Meta-Heuristic Combining Prior, Online and Offline Information for the Quadratic Assignment Problem

Jianyong Sun, Qingfu Zhang and Xin Yao

***Abstract*— The construction of promising solutions for $\mathcal{NP}$-hard combinatorial optimisation problems (COPs) in meta-heuristics is usually based on three types of information, namely *a priori* information, *a posteriori* information learned from visited solutions during the search procedure, and online information collected in the solution construction process. Prior information reflects our domain knowledge about the COPs. Extensive domain knowledge can surely make the search effective, yet it is not always available. Posterior information could guide the meta-heuristics to globally explore promising search areas, but it lacks of local guidance capability. On the contrary, online information can capture local structures, its application can help exploit the search space. In this paper, we studied the effects of using these information on meta-heuristic's algorithmic performances for the COPs. The study was illustrated by a set of heuristic algorithms developed for the quadratic assignment problem (QAP). We first proposed an improved scheme to extract online local information, then developed a unified framework under which all types of information can be combined readily. Finally, we studied the benefits of the three types of information to meta-heuristics. Conclusions were drawn from the comprehensive study, which can be used as principles to guide the design of effective meta-heuristic in the future.**

***Index Terms*— Meta-heuristics, quadratic assignment problem, fitness landscape analysis, online local information, offline global information.**

I. Introduction

One of the key points in designing effective meta-heuristics for $\mathcal{NP}$-hard combinatorial optimisation problems is how to acquire useful information to build promising solutions. Acquired information in literatures is usually fallen into three categories, namely *a priori* information, *a posteriori* information learned from visited solutions, and online information collected during the solution construction process.

First of all, it is intuitive that if we have as much as possible extensive knowledge about the considered problem, we can develop an optimisation algorithm as effective as possible [25]. The *a priori* information reflects our domain knowledge about the considered problem. A useful prior information, if any, can significantly improve the search effectiveness and efficiency[1]. For example, the location-assignment desirability has been successfully applied in ant systems for the quadratic assignment problem [7]. In [62], the prior information about the characteristics of the cardinality of the maximum clique problems has shown its significant usefulness to the effectiveness of the designed meta-heuristics. Moreover, it is claimed in [21] that the heuristic domain knowledge is very helpful for evolutionary algorithms to find good approximate solutions of the node covering problem. Unfortunately, the *a priori* information is not always available for $\mathcal{NP}$-hard COPs: we usually do not have any knowledge about the characteristics of the global optimum solutions.

Second, most widely-used information in existing heuristics is the information extracted from the visited solutions during the search, i.e. the posterior information. First, in genetic algorithms (GA) [19], selection operation is used to harvest promising solutions from the visited solutions, while crossover and mutation operators attempt to create better-fit offspring. The posterior information takes effect implicitly through the use of selection, crossover and mutation. On the other hand, in tabu search (TS) [17] and its variants [10][24], the visited solutions (or moves) are forbidden in further search in order to avoid cycling; while in particle swarm optimisation (PSO) [16][26], new offspring are created based on the location information of the current local best and global best solutions. The posterior information, i.e. the found best solutions, is applied directly. In some recently developed algorithms for the QAP, such as the self controlling tabu search algorithm [13] and the consultant-guided search algorithm [23], kinetic global information is collected, shared and propagated among individuals. In the probability model based evolutionary algorithms (PMBEAs), such as estimation of distribution algorithms (EDAs) [29], ant colony optimisation (ACO) [6][8], cross entropy [53], useful information is extracted and represented as a posterior probability distribution. The posterior probability distribution models the visited promising solutions during the search. It represents the global statistical information extracted from previous search.

To create a solution with discrete variables by the probability distribution, one needs to select proper elements for the components of the solution following an order of these components. The selection of elements depends on the order and the probability values of the variables. In the context

J. Sun is with School of Engineering, Computing and Applied Mathematics, The University of Abertay Dundee, Dundee, DD1 1HG. E-mail: j.sun@abertay.ac.uk.

Q. Zhang is with School of Computing and Electrical Systems, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ. E-mail: qzhang@essex.ac.uk.

Xin Yao is with CERCIA, School of Computer Science, The University of Birmingham, Edgbaston, Birmingham, B15 2TT. E-mail: x.yao@cs.bham.ac.uk.

[1]As clarified in [42], the 'effectiveness' of an optimisation algorithm refers to the quality of the solutions found or its robustness in finding desired solutions. The 'efficiency' characterizes the runtime behavior of the optimisation algorithm. In this paper, we focus on developing an effective optimisation algorithm, but do not consider its efficiency.

of EDAs for the COPs, the order of the components implies the structure of variable dependencies. The dependency structure is often represented by a Bayesian network [29]. In EDAs, such as the univariate marginal distribution algorithm (UMDA) [43] [45] and FDA [44] and others, the structure of the variable dependencies is fixed. While in most other EDAs [47] [29], the Bayesian network needs to be inferred from the promising solutions. As well known, the structure inferring is an $\mathcal{NP}$-hard optimisation problem itself, it is then not wise to introduce an auxiliary $\mathcal{NP}$-hard problem for the aim of solving an $\mathcal{NP}$-hard problem. Alternatively, some probability model based heuristics, such as ant colony optimisation and guided mutation [62] [64], have been proposed while the structure of the variable dependencies is not learned, but is applied in a random manner when sampling solutions. These algorithms have shown their successes in solving a number of combinatorial optimisation problems [9] [64].

Specifically, focusing on the solution construction procedure, a discrete solution $\mathbf{x} = [x_1, \cdots, x_n]^T$ is usually constructed starting from an empty solution. Following a fixed, learned or random dependency structure, i.e. a Bayesian network π, elements are sequentially selected, proportionally to its probability or conditional probability values specified by the probability distribution p. Suppose that a *partial solution* $\tilde{\mathbf{x}} = [x_{\pi_1}, \cdots, x_{\pi_k}]$ ($k = 0$ means that we need to build a solution from scratch) has been created, the selection of an element for $x_{\pi_{k+1}}$ is proportional to conditional probability $p(x_{\pi_{k+1}}|x_{pa_{\pi_{k+1}}})$, where $pa_{\pi_{k+1}}$ is the parent variables of $x_{\pi_{k+1}}$ specified by the structure. Obviously, the selection of an element based on $p(x_{\pi_{k+1}}|x_{pa_{\pi_{k+1}}})$ has no awareness about the local structure of the search space.

In literature, apart from the probability information used in solution construction process, information on local structure has been applied. For example, in greedy adaptive randomised search procedure (GRASP), developed by Feo et al. [12][51], the solution construction is based on some sort of local information. To select an element for the π_{k+1}-th variable, GRASP computes problem-specific greedy function values of the candidate elements (i.e. those elements that can make the solution feasible) and selects candidates according to these values. GRASP attempts to exploit the local structure of the search space. It can be seen that the the selection of a candidate element is based on the instantaneous computation of the greedy function during the construction process rather than learned from visited solutions as in the above PMBEAs. We therefore call this information as *online local information*. In contrast to the online information, the statistical information used in PMBEAs is referred to as *offline global information*. A different type of online local information could be captured through approximation of (local) fitness landscape [33].

In this paper, we intend to study the effectiveness of the three kinds of information to find high quality solutions, and the possibility of combing these information to develop effective meta-heuristics. The study was illustrated by solving the well-known $\mathcal{NP}$-hard QAP. We developed a unified multistart algorithmic framework, in which prior information, online local information and offline global information are incorporated. Various meta-heuristics with different incorporated information can be derived from the unified framework. This allows us to readily study the effects of these information to the performance of the meta-heuristics.

The rest of the paper is organised as follows. Firstly, a new greedy function for the QAP was defined for the online information collection in Section II. It improves the original greedy function developed by Li et al. [32]. Secondly, the guided mutation operator was briefly introduced in Section III which is able to cooperate different information. The underlying assumption of the guided mutation was also verified statistically in this section. Thirdly, a variant of ant system called Max-Min ant system ($\mathcal{MM}$AS) [56] for the QAP, which is used to represent the offline global information, was briefly described in Section IV. The developed unified meta-heuristic framework was described in Section V. Finally, controlled experimental results were given in Section VI to study the effects of the new proposed greedy function, the guided mutation, and the online and offline information. The proposed heuristics were also compared with some other known meta-heuristics for the QAP, including the robust tabu search (Ro-TS) [58], the fast ant colony algorithm (FANT) [60], a variant of iterated local search algorithm (ILS) [55], the GRASP with path-relinking [46] (GPATH), and $\mathcal{MM}$AS [56]. Section VII concluded the paper.

II. Online Local Information

In literatures, few algorithms have been developed based on online local information. Greedy adaptive randomised search procedure (GRASP), initially developed by Feo et al. [12] [51], is an excellent example of these algorithms particularly for a class of COPs, such as assignment problems, routing problems, and others. We follow the definition of these problems by Resende and Riberio [51]. That is, these problems can be defined by a finite set $E = \{1, 2, \cdots, n\}$, a set of feasible solutions $F \subset 2^{|E|}$, and an objective function $f : 2^{|E|} \rightarrow \mathcal{R}$ such that $f(S) = \sum_{e \in S} c(e)$, $\forall S \in 2^{|E|}$, where $c(e)$ is the cost associated with the inclusion of element $e \in E$ in the solution S. The quadratic assignment problem is one of them.

As shown in Alg. 1, GRASP constructs a solution step by step from scratch. To construct a solution S, a problem-specific greedy function g is firstly defined. The solution construction process starts (step 1) from an empty solution $S = \Phi$. Candidate element set (denoted by C) is initialised to be all possible elements in E. In step 3, the greedy function values of all candidate elements in C, i.e. $g(c), c \in C$, are calculated. A restricted set of candidate elements, called restricted candidate list (RCL), is then selected from C based on the values in step 4. In step 5, an element s is randomly selected from the RCL. Step 6 updates the set of available elements, and the partial solution. The solution construction is proceed until a full solution is constructed. To improve the generated solution, local search is applied. It can be seen that local information (the benefits of selecting a certain elements measured by the greedy function) collected online (in step 3) is used to construct a solution in GRASP. Basic GRASP iterates the construction process until satisfying solutions have been met.

Algorithm 1 The solution construction procedure of GRASP.

```
Set S := Φ; C := E.
while S is not complete do
    Evaluate g(c) for all c ∈ C;
    Build RCL from C based on g(c), c ∈ C;
    Select element s from the RCL at random;
    S := S ∪ s, C := C \ {s};
end while
return solution S.
```

In the following, GRASP developed for the QAP by Li et al. [32] is briefly introduced, following the proposed greedy function. Given two $n \times n$ matrices $F = (f_{ij})$ and $D = (d_{kl})$, and a set $E = \{1, 2, \cdots, n\}$, the QAP can be stated as follows:

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

where Π is the set of all permutations of E, $F = (f_{ij})$ is the flow matrix where f_{ij} is the flow of materials from facility i to facility j; and $D = (d_{kl})$ is the distance matrix with d_{kl} denotes the distance from location k to location l. In a permutation π, $\pi_j = i$ means that facility i is assigned at location j. The objective of the QAP is to find the optimal assignment of all facilities to all locations.

