setting Population Size: N .

The number of new solutions generated at each generation: M . The control parameter in **GuidedMutation**: δ . The control parameter in **GuidedLocalSearch**: λ .

Step 1 Initialization Set $t := 0$. Generate the initial population $Pop(0)$ by the method of Section. Compute the probability matrix $P(0)$ using (6). Set π^* to be the solution with the lowest cost c in $Pop(0)$.

Step 2 Guided Mutation For $j = 1, 2, \dots, M$, do:

Pick up a solution π from $Pop(t)$.

Mutation

$$\sigma = \mathbf{GuidedMutation}(\pi, P(t), \delta).$$

Guided Local Search

$$\sigma^j = \mathbf{GuidedLocalSearch}(\sigma, c, \lambda).$$

Step 3 New Population Choose the N best solutions from $\{\sigma^1, \dots, \sigma^M\} \cup Pop(t)$ to form $Pop(t+1)$. Set $t := t+1$. Set π^* to be the solution with the lowest cost c in $Pop(t)$. Computing the probability matrix using (6).

Step 4 Stopping Condition If the stopping condition is met. Stop. Return π^* .

Step 5 Restart Condition If the Restart Condition is not met, goto Step 2.

Step 6 Restart For $j = 1, 2, \dots, N$, set $\pi^j = \mathbf{REstart}(P(t))$. Set $Pop(t) = \{\pi^1, \pi^2, \dots, \pi^N\}$. Find the lowest cost solution σ^* in $Pop(t)$. If $c(\pi^*) > c(\sigma^*)$, set $\pi^* = \sigma^*$. Computing the probability matrix using (6). Goto Step 2.

To generate a new solution, Step 2 calls **GuidedMutation** to generate a solution σ and then applies GLS on σ . Thus, Step 2 mainly exploits the area where the best solutions found so far are from. When the Restart Condition is met, i.e., EDA/GLS has been trapped in a local area, it calls **REstart** to generate a new population in Step 6, which is far from the old population. Therefore, Step 6 is mainly for exploration.

5 Computational Experiments

In this section, EDA/GLS is compared with GLS through experiments on test problems from QAPLIB. Both algorithms were implemented in C++. All experiments mentioned in were performed on identical PCs (AMD Athlon 750MHZ) running Linux. The parameter settings were the following:

EDA/GLS:

- Population Size $N = 40$, the number of new solutions generated at each generation: $M = 20$. The control parameter in the guided mutation $\delta = 0.3$.
- The control parameter in GLS $\lambda = \frac{3}{5n^4} \sum_{i=1}^n \sum_{j=1}^n a_{ij} \times \sum_{i=1}^n \sum_{j=1}^n b_{ij}$ as suggested in [8]. The stopping condition of GLS is that the number of swapping exceeds $20n$.
- The restart condition is that the average cost of the population does not decrease for consecutive 30 generations.
- The stopping condition is that the CPU time exceed a time limit T .

GLS:

- The control parameter $\lambda = \frac{3}{5n^4} \sum_{i=1}^n \sum_{j=1}^n a_{ij} \times \sum_{i=1}^n \sum_{j=1}^n b_{ij}$.
- The stopping condition is that the CPU time exceed a time limit T .

The experimental results is given in the following time.

Table 1: Experimental Results of EDA/GLS vs. GLS

instance	best known	GLS	EDA/GLS	t/s
els19	17212548	0.000	0.000	5
chr25a	3796	14.54	2.391	30
bur26a	5426670	0.007	0.000	40
nug30	6124	0.182	0.000	40
kra30a	88900	1.347	0.000	40
ste36a	9526	1.274	0.041	60
tai60a	7208572	1.435	1.209	180
tai80a	13557864	0.980	0.887	360
tai100a	21125314	0.915	0.779	600
sko100a	152002	0.226	0.066	600
tai60b	608215054	1.135	0.132	180
tai80b	818415043	1.207	0.513	360
tai100b	1185996137	0.888	0.135	600
tai150b	498896643	0.765	0.351	1200
tho150	8133484	0.238	0.091	1200
tai256c	44759294	0.100	0.042	2400

In this table, *instance* denotes the name of QAP instance from QAPLIB (the number in the name indicates its size). QAPLIB is a public library for the QAP (www.opt.math.tu-graz.ac.at/qaplib). The *best known* solution is provided in the second column. For GLS and EDA/GLS columns, the number means the average percentage excess over the best-known solution obtained within 10 runs, and t/s indicates the time in seconds used in each run. The results indicate that EDA/GLS is better than GLS for all test problem in terms of solution quality within a given time limit.

