Diversity-Adaptive Parallel Memetic Algorithm for Solving Large Scale Combinatorial Optimization Problems

JING TANG [1], MENG HIOT LIM [1], YEW SOON ONG [2]

[1] *School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798*

[2] *School of Computer Engineering, Nanyang Technological University, Singapore 639798*

{pg04159923, emhlim, asysong}@ntu.edu.sg

Abstract Parallel Memetic Algorithms (PMAs) are a class of modern parallel meta-heuristics that combine evolutionary algorithms, local search, parallel and distributed computing technologies for global optimization. Recent studies on PMAs for large-scale complex combinatorial optimization problems have shown that they converge to high quality solutions significantly faster than canonical GAs and MAs. However, the use of local learning for every individual throughout the PMA search can be a very computationally intensive and inefficient process. This paper presents a study on two diversity-adaptive strategies, i.e., 1) diversity-based static adaptive strategy (PMA-SLS) and 2) diversity-based dynamic adaptive strategy (PMA-DLS) for controlling the local search frequency in the PMA search. Empirical study on a class of NP-hard combinatorial optimization problem, particularly large-scale quadratic assignment problems (QAPs) shows that the diversity-adaptive PMA converges to competitive solutions at significantly lower computational cost when compared to the canonical MA and PMA. Furthermore, it is found that the diversity-based dynamic adaptation strategy displays better robustness in terms of solution quality across the class of QAP problems considered. Static adaptation strategy on the other hand requires extra effort in selecting suitable parameters to suit the problems in hand.

Keywords: island model parallel memetic algorithm; adaptive local search frequency; quadratic assignment problem

1 Introduction

Genetic algorithms (GAs), first proposed by Holland in 1975 [11], are meta-heuristic methods that have been successfully used for solving large-scale optimization problems. Their popularity lies in the ease of implementation and their ability to converge close to the global optimum. Nevertheless, canonical GAs generally suffers

from excessively slow convergence to locate a precise enough solution because of their failure to exploit local information. This often limits the practicality of GAs on many large-scale real world problems where computational time is a crucial consideration. Hence, it is now well established that canonical GAs are not suited for fine-tuning in complex search spaces, and that hybridization with other local learning improvement techniques can greatly improve the efficiency of search [9].

One of the recent growing areas in evolutionary algorithm (EAs) research is *memetic algorithms* (MAs) [21]. MAs are population-based meta-heuristic search methods inspired by Darwinian's principles of natural evolution and Dawkins' notion of a meme defined as a unit of cultural evolution that is capable of local refinements [5]. Hence, a memetic model of adaptation exhibits the plasticity of individuals that a strictly genetic model fails to capture. Recent studies on MAs have revealed their successes on a wide variety of optimization problems [1, 10, 12, 15, 16, 17, 22, 25, 33]. They not only converge to high quality solutions, but also search more efficiently than their conventional counterparts. Some theoretical and empirical investigations on MAs can be found in [9, 10, 15, 17, 22, 25]. In a more diverse context, MAs are also commonly known as hybrid EAs, Baldwinian EAs, Lamarkian EAs, cultural algorithms and genetic local search.

With evolutionary algorithms (EAs), there is flexibility to partition the population of individuals or islands of EA subpopulations among multiple compute nodes. It is important that the intrinsic parallelism of EAs is retained when designing any MAs. Best of all, parallel EAs possess diversity preservation capabilities that alleviate the effect of premature convergences. In our recent work, some extensions of MAs to parallel MAs (PMAs) have been proposed [34, 35]. It is worth noting that a crucial aspect of MAs or PMAs is to strike an optimum balance between the level of exploration provided by the GA, against the level of exploitation posed by the local search procedure throughout the memetic search. However, in canonical MAs or PMAs, it is common practice for the local search procedure to be applied on every individual/chromosome in the GA population(s). This is a very computationally intensive and inefficient search process. At the same time, exhaustive local search may lead to ineffective search due to premature fall in diversity during the PMA search.

To control the local search frequency during a PMA search, we focus on two diversity-adaptive strategies, i.e., PMA-SLS and PMA-DLS. In contrast to canonical

MAs and PMAs, the diversity-based adaptive approaches control the number of individuals undergoing the local search procedure throughout the PMA evolutionary search process. PMA-SLS uses a static adaptation strategy, maintaining population diversity throughout the PMA search by using a pre-defined Gaussian distribution to adjust the local search frequency. PMA-DLS's adaptation is based on online monitoring of population diversity during the PMA search for controlling the local search frequency. Empirical studies on the two diversity-adaptive PMAs are conducted for a class of NP-hard combinatorial optimization problem, particularly on large-scale quadratic assignment problems (QAPs). Results obtained show that both PMA-SLS and PMA-DLS converge to competitive solutions at significantly lower computational cost compared to canonical MA and PMA. Furthermore, the diversity-based dynamic adaptation strategy is shown to display better robustness in terms of solution quality on the class of QAP problems considered. Static adaptation strategy on the other hand is parameters sensitive. Therefore, suitable parameters should be chosen to suit the problems in hand.

This paper is organized as follows. Section 2 provides a brief overview of the recent research activities on memetic algorithm. The proposed static and dynamic diversity-adaptive approaches for controlling the local search frequency in the island model parallel memetic algorithm are described in Section 3. Section 4 presents the numerical results obtained from empirical study and provides a comprehensive quantitative/statistical comparison of PMA-DLS, PMA-SLS and PMA in the context of large scale QAPs. The search performances of the various algorithms in terms of solution quality, computational time, and solution precision are also reported in the section. Finally, we conclude the paper in Section 5.

2 Overview on Adaptive Memetic Algorithms

Memetic algorithm may be regarded as a marriage between a population-based global search and the local improvement made by each of the individuals. This has the potential to exploit the complementary advantages of EAs (generality, robustness, global search efficiency), and problem-specific local search (exploiting application-specific problem structure, rapid convergence toward local minima). In recent years, a number of independent researchers have addressed several issues relating to the trade-

off between exploration and exploitation in MAs. Some of the typical issues considered in literature are as follow:

1) How often should local learning be applied for, i.e., local search frequency?
2) On which solutions should the local learning be applied?
3) How long should the local learning be run, i.e., local search intensity?
4) Which local learning procedure or local search or meme to use?

The first issue pertinent to memetic algorithm design is to consider how often the local search should be applied for, i.e., local search frequency. Hart [10] investigated the effect of local search frequency on MA search performance. He studied various configurations of the local search frequency at different stages of the MA search. Conversely, it was shown in Ku et al. [16] that it may be worthwhile to apply local search on every individual if the computational complexity of the local search is relatively low. Hart [10] also studied the issue on how to best select the individuals among the EA population that should undergo local search. In his work, fitness-based and distribution-based strategies were studied for adapting the probability of applying local search on the population of chromosomes in continuous parametric search problems. Land [17] extended his work to combinatorial optimization problems and introduced the concept of "sniff" for balancing genetic and local search, also known as the local/global ratio. In [9], Goldberg and Voessner provide a theoretical alternative for efficient global-local hybrid search and characterize the optimum local search time that maximizes the probability of achieving a solution of a specified accuracy. Recently, Bambha et al. [1] introduced a simulated heating technique for systematically integrating parameterized local search into evolutionary algorithms to achieve maximum solution quality under a fixed computational time budget.

It is worth noting that the performance of MA search is also greatly affected by the choice of neighborhood structures. Fitness landscape analysis [22] provided a way for identifying the structure of a given problem and thus a selection of local search algorithms. Krasnogor [15] investigated how to change the size and the type of neighbourhood structures dynamically in the framework of multi-meme memetic algorithms where each meme had a different neighbourhood structure, a different acceptance rule and different local search intensity. The choice of multiple local learning procedure or memes during a memetic algorithm search in the spirit of Lamarckian learning, otherwise, known as meta-Lamarckian learning, on continuous

optimization problems was also considered in Ong et al. [25]. For a detailed taxonomy and comparative study on adaptive choice of memes in memetic algorithms, the reader may refer to [26].

In the context of multi-objective MA, issues relating to the frequency and intensity of local search have also been studied. The importance of striking a balance between genetic and local search in the multi-objective optimization was emphasized in Ishibuchi et al. [12]. Tan et al. [33] also incorporated the concept of fuzzy boundary local perturbation (FBLP) with interactive local fine tuning to facilitate broader neighborhood exploration in the context of multi-objective optimization. An excellent exposition on the design on multi-objective MAs can be found in [13].

A variety of parallel memetic algorithm (PMA) models which are extensions of canonical PGA have also been studied recently. These include the blackboard parallel asynchronous memetic algorithm [2], master/slave PMA [6] and the island model PMA [4]. The issue on which individuals should local learning be applied was also recently considered in the context of island model parallel memetic algorithm [4]. From literature survey, there has not been much work that considered balancing global and local search in the context of parallel MA. In particular, there is very little focus in literature on how local search frequency affects the diversity level of PMAs. We will first show that excessive local search in island model PMA can be counter-productive in the sections that follow. To address this issue, we proposed diversity-adaptive strategies for controlling the local search frequency in island model parallel memetic algorithms.