The GRASP implementation proposed by Li et al. includes two stages [32]. In the first stage, GRASP selects two pairs of facility-location. To proceed, the $n^2 - n$ distance entries in D are firstly sorted in an ascending order. We keep the smallest $\lfloor \beta(n^2 - n) \rfloor$, where $\lfloor x \rfloor$ is the largest integer smaller or equal to x. Secondly, the $n^2 - n$ flow entries in F are sorted in descending order, and the $\lfloor \beta(n^2 - n) \rfloor$ largest are kept. Thirdly, the cost of the interactions $f_{ij} \times d_{kl}, 1 \leq i, j, k, l \leq n, i \neq j, k \neq l$ are sorted and the smallest $\lfloor \alpha\beta(n^2 - n) \rfloor$ elements are reserved for the facility-location pair selection, where α and β are parameters of the GRASP. Note that the above list only need to be done once for the use in the rest GRASP iterations. Two facility-location pairs, say $(j_1, l_1), (j_2, l_2)$, are randomly selected from the $\lfloor \alpha\beta(n^2 - n) \rfloor$ candidate pairs. In stage 2 of the GRASP, the rest facilities are assigned to the rest locations sequentially.

Suppose that at an intermediate construction step, a set of r facility-location pairs $\Gamma = \{(j_1, l_1), (j_2, l_2), \cdots, (j_r, l_r)\}$ has been assigned. The pairs in Γ indicate a partial solution. To select a new facility-location pair, the greedy function of assigning facility i to location k w.r.t.. the already-made assignments is computed as follows [32]:

$$\mathcal{C}_{ik} = \sum_{(j,l) \in \Gamma} f_{ij} d_{kl} \text{ for } (i, k) \notin \Gamma$$

The RCL in the present step includes the $\lceil \alpha(n^2 - n - r) \rceil$ facility-location pairs with the smallest $\mathcal{C}_{ik}$ values. The $(r+1)$-th facility-location pair is then randomly selected from the RCL, say (j_{r+1}, l_{r+1}). The already-made assignment set Γ will be updated as $\Gamma = \Gamma \cup \{(j_{r+1}, l_{r+1})\}$. The above procedure will terminate until a full solution is constructed.

In the above construction procedure, the time complexity of the first stage is $\mathcal{O}(n^2)$. In stage two, at each step, the greedy function values of all the unassigned facility-location pairs need to be calculated and sorted. This would make the solution construction computationally time-consuming. However, it is doubtable whether the calculation is necessary or not. We here propose a new greedy function without requiring the intensive computing as follows. To generate a permutation π, we iteratively assign randomly selected unfilled-location with unassigned facility, until all locations are filled. The benefit of assigning a facility to a location is measured by a new greedy function described as follows. Suppose that at an intermediate construction step, the set of already-assigned facilities is $\mathcal{A}$, and the set of already-filled locations is $\mathcal{B}$. To select a facility for a randomly unassigned location $k \in E \setminus \mathcal{B}$, we compute the greedy function of assigning a facility $i \notin \mathcal{A}$ to location k as follows:

$$\ell_{ik} = \sum_{s \in \mathcal{A}} f_{i\pi_s} d_{ks}. \tag{2}$$

If $i \in \mathcal{A}$, then the cost ℓ_{ik} is set to ∞. The same as general GRASP implementation, the facility to be assigned to location k can only come from the facilities with $\alpha(n - |\mathcal{B}|)$ smallest values, i.e., the RCL for location k, RCL_k is defined as:

$$RCL_k = \{i \notin \mathcal{A} | \ell_{ik} \leq \underline{\ell}_k + \alpha(\bar{\ell}_k - \underline{\ell}_k)\}. \tag{3}$$

where $\underline{\ell}_k = \min_{i \notin \mathcal{A}} \ell_{ik}$ and $\bar{\ell}_k = \max_{i \notin \mathcal{A}} \ell_{ik}$.

In the above greedy solution construction procedure, only $\mathcal{O}(n)$ values need to be calculated in each construction step rather than $\mathcal{O}(n^2)$ as in the original one. Based on the new greedy function, a new GRASP can thus be proposed. We refer to the new GRASP as G^{new} in this paper.

GRASP also incorporates a local search algorithm which is used to improve the quality of the constructed solution to local optimum. A number of local search algorithms have been applied in literature, such as variable neighbourhood search [20][61], tabu search [18], k-opt local search [32], simulated annealing [61], etc. In G^{new} and all the proposed heuristics in this paper, the first-improvement 2-opt local search algorithm (called $\mathbf{LS}_2$) is applied, its pseudo code is shown in Alg. 2. It searches its neighborhood $\mathcal{N}_2(s)$, and updates the current solution with the firstly-found better solution in the neighbourhood. For the QAP, $\mathcal{N}_2(s)$ is defined as the set of all solutions that can be obtained by swapping two elements in solution s.

Algorithm 2 The first-improvement 2-opt local search.

```
for s not locally optimal → do
    Find a better solution t ∈ N_2(s);
    Let s := t;
end for
return s as a local optimum.
```

The benefit of the new greedy function will be shown in Section VI by comparing G^{new} and the old version GRASP (called G^{old}) on a set of QAP instances within a given time limit.

Basic GRASP iterates the above solution construction process until satisfied solutions have been found. There are no cooperation among these constructed solutions. This could

be a possible shortcoming to the effectiveness of GRASP. A number of GRASP variants have been proposed attempting to incorporate learning mechanisms in the solution construction. For examples, in reactive GRASP, the decision of the RCL is based on the found solutions [49]. Cost perturbation methods introduce noise into the greedy function, where the noise is based on the appearance of elements in the visited solutions [52]. Bias functions have been used to bias the selection of element from the RCL toward some particular candidates [3]. In the intelligent construction method, adaptive memory strategies and the proximate optimality principle (POP) were incorporated [14]. Many techniques, such as hashing [38], filtering [11], variable neighborhood search [20] [52], genetic algorithm [1], tabu search [27], and path-relinking [28], and others, are also hybridized with GRASP. Among these GRASP variants, some of them (e.g. [3][49]) made attempt to incorporate the search history to improve the search efficiency, but only in an implicit way.

III. Guided Mutation

To the best of our knowledge, there are two kinds of prior information for the QAPs adopted in literatures. In an ant systems to the QAP, called AS-QAP [37], the desirability of facility-location assignments is computed according to the flow and distance matrices [37]. This prior (heuristic) information has been adopted in a substantially improved ACO, ANTS-QAP [36]. But most recent ACO variants [15][57][60] choose not to use this information, which implies that it is not as effective as we thought. On the other hand, the proximate optimality principle (POP) proposed by Grover [17], has been implicitly applied in meta-heuristics for the QAP. The POP states that good solutions have similar structure. It is considered as the underlying assumption for most, if not all, heuristics. We have developed the so-called "guided mutation" operator to apply the principle in creating promising solutions. Evolutionary algorithms based on guided mutation have been successfully applied to the maximum clique problem (MCP) [62] and the quadratic assignment problem [63]. Its superiority over other mutation operators has been demonstrated in solving the MCP [62]. In this paper, we will only use the POP as a prior information.

Guided mutation generates a solution by the combination of posterior probability distribution and the location information of solutions found so far (the actual position of the solutions in the search space). For the QAP, guided mutation builds a permutation based on a probability matrix p and a found best solution π with parameter β. The probability matrix entry p_{ij} indicates the probability of assigning an element i to location j. The construction of a permutation σ was shown in Alg. 3 (cf. [62]).

The success of the guided mutation to solve a certain COP depends on whether the COP holds the principle or not. In literature, there are no theoretical results on the verification of the assumption. We propose to use a fitness landscape analysis method to empirically investigate whether the POP holds to the QAP. In [40] [41], fitness landscape analysis, including autocorrelation analysis and the fitness distance

Algorithm 3 Guided mutation for constructing a permutation.

1: Set $U = I = E$. Randomly pick up $\lfloor \beta n \rfloor$ integers uniformly from U and let these integers constitute a set $K \subset I$. $V := I \setminus K$, i.e. the set of already-filled locations.
2: **for** each $i \in K$, $\sigma_i := \pi_i$ and $U := U \setminus \{\pi_i\}$. **do**
3: **while** $U \neq \Phi$ **do**
4: Select randomly an s from V, then pick up a $k \in U$ with probability $\frac{p_{sk}}{\sum_{j \in U} p_{ij}}$.
5: Set $\sigma_s := k$, $U := U \setminus \{k\}$ and $V := V \setminus \{s\}$.
6: **end while**
7: **end for**
8: **return** σ.

analysis, has been conducted for several QAP instances for the classification of these problem instances. The autocorrelation analysis is applied to investigate the local properties of the fitness landscapes, while the global structures are investigated by employing a fitness distance correlation analysis. However, the fitness analysis in [41] cannot be employed to investigate the proximate optimality principle. A new fitness landscape analysis method was proposed recently [30], [31], but no comprehensive studies have been carried out. The fitness landscape analysis we employed in this paper is as follows.

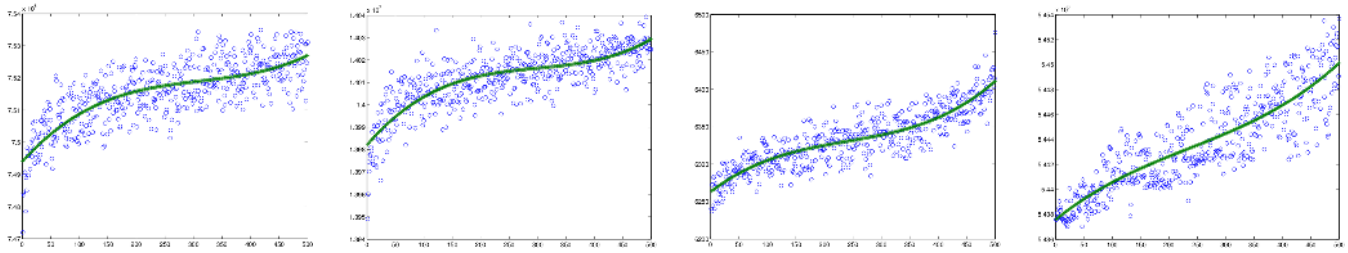
To carry out the analysis, we first randomly generate a set of permutations, and apply the first-improvement 2-opt local search to result in 500 distinctive local optima $\Lambda = \{\pi_1, \cdots, \pi_{500}\}$ with cost values $\mathcal{H} = \{f_1, \cdots, f_{500}\}$ sorted in an ascending order. We perturb each local optimum $\pi_i, 1 \leq i \leq 500$ to form a new permutation. In our experiments, we perturb π_i by randomly exchanging $0.2n$ elements of π_i. The newly-formed solution is then improved to a local optimum. We iterate the perturbation until 1000 distinct local optima $\Sigma_i = [\sigma_{i,1}, \cdots, \sigma_{i,1000}]$ are created. The average fitness of these local optima $\Sigma_i, 1 \leq i \leq 500$ around the original local optima π_i:

$$g_i = \frac{1}{1000} \sum_{j=1}^{1000} c(\sigma_{i,j}) \tag{4}$$

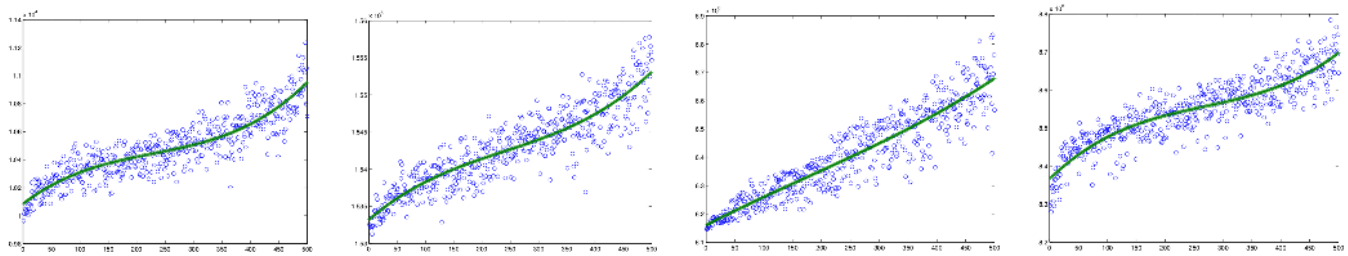
is computed. We then plot $G = \{g_1, \cdots, g_{500}\}$ as a function of the ascending order of the fitness values $\mathcal{H}$ as shown in Fig. 1, taking some QAP instances as examples.

From Fig. 1, it can be seen that the average cost values are highly correlated with the costs of the template solutions. The positive correlation indicates that the proximate optimality principle holds in these QAPs. Moreover, we can claim that statistically there is a reasonable higher chance to find an even better solutions around good solutions rather than around bad solutions. This implies that the guided mutation has the potential to generate promising solutions if we take the best solution found so far as the template solution.

Beside the guided mutation, the POP has also been applied in the iterated local search algorithm (ILS) [55]. ILS mutates a solution by randomly exchanging some items of the solution, rather than by sampling some items of the solution from a probability model as in the guided mutation. We will show the superiority of the guided mutation over ILS in Section VI.



(a) tai60a (b) tai80a (c) nug30 (d) bur26a



(e) ste36a (f) sko100a (g) tai60b (h) tai80b

Fig. 1. The fitness landscape analysis for eight QAP instances: **tai60a**, **tai80a**, **nug30**, **bur26a**, **ste36a**, **sko100a**, **tai60b** and **tai80b**. The x-axis is the ascending order of the generated 500 local optima w.r.t. their fitness values; the y-axis is the average fitness values of 1000 local optima generated around each local optima. The solid lines in the plots are the order-3 interpolation curves.

IV. Offline Global Information

We used a variant of ant colony optimisation (ACO), called $\mathcal{M}$ax-$\mathcal{M}$in Ant System ($\mathcal{MMAS}$) [56], as an example to show the extraction of offline global information (please refer to [5] for an excellent review of ant systems for the QAP). A generic ACO heuristic comprises mainly three components: *heuristic information assignment*, *pheromone trail update* and *solution construction process*. The heuristic information represents *a priori* problem-specific information, and the pheromone trail is the offline global information acquired by ants about the distribution of promising solutions in the evolution process. The pheromone trail will be updated along the evolution process. It is used in the solution construction process to simulate ant's foraging behaviour.

In $\mathcal{MMAS}$, the heuristic information is not used. Suppose at generation t, the best solution found so far is π with cost c_π. The pheromone trail is denoted by $\mathcal{G} = (g_{ij})$, where g_{ij} indicates the probability of assigning facility i at location j. The pheromone trail $g_{ij}(t+1)$ is updated as follows:

$$g_{ij}(t+1) = \rho \cdot g_{ij}(t) + \Delta g_{ij}; \tag{5}$$

where

$$\Delta g_{ij} = \begin{cases} 1.0/c_\pi, & \text{if facility } i \text{ is in location } j \text{ in solution } \pi; \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

and $0 < \rho \leq 1$ is a coefficient representing the so-called *pheromone evaporation* phenomenon and ρ is called the *evaporation coefficient*. The pheromone evaporation is used to avoid the unlimited accumulation of the pheromone trails. Moreover, to avoid the premature convergence of the search, the range of possible pheromone trails on each solution component is limited, i.e. $\tau_{\min} \leq g_{ij}(t) \leq \tau_{\max}$. At each generation, one has to make sure the pheromone trails fall into the range of limits, that is, if $g_{ij}(t) > \tau_{\max}$, set $g_{ij}(t) = \tau_{\max}$; if $g_{ij}(t) < \tau_{\min}$, set $g_{ij}(t) = \tau_{\min}$. As suggested in [57], $\tau_{\max}$ can be adaptively updated as:

$$\tau_{\max} = \frac{1}{1-\rho}\frac{1}{c_\pi}. \tag{7}$$

while $\tau_{\min}$ can be set as some constant factor lower than $\tau_{\max}$.

New offspring is sampled from the probability model $\mathcal{G}$ step by step: at construction step t, firstly a randomly location j is picked from those unassigned, the probability of facility i to be assigned at location j is given by:

$$p_{ij}(t) = \begin{cases} \frac{g_{ij}(t)}{\sum_{k \in \mathcal{S}(j)} g_{kj}(t)}, & i \in \mathcal{S}(j) \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

where $\mathcal{S}(j)$ is the set of facilities that can be placed at location j at the present construction step.

V. The Unified Algorithmic Framework

In this section, we will present a unified algorithmic framework which can facilitate our study on the effectiveness of these information to solve the QAP. To begin with, we first present three approaches to combine the global and local information.

A. Information Combination

On one hand, the success of GRASP for combinatorial optimisation is mainly because of the online local information collected through the use of greedy function (except the local search). To the best of our knowledge, online local information is ignored in almost all probability model based evolutionary algorithms (PMBEAs) except in [54] where solution is constructed dynamically. On the other hand, the success of the PMBEAs is indebted to the use of offline global information. Both GRASP and PMBEAs have achieved great success in solving difficult COPs. However, GRASP lacks of learning mechanism, and PMBEAs do not use local information except for some rare cases where clustering was used in numerical optimisation [35]. It is then worthwhile studying whether we can take the advantages of both information to produce effective algorithms or not.

Recalling the solution construction process in GRASP and $\mathcal{MM}$AS, at a certain step, the filling of a location j depends on the online local information ℓ_{ij} (Eq. 2) and offline global information g_{ij} (Eq. 5) respectively, for those unassigned facilities $i \in S(j)$. However, the online local information and the offline global information are applied in different ways. When the online information is applied, we want to select a facility with relatively small ℓ_{ij} value. On the contrary, the larger the g_{ij} value, the higher the probability of selecting facility i. In order to combine the two kinds of information properly, at a certain construction step, we first change the probability value g_{ij} to $\tau_{\max} - g_{ij}$ for all $i \in S(j)$ and normalise it:

$$q_{ij} = \frac{\tau_{\max} - g_{ij}}{\sum_{k \in S(j)} (\tau_{\max} - g_{kj})}. \tag{9}$$

After the transformation, we see that the smaller value of q_{ij}, the higher probability of selecting facility i. We also normalise $\ell_{ij}, i \in S(j)$ values:

$$r_{ij} = \frac{\ell_{ij}}{\sum_{k \in S(j)} \ell_{ik}}. \tag{10}$$

The normalisation of $\ell_{ij}, i \in S(j)$ is to make a balanced combination of online and global information since the normalized q_{ij} and r_{ij} are both in the range $[0, 1]$, while the un-normalised g_{ij} and ℓ_{ij} will be in different ranges.

To combine online and global information, we propose the following three methods. Suppose that the values provided for the solution construction are represented by $\psi_{ij}, i \in S(j)$ at a construction step for assigning facility to location j, we summarize the three methods as follows.

(1) Crossover. That is, for each $\psi_{ij}, i \in S(j)$,

$$\psi_{ij} = \begin{cases} r_{ij}, & \text{if } rand() < \delta; \\ q_{ij}, & \text{otherwise.} \end{cases} \tag{11}$$

where $rand()$ is a uniform random number in [0,1], δ is a parameter to balance the contribution of the online local information and offline global information.

(2) Linear. For each $\psi_{ij}, i \in S(j)$, it is obtained by combining the two kinds of information in a linear way. That is,

$$\psi_{ij} = \lambda \cdot r_{ij} + (1.0 - \lambda) \cdot q_{ij}. \tag{12}$$

where $\lambda \in [0, 1]$ is a parameter.

(3) Power-law. That is, for each $\psi_{ij}, i \in S(j)$,

$$\psi_{ij} = r_{ij}^{\kappa} q_{ij}^{\gamma}. \tag{13}$$

where κ, γ are parameters.

The control parameters, including δ,λ,κ and γ, in the three combination methods, are used to control the intensities of the online and offline information in calculating ψ. Note that in special case of the parameter settings, it can be degenerated to using only online or offline information (cf. Remark 2 of subsection V-B).

B. The Algorithm

The pseudocode of the proposed meta-heuristics, called Information Combination based Evolutionary Algorithm (**ICEA**), is summarised in Alg. 4.

Algorithm 4 The pseudo-code of the developed ICEA.

Input: Termination Criterion, Restart Criterion
Output: Best solution

```
1:  Compute prior information if possible.
2:  while Termination Criterion not met do
3:      Randomly create a set of solution, set the best solution
        to be empty, global information to be null.
4:      Perform local search over initialised solutions.
5:      Construct new solutions combining available informa-
        tion; Perform local search over newly created solutions.
6:      Update global information.
7:      if Improved strictly then
8:          Update best solution.
9:      else
10:         if Restart Criterion met then
11:             Perform diversification.
12:             Goto line 4.
13:         end if
14:     end if
15: end while
16: return best solution.
```

In the pseudocode, the algorithm iterates until termination criterion has been met. For example, the algorithm could terminate if a maximal number of fitness evaluations or a time limit has been reached. The algorithm starts a new search if the restart criterion has been met. In our implementation, if in consecutive some iterations, there is no update on the best solution, we restart the search. The prior information is computed in line 1, the global information is updated in line 6 after local search over newly created solutions. Online information is computed during the solution construction procedure. Particularly for the QAP, the algorithm can be detailed as in Alg. 5.

