Ann Oper Res (2011) 186:231–262
DOI 10.1007/s10479-011-0894-3

A unified framework for population-based metaheuristics

Bo Liu · Ling Wang · Ying Liu · Shouyang Wang

Published online: 13 May 2011

Abstract Based on the analysis of the basic schemes of a variety of population-based metaheuristics (PBMH), the main components of PBMH are described with functional relationships in this paper and a unified framework (UF) is proposed for PBMH to provide a comprehensive way of viewing PBMH and to help understand the essential philosophy of PBMH from a systematic standpoint. The relevance of the proposed UF and some typical PBMH methods is illustrated, including particle swarm optimization, differential evolution, scatter search, ant colony optimization, genetic algorithm, evolutionary programming, and evolution strategies, which can be viewed as the instances of the UF. In addition, as a platform to develop the new population-based metaheuristics, the UF is further explained to provide some designing issues for effective/efficient PBMH algorithms. Subsequently, the UF is extended, namely UFmeme to describe the Memetic Algorithms (MAs) by adding local search (memetic component) to the framework as an extra-feature. Finally, we theoretically

B. Liu (✉) · S. Wang
Center for Forecasting Science, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China
e-mail: liub01