3 Diversity-Adaptive Parallel Memetic Algorithms

In this section, we begin with a brief overview on the diverse forms of parallel evolutionary algorithms in literature. Parallel evolutionary algorithm (PEA) represents an extension of the canonical EA. The basic concept of PEA is based on principle of tasks division of a classical EA across multiple processing nodes. The other advantage of PEA is that it facilitates speciation, a process by which different subpopulations evolve in diverse directions simultaneously. They have been shown to speed up the search process, attaining higher quality solutions on complex design problems. Several types of PEAs are briefly discussed. Subsequently, the PMA, PMA-SLS and

PMA-DLS for solving combinatorial optimization are introduced and described in this section.

3.1 Parallel Evolutionary Algorithm

Master-slave PEA — In master-slave PEAs, it is assumed that there is only a single panmictic population, i.e., a canonical EA. Like the canonical EA, each individual competes and reproduces with others. However, unlike the canonical EA, evaluations of individuals are distributed by scheduling fractions of the population among the processing slave nodes. In addition, master-slave PEA uses parallel computing to speed up the operation of the simple EA without changing the basic operations of the sequential EA. Such a model has the advantage of easy implementation since it does not alter the protocol of canonical EA search, i.e., the existing theory of simple EA still applies. Furthermore, it serves as an efficient method of parallelization when the fitness evaluation is computationally expensive.

Fine-grained PEA — Fine-grained parallel EA consists of a single population pool, which is spatially structured. It is designed to run on closely-link massively parallel processing system, i.e., a computing system consisting of large number of processing elements and connected in a specific high-speed topology. For instance, the population of individuals in a fine-grained PEA may be organized as a two-dimensional grid, since many massively parallel computers have processing elements that are connected using this topology. Consequently, selection and mating in a fine-grained parallel EA are restricted to small groups. Nevertheless, groups overlap to permit some interactions among all the individuals so that good solutions may disseminate across the entire populations. Sometimes, fine-grained parallel EA is also termed as the cellular model.

Multi-population PEA — Multiple population (or deme) EA may be more sophisticated, as it consists of several subpopulations that exchange individuals occasionally. This exchange of individuals is called migration and it is controlled by several parameters. Multi-population EAs are characterized by the use of multiple subpopulations and migration operation. Multi-population PEAs are also known by various names. Since they resemble the “island model” in population genetics that considers relatively isolated demes, it is often known as “Island EA”.

Hierarchical PEA — Various PEA models may also be hybridized to produce other new hierarchical PEA (HPEA) models. One may form a hierarchical PEA that combines a multi-population PEA (at the upper level) and a fine-grained PEA or master-slave PEA (at the lower level). On the other hand, multi-population PEA may also be designed with multiple levels in a manner such that migration rate is faster at the lower level and possessing a communication topology which is much denser than the upper level. In general, hierarchical PEA is considered to be more effective in generating significant speed up than standalone PEA models.

3.2 Canonical Island Model Parallel Memetic Algorithm (PMA)

We focus on island model parallel memetic algorithm for solving large-scale combinatorial optimization problems. The pseudo-code of a canonical PMA is outlined in Fig. 3-1.

BEGIN
Initialize *M* subpopulations of size *N* each
WHILE (termination condition not met)
  FOR each subpopulation or island
    Evaluate all individuals in the subpopulation
    For each individual in the subpopulation
      - Apply local search to the individuals in the subpopulation.
      - Proceed with local improvement and replace the genotype in the subpopulation with the improved solution.

    End For
    Create a new population based on Selection, Mutation and Crossover.
  END FOR
  For every *P* migration interval
    Send $K < N$ best individuals to a neighbouring subpopulation
    Receive *K* individuals from a neighbouring subpopulation
    Replace *K* individuals in the subpopulation
  END For
END WHILE
END

Fig. 3-1 Pseudo-code of the canonical island model parallel memetic algorithm

Initially, *M* subpopulations are randomly generated. Individuals in the subpopulations will then undergo the local search learning procedure in the spirit of Lamarckian learning. This form of learning forces the genotype to reflect the result of improvement through the placement of locally improved individual back into the population in order to compete for reproductive opportunities. The local search procedure considered here is based on the *k*-gene exchange [19, 20, 34, 35]. Subsequently new subpopulations are created through selection, mutation and crossover. For every *P* migration interval, the *K* best performing individuals in each

subpopulation migrate to its neighbouring subpopulation based on the one-way ring topology [35]. Meanwhile, the subpopulation being considered will receive *K* individuals from a neighbouring subpopulation. The replacement scheme may be a random walk. Alternatively, the worst performing *K* individuals are replaced with the *K* migrants from its neighbour. The entire procedure repeats until the stopping conditions are satisfied.

It is generally accepted that good diversity profile over the entire evolutionary process is a primary advantage of using island model parallel memetic algorithm for solving global optimization problems. As preliminary study, we consider using the 1) 2-island PMA and 2) 2-island PGA for solving the *sko100b* QAP benchmark. Note that PGA represents a canonical parallel GA. In contrast to PMA, no form of local search is used throughout the PGA search. The diversity of each subpopulation can be measured by various means. One simple approach is based on Shannon's information entropy, which represents an overall measure for describing the state of the dynamical system represented by the population. This is analogous to the state of a physical or information system.

Let *S* denotes the set of individuals that make up a subpopulation. The set *S* is further divided into partitions or subsets $S_1, S_2, \cdots, S_Q$. Each subset S_j is a grouping of individuals with the same fitness value. The ratio of the number of individuals in a partition S_j over the entire subpopulation can therefore be written as follows:

$$p_j = \frac{\left|S_j\right|}{\sum_{i=1}^{Q}\left|S_i\right|} \tag{1}$$

where $\left|S_j\right|$ is the cardinality of the set S_j. Based on partitioning of individuals according to the fitness values, one approach to describe the state of the dynamical system is based on Shannon's information entropy *E* as follows [27].

$$E = -\sum_{j=1}^{Q} p_j \log(p_j) \tag{2}$$

For illustration, the diversity of each subpopulation in the 2-island PMA and PGA based on the entropy measure is depicted in Fig. 3-2. From Fig. 3-2, it is worth highlighting that the significant drop in the entropy measure of the PMA in comparison to the PGA. It appears that PMA loses search diversity much earlier than

PGA due to possible excessive local searches. This significant drop in diversity for the PMA is indicative of the benefits derived from using local search in speeding up convergence rate of the search. However, it also implies the high risk of the PMA, losing out on search diversity prematurely as a result of the extensive local searches. It can be observed that this effect is more significant at the later stage of the search.

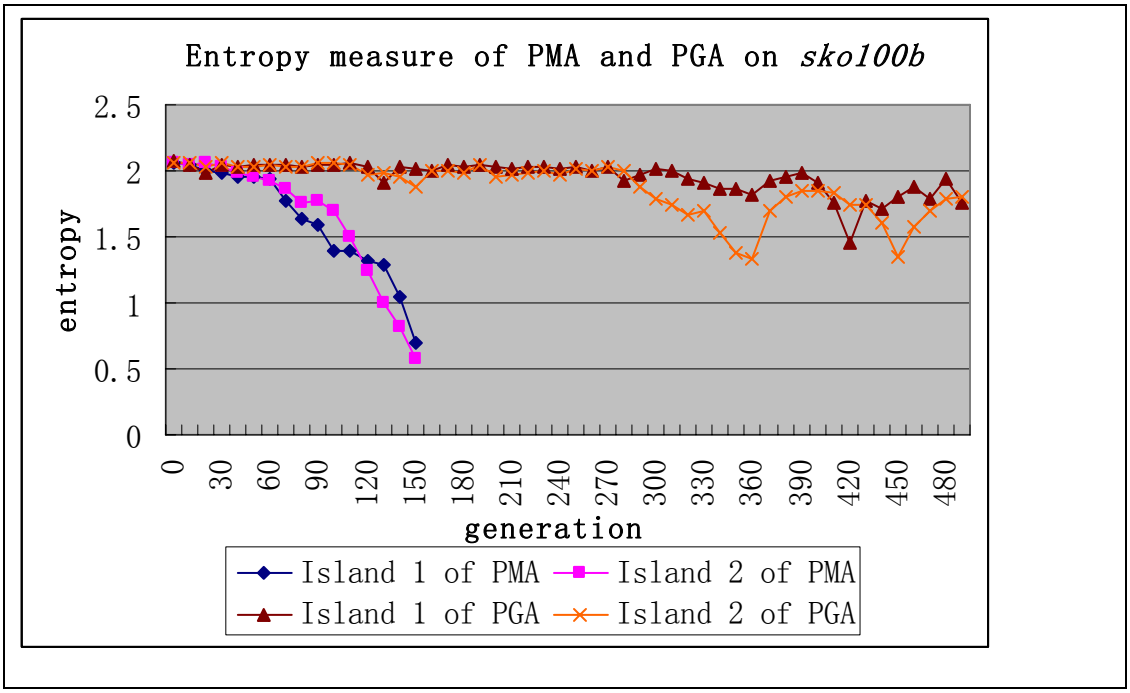


Fig. 3-2 Entropy measure for PMA and PGA on the *sko100b* QAP problem

To minimize the risk of premature convergence in the PMA, it is reasonable to ask whether the effects on performance might be reduced by adapting the local search frequency in the PMA search. Here, we present two diversity-adaptive strategies, 1) diversity-based static adaptive strategy (PMA-SLS) and 2) diversity-based dynamic adaptive strategy (PMA-DLS) for controlling the local search frequency in the PMA search. The pseudo-codes of PMA-SLS and PMA-DLS are outlined in Fig. 3-3 and Fig. 3-4, respectively.