In ICEA, the population size is set to 1 as in $\mathcal{MM}$AS [57] and fast any colony [60]. In step 1 of ICEA, the algorithmic parameters are set; the global best solution π_g^* is set to empty with infinite cost. In step 2, the matrix $\mathcal{G}$ representing the global offline information is initialised where the probability of assigning a facility to a location is set equally for all

Algorithm 5 The Developed ICEA for the QAP

Input: Termination Criterion, Restart Criterion

Output: Best solution

1: Set π_g^* to be an empty permutation with cost $c(\pi_g^*) = \infty$. Set the algorithmic parameters, including the restrict candidate list parameter α, the guided mutation parameter β, the evaporation parameter ρ, and the information combination parameter δ (or λ, or κ, γ).

2: Initialisation.

- Initialize the offline information $\mathcal{G}$ with element $g_{ij} = 1.0/n$ for all $1 \leq i, j \leq n$.
- Randomly generate a solution π and improve the solution π = $\mathbf{LS}_2(\pi)$. Update $\mathcal{G}$ according to Eq. (5). Let $\pi^* := \pi$ and $s = \lceil \beta \cdot n \rceil$.

3: Solution construction.

3.1 Randomly select a set of indices $I = \{i_1, i_2, \cdots, i_s\}$ as the set of fixed locations. Copy facilities assigned in π^* to the new solution ν in the fixed locations I, i.e.

$$\nu_i = \pi_i^*, i \in I. \tag{14}$$

Let J be the set of facilities that has been assigned: $J = \{\pi_{i_1}^*, \cdots, \pi_{i_s}^*\}$.

3.2 While $E \setminus I \neq \Phi$, do

3.2.1 Random select an $j \in E \setminus I$;

3.2.2 Calculate the online local information, i.e., ℓ_{ij}, the greedy function value of placing facility i ($i \in S(j) = E \setminus J$) at location j according to Eq. (2).

3.2.3 Normalize $\ell_{ij}, i \in S(j)$ as r_{ij} (cf. Eq. (10)), and $g_{ij}, i \in S(j)$ as q_{ij} (cf. Eq. (9)).

3.2.4 Combine the local information and global information by using Eq. (11) or (12) or (13) to obtain ψ_{ij} for $i \in S(j)$.

3.2.5 Decide the restricted candidate list $\mathcal{T}$, the set of potential facilities which can be assigned at location j; set $\theta = \underline{\psi}_j + \alpha \cdot (\overline{\psi}_j - \underline{\psi}_j)$, where $\underline{\psi}_j = \min_i \psi_{ij}$ and $\overline{\psi}_j = \max_i \psi_{ij}$, and $\mathcal{T} = \{i : \underline{\psi}_i \leq \psi_{ij} \leq \theta\}$

3.2.6 Randomly select a facility $t \in \mathcal{T}$, update $\nu_j := t$ and $I := I \cup \{j\}$ and $J := J \cup \{t\}$.

3.3 Return the new solution ν.

4: Improve the constructed solution ν, let $\nu = \mathbf{LS}_2(\nu)$.

5: Update the global offline information $\mathcal{G}$ by Eq. (5). If any $g_{ij} > \tau_{\max}$, set $g_{ij} = \tau_{\max}$; if $g_{ij} < \tau_{\min}$, set $g_{ij} = \tau_{\min}$. Update the best solution so far π^*: If $c(\nu) < c(\pi^*)$, set $\pi^* := \nu$.

6: Restart. If the restart criterion has been met, set $\pi_g^* := \arg\min(c(\pi^*), c(\pi_g^*))$, goto step 2.

7: Stop Criterion. If the stop criterion is met, stop and return the found best solution π_g^* and its cost. Otherwise goto step 3.

the facility-location pairs. An initial solution is randomly generated and improved by local search to a local optimum π^*. The probability matrix $\mathcal{G}$ is updated, and the current best solution is set to π^*. In step 3, first a set of locations is randomly selected. These locations are filled by facilities copied from the current best solution π^*. The greedy function values will be calculated for these unassigned facilities. It will be combined with the probability model value $\mathcal{G}$ for producing new offspring. This step returns a new solution. The new solution is then improved by local search in step 4. Then $\mathcal{G}$ and the current best solution π^* are updated in step 5. The algorithm iterates from steps 3 and 5 until the restart criterion has been met. The iterates from steps 2 to 5 are called a *cycle*. The algorithm restarts its search in step 6 if current search has been converged, and the global best solution π_g^* is updated by comparing with the current best solution π^*. When the stop criterion has been met, the algorithm stops and returns the best solution found in all the cycles. The algorithm will terminate when the maximum number of fitness evaluations or a given time limit, have been reached.

Remark 1: In the described algorithmic framework, the backbone is GRASP. An element is selected randomly from RCL, which depends on the combined information. Alternatively, we can select an element proportionally to the combined information. That is, step 3.2.5 can be written as follows:

3.2.5 Select an element i for location j proportionally to

$$p_{ij} = \frac{g_{ij}}{\sum_{k \in S(j)} g_{kj}}, i \in S(j) \tag{15}$$

Note that g_{ij} is applied rather than q_{ij} as in the old step 3.2.5. In the following, we do not experimentally study this variants, but leave it as an alternative for future study.

Remark 2: According to different algorithmic parameter settings, ICEA has the following five variants. In case that $\delta = 1$ (or $\lambda = 1$, or $\kappa = 1, \gamma = 0$), no global information is incorporated. The resultant algorithm is called "ILSOL" (Iterated Local Search with Online Local information). Note that the only difference of ILSOL and GRASP is that ILSOL uses guided mutation to create offspring. In case that $\delta = 0$ (or $\lambda = 0$, or $\kappa = 0, \gamma = 1$), no local information is used. If we replace step 3.2.5 with the new step 3.2.5 in Remark 1, the resultant algorithm is similar to the $\mathcal{MM}$AS, except that the guided mutation is applied. We call the resultant algorithm "GANT" (Guided mutation based ant system). Except ILSOL and GANT, three ICEA variants, called "ICEA-CRO" (the cross combination), "ICEA-LIN" (the linear combination) and "ICEA-POW" (the power-law combination), can be derived by setting different δ, λ, or κ and γ values. Moreover, if the guided mutation parameter β is set to zero, no prior information is combined. If we set $\delta = 1$ (or $\lambda = 1$, or $\kappa = 1, \gamma = 0$), G^{new} is recovered. If we set $\delta = 0$ (or $\lambda = 0$, or $\kappa = 0, \gamma = 1$), $\mathcal{MM}$AS is obtained. Table I shows the relationship among these variants and the information used.

Remark 3: The restart, or diversification, strategy is used to avoid the '*closed orbits*" and "*chaotic attractor*" phenomena as discussed in [58]. In some cases, the evolution trajectory endlessly repeat a sequence of states, which indicates that the algorithm is trapped in a closed orbit. Chaotic attractor

TABLE I
THE INFORMATION USED BY THE HEURISTICS.

type of information	$\mathcal{MM}$AS	G^{new}/GRASP	ICEA				
			GANT	ILSOL	-LIN	-CRO	-POW
global	Yes	No	Yes	No	Yes	Yes	Yes
local	No	Yes	No	Yes	Yes	Yes	Yes
prior	No	No	Yes	Yes	Yes	Yes	Yes

indicates that the search trajectory is confined in a limited portion of the search space. To avoid the phenomena, a simple method is to restart the search procedure. In the proposed algorithms, if in executive 100 generations no better solution can be found, the algorithms will restart.

The experimental comparison among the ICEA variants, and some other heuristics on some QAP instances will be given in the following section.

VI. EXPERIMENTAL RESULTS

To study the proposed algorithms and compare them with some best-known algorithms, the proposed algorithms will be applied to solve a set of QAP instances with size up to 256 from QAPLIB [4]. As claimed in [59], the type of a QAP instance has very significant influence to the performance of meta-heuristics. In this paper, the proposed algorithms have several parameters. The performances of these algorithms would depend on these parameters, and consequently the selection of these parameters may depend on the types of the QAP instances. We follow the QAP instances' classification in [59] to classify these instances as follows:

(I) Unstructured, randomly generated instances. In these QAP instances, elements in the distance and flow matrices are randomly generated according to a uniform distribution.
(II) Real-life instances. Instances of this class are from real-life applications of the QAP.
(III) Grid-based distance matrix. Elements in the distance matrix are defined as the Manhattan distance between grid points.
(IV) Real-life like instances. Elements in the matrices of this class are generated from a distribution which is similar to the distribution found in real-life problem.

In this section, instances from the above four classes were used to compare among different algorithms and the parameters were determined according to the classes. The comparison was divided into three stages.

- Stage 1. G^{new} was compared with G^{old}, and a purely random start algorithm (called pRand). In pRand, solutions were independently and randomly generated, and were improved to local optima by $\mathbf{LS}_2$. The comparison was used to show the benefit of the proposed greedy function (cf. subsection VI-A).
- Stage 2. ILSOL and GANT were compared with G^{new}, $\mathcal{MM}$AS with $\mathbf{LS}_2$, and a variant of the iterated local search (ILS) algorithm. Note that the only difference between ILSOL & G^{new}, and GANT & $\mathcal{MM}$AS, is that the prior information (guided mutation) is applied in the previous ones, but not in the later ones. Hence, the comparison can be used to show the benefit of the prior information to the performance of the heuristics (cf. subsection VI-B). The comparison with ILS was to answer whether or not the online and offline information does improve the search capability.
- Stage 3. The ICEA variants, including ILSOL, GANT, ICEA-CRO, ICEA-LIN and ICEA-POW, were compared with each other in order to study the benefits of the global, local information, and their combination to effective search. They were also compared with some other known algorithms, including the robust tabu search (Ro-TS) [58], FANT, the GRASP with path relinking [46] (GPATH), and the $\mathcal{MMAS}$ with tabu search (cf. subsection VI-C). The comparison was to investigate whether the information combination could result in effective meta-heuristics.

To fairly compare the considered algorithms, we have used the codes developed by the corresponding authors, and run them on our machines with the algorithmic parameters suggested in these papers.

A. The New Implementation of GRASP

The authors in [14] [55] claimed that GRASP outperforms pRand when both the algorithms terminate at a given number of solution evaluations. However, it is not clear which one perform better in terms of solution quality within a given time, since it is obvious that the time used for solution generation by pRand is shorter than GRASP. The comparison in this section is used to clear the open question. Moreover, we concern on the performance of G^{new}, in which the new greedy function is adopted.

In Table II, the experimental results of pRand, G^{old} (the basic GRASP for the QAP developed in [32]), and G^{new} are listed. There are two versions of G^{old} by Oliveria et al. [46] and Li et al. [32], respectively. We found that the implementation by Oliveria et al. achieved better performance. Hence the implementation of Oliveria et al. was used to carry out the comparison. The parameter settings of G^{old} is the same as in [46], while the algorithmic parameters of G^{new} is the same as those of G^{old}.

In the table (and the following tables), t indicates the time given for different QAP instances, and the best results are typeset in bold. The algorithms terminate when the given time has been reached. Entries in the table and the following tables are the average percentage of the found solutions excess over the best-known solutions over 30 runs, $avg.\%$ is the average value of the entries in each column for the corresponded algorithm.