6 Conclusion

Guided Local Search (GLS) [1] is a general meta-heuristic that sits on top of local search procedures and help them escape from local minima. Often, GLS there is a high possibility that GLS is trapped in a relatively small region in the search space. This paper proposed a combination of Estimation of Distribution Algorithm (EDAs) and GLS (EDA/GLS) for solving the QAP. A novel reproduction operator called guided mutation was used in the proposed algorithm. EDA/GLS use the guided mutation and GLS to search a local search

extensively. The restart strategy leads EDA/GLS to a new area which is far from the area EDA/GLS has been trapped. We compared EDA/GLS with GLS on several QAP instances with different sizes up to 256. Results show that the EDA/GLS outperforms GLS and in all the test instances cases. Our future work will be to apply EDA/GLS for solving other combinatorial optimization problems and compare it with other algorithms.

Acknowledgement: This work was supported by EPSRC under Grant GR/R64742/01.

References

- [1] Voudouris, C.: *Guided Local Search for Combinatorial Optimization Problems*. Ph.D. thesis, Department of Computer Science, University of Essex, (1997).
- [2] Mills, P., Tsang, E.P.K.: *Guided Local Search for Solving SAT and Weighted MAX-SAT Problems*. Journal of Automatic Reasoning, Special Issue on Satisfiability Problems, Kluwer, Vol.24, (2000) 205-223
- [3] Kilby, P., Prosser, P., Shaw, P.: *Guided Local Search for the Vehicle Routing Problem*. Proceedings of the 2nd International Conference on Metaheuristics, July 1997.
- [4] Tsang, E.P.K., Voudouris, C.: *Fast Local Search and Guided Local Search and their application to British Telecom's Workforce Scheduling Problem*. Operations Research Letters, Elsevier Science Publishers, Amsterdam, Vol.20, No.3, 119-127, March 1997.
- [5] Voudouris, C., Tsang, E.P.K.: *Solving the Radio Link Frequency Assignment Problem using Guided Local Search*. Proceedings of NATO Symposium on Radio Length Frequency Assignment, Sharing and Conservation System (Aerospace), Aalborg, Demark, October 1998,
- [6] Voudouris, C.: *Guided Local Search - An Illustrative Example in Function Optimization*. BT Technology Journal, Vol.16, No.3, July 1998, 46-50
- [7] Voudouris, C., Tsang, E.P.K.: *Guided Local Search and Its Application to the Travelling Salesman Problem*. European Journal of Operational Research, Anbar Publishing, Vol.113, Issue 2, March 1999, 469-499
- [8] Mills, P., Tsang, E.P.K., Ford, J.: *Applying an Extended Guided Local Search to the Quadratic*

Assignment Problem, submitted to Annals of OR (revised June 2002).

- [9] Lau, T.L.: *Guided Genetic Algorithm*. Ph.D. Thesis, Department of Computer Science, University of Essex, 1999.
- [10] Corne D., Dorigo, M., Glover, F. (eds.): *New Ideas in Optimisation*, McGraw-Hill (1999)
- [11] Mühlenbein, H., Paaß, G.: *From Recombination of Genes to the Estimation of Distribution Algorithms I. Binary Parameters*, H.M.Voigt, et al.(eds.): Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature - PPSN IV (1996) 178-187.
- [12] Shorin-Kapov,J.: *Tabu Search Applied to the Quadratic Assignment Problem*, ORSA Journal on Computing, Vol.2, No.1, 91990)33-45
- [13] Maniezzo, V, Colormi, A., Dorigo, M.: *The Ant System Applied to the Quadratic Assignment Problem*, Tech. Rep. 94/28, IRIDIA, Université de Bruxelles (1994)
- [14] Nissen, V.: *Solving the Quadratic Assignment Problem with Clues from Nature*. IEEE Transactions on Neural Networks, Vol.5, No.1, (1994) 66-72
- [15] Tate., D.M., Smith, A.E.: *A Genetic Approach to the Quadratic Assignment Problem*. Computers and Operations Research, Vol.22, No.1 (1995) 78-83
- [16] Ishii, S., Sato, M.: *Constrained Neural Approaches to Quadratic Assignment Problems*. Neural Networks, Vol.11, (1998) 1073-1082.
- [17] Merz, P., Freisleben, B.: *Fitness Landscapes and Memetic Algorithm Design*, in D. Corne, M. Dorigo, F. Glover (eds.): *New Ideas in Optimisation*, McGraw-Hill (1999).