BEGIN
Initialize *M* subpopulations of size *N* each
WHILE (termination condition not met)
FOR each subpopulation or island
Evaluate all individuals in the subpopulation
Calculate the number of individuals undergoing local search using PMA-SLS strategy
{

$$\gamma(gen;\mu,\sigma,\eta)=\frac{1}{\sqrt{2\pi}\bullet\sigma}\exp(-\frac{1}{2}(\frac{gen-\mu}{\sigma})^2)*\eta$$

$$\phi(gen,\xi;\mu,\sigma,\eta)=\gamma*\xi$$

}
For randomly selective $\phi(.)$ individuals in the subpopulation

- Apply local search to the selective individuals in the subpopulation.
- Proceed with local improvement and replace the genotype in the subpopulation with the improved solution.

End For
Create a new population based on Selection, Mutation and Crossover.
END FOR
For every *P* migration interval
Send $K < N$ best individuals to a neighbouring subpopulation
Receive *K* individuals from a neighbouring subpopulation
Replace *K* individuals in the subpopulation
END For
END WHILE
END

Fig. 3-3 Pseudo-code of PMA-SLS

BEGIN
Initialize *M* subpopulations of size *N* each
WHILE (termination condition not met)
FOR each subpopulation or island
Evaluate all individuals in the subpopulation
Calculate the number of individuals undergoing local search using PMA-DLS strategy
{

$$\beta(gen) = 1 + \frac{E(gen) - E(gen-k)}{E(gen-k)}$$

$$\phi(gen) = \begin{cases} \xi, & gen = 0 \\ Min\left[\phi(gen-k) * \lfloor \beta(gen) \rfloor, \xi\right], & gen > 0 \end{cases}$$

}
For randomly selective $\phi(.)$ individuals in the subpopulation
- Apply local search to the selective individuals in the subpopulation.
- Proceed with local improvement and replace the genotype in the subpopulation with the improved solution.

End For
Create a new population based on Selection, Mutation and Crossover.
END FOR
For every *P* migration interval
Send $K < N$ best individuals to a neighbouring subpopulation
Receive *K* individuals from a neighbouring subpopulation
Replace *K* individuals in the subpopulation
END For
END WHILE
END

Fig. 3-4 Pseudo-code of PMA-DLS

3.3 Diversity-adaptive strategy for local search frequency

A. Diversity-based static adaptive strategy (PMA-SLS)

In Fig. 3-2, it is noted that the population diversity degrades gradually with time. Since poor diversity generally occurs at the final stages of the parallel evolutionary search, it makes sense to consider reducing the local search frequency as the search progresses. In this manner, it is hoped that the high search efficiency of the canonical PMA at the initial stages of the search is preserved, while at the same time, greater explorations are enforced at the later stages of the search to reduce the risk of premature convergence. In particular, we model the local search frequency γ of the parallel MA search process as a normal or Gaussian distribution:

$$\gamma(gen; \mu, \sigma, \eta) = \frac{1}{\sqrt{2\pi} \bullet \sigma} \exp\left(-\frac{1}{2}\left(\frac{gen - \mu}{\sigma}\right)^2\right) * \eta \qquad (3)$$

where *gen* is the evolution generation ($gen \geq 0$), μ and σ are the mean and standard deviation of the specified Gaussian distribution, respectively. η represents a scaling factor for the number of chromosomes local search is applied. Using the taxonomy of adaptive evolutionary algorithm provided in [7], the present adaptive strategy is termed here as diversity-based static adaptive strategy.

Using Equation (3) and a subpopulation size, ξ, the number of chromosomes to apply local search, $\phi(.)$, at gen^{th} generation is then defined as

$$\phi(gen, \xi; \mu, \sigma, \eta) = \gamma * \xi \tag{4}$$

where $\phi(.)$ denotes the number of selected candidates whereby local search is applied at the gen^{th} generation.

B. Diversity-based dynamic adaptive strategy (PMA-DLS)

Online entropy measure may also be used to provide dynamic information about the stage of the evolutionary search process and the degree of diversity of each subpopulation. Since population diversity represents a crucial characteristic of the PMA, the approach considered here makes use of online entropy measure to adapt the local search frequency. The method considered here is the diversity-based dynamic adaptive strategy or PMA-DLS in short. Hence, the dynamic local search frequency β in PMA-DLS can be defined based on the online entropy ratio given by

$$\beta(gen) = 1 + \frac{E(gen) - E(gen-k)}{E(gen-k)} \tag{5}$$

where $E(gen)$ and $E(gen-k)$ $(gen \geq k)$ are the population entropy measure at the gen^{th} and $(gen\text{-}k)^{th}$ generation, respectively.

The PMA-DLS search thus begins by initializing all subpopulations randomly with ξ chromosomes, i.e., $\phi(0) = \xi$. Subsequently, the number of chromosomes that undergo local learning is defined based on online diversity of the subpopulations as per Equation (6).

$$\phi(gen) = \begin{cases} \xi, & gen = 0 \\ Min\left[\phi(gen-k) * \lfloor \beta(gen) \rfloor, \xi\right], & gen > 0 \end{cases} \tag{6}$$

4 Empirical Study

To demonstrate the capability of the proposed strategies described in the above section, a series of empirical studies on solving complex combinatorial optimization problem, in particular the quadratic assignment problem (QAP) were conducted. First, we briefly introduce the QAP and the experimental setup. Then, we present a series of empirical comparison of results for PMA-DLS, PMA-SLS, PMA and PMA-FLS on several large scale QAP benchmarks. PMA-FLS is a parallel memetic algorithm with fixed local search strategy whereby local search is applied only on individuals that have undergone modification by the evolutionary operators [34, 35].

4.1 QAP and experimental setup

The quadratic assignment problem (QAP) is one of the hardest combinatorial optimization problems and is frequently used as benchmark to study various heuristic algorithms such as greedy randomized search [18], tabu search [28, 31], ant colony optimization algorithms [29], simulated annealing [37], genetic algorithms (GA) [19, 20], memetic algorithms [23, 24], etc.. In general, the QAP can be described by two $n \times n$ matrices $A = [a_{ij}]$ and $B = [b_{ij}]$ [14]. The goal is to find a permutation π of the set M={1,2,3,…,n}, which minimizes the objective function C(π) as in Eq.(7).

$$C(\pi) = \sum_{l=1}^{n} \sum_{t=1}^{n} a_{lt} b_{\pi(l)\pi(t)} \tag{7}$$

In solving QAP, two issues are of primary concerns. One is the solution quality which depends on the algorithm effectiveness; the other being the computational cost which depends on the algorithm efficiency. Many algorithms generate good solutions while incurring huge computational cost. On the other hand, those that converge to solutions quickly tend to produce poor results. It is therefore necessary to strike a balance between these two factors, a primary focus of our previous work [19, 20, 34, 35]. In particular, we evaluate the performance of different algorithms both in terms of computation time and solution quality. The statistical significance based on *t*-test for PMA-SLS and PMA-DLS compared with PMA is evaluated for its performance in terms of computation cost. For convenience, the abbreviations for the different algorithms used in our study are summarized below:

PMA-DLS —Island model parallel memetic algorithm with diversity-based dynamic adaptive strategy;

PMA-SLS —Island model parallel memetic algorithm with diversity-based static adaptive strategy;

PMA-FLS —Island model parallel memetic algorithm with fixed local search strategy in our previous work [34, 35];

PMA —Island model parallel memetic algorithm with complete local search strategy;

MA —Canonical memetic algorithm.

The algorithms were coded in C programming language and simulations were carried out on a cluster of Pentium IV 1.9 GHz workstations. Each computing node is configured with 256MB of RAM, running on Linux Redhat 7.0 operating system. For each QAP benchmark problem, we carried out 10 optimization runs and the algorithms were evaluated based on their average performance.

In our empirical study, a grid-enabled solver is used to facilitate the implementation of the PMA, such that islands of MA individuals are executed on multiple computing nodes within a distributed computing framework. The configuration of the PMA control parameters is summarized in Table 4-1.

Table 4-1PMA parameters setting

MA parameters	Multi-island PMA
Population size	240
Subpopulation size	240/M
Elite size	2 (M=2)
	1 (M≥3)
Maximum number of generations	180
Fitness scaling factor S_f	3
Crossover probability P_c	0.8
Mutation probability P_m	0.05
Zerofit threshold constant K_z	5

M ≡ number of islands (processing nodes)

Except for S_f and K_z, all the above are standard GA parameters. A feasible solution of QAP of size n can be genetically coded as a permutation string of n integers, which are evaluated based on the objective function described by Eq.(7). For

each permutation string denoted as π, the objective value obtained is normalized to obtain a fitness value f as follows:

$$f = 1 - C(\pi)/F_z \tag{8}$$

The parameter F_z is a threshold value specified for each problem and it is akin to the upper bound value used in many deterministic search algorithms. Since we require that $f \geq 0$, Eq. (8) is clipped at $f = 0$ for $C(\pi) \geq F_z$. For this reason, we refer to F_z as a *zerofit* threshold. In our implementation of the algorithm, the threshold is defined as $F_z = K_z \times \Omega$; K_z being an integer constant while Ω refers to the known optimum or lower bound of the problem. Subsequently, the fitness values of the general population are scaled in order to avoid any unintentional bias. The approach for scaling is based on the commonly used linear scaling model [8] as follows:

$$F = k_1 + k_2 f \tag{9}$$

where f and F denote the fitness values before and after scaling respectively. The coefficients k_1 and k_2 are chosen such that $F_{ave} = f_{ave}$ and $F_{\max} = S_f \times F_{\min}$. S_f refers to the scaling factor, f_{ave} the average fitness before scaling while F_{ave}, $F_{\max}$ and $F_{\min}$ are respectively the average, maximum and minimum fitness values after scaling. Based on these conditions, the coefficients are determined as follows:

$$k_1 = \frac{(1 - S_f) \times f_{ave}}{(f_{\min} \times S_f - f_{\max}) + (1 - S_f) \times f_{ave}} \tag{10}$$

$$k_2 = (1 - k_1) \times f_{ave} \tag{11}$$

where $f_{\min}$ is the minimum fitness and $f_{\max}$ the maximum fitness before scaling.