According to the $avg.\%$ values in the table, we can see that on average, (1) the new GRASP is superior to all the others;

TABLE II

THE COMPARISON RESULTS OBTAINED BY G^{old}, G^{new} AND PRAND WITHIN A GIVEN TIME ON SOME QAP INSTANCES. THE BEST RESULTS ARE TYPESET IN BOLD.

instances	pRand	G^{new}	G^{old}	t
randomly generated instances				
tai20a	0.447	**0.198**	0.442	5
tai25a	0.936	**0.792**	1.272	15
tai30a	**1.302**	1.414	1.464	20
tai35a	**1.621**	1.627	1.768	60
tai40a	**1.867**	1.901	2.124	60
tai50a	2.474	**2.440**	2.494	90
tai60a	2.632	**2.442**	2.736	90
tai80a	**2.166**	2.174	2.208	180
tai100a	2.032	**2.026**	2.130	300
tai256c	2.127	**0.210**	0.237	1200
avg. %	1.764	**1.668**	1.687	/
real-life instances				
chr25a	7.444	**5.389**	6.870	15
bur26a	0.000	0.000	0.000	15
kra30a	0.601	**0.156**	0.572	20
kra30b	**0.120**	0.126	0.236	20
ste36a	**0.982**	1.077	1.440	30
ste36b	**0.126**	0.272	1.090	30
avg. %	1.545	**1.170**	1.701	/
instances with grid-based distance matrix				
nug30	0.209	**0.150**	0.317	20
sko42	0.413	**0.407**	0.508	60
sko49	0.560	**0.519**	0.569	60
sko56	0.520	**0.516**	0.670	90
sko64	**0.589**	0.622	0.713	90
sko72	0.679	**0.627**	0.823	120
sko81	0.690	**0.666**	0.837	120
sko90	0.682	**0.670**	0.818	180
sko100a	0.647	**0.590**	0.779	300
avg. %	0.554	**0.529**	0.670	/
real-life like Instances				
tai20b	0.000	0.000	0.000	5
tai25b	0.007	**0.000**	0.007	15
tai30b	**0.032**	0.090	0.111	20
tai35b	1.437	**0.171**	0.200	60
tai40b	0.021	**0.008**	0.010	60
tai50b	0.182	**0.166**	0.273	90
tai60b	0.226	**0.202**	0.267	90
tai80b	**0.883**	0.969	1.033	180
tai100b	**0.595**	0.698	0.825	300
tai150b	1.373	**1.313**	1.529	600
avg. %	0.476	**0.348**	0.399	/

(2) GRASP is only comparable with the pure random restart algorithm. It seems that the online information does not really contribute to the search. This observation seems contradict the claim in the beginning of this section that GRASP outperforms pRand. To explain this, we record the number of local search the algorithms carried out within the given time and plot the results in Fig. 2. In the figure, the x-axis shows the QAP instances sorted in ascending order w.r.t. the number of local optima visited by the new GRASP (G^{new}); the y-axis is the number of visited local optima. From the figure, we can see that within a given time, pRand conducts much more $\mathbf{LS}_2$ than G^{old}, but only a few more than G^{new}. This indicates that the extra time used by the solution construction procedure in G^{old} cannot compensate for the solution quality.

On the other hand, although the number of local optima that G^{new} conducted is always smaller than those of pRand, the performance of G^{new} is still better than that of pRand on average. Note that G^{new} chooses the locations to be assigned randomly rather than being decided along with the construction step as in G^{old}, we can then claim that the random order of locations does not necessarily deteriorate the search performance, and the online local information collected by the new greedy function can indeed benefit the search.

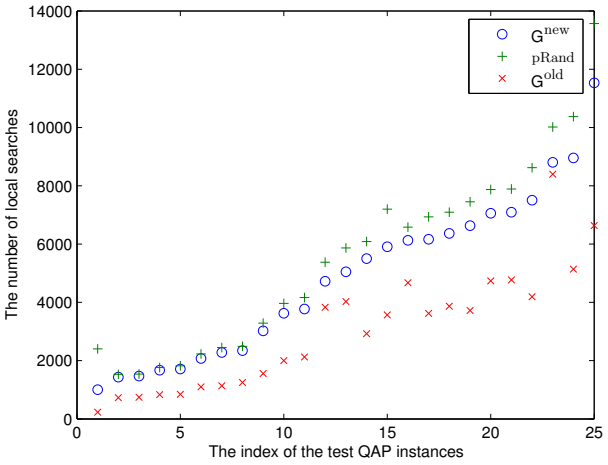


Fig. 2. The comparison of the number of local searches performed by G^{new}, G^{old}, and pRand within given time limits.

B. ILSOL and GANT

In the ICEA variants, ILSOL and GANT create new offspring by applying the guided mutation. The online local information and the offline global information are used, respectively. This section is to answer the following two questions. The first is whether the search can benefit from the guided mutation operator. The second is whether or not the incorporation of the guiding information as in ILSOL and GANT is better than random exchanging as used in ILS, as already proposed in section III.

To answer the questions, we compare the related algorithms, including ILSOL, G^{new}, GANT, a variant of $\mathcal{MM}$AS and a variant of ILS. In the ILS variant, the "better" acceptance criterion and the restart strategy are applied (please see [56] for details). That is, when a better solution is found during the search, the current best solution is updated and mutated for a new solution; when the algorithm cannot find a better solution in some executive generations, the algorithm will restart. The variant is called "ILS_r^b" following [56]. $\mathcal{MM}$AS developed in [57] has two variants with different local search approaches, including the first-improvement 2-opt local search ($\mathcal{MM}\text{AS}_{LS}$), and tabu search ($\mathcal{MM}\text{AS}_{TS}$). In this section, $\mathcal{MM}\text{AS}_{LS}$ was used. The reason that we used ILS_r^b and $\mathcal{MM}\text{AS}_{LS}$ to carry out the comparison is that they are the same as GANT and ILSOL respectively except the guiding information. The comparison between ILSOL & G^{new}, and GANT & $\mathcal{MM}\text{AS}_{LS}$ can answer the first question, while the comparison among ILSOL, GANT and ILS_r^b can be used to answer the second question.

The algorithmic parameters of these algorithms were set as follows:

- The algorithmic parameters of ILSOL include the restricted candidate list parameter α and the guided mutation parameter β. They are set differently with respect to different QAP classes. For instances in the unstructured, randomly generated class, $\alpha = 0.1$, $\beta = 0.9$ (i.e., 90% items of the current best solution are fixed). For most instances in the other three classes, $\alpha = 0.3$, $\beta = 0.9$. For the QAP instances with smaller problem sizes including tai20a, tai20b, tai25b, chr25a, bur26a, $\beta = 0.8$.
- For $\mathcal{MM}$AS$_{LS}$, $\tau_{\max}$ is set adaptively as in Eq. (7), $\tau_{\min} = 2\tau_{\max}/n$, the evaporation parameter $\rho = 0.8$.
- For GANT, the guided mutation parameter β is set to be 0.9 for large-size QAP instances, and 0.8 for the QAP instances with size less than 26. The evaporation parameter ρ is 0.8. $\tau_{\max}$ and $\tau_{\min}$ are the same as in $\mathcal{MM}$AS$_{LS}$.
- The parameter settings of ILS$_r^b$ follow the settings in [56].

All the algorithms terminate when the given time limits as shown in Table II have been reached. Experimental results obtained by these algorithms are listed in Table III.

From the average values listed in the tables, we have the following observations:

(1) ILSOL clearly outperforms G^{new} in all the four classes;
(2) GANT outperforms $\mathcal{MM}$AS$_{LS}$ for the QAP instances in type I, and shows a slightly better performance for the QAP instances in other types;
(3) both ILSOL and GANT are superior to ILS$_r^b$ for all the QAP instances;
(4) The performance of ILSOL is slightly better than GANT for the QAP instances in type I and IV, is slightly worse in type III, and is the same as in type II.

Since the only difference between ILSOL and G^{new} (GANT and $\mathcal{MM}$AS$_{LS}$) is that POP is applied in ILSOL and GANT, we can claim that the use of POP can indeed improve the search effectiveness according to observations (1) and (2). Moreover, since ILS$_r^b$ uses random exchanging to mutate a solution, while ILSOL and GANT use collected information as guidance, we can claim that the incorporation of the guiding information indeed improve the performances of meta-heuristics according to observation (3). Finally observation (4) indicates that the capabilities of the online and offline information are generally about the same. Specifically, it seems that the the local information (ILSOL) works better if the QAP instances have complex structures as in types I and IV.

TABLE III
THE EXPERIMENTAL RESULTS OF ILS$_r^b$, ILSOL, G^{new}, GANT AND $\mathcal{MM}$AS$_{LS}$ ON THE TEST QAP INSTANCES WITH DIFFERENT CLASSES. THE BEST RESULTS ARE TYPESET IN BOLD.

instances	ILS$_r^b$	ILSOL	G^{new}	GANT	MMAS$_{LS}$	t
randomly generated instances						
tai20a	0.248	0.192	0.198	0.345	**0.183**	5
tai25a	0.785	**0.262**	0.792	0.298	0.284	15
tai30a	0.958	**0.276**	1.414	0.289	0.394	20
tai35a	1.046	0.529	1.627	**0.480**	0.698	60
tai40a	1.015	0.790	1.901	**0.743**	0.777	60
tai50a	1.577	1.211	2.440	**1.178**	1.217	90
tai60a	1.619	1.408	2.442	**1.226**	1.353	90
tai80a	1.501	**1.009**	2.174	1.111	1.891	180
tai100a	1.396	0.996	2.026	**0.947**	1.061	300
tai256c	0.098	**0.082**	0.210	0.261	1.245	1200
avg.%	1.127	**0.675**	1.668	0.688	0.910	
real-life instances						
chr25a	2.718	**0.000**	5.389	**0.000**	**0.000**	15
bur26a	0.000	0.000	0.000	0.000	0.000	15
kra30a	0.101	**0.000**	0.156	**0.000**	**0.000**	20
kra30b	0.040	**0.000**	0.126	**0.000**	**0.000**	20
ste36a	0.283	**0.000**	1.077	**0.000**	**0.000**	30
ste36b	**0.000**	**0.000**	0.272	**0.000**	**0.000**	30
avg.%	0.524	**0.000**	1.170	**0.000**	**0.000**	
instances with grid-based distance matrix						
nug30	0.042	**0.000**	0.150	**0.000**	**0.000**	20
sko42	0.074	**0.000**	0.407	**0.000**	**0.000**	60
sko49	0.137	0.053	0.519	**0.049**	0.056	60
sko56	0.247	0.080	0.516	0.064	**0.062**	90
sko64	0.174	0.057	0.622	**0.009**	0.011	90
sko72	0.227	0.159	0.627	**0.137**	0.149	120
sko81	0.326	0.188	0.666	0.185	**0.179**	120
sko90	0.270	0.239	0.670	**0.200**	0.224	180
sk0100a	0.284	**0.153**	0.590	0.246	0.270	300
avg.%	0.198	0.102	0.129	**0.098**	0.106	
real-life like Instances						
tai20b	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	5
tai25b	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	15
tai30b	**0.000**	**0.000**	0.090	**0.000**	**0.000**	20
tai35b	0.018	**0.000**	0.171	**0.000**	**0.000**	60
tai40b	**0.000**	**0.000**	0.008	**0.000**	**0.000**	60
tai50b	0.233	**0.008**	0.166	0.115	0.013	90
tai60b	0.483	**0.002**	0.202	0.004	0.021	90
tai80b	0.367	**0.221**	0.969	0.254	0.291	180
tai100b	0.193	**0.153**	0.698	0.211	0.190	300
tai150b	0.616	0.614	1.313	**0.580**	0.691	600
avg.%	0.182	**0.086**	0.348	0.116	0.121	

C. ICEA: Study of Information Incorporation

It is shown in the last section that the incorporation of both online information and offline information does benefit the performance of the corresponding algorithms. In this section, we intended to study whether the combined information can be more useful for designing effective heuristics.

The proposed ICEA provides us a chance to study the effects of information incorporation. The ICEA variants, including ILSOL, GANT, ICEA-CRO, ICEA-LIN and ICEA-POW, use different probability information to produce new offspring. In this section, we compare them with each other, and with some known heuristics, including *robust tabu search algorithm* (Ro-TS) [58], *fast ant colony algorithm* (FANT) [60], *GRASP with path-relinking algorithm* (GPATH) [46], *$\mathcal{M}$ax-$\mathcal{M}$in ant system* with tabu search ($\mathcal{MM}$AS$_{TS}$) [56].

For the ICEA variants, different parameter settings were used. The parameter settings of ILSOL and GANT have been described in Section VI-B. For ICEA-POW, $\alpha = 0.6$, $\rho = 0.8$; κ and $\gamma = 1.0$. For ICEA-CRO, the parameter settings are: $\alpha = 0.3$, $\rho = 0.8$ and $\delta = 0.6$. For ICEA-LIN, $\alpha = 0.5$, $\rho = 0.6$ and $\lambda = 0.5$.

The comparison results are listed in Table IV. In the table, the given time limits are the same as in the above experiments. Figs. 3 and 4 show the evolution process of the algorithms taking four QAP instances from each class as examples. In the figures, the x-axis is the time steps, while the y-axis shows the objective function values. From the figures, it can be observed that the performances of the developed algorithms vary along with the different QAP classes. Detailed observations are as

follows:

(1) On average, all the ICEA variants perform clearly better than Ro-TS for the QAP instances in types (II), (III), but worse for the QAP instances in type I in terms of solution quality within a given time limit. The linear combination variant ICEA-LIN performs better than Ro-TS for type IV instances. But the other variants, especially GANT, performs worse than Ro-TS in type IV. The performance of Ro-TS for the QAP instances from type I is the best among these algorithms except $\mathcal{MM}\text{AS}_{TS}$.
(2) The ICEA variants clearly compare favorably against FANT and GPATH. Actually, on average, among the nine algorithms, GPATH performs the worst. FANT performs better than Ro-TS for the real-life instances (type II).
(3) Among the nine algorithms, the tabu search variant of $\mathcal{MM}$AS achieved the best performance. The performance of $\mathcal{MM}\text{AS}_{TS}$ is significantly better than the ICEA variants for type I and III QAP instances, but similar to those of the ICEA variants for type IV instances.
(4) Comparing the ICEA variants, we can see that for type I instances, ILSOL obtained the best average results, and the combined versions of ICEA are worse than ILSOL and GANT; for type III instances, ICEA-POW and ICEA-CRO perform better than ICEA-LIN and GANT but worse than ILSOL; for type IV instances, ICEA-POW and ICEA-LIN work better than the other three variants.

Observation (1) indicates that Ro-TS is superior to most of the ICEA variants in terms of solution quality within a given time limit for the QAP instances in type I. The observation is actually not surprising. As we already knew [42], the QAP instances in type I are completely unstructured (high epistasis and high landscape ruggedness). Since the proposed algorithms are learning-based heuristics which are designed to exploit some kind of global / local structure through a learned distribution model of the promising solutions in the QAP search space, it can be expected that these algorithms would perform better on the structured problem instances like the QAP instances in type II, than in the QAP instances with complicated structures (type I). The observation that FANT performs better than Ro-TS for type II QAP instances (observation (2)), and the better performance of the ICEA variants for the type II-IV QAP instance provide us more evidences to verify that learning-based meta-heuristics can indeed work well on well-structured (type II) and unstructured (but not completely unstructured, e.g. types III and IV) QAP instances.

Observation (2) indicates that the variants of ICEA are superior to FANT and GPATH in terms of solution quality on all the QAP instances. This again proves the effectiveness of the application of prior information (cf. Section VI-B). This also shows that the information combination has the potential to develop effective meta-heuristics.

Focusing on the comparison among the combined information variants of ICEA (observation (4)), the better performance of ILSOL against GANT for the type I QAP instances indicates that the online local information is significantly useful in solving unstructured QAP instances. While the observation that combined information variants (ICEA-POW and ICEA-LIN) are superior to GANT and ILSOL in solving type III and IV instances indicates that the global information is useful in solving structured problem instances. This can be explained by noticing that the global properties of the structured QAP instances can be easily learned. For the unstructured instances, the learned global information can be misleading for the new offspring generation since the local optima obtained by $\mathbf{LS}_2$ in the search space can be totally uncorrelated.

Observation (4) states that ILSOL is superior to GANT, and GANT is superior to the ICEA variant with combined information for type I QAP instances. This tells us that the combined information is not useful for completely unstructured QAP instances. It is probably because that the combined global information distracts the online information which can guide the search to a promising area. Moreover, we found that the crossover combination is not as effective as the linear and power-law combination for the QAP instances with complex structures (type I and type IV).

Comparing the results of $\mathcal{MM}\text{AS}_{LS}$ listed in Table III with those of the ICEA variants in Table IV, we can see that the ICEA variants obtained better results than $\mathcal{MM}\text{AS}_{LS}$ for all the QAP instances (observation (3)). On the contrary, the $\mathcal{MM}$AS with tabu search performs favorably over the ICEA variants. Hence, the performance difference between $\mathcal{MM}\text{AS}_{LS}$ and $\mathcal{MM}\text{AS}_{TS}$ must due to the use of the different local search algorithms. The use of tabu search can gain much better local solutions since it does not stop in local optima. Due to the better solutions, offspring sampled from the extracted posterior probability model could be more accurate since we have a better understanding of the fitness landscape of the QAP search space. Note that tabu search has been proven to be more effective on the QAP landscapes [42]. Moreover, even though the extracted probability model is not accurate, the application of tabu search on offspring solutions can still guarantee a fairly good local optima. That explains why $\mathcal{MM}\text{AS}_{TS}$ can perform better than the others. In the future, it is worth studying the hybridization of tabu search in the proposed algorithmic framework.

D. Algorithmic Parameter Settings

As well known, to decide on an appropriate set of parameter values is always a difficult problem for meta-heuristics [34]. In our study, we adjusted the parameters of the developed algorithms using a greedy method similar to the coordinate descent method [2]. The parameters are considered as discrete variables, that is, each parameter can only take from a set of predefined values. The average objective function value over 10 runs for a set of parameters is considered as the objective function for parameter tuning. The search for appropriate parameter set is iterated as follows. At each iteration, the setting of a particular parameter is to choose the parameter value with the least objective function value. In the following iteration, this parameter is fixed. The iteration continues until all the parameters do not vary.

According to our experiences in adjusting the parameters of the ICEA variants, it seems that the guided mutation parameter

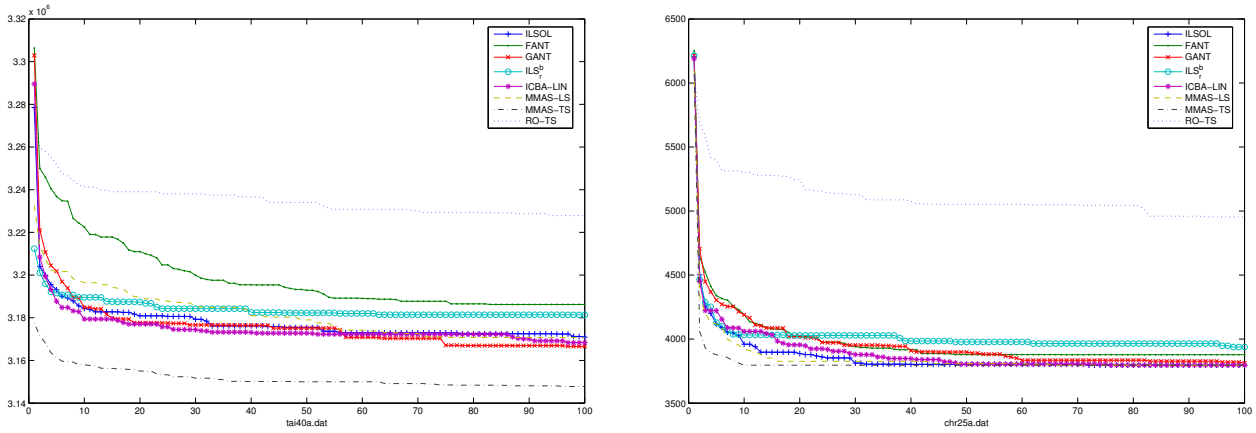


Fig. 3. The evolution process of the considered algorithms, including ILSOL, FANT, GANT, ILS$_r^b$, ICEA-LIN, $\mathcal{MM}$AS-LS, $\mathcal{MM}$AS-TS and Ro-TS, for tai40, chr25a. The x-axis is the time, while the y-axis is the objective function.

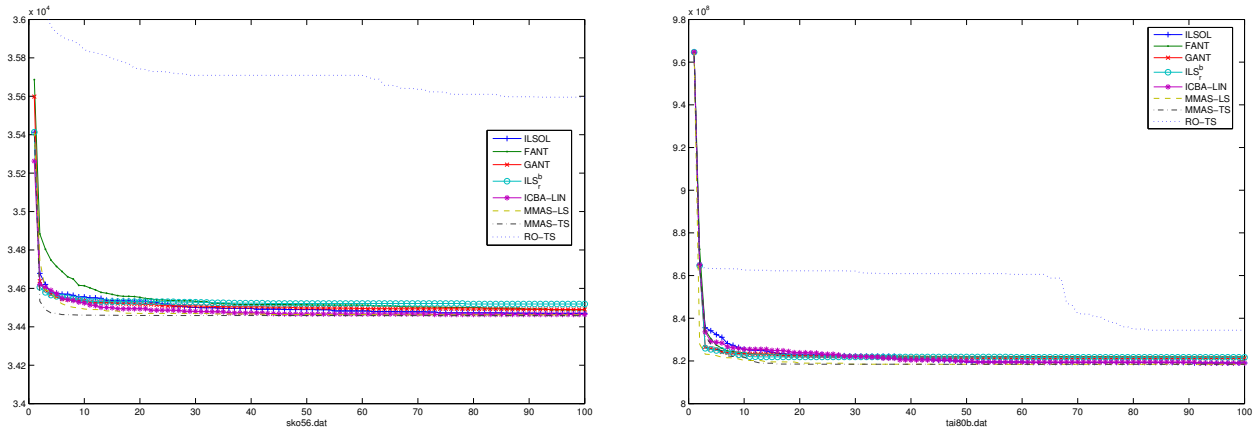


Fig. 4. The evolution process of the considered algorithms, including ILSOL, FANT, GANT, ILS$_r^b$, ICEA-LIN, $\mathcal{MM}$AS-LS, $\mathcal{MM}$AS-TS and Ro-TS, for sko56 and tai80b. The x-axis is the time, while the y-axis is the objective function.