Through a series of empirical study and based on results and experience from previous work [34, 35], the following migration control parameters have been adopted in the PMA.

Migration Interval ─ Migration occurs every 10 generations;

Migration Rate ─ One chromosome per migration phase;

Migration Policy ─ Elitist strategy, whereby the best individual in one subpopulation replaces the worst in the other;

Migration Topology ─ One-way ring topology.

The search stops or terminates when either one of the following criteria is satisfied:

i. Solution stalls for more than 70 successive generations;

ii.Maximum number of generations has been reached.

Several criteria defined to measure the performance are listed as follows:

CPU time – Average computation time in seconds upon termination of the algorithm;

Generation – Average number of generations elapsed before the occurrence of the best solution;

TG – Average number of generations elapsed before the algorithm terminates;

Average – Average objective value of the solutions obtained for all the simulation runs;

Average gap – Difference between the *Average* and the best-known value of the objective function;

Best – Best solution obtained among all the simulation runs;

Gap – Difference between the best-found value and the best-known value of a benchmark problem;

*S*uccess rate – Number of times the algorithm finds the best-known solution out of all the simulation runs.

Among these criteria, *CPU time* is used to measure the computational cost of the algorithms in wall-clock time. *Generation* and *TG* provide a measure on the convergence rate of the algorithms in terms of the number of iterations rather than the wall-clock time. *Average*, *Average gap*, *Best*, *Gap* and *Success rate* serve as the criteria for measuring the solution quality of the algorithm.

4.2 Results comparison – PMA-DLS vs. PMA-SLS

For parameters pertaining to PMA-SLS in Equation (4), the subpopulation size, ξ, is a constant for certain number of islands in the PMA and μ is set to zero. The other two parameters (σ, η) were tuned in order to adjust the local search frequency for each generation *gen*. To decide on the appropriate configuration, significant effort was expended on parameters tuning in order to achieve a desirable level of performance. Various parameters setting for Gaussian function were experimented to configure the PMA-SLS. For example, in Fig. 4-1, three Gaussian functions denoted as γ_1, γ_2 and γ_3 with different parameters setting are shown. The corresponding number (*Num*) of individuals where local search is applied can be determined based on Eq. (4). Local

search frequency γ in Eq. (4) was updated every 10 generations. According to Fig. 4-1, application of γ_1, γ_2 and γ_3 will result in different local search frequency applied in the PMA. γ_3, the upper curve results in higher frequency of local search while γ_2, the lower curve indicates a lower frequency. Based on the application of γ_1, γ_2 and γ_3, the corresponding PMA-SLS-1, PMA-SLS-2, and PMA-SLS-3 were derived respectively. Meanwhile, PMA-DLS is more straightforward, with fewer parameters setting required. Only parameter *k* is required to be set in Equation (6). Here, *k* is set to 10.

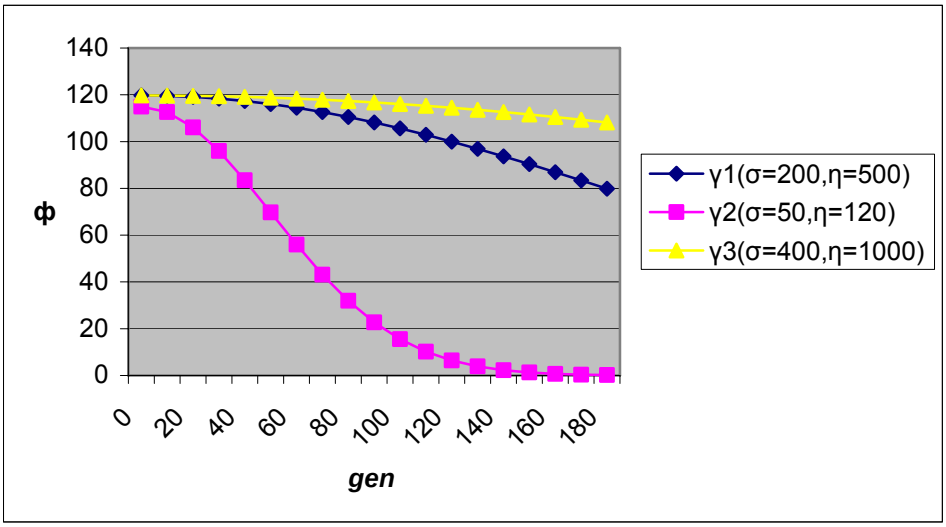


Fig. 4-1 Application of Gaussian functions to determine number of chromosomes selected for local search according to PMA-SLS

Table 4-2 Comparison of PMA-DLS and PMA-SLS with γ_1, γ_2 and γ_3

			CPU time	Generation	TG	Average	Average gap	Best	Gap	Success rate
sko100b	2-island	PMA-SLS-1	875.20	113.60	168.40	154012.80	0.08%	153904	0.01%	0.00%
153890		PMA-SLS-2	563.80	134.10	174.70	154114.60	0.16%	153962	0.05%	0.00%
		PMA-SLS-3	903.20	119.20	171.30	154016.60	0.08%	153904	0.01%	0.00%
		PMA-DLS	**859.40**	**125.30**	**169.00**	**154020.80**	**0.08%**	**153920**	**0.02%**	**0.00%**

We first carried out experimental study to gauge the effect of the choice of Gaussian function on the performance of PMA-SLS. The results presented in Table 4-2 are based on comparison of PMA-SLS and PMA-DLS on one particular benchmark. This experiment shows that PMA-DLS could produce good solutions with 0.08% average gap, consuming 859.40 seconds of CPU time. In comparison, the 3 variants of PMA-SLS vary in terms of solution quality and CPU time. In terms of CPU time, PMA-SLS-3 requires as much as 903.20 seconds while PMA-SLS-2 takes up 563.80 seconds of CPU time. On solution quality, the average gap of the PMS-SLS with the

three configurations falls into the range of 0.08% to 0.16%. This may be due to the different number of individuals undergoing local search in PMA-SLS, especially at the later stages of the evolution process. For example, the number of individuals whereby local search is applied in PMA-SLS-3 is much larger than that for PMA-SLS-2 (when *gen* > 100). Consequently, PMA-SLS-3 produced better solution (0.08%) quality than PMA-SLS-2 (0.16%). However, PMA-SLS-3 takes up more computational time. Between the 3 selection functions experimented, it appears that γ_1 ($\sigma = 200, \eta = 500$) produced the most competitive results in terms of solution quality and computational cost.

4.3 Results comparison – PMA-SLS, PMA-DLS and PMA

To demonstrate the advantage of PMA-SLS and PMA-DLS, a comparison among PMA-SLS, PMA-DLS and PMA on the two-island model for the same benchmark, *sko100b*, is shown in Table 4-3.

Table 4-3 Comparison among PMA-SLS, PMA-DLS and PMA

			CPU time	Generation	TG	Average	Average gap	Best	Gap	Success rate
sko100b	2-island	PMA	1350.00	94.70	145.90	153950.40	0.04%	153890	0.00%	20.00%
153890		PMA-SLS	875.20	113.60	168.40	154012.80	0.08%	153904	0.01%	0.00%
		PMA-DLS	859.40	125.30	169.00	154020.80	0.08%	153920	0.02%	0.00%

In Table 4-3, PMA-SLS and PMA-DLS produce competitive solutions although the frequency of local search of PMA-SLS and PMA-DLS never exceed that of PMA which maintain the highest local search frequency throughout the evolution process. This is due mainly to the ability of the PMA-SLS and PMA-DLS to manage a more desirable diversity profile as the search progresses, especially at the later stage of the evolution process, compared to the poor diversity profile in PMA. The diversity of each subpopulation for PMA-SLS, PMA-DLS and PMA, measured by the entropy, was traced in our simulation and shown in Fig. 4-2.