β is the easiest to be set. The usual value for β is 0.9 for the QAP instances with size $n \geq 30$. For small-size QAP instances, β can be set to 0.8. Note that the β value indicates the number of components to be statistically fixed during the search. This large value setting implies that the number of local minima in the QAP instances are extremely large even for the small-size QAP instances. The reason is that if there is less local minima, the large β value will cause the improvement of a perturbed solution to an already-visited local minima which will no doubt deteriorate the performance of the algorithm.

For ρ, the evaporation coefficient, it can be set between [0.6,0.8]. The large value indicates that we need to be careful about the forgetting of previous history and including of new knowledge. In many ant systems for the COPs, the ρ is set a similar value.

For α, the parameter that controls the restricted candidate list, its setting depends on what information combination method is used. If there is no global information as in ILSOL, the value is in the range of [0,0.3]. If the information combination method is used, its setting seems correlated to the other parameters, such as δ, or κ, γ, λ w.r.t. the information combination method. It is intuitive since the performance of the algorithm largely depends on the balance of the global and local information. It is these parameters that defines the compromise.

VII. Conclusion and Future Work

In this paper, the effects of different kinds of information to the performances of the heuristics were investigated. A unified framework, named ICEA, was proposed in which different kinds of information are incorporated to produce promising solutions. The incorporated information types are *a priori information*, *online local information* and *offline global information*.

The developed ICEA was applied to the QAP as a case study. In the ICEA variants, the online local information collection method by GRASP (greedy randomised adaptive search procedure) was improved and adopted. The pheromone trail update method in $\mathcal{MM}$AS was adopted as the offline

TABLE IV

THE EXPERIMENTAL RESULTS OF ICEA VARIANTS, RO-TS, FANT, GRASP WITH PATH RELINKING, $\mathcal{MM}$AS WITH TABU SEARCH. THE BEST RESULTS ARE TYPESET IN BOLD.

instances	Ro-TS	FANT	GPATH	ICEA					MMAS$_{TS}$	t
				ILSOL	GANT	POW	CRO	LIN		
randomly generated instances										
tai20a	**0.000**	0.405	0.345	0.192	0.345	0.182	0.168	0.182	**0.000**	5
tai25a	0.129	0.968	0.872	0.262	0.298	0.216	0.268	0.201	**0.000**	15
tai30a	0.187	1.023	1.183	0.276	0.289	0.568	0.454	0.393	**0.000**	20
tai35a	0.389	1.195	1.267	0.529	0.480	0.518	0.707	0.705	**0.006**	60
tai40a	0.628	1.171	1.760	0.790	0.743	0.883	0.964	0.834	**0.402**	60
tai50a	1.083	1.906	2.026	1.211	1.178	1.279	1.300	1.232	**0.682**	90
tai60a	1.169	2.646	2.460	1.408	1.226	1.390	1.271	1.405	**0.902**	90
tai80a	0.989	2.562	2.171	1.009	1.111	1.148	1.510	0.998	**0.746**	180
tai100a	0.899	2.561	2.205	0.996	0.947	0.987	1.080	0.984	**0.696**	300
tai256c	0.234	0.263	0.276	**0.082**	0.261	0.115	0.120	0.112	0.271	1200
Avg.	0.570	1.470	1.456	0.675	0.688	0.728	0.784	0.705	**0.371**	
real-life instances										
chr25a	2.808	1.243	3.746	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	15
bur26a	**0.000**	0.008	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	15
kra30a	**0.000**	0.552	0.090	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	20
kra30b	**0.000**	0.015	0.040	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	20
ste36a	0.010	0.487	0.613	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	30
ste36b	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	30
Avg.	0.469	0.384	0.748	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	
instances with grid-based distance matrix										
nug30	**0.000**	0.091	0.072	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	20
sko42	**0.000**	0.038	0.193	**0.000**	**0.000**	**0.000**	**0.000**	0.002	**0.000**	60
sko49	0.073	0.155	0.355	0.053	0.049	0.082	0.063	0.055	**0.000**	60
sko56	0.037	0.094	0.365	0.080	0.064	0.073	0.065	0.052	**0.035**	90
sko64	0.043	0.203	0.468	0.057	0.009	0.022	0.041	0.107	**0.009**	90
sko72	0.126	0.299	0.503	0.159	0.137	0.170	0.133	0.158	**0.083**	120
sko81	**0.113**	0.272	0.499	0.188	0.185	0.157	0.135	0.186	0.121	120
sko90	**0.103**	0.403	0.512	0.239	0.200	0.158	0.230	0.209	0.108	180
sko100a	0.109	0.347	0.496	0.153	0.246	0.186	0.175	0.204	**0.107**	300
Avg.	0.168	0.214	0.385	0.088	0.098	0.094	0.093	0.108	**0.051**	
real-life like Instances										
tai20b	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	5
tai25b	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	15
tai30b	**0.000**	**0.000**	0.003	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	20
tai35b	**0.000**	0.047	0.023	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	60
tai40b	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	60
tai50b	0.008	0.239	0.062	0.008	0.115	**0.000**	0.001	0.001	0.010	90
tai60b	0.019	0.008	0.088	**0.002**	0.004	0.005	0.012	0.014	0.009	90
tai80b	0.195	0.385	0.259	0.221	0.254	0.168	0.195	0.154	**0.150**	180
tai100b	0.172	0.193	0.432	0.153	0.211	0.180	0.305	0.160	**0.115**	300
tai150b	0.441	0.818	1.022	0.674	0.580	0.524	0.597	**0.432**	0.481	600
Avg.	0.084	0.169	0.187	0.106	0.116	0.087	0.111	**0.076**	0.077	

global information collection method. The guided mutation, with the proximate optimality principle (the prior information) as the underlying assumption, was used to produce new offspring. In the ICEA variants, either only online local information (ILSOL), or offline global information (GANT), or the combination of online and offline information (ICEA-CRO, ICEA-LIN and ICEA-POW), was used. The prior information, i.e. the guided mutation, was used in all the ICEA variants.

In the experimental studies, firstly, we investigated the significance of the proposed new greedy function to the performance of the meta-heuristic. Secondly, the benefits of different types of information were compared. Finally, experiments were carried out to compare algorithms among the variants of ICEA and other known algorithms. From the experiment results, we have the following conclusions:

- The new proposed greedy function is more effective than the old greedy function on collecting online local information;
- The prior information (POP) has significant effect on generating promising offspring;
- The online local information has more significant effect on designing efficient heuristics for completely unstructured QAP instances, while the offline global information extracted from the $\mathbf{LS}_2$ local optima is more suitable for the QAP instances with relatively simple structure.
- It is not wise to combine the global information with the online information in solving completely unstructured QAP instances in case a cheap local search algorithm like $\mathbf{LS}_2$ is applied. However, the information combination can indeed result in effective meta-heuristics for structured QAP instances.
- The linear and power-law information combination methods perform better than the crossover method on average.

Since there are many other global information collection methods rather than $\mathcal{MM}$AS, we will embed these methods into the ICEA framework for information combination in

the future. Furthermore, the observation that $\mathcal{MM}$AS with tabu search performs well implies that the global structure of completely unstructured QAP instances could be learned from the local optima obtained by expensive local search algorithm e.g. tabu search. Therefore, we will study the information combination by hybridising other local search or even meta-heuristic methods, such as tabu search and guided local search [63] in ICEAs. Moreover, the effect of the information combination on the performance of probability model based evolutionary algorithms will be further investigated by applying the information combination method to other difficult COPs, such as traveling salesman problem, frequency assignment problem, routing problems [39], etc.

To apply the information combination framework to a COP, we are expecting that the solution of the COP can be constructed gradually from partial solution. Moreover, we expect some sort of local information that can be extracted during the solution construction process, and global information as references from visited solutions. The global information may not be limited to the form of probabilistic model as presented in the paper. Other forms, such as crossover and mutation, can also be applied. With the local and global information, a proper method to combine them is required. From our study, we can say that the investigation of the information combination has the potential to achieve highly effective and efficient meta-heuristics for COPs.

Finally, since a multi-start strategy is applied in the developed framework, the diversification scheme will have a critical effect on the algorithmic performance. In this paper, only a random diversification scheme is applied. In the future, we will work on developing a sophisticated diversification scheme, including a portfolio of complementary algorithms [48]. Moreover, the developed algorithm will be applied to some real application, such as the hybrid electric vehicle [22], fuzzy controllers [50], and others.

ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for their helpful and constructive suggestions and comments. JS was supported by the National Natural Science Foundation of China (No.61273313). XY was supported by EPSRC (grant no. EP/I010297/1) and a Royal Society Wolfson Research Merit Award.

REFERENCES

[1] R.K. Ahuja, J.B. Orlin, and A. Tiwari. A greedy genetic algorithm for the quadratic assignment problem. *Computers and Operations Research*, 27:917–934, 2000.

[2] J.C. Bezdek, R.J. Hathaway, R.E. Howard, C.A. Wilson, and M.P. Windham. Local Convergence Analysis of a Grouped Variable Version of Coordinate Descent. *Journal of Optimization Theory and Applications*, 54(3):471–477, 1987.

[3] J.L. Bresina. Heuristic-biased stochastic sampling. In *Proc. of the Thirteenth National Conference on Artificial Intelligence*, pages 271–278, Portland, USA, 1996.

[4] R.E. Burkard, S. Karisch, and F. Rendl. QAPLIB - a quadratic assignment problem library. *European Journal of Operational Research*, 55:115–119, 1991. Updated version: http://www.imm.dtu.dk/ sk/qaplib.

[5] M. Dorigo, M. Birattari, and T. Stützle. Ant colony optimization. Technical Report TR/IRIDIA/2006-023, IRIDIA, 2006.

[6] M. Dorigo, V. Maniezzo, and A. Colorni. Positive feedback as a search strategy. Technical Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1991.

[7] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: Optimization by a colony of cooperating agents. *IEEE Trans. on Systems, Man, and Cybernetics: Part B*, 26(1):29–41, 1996.

[8] M. Dorigo and T. Stützle. *The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances*, chapter 9, pages 251–285. International Series in Operations Research and Management Science. Boston, MA: Kluwer Academic Publisher, 2003.

[9] M. Dorigo and T. Stützle. *Ant Colony Optimization*. The MIT Press, 2004.

[10] Z. Drezner. Extensive experiments with hybrid genetic algorithms for the solution of the quadratic assignment problem. *Computers & Operations Research*, 35:717–736, 2008.

[11] T.A. Feo, K. Sarathy, and J. McGahan. A greedy randomized adaptive search procedure for maximum independent set. *Operations Research*, 42:860–878, 1994.

[12] T.A. Feo, K. Venkatraman, and J. Bard. A GRASP for a difficult single machine scheduling problem. *Computers & Operations Research*, 18(8):635–643, 1991.

[13] N. Fescioglu-Unver and M. M. Kokar. Self controlling tabu search algorithm for the quadratic assignment problem. *Computers & Industrial Engineering*, 60:310–319, 2011.

[14] C. Fleurent and F. Glover. Improved constructive multistart strategies for the quadratic assignment problem using adaptive memory. *INFORMS Journal of Computing*, 11:198–204, 1999.

[15] L. Gambardella, É. Taillard, and M. Dorigo. Ant colonies for the QAP. *Journal of the Operations Research Society*, 50:167–176, 1999.

[16] H. Gao and W. Xu. A New Particle Swarm Algorithm and Its Globally Convergent Modifications. *IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics*, 41(5):1334–1351, October 2011.

[17] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, 1998.

[18] F. W. Glover and G. A. Kochenberger, editors. *Handbook of Metaheuristics*. Kluwer Academic Publishers, October, 2003.

[19] D. E. Goldberg, K. Deb, H. Kargupta, and G. Harik. Rapid, accurate optimization of difficult problems using fast messy genetic algorithms. In S. Forrest, editor, *Proc. of the Fifth International Conference on Genetic Algorithms*, pages 56–64. Morgan Kauffman, 1993.

[20] P. Hansen and N. Mladenović. Developments of variable neighborhood search. In G.C. Ribeiro and P. Hansen, editors, *Essays and Surveys in Metaheuristics*, pages 415–439. Kluwer Academic Publishers, 2002.

[21] J. He, X. Yao, and J. Li. A comparative study of three evolutionary algorithms incorporating different amount of domain knowledge for node covering problems. *IEEE Transactions on Systems, Man and Cybernetics, Part C*, 35(2):266–271, May 2005.

[22] X. Huang, Y. Tan, and X. He. An Intelligent Multifeature Statistical Approach for the Discrimination of Driving Conditions of a Hybrid Electric Vehicle. *IEEE Transactions on Intelligent Transportation System*, 12(2):453–465, June 2011.

[23] S. Iordache. Consultant-guided search algorithms for the quadratic assignment problem. *Lecture Notes on Computer Science*, 6373:148–159, 2010.

[24] T. James, C. Rego, and F. Glover. Multistart Tabu Search and Diversification Strategies for the Quadratic Assignment Problem . *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 39(3):579–596, 2009.

[25] Y. Jin, editor. *Knowledge Incorporation in Evolutionary Computation*. Springer, Heidelberg, Germany, 2005.

[26] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proc. of IEEE International Conference on Neural Networks (Perth, Australia)*, volume IV, pages 1942–1948. IEEE Service Center, Piscataway, NJ, 1995.

[27] M. Laguna and J.L. González-Velarde. A search heuristic for just-in-time scheduling in parallel machines. *Journal of Intellignent Manufacturing*, 2:253–260, 1991.

[28] M. Laguna and R. Marti. GRASP and path relinking for 2-layer straight line crossing minimization. *INFORMS Journal on Computing*, 11:44–52, 1999.

[29] P. Larrañaga and J. A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2002.

[30] J. Li, G. Lu, and X. Yao. Fitness landscape-based parameter tuning method for evolutionary algorithms for computing unique input output sequences. In *Proc. of the 18th International Conference on Neural*

Information Processing (ICONIP'11), volume 7063 of *Lecture Notes in Computer Science*, pages 453–460, Shanghai, China, Nov. 2011.

[31] J. Li, G. Lu, and X. Yao. Fitness landscape-based parameter tuning method for evolutionary algorithms for computing unique input output sequences. In *Proc. of the 18th International Conference on Neural Information Processing (ICONIP'11)*, volume 7063 of *Lecture Notes in Computer Science*, pages 453–460, Shanghai, China, Nov. 2011.

[32] Y. Li, R. Pardalos, and M. Resende. A greedy randomized adaptive search procedure for the quadratic assignment problem. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 237–261, 1994.

[33] K.-H. Liang, X. Yao, and C. Newton. Evolutionary search of approximated n-dimensional landscapes. *International Journal of Knowledge-Based Intelligent Engineering Systems*, 4(3):172–183, July 2000.

[34] F.J. Lobo, C.F. Lima, and Z. Michalewicz, editors. *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*. Springer, 2007.

[35] Q. Lu and X. Yao. Clustering and learning gaussian distribution for continuous optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 35(2):195–204, May 2005.

[36] V. Maniezzo and A. Colorni. The ant system applied to the quadratic assignment problem. *IEEE Transactions on Knowledge and Data Engineering*, 11(5):769–778, 1999.

[37] V. Maniezzo, A. Colorni, and M. Dorigo. The ant system applied to the quadratic assignment problem. Technical Report 94/28, IRIDIA, Université de Bruxelles, 1994.

[38] S.L. Martins, M.G.C. Resende, C.C. Ribeiro, and P. Pardalos. A parallel GRASP for the steiner tree problem in graphs using a hybrid local search strategy. *Journal of Global Optimization*, 17:267–283, 2000.

[39] Y. Mei, K. Tang, and X. Yao. A memetic algorithm for periodic capacitated arc routing problem. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 41(6):1654–1667, Dec. 2011.

[40] D.P. Merz. *Memetic Algorithms for Combinatorial Optimization Problems: Fitness Landscapes and Effective Search Strategies*. PhD thesis, Department of Electrical Engineering and Computer Science, University of Siegen, Germany, 2000.

[41] D.P. Merz. Advanced fitness landscape analysis and the performance of memetic algorithms. *Evolutionary Computation, Special Issue on Memetic Evolutionary Algorithms*, 12(3):303–326, 2004.

[42] D.P. Merz and B. Freisleben. Fitness landscape, memetic algorithms and greedy operators for graph bi-partitioning. *Evolutionary Computation*, 8(1):61–91, 2000.

[43] H. Mühlenbein. The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5:303–346, 1998.

[44] H. Mühlenbein and T. Mahnig. Convergence theory and applications of the factorized distribution algorithm. *Journal of Computing and Information Technology*, 7:19–32, 1999.

[45] H. Mühlenbein and T. Mahnig. Evolutionary algorithms: From recombination to search distributions. *Theoretical Aspects of Evolutionary Computing. Natural Computing*, pages 137–176, 2000.

[46] C.A.S. Oliveira, P.M. Pardalos, and M.G.C. Resende. Grasp with path-relinking for the quadratic assignment problem. Technical report, AT & T labs, 2004.

[47] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. Linkage problem, distribution estimation and Bayesian networks. *Evolutionary Computation*, 8(3):311–340, 2000.

[48] F. Peng, K. Tang, G. Chen, and X. Yao. Population-based algorithm portfolios for numerical optimization. *IEEE Transactions on Evolutionary Computation*, 14(5):782–800, Oct. 2010.

[49] M. Prais and C. G. Ribeiro. Reactive grasp: An application to a matrix decomposition problem in tdma traffic assignment. *INFORMS Journal on Computing*, 12:164–176, 2000.

[50] R.-E. Precup, R.-C. David, E.M. Petriu, M.-B. Radac, S. Preitl, and J. Fodor. Evolutionary Optimization-based Tuning of Low-cost Fuzzy Controllers for Servo Systems. *Knowledge-Based Systems*, 2011. Available online 12 July 2011.

[51] M. Resende and C.C. Ribeiro. Greedy randomized adaptive search procedures. In F. Glover and G. Kochenberger, editors, *State-of-the-Art Handbooks in Meta-Heuristics*. Kluwer Academic Publishers, 2003.

[52] C.C. Ribeiro and M.C. Souza. Variable neighbourhood search for the degree constrained minimum spanning tree problem. *Discrete Applied Mathematics*, 118:43–54, 2002.

[53] R. Rubinstein. The cross-entropy method for combinatorial optimization and continuous optimization. *Methodology and Computing in Applied Probability*, 1(2):127–190, 1999.

[54] S.J. Shyu, P.Y. Yin, B.M.T. Lin, and T.S. Hsiao. An Ant Colony Optimization Algorithm for the Minimum Weight Vertex Cover Problem. *Annals of Operations Research*, 131:283–304, 2004.

[55] T. Stützle. Iterated local search for the quadratic assignment problem. Technical Report AIDA-99-03, Darmstadt University of Technology, Computer Science Department, Intellectics Group, 1999.

[56] T. Stützle and H. Hoos. MAX-MIN ant system and local search for combinatorial optimization problem. In S. Voss, S. Martello, I. H. Osman, and C. ROucairol, editors, *Meta-Heuristic: Advances and Trends in Local Search Paradigms for Optimization*, pages 313–329. Kluwer Academic Publishers, Boston, 1999.

[57] T. Stützle and H. Hoos. MAX-MIN ant system. *Future Generation Computer Systems*, 16(8):889–914, 2000.

[58] É.D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17:443–455, 1991.

[59] É.D. Taillard. Comparison of iterative searches for the quadratic assignment problem. *Location Science*, 3:87–105, 1995.

[60] É.D. Taillard. FANT: Fast ant system. Technical Report IDSIA 46-98, DSIA, Lugano, Switzerland, 1998.

[61] X. Yao. Dynamic neighbourhood size in simulated annealing. In *Proc. of Int'l Joint Conf. on Neural Networks (IJCNN'92 Beijing)*, volume 1, pages 411–416, Piscataway, NJ, USA, 1992. IEEE Press.

[62] Q. Zhang, J. Sun, and E.P.K. Tsang. Evolutionary algorithm with the guided mutation for the maximum clique problem. *IEEE Transactions on Evolutionary Computation*, 9(2):192–200, 2005.

[63] Q. Zhang, J. Sun, E.P.K. Tsang, and J.A. Ford. Combination of guided local search and estimation of distribution algorithm for solving quadratic assignment problem. In *Proc. of the Bird of a Feather Workshops, Genetic and Evolutionary Computation Conference*, pages 42–48, 2004.

[64] Q. Zhang, J. Sun, G. Xiao, and E.P.K. Tsang. Evolutionary algorithms refining a heuristic: Hyper-heuristic for shared-path protections in wdm networks under srlg constraints. *IEEE Trans on Systems, Man and Cybernetics, Part B*, 37(1):51–61, 2007.