According to Fig. 4-2, PMA-SLS and PMA-DLS can consistently maintain a good level of diversity as the evolution progresses. However, the diversity of PMA shows a significant drop in entropy, especially at the later stages, indicating that local search has a tendency to speed up convergence significantly. From an evolutionary process point of view, PMA results in poorer diversity due to excessive localized searches, especially at the later stage of evolution. On the other hand, PMA-SLS adjusts the local search frequency according to a specific Gaussian function. PMA-DLS adjusts the local search frequency based on changes in population diversity. The

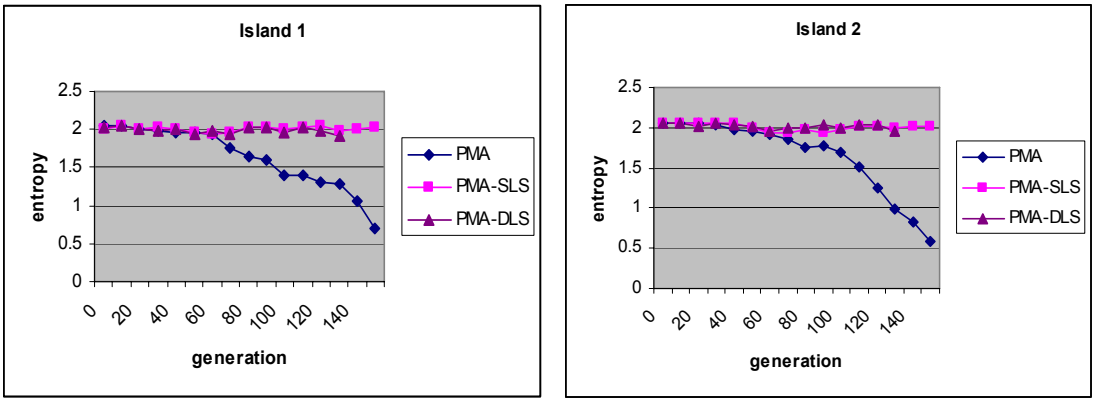


Fig. 4-2 Comparison of diversity among PMA-SLS, PMA-DLS and PMA

number of individuals to apply local search is then adjusted dynamically, enabling PMA-DLS to maintain a consistent level of population diversity. This in turn enhances the capacity of PMA-DLS to produce good solutions. A significant observation from Table 4-3 is that PMA-SLS, PMA-DLS and PMA achieved almost the same level of solution quality, with PMA incurring higher computational cost due to the intensive local search. PMA-SLS and PMA-DLS therefore show a potential for reducing computational time significantly with little or no lost of solution quality. This is mainly attributed to its capability to maintain a higher level of population diversity.

4.4 Overall comparison of results and analysis

For the purpose of detailed comparison among PMA-SLS, PMA-DLS and PMA, Tables 4-4 to 4-9 summarize the empirical results of testing on a diverse set of large scale QAP benchmarks. The benchmark problems considered in the present study are classes of synthetic problems randomly generated or created to study the robustness of algorithms for solving QAPs [3]. The best-known value corresponding to each instance of QAP is shown in the first column of Tables 4-4 to 4-9. The characteristics of these benchmark problems are summarized below:

sko — This group of benchmarks was proposed by Skorin-Kapov [28]. The distance matrices of these problems are rectangular and the entries of the flow matrices are pseudo-random numbers.

tai — The instances *tai-a* are uniformly generated and was proposed in [31], while the instances *tai-b* were introduced in [32]. Problems of *tai-b* group are asymmetric and randomly generated.

wil — This group of benchmarks was proposed by Wilhelm and Ward [37], the distance matrices of these problems are rectangular.

tho — This group of benchmarks was proposed by Thonemann and Bolte [36], the distance matrices of these instances are rectangular.

Tables 4-4 and 4-5 present a detailed simulation results for PMA-DLS, PMA, PMA-SLS and PMA-FLS [34, 35] on *sko100b* and *tai100b* benchmarks, respectively. Tables 4-6 to 4-9 show the simulation results on the other classes of QAP, namely, *sko100**, *tai100a*, *wil100* and *tho150*, respectively. The size of all these problems considered in the study is relatively large.

Table 4-4 Results of testing on *sko100b* benchmark

			CPU time	Generation	TG	Average	Average gap	Best	Gap	Success rate
sko100b		MA	3096.50	127.30	160.50	153955.60	0.04%	153890	0.00%	20.00%
153890	2-island	PMA-DLS	859.40	125.30	169.00	154020.80	0.08%	153920	0.02%	0.00%
		PMA-SLS	875.20	113.60	168.40	154012.80	0.08%	153904	0.01%	0.00%
		PMA-FLS[34]	266.80	182.70	252.70	154494.20	0.39%	154160	0.18%	0.00%
		PMA	1350.00	94.70	145.90	153950.40	0.04%	153890	0.00%	20.00%
	4-island	PMA-DLS	885.70	131.40	170.00	153977.40	0.06%	153900	0.01%	0.00%
		PMA-SLS	898.00	137.10	178.10	153990.80	0.07%	153902	0.01%	0.00%
		PMA-FLS[35]	174.50	282.50	352.50	154213.80	0.21%	153952	0.04%	0.00%
		PMA	1445.90	122.20	174.60	153952.20	0.04%	153898	0.01%	0.00%
	6-island	PMA-DLS	413.80	126.80	164.20	153951.20	0.04%	153890	0.00%	20.00%
		PMA-SLS	429.40	130.20	168.20	153985.00	0.07%	153890	0.00%	10.00%
		PMA-FLS[35]	148.80	213.30	283.30	154254.60	0.24%	154074	0.12%	0.00%
		PMA	694.30	104.80	154.50	153925.40	0.02%	153890	0.00%	20.00%
	10-island	PMA-DLS	276.80	98.30	159.50	153974.40	0.05%	153924	0.02%	0.00%
		PMA-SLS	289.30	95.20	148.80	153987.80	0.06%	153890	0.00%	10.00%
		PMA-FLS[35]	119.60	150.80	220.80	154195.80	0.20%	153936	0.03%	0.00%
		PMA	439.00	111.20	144.40	153942.60	0.04%	153890	0.00%	30.00%

Table 4-5 Results of testing on *tai100b* benchmark

			CPU time	Generation	TG	Average	Average gap	Best	Gap	Success rate
tai100b	2-island	PMA-DLS	694.70	94.70	124.30	1186119285.20	0.01%	1185996137	0.00%	50.00%
1185996137		PMA-SLS	782.40	106.70	134.00	1186275856.50	0.02%	1185996137	0.00%	40.00%
		PMA-FLS[34]	186.90	175.30	245.30	1188882832.20	0.24%	1186007112	0.00%	0.00%
	4-island	PMA-DLS	633.40	105.00	122.00	1186121434.00	0.01%	1185996137	0.00%	80.00%
		PMA-SLS	647.50	92.60	102.30	1186007361.40	0.00%	1185996137	0.00%	80.00%
		PMA-FLS[35]	178.10	268.80	332.00	1187539521.00	0.13%	1186007112	0.00%	0.00%
	6-island	PMA-DLS	342.40	65.20	107.20	1186132401.50	0.01%	1185996137	0.00%	70.00%
		PMA-SLS	356.60	88.30	104.90	1186058956.40	0.01%	1185996137	0.00%	70.00%
		PMA-FLS[35]	160.10	233.70	296.70	1187892570.00	0.16%	1185996137	0.00%	10.00%
	10-island	PMA-DLS	214.50	87.00	112.80	1186064568.60	0.01%	1185996137	0.00%	80.00%
		PMA-SLS	220.70	68.70	82.70	1186053344.20	0.00%	1185996137	0.00%	80.00%
		PMA-FLS[35]	148.00	250.00	320.00	1187927883.00	0.16%	1186052259	0.00%	0.00%

Table 4-6 Results of testing on *sko100** benchmarks

			CPU time	Generation	TG	Average	Average gap	Best	Gap	Success rate
sko100a	2-island	PMA-DLS	852.80	118.80	164.50	152156.10	0.11%	152069	0.03%	0.00%
152002		PMA-SLS	883.60	133.80	175.10	152188.20	0.12%	152042	0.03%	0.00%
		PMA-FLS[34]	194.00	203.40	273.40	152322.80	0.21%	152122	0.08%	0.00%
	4-island	PMA-DLS	855.30	132.40	171.60	152104.10	0.07%	152059	0.04%	0.00%
		PMA-SLS	885.20	142.40	176.80	152119.00	0.08%	152058	0.04%	0.00%
	6-island	PMA-DLS	416.60	131.60	170.80	152126.20	0.08%	152044	0.03%	0.00%
		PMA-SLS	431.90	138.90	176.90	152109.40	0.07%	152067	0.04%	0.00%
	10-island	PMA-DLS	273.50	134.20	174.20	152093.60	0.06%	152035	0.03%	0.00%
		PMA-SLS	283.20	98.10	156.00	152102.80	0.06%	152042	0.03%	0.00%
sko100c	2-island	PMA-DLS	847.30	120.60	167.90	147928.60	0.05%	147862	0.00%	10.00%
147862		PMA-SLS	939.30	121.80	168.40	147934.80	0.05%	147862	0.00%	10.00%
		PMA-FLS[34]	184.40	205.80	275.80	148140.40	0.18%	148050	0.13%	0.00%
	4-island	PMA-DLS	826.30	112.20	163.00	147894.00	0.02%	147862	0.00%	20.00%
		PMA-SLS	845.90	111.40	160.50	147908.20	0.03%	147862	0.00%	10.00%
	6-island	PMA-DLS	401.20	124.20	173.00	147887.20	0.02%	147868	0.00%	30.00%
		PMA-SLS	416.80	106.40	151.80	147885.60	0.02%	147862	0.00%	20.00%
	10-island	PMA-DLS	258.40	102.40	125.60	147885.20	0.02%	147862	0.00%	40.00%
		PMA-SLS	284.20	107.90	151.00	147895.40	0.02%	147862	0.00%	10.00%
sko100d	2-island	PMA-DLS	869.60	136.10	170.80	149742.20	0.11%	149656	0.05%	0.00%
149576		PMA-SLS	883.00	111.00	166.90	149803.60	0.15%	149618	0.03%	0.00%
		PMA-FLS[34]	232.10	259.90	327.40	150036.80	0.31%	149732	0.10%	0.00%
	4-island	PMA-DLS	813.50	137.00	177.20	149729.20	0.10%	149648	0.05%	0.00%
		PMA-SLS	881.20	146.70	180.00	149752.00	0.12%	149630	0.04%	0.00%
	6-island	PMA-DLS	429.00	134.20	168.40	149707.60	0.09%	149620	0.03%	0.00%
		PMA-SLS	436.80	135.80	173.80	149699.40	0.08%	149578	0.00%	0.00%
	10-island	PMA-DLS	279.60	154.60	180.00	149685.20	0.07%	149608	0.02%	0.00%
		PMA-SLS	312.60	120.10	167.20	149681.60	0.07%	149584	0.01%	0.00%
sko100e	2-island	PMA-DLS	809.40	111.30	148.00	149198.20	0.03%	149150	0.00%	30.00%
149150		PMA-SLS	845.40	121.00	166.70	149205.80	0.04%	149150	0.00%	10.00%
		PMA-FLS[34]	235.50	252.90	322.90	149642.20	0.33%	149198	0.03%	0.00%
	4-island	PMA-DLS	864.80	119.80	173.60	149188.80	0.03%	149150	0.00%	10.00%
		PMA-SLS	898.50	114.30	164.50	149202.60	0.04%	149150	0.00%	10.00%
	6-island	PMA-DLS	425.00	107.80	144.80	149183.60	0.02%	149150	0.00%	40.00%
		PMA-SLS	452.10	113.70	156.90	149179.20	0.02%	149150	0.00%	30.00%
	10-island	PMA-DLS	251.20	61.00	131.00	149180.80	0.02%	149150	0.00%	40.00%
		PMA-SLS	274.20	91.20	130.00	149176.40	0.02%	149150	0.00%	40.00%
sko100f	2-island	PMA-DLS	825.70	98.90	155.20	149218.03	0.12%	149096	0.04%	0.00%
149036		PMA-SLS	888.40	104.60	153.70	149232.80	0.13%	149126	0.06%	0.00%
		PMA-FLS[34]	206.50	214.80	284.80	149496.60	0.31%	149228	0.13%	0.00%
	4-island	PMA-DLS	813.90	130.20	173.40	149144.20	0.07%	149036	0.00%	20.00%
		PMA-SLS	872.10	126.10	166.70	149150.40	0.08%	149036	0.00%	10.00%
	6-island	PMA-DLS	423.20	151.40	180.00	149145.20	0.07%	149092	0.04%	0.00%
		PMA-SLS	451.70	136.10	172.30	149205.40	0.11%	149078	0.03%	0.00%
	10-island	PMA-DLS	282.40	111.20	158.80	149162.40	0.08%	149104	0.05%	0.00%
		PMA-SLS	300.30	107.00	161.40	149203.40	0.11%	149114	0.05%	0.00%

Table 4-7 Results of testing on *tai100a* benchmark

			CPU time	Generation	TG	Average	Average gap	Best	Gap	Success rate
tai100a	2-island	PMA-DLS	866.20	127.90	161.80	21442193.71	1.50%	21379594	1.18%	0.00%
21125314		PMA-SLS	860.00	127.20	164.60	21458262.60	1.58%	21382118	1.22%	0.00%
		PMA-FLS[34]	222.80	238.50	308.50	21464686.20	1.61%	21335594	1.00%	0.00%
	4-island	PMA-DLS	889.20	156.20	180.00	21380930.80	1.21%	21362016	1.12%	0.00%
		PMA-SLS	889.60	140.20	170.90	21420954.60	1.40%	21352956	1.08%	0.00%
	6-island	PMA-DLS	433.60	146.40	180.00	21368255.10	1.15%	21237278	0.53%	0.00%
		PMA-SLS	451.40	152.50	180.00	21373508.00	1.17%	21270370	0.69%	0.00%
	10-island	PMA-DLS	294.80	120.80	168.40	21347190.00	1.05%	21306288	0.86%	0.00%
		PMA-SLS	309.60	123.60	169.50	21382655.00	1.21%	21295312	0.80%	0.00%

Table 4-8 Results of testing on *wil100* benchmark

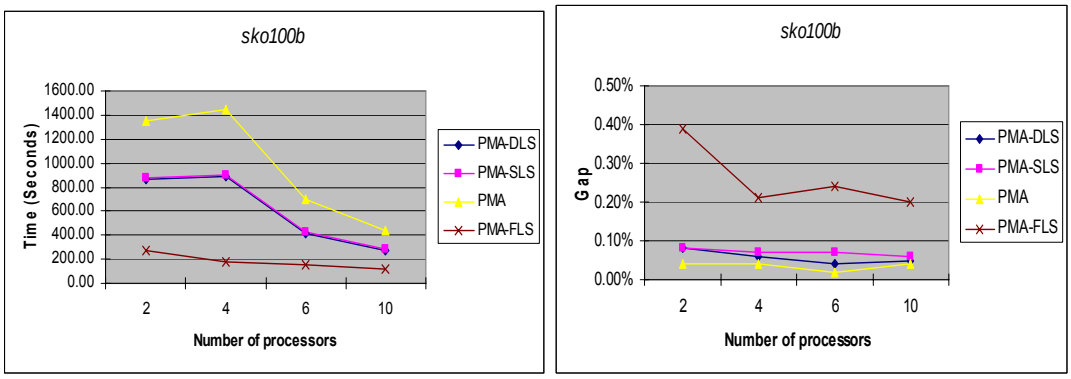
			CPU time	Generation	TG	Average	Average gap	Best	Gap	Success rate
wil100	2-island	PMA-DLS	833.90	120.10	167.10	273147.20	0.04%	273078	0.01%	0.00%
273038		PMA-SLS	882.10	114.50	166.20	273198.80	0.06%	273054	0.01%	0.00%
		PMA-FLS[34]	218.00	226.10	292.80	273458.20	0.15%	273236	0.07%	0.00%
	4-island	PMA-DLS	881.40	137.00	180.00	273101.60	0.02%	273066	0.01%	0.00%
		PMA-SLS	895.20	115.20	165.60	273228.60	0.07%	273054	0.01%	0.00%
	6-island	PMA-DLS	433.40	128.20	178.00	273092.80	0.02%	273056	0.01%	0.00%
		PMA-SLS	445.00	99.30	164.30	273103.80	0.02%	273044	0.00%	0.00%
	10-island	PMA-DLS	272.40	88.20	153.00	273102.60	0.02%	273054	0.01%	0.00%
		PMA-SLS	295.20	93.90	161.20	273128.60	0.03%	273054	0.01%	0.00%

Table 4-9 Results of testing on *tho150* benchmark

			CPU time	Generation	TG	Average	Average gap	Best	Gap	Success rate
tho150	2-island	PMA-DLS	7136.20	139.10	177.00	8146734.00	0.16%	8142504	0.11%	0.00%
8133398		PMA-SLS	7290.30	140.30	174.00	8148332.60	0.18%	8142700	0.11%	0.00%
		PMA-FLS[34]	1428.50	308.50	368.20	8158144.60	0.30%	8140370	0.09%	0.00%
	4-island	PMA-DLS	4993.80	141.60	172.00	8143158.00	0.12%	8137465	0.05%	0.00%
		PMA-SLS	5258.40	148.40	180.00	8144249.60	0.13%	8138428	0.06%	0.00%
		PMA-FLS[35]	935.10	317.70	376.80	8162408.00	0.36%	8145990	0.15%	0.00%
	6-island	PMA-DLS	3027.20	156.80	180.00	8142624.80	0.11%	8137004	0.04%	0.00%
		PMA-SLS	3267.40	156.20	180.00	8145297.20	0.15%	8142554	0.11%	0.00%
		PMA-FLS[35]	885.30	332.50	386.00	8157363.67	0.29%	8151408	0.22%	0.00%
	10-island	PMA-DLS	1953.80	155.40	180.00	8140869.60	0.09%	8139868	0.07%	0.00%
		PMA-SLS	2004.20	136.00	177.20	8144993.20	0.14%	8142102	0.11%	0.00%
		PMA-FLS[35]	605.10	418.50	462.40	8165464.60	0.39%	8154998	0.27%	0.00%

In Table 4-4, from the viewpoint of computational time, compared to the serial MA, much shorter computational time is consumed by PMA-SLS, PMA-DLS, PMA-FLS and PMA, indicating the advantage of employing parallel memetic algorithms. From a solution quality point of view, PMA, PMA-DLS and PMA-SLS can achieve much better quality than PMA-FLS. This is evident from the much improved solution gap and the higher success rate achieved, which can be attributed to the powerful search capability of memetic algorithm. Furthermore, PMA-DLS and PMA-SLS can reduce the computational time significantly with little or no lost in solution quality compared to PMA which benefited from the more desirable population diversity profile as a result of the diversity-adaptive strategies employed. The comparison

among PMA-SLS, PMA-DLS, PMA and PMA-FLS on *sko100b* benchmark is shown in Fig. 4-3.



(a) Computation time (b) Solution quality

Fig. 4-3 Comparison among PMA-SLS, PMA-DLS, PMA and PMA-FLS on *sko100b* benchmark

The plot in Fig. 4-3(b) shows that PMA-DLS, PMA-SLS and PMA improve the solution quality significantly compared to PMA-FLS. It is noted that the maximum number of generations for PMA-FLS was set at 500. Instead, the maximum number of generations for PMA-DLS, PMA-SLS and PMA was set to 180. This is indicative of the powerful search capability and quick convergence speed of the PMA. As for the computational time shown in Fig. 4-3(a), the greater reliance on local search makes PMA more time-consuming than the PMA-FLS. However, with the island model paradigm of the parallel memetic algorithm, a distributed computing framework can help to reduce the computational time significantly. Furthermore, the diversity-based adaptive local search strategy both in static and dynamic manner used in PMA-SLS and PMA-DLS, respectively, improves the efficiency of the PMA remarkably.

In addition, it is observed from experimental results that lower accuracy solutions are obtained using shorter CPU time, and higher accuracy solutions are obtained using longer CPU time as shown in Fig. 4-4 for *sko100b* as an example. The data points on each line from the first data point to the last one denote MA, PMA, PMA-DLS, PMA-SLS and PMA-FLS, respectively. From Fig. 4-4, it is also obvious that PMA-DLS and PMA-SLS produce solutions that are competitive with that obtained in PMA and MA at significantly less computational cost. Moreover, PMA-DLS and PMA-SLS achieve much higher solution quality than PMA-FLS with little

increase in CPU time. The trend is more evident when the number of processors increases.

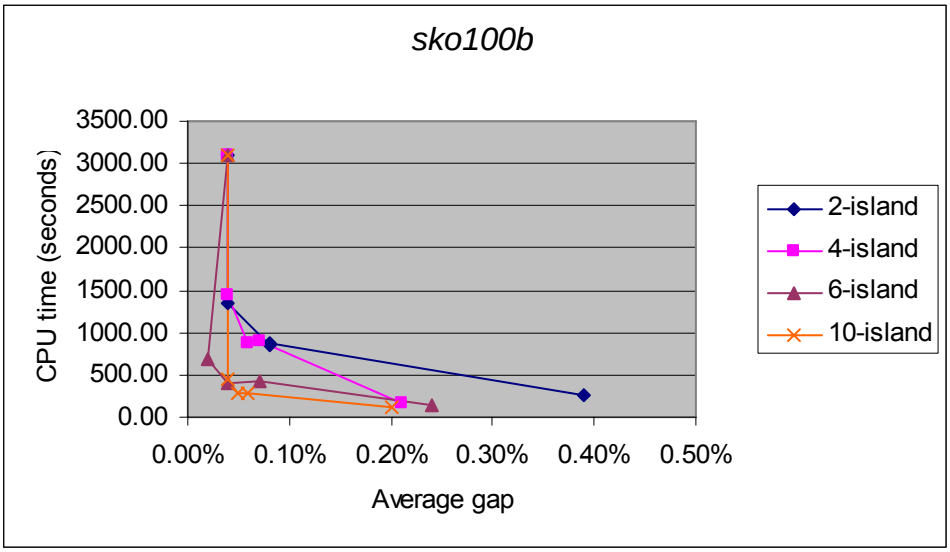


Fig. 4-4 Two-dimensional plot of MA, PMA, PMA-DLS, PMA-SLS and PMA-FLS on *sko100b* benchmark

To determine the significance of the reduced computational time by PMA-DLS and PMA-SLS, a statistical *t*-test was used ($p<0.05$), with the null hypothesis of having no difference between PMA-DLS and PMA, as well as between PMA-SLS and PMA. Based on *t*-test for statistical significance, the mean and the associated results of the one tail difference of two independent means, taken over 10 independent trials for *sko100b* for PMA-DLS, PMA-SLS and PMA, are presented in Table 4-10.

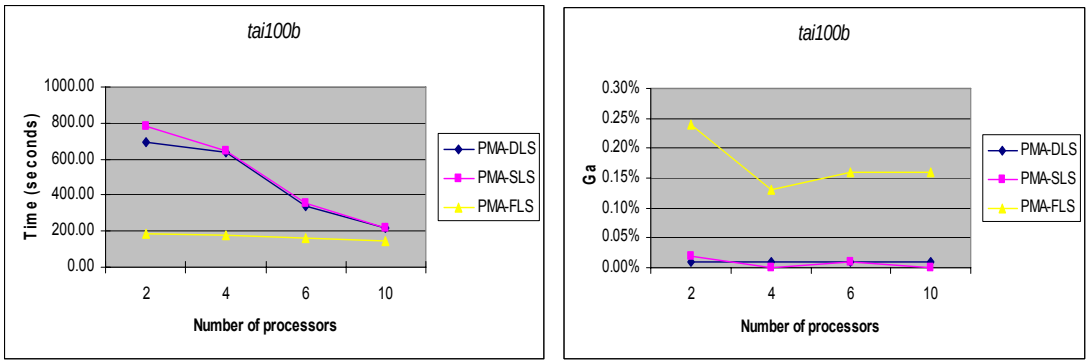
Table 4-10 Mean and the one tail difference of *t*-test between PMA-DLS and PMA as well as between PMA-SLS and PMA

	PMA-SLS Mean	PMA-DLS Mean	PMA Mean	*p*- PMA-SLS *vs.* PMA	*p*- PMA-DLS *vs.* PMA
2-island	875.2	859.4	1350	0.00010793	4.17256E-05
4-island	898	885.7	1445.9	1.0808E-11	3.96277E-11
6-island	429.4	413.8	694.3	1.331E-05	1.36381E-05
10-island	289.3	276.8	439	0.00010352	3.83569E-05

In Table 4-10, for all cases, the mean of both PMA-DLS and PMA-SLS is smaller than that of PMA and the value $p<0.05$ indicates that PMA-DLS and PMA-SLS indeed was able to reduce the computational time compared to PMA with high level of statistical significance. This validates the notion of PMA-DLS and PMA-SLS being able to search more efficiently than PMA to achieve a comparable solution quality.

Similar to the *sko100b* benchmark, the effect of multiple islands processing is plotted as in Fig. 4-5 for the *tai100b* benchmark. The results show that PMA-DLS and

PMA-SLS can achieve much better solution quality with comparable computational time, especially for the case where the number of processors increases to 10 machines. It is also observed that the *tai100b* QAP benchmark shows a much higher *Success rate*, indicating that the PMA-DLS has greater success in locating the global optimum. This implies that the PMA-DLS is capable of locating the best-known solution more frequently than the PMA-FLS.



(a) Computation time (b) Solution quality

Fig. 4-5 Comparison among PMA-DLS, PMA-SLS and PMA-FLS on *tai100b* benchmark

In Tables 4-6 and 4-8, based on observations of the two criteria, *Gap* and *Success Rate*, the results of the different benchmarks (*sko100** and *wil100*) show that PMA-DLS and PMA-SLS can significantly improve the solution quality with comparable computational time, especially so when the number of processors increases to 10 machines. In addition, the results in Table 4-7 show that PMA-DLS and PMA-SLS are even more superior compared to PMA-FLS, even for the seemingly difficult class of benchmarks, *tai100a*. Remarkable improvement in terms of solution quality was observed. The *tai100a* corresponds to a class of problems randomly generated by Taillard using a uniform distribution. In [32], Taillard noted that for this type of randomly generated instances, finding good solutions (about 1% and 2%) is easy, but it is extremely difficult to find the optimum. This type of randomly generated instances is not that significant for practical applications of the QAP. As such, a set of non-uniformly generated random instances (*tai*b*) with the same characteristics as real-life problems was defined. As shown in Table 4-9, for the very large scale benchmark, *tho150*, both PMA-DLS and PMA-SLS can improve the solution quality significantly.

4.5 Discussion

When judged against existing results available in the literature, it is noted that the results for PMA-SLS and PMA-DLS of several instances are much better than that of the MAs developed by other authors. For example, our results on the *tai100a* benchmark problem in Table 4-7 using the PMA-DLS are better than that found in [24]. The *Average gap* of 1.089% for *tai100a* was reported in [24], while the *Average gap* of PMA-DLS for *tai100a* on 10 machines is 1.05%. Also the results of *tai100b* for PMA-SLS are much better than that shown in [23]. The *Average gap* of *tai100b* was reported as 0.026% in [23], with the *Success rate* being less than 50%. On the other hand, *Average gap* achieved by our PMA-SLS (0.01%) and PMA-DLS (0.01%) is much better than that in [23], and the *Success rate* is very commendable, being as high as 80%. Furthermore, it is worth nothing that the PMA-SLS and PMA-DLS are also capable of attaining search quality that is significantly better than that obtained in [24] on the *sko100a* problem. As shown in Table 4-6, on the *sko100a* benchmark, the *Average gap* obtained in [24] was 0.096%, while we were able to reduce this value to 0.06%. As for the very large benchmark, *tho150*, the empirical results in Table 4-9 show that the best solution quality on average, 0.09%, is much better compared to that reported in [24], which was 0.151%.

Furthermore, based on the comparison between PMA-SLS and PMA-DLS, there are three advantages of PMA-DLS which determine the number of individuals undergoing local search based on online dynamic population diversity. First, the number of individuals to be selected for local search is made dynamic and adaptive to online fluctuation of population diversity. Using this diversity-based dynamic adaptation mechanism, it is able to set a high number of individuals for undergoing local search if the population diversity is high. If the population diversity is very low, it is able to decrease the number of candidates undergoing local search to reduce the additional computational effort. In addition, for the island model PMA, the diversity-based dynamic adaptive local search is able to adjust the number of individuals for local search according to the different diversity fluctuation tendency in each island.

Secondly, the PMA-DLS adjusts the local search frequency online, avoiding the laborious task of parameters tuning. Therefore, PMA-DLS is desirable in producing more robust search performance, resulting in overall improvement in solution quality.

Thirdly, an intrinsic characteristic of PMA-DLS is the Markovian property, in deciding the frequency of applying local search. Equation (6) computes the number of chromosomes that undergo local learning in the current generation based on the previous k generations and the current generation. This property is consistent with the theoretical foundation of various evolutionary algorithms, such as genetic algorithms and memetic algorithms [30].

5 Conclusion and future work

This paper proposes two diversity-based adaptive strategies in the island model parallel memetic algorithm with adaptive local search frequency. Instead of having a constant local search frequency, PMA-SLS adopted a diversity-based static adaptive local search strategy based on parameterized Gaussian distribution. However, in the PMA-SLS strategy, its efficient use presupposes tedious tuning of the parameters for the Gaussian function. The Gaussian function used to decide on the local search frequency was problem specific. It was configured through trial-and-error experimentation without generalization or analysis of the characteristics of the PMA with respect to population diversity, an important characteristic indicative of the population convergence level. Furthermore, instead of fastidious tuning of the parameters setting for the Gaussian Function, a diversity-based dynamic adaptive local search strategy is employed in PMA-DLS such that the local search frequency is adaptively adjusted based on the online fluctuation of population diversity. This diversity-adaptive approach avoids premature convergence resulting from fast decreasing population diversity, as well as reduces the computational effort.

The experimental study verifies that PMA-SLS and PMA-DLS show the ability of producing solutions that are competitive with that obtained in PMA at significantly less computational cost for solving large scale QAP. The higher success rate of PMA-DLS also indicates the improved solution precision due to the intrinsic parallelism and the higher level of diversity maintained during the evolutionary process. Furthermore, PMA-DLS achieves more reliable solutions than PMA-SLS and is more robust, being less sensitive to the parameters setting. Hence, there is not much effort expended on tedious parameters tuning which quite often frustrates the setting-up process for PMA-SLS.

Without doubt, this paper will elicit more relevant research work on this topic. The issue on applying different levels of the local search frequency in selecting memes in multi-meme PMA paradigm is meaningful and challenging, deserving further study. The diversity-based adaptive PMA also demonstrates its potential in solving other computationally demanding optimization problems.

Acknowledgements The authors acknowledge the funding support of Singapore Technologies (ST) Engineering Pte Ltd in this work. We wish to specially acknowledge the help and support of two individuals, Mr. David Chin and Mr. Yew Kong Leong from ST Engineering in the work reported here.

References

1. Bambha NK, Bhattacharyya SS, Teich J, Zitzler E (2004) Systematic integration of parameterized local search into evolutionary algorithms. IEEE Transactions on Evolutionary Computation, 8(2): 137-155
2. Bradwell R, Brown K (1999) Parallel asynchronous memetic algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference, Evolutionary Computation and Parallel Processing Workshop, Orlando, Florida
3. Burkard RE, Karisch SE, Rendl F (1997) QAPLIB—A quadratic assignment problem library, Journal of Global Optimization, 10: 391-403, Available from <http://www.opt.math.tu-graz.ac.at/qaplib/>
4. Cotta C, Mendes A, Garcia V, Franca P, Moscato P (2003) Applying memetic algorithms to the analysis of microarray data. Application of Evolutionary Computing, G. Raidl et al (Eds.), Lecture notes in computer science, Springer-Verlag, 2611: 22-32
5. Dawkins R (1976) The selfish gene. Oxford University Press, New York
6. Digalakis JG, Margaritis KG (2004) Performance comparison of memetic algorithms. Journal of Applied Mathematics and Computation, Elsevier Science, 158(25): 237-252
7. Eiben AE, Hinterding R, Michalewicz Z (1999) Parameter control in evolutionary algorithm. IEEE Transactions on Evolutionary computation, 3(2): 124-141
8. Goldberg DE (1989) Genetic Algorithms in Search, Optimization and Machine Learning, MA: Addison-Wesley: Reading
9. Goldberg D, Voessner S (1999) Optimizing global-local search hybrids. In: Banzhaf W et al (Eds.) Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann, pp. 220-228
10. Hart WE (1994) Adaptive global optimization with local search. Ph. D. Thesis, University of California, San Diego
11. Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor
12. Ishibuchi H, Yoshida T, Murata T (2003) Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Transactions on Evolutionary Computation, 7(2): 204-223
13. Knowles J, Corne D (2004) Memetic algorithms for multiobjective optimization: issues, methods and prospects. In: Krasnogor N et al (Eds) Recent Advances in Memetic Algorithms, Springer, pp. 313-352
14. Koopmans TC, Beckmann MJ (1957) Assignment problems and the location of economic activities, Econometrica, 25: 53-76

15. Krasnogor N (2002) Studies on the theory and design space of memetic algorithms. Ph. D. Thesis, University of the West of England, Bristol
16. Ku KWC, Mak MW, Siu WC (2000) A study of the Lamarckian evolution of recurrent neural networks. IEEE Transactions on Evolutionary Computation, 4(1): 31-42
17. Land MWS (1998) Evolutionary algorithms with local search for combinatorial optimization. Ph. D. Thesis, University of California, San Diego
18. Li Y, Pardalos PM, Resende MGC (1994) A greedy randomized adaptive search procedure for the quadratic assignment problem. In: Pardalos P et al (Eds.) Quadratic Assignment and Related Problems, AMS: Providence, Rhode Island, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 16: 173-187
19. Lim MH, Yuan Y, Omatu S (2000) Efficient genetic algorithms using simple genes exchange local search policy for the quadratic assignment problem. Computational Optimization and Applications, 15: 249-268
20. Lim MH, Yuan Y, Omatu S (2002) Extensive testing of a hybrid genetic algorithm for quadratic assignment problem. Computational Optimization and Applications, 23: 47-64
21. Mascato P (1989) On evolution, search, optimization, genetic algorithms and martial arts: Toward memetic algorithms. Tech. Rep. Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA
22. Merz P, Freisleben B (1999) Fitness landscapes and memetic algorithm design. In: Corne D et al (Eds.) New ideas in optimization, McGraw-Hill, London, pp. 245-260
23. Merz P, Freisleben B (1999) A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem. In: Proceedings of the 1999 International Congress of Evolutionary Computation, IEEE Press, pp. 2063-2070
24. Merz P, Freisleben B (2000) Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. IEEE Transactions on Evolutionary Computation, 4(4): 337-352
25. Ong YS, Keane AJ (2004) Meta-lamarckian in memetic algorithm. IEEE Transactions on Evolutionary Computation, 8(2): 99-110
26. Ong YS, Lim MH, Zhu N, Wong KW (2006) Classification of adaptive memetic algorithms: a comparative study. IEEE Transactions on Systems, Man and Cybernetics - Part B, 36(1): 141-152
27. Rosca J (1995) Entropy-driven adaptive representation. In: J. Rosca (Ed.) Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications, Tahoe City, California, USA, pp. 23-32
28. Skorin-Kapov J (1990) Tabu search applied to the quadratic assignment problem. ORSA Journal of Computing, 2(1): 33-45
29. Solimanpur M, Vrat P, Shankar R (2004) Ant colony optimization algorithm to the inter-cell layout problem in cellular manufacturing. European Journal of Operational Research, 157(3): 592-606
30. Suzuki J (1999) A Markov chain analysis on simple genetic algorithms. IEEE Transactions on System, Man, and Cybernetics, 25(4): 5655-659
31. Taillard ED (1991) Robust tabu search for the quadratic assignment problem. Parallel Computing, 17: 443-455
32. Taillard ED (1995) Comparison of iterative searches for the quadratic assignment problem. Location Science, 3: 87-105
33. Tan KC, Lee TH, Khor EF (2001) Evolutionary algorithms with dynamic population size and local exploration for multiobjective optimization. IEEE Transactions on Evolutionary Computation, 5(6): 565-588
34. Tang J, Lim MH, Ong YS (2003) A parallel hybrid GA for combinatorial optimization using grid technology. In: Proceedings of IEEE Congress on Evolutionary Computation, Canberra, Australia, IEEE, 3: 1895-1902

35. Tang J, Lim MH, Ong YS, Er MJ (2004) Study of migration topology in island model parallel hybrid-GA for large scale quadratic assignment problems. In: The Eighth International Conference on Control, Automation, Robotics and Vision (ICARCV2004), Special Session on Computational Intelligence on the Grid, December 6-9, Kunming, China
36. Thonemann UW, Bolte A (1994) An improved simulated annealing algorithm for the quadratic assignment problem. Technical report, School of Business, Department of Production and Operations Research, University of Panderborn, Germany
37. Wilhelm MR, Ward TL (1987) Solving quadratic assignment problems by simulated annealing. IIE Transactions, 19(1): 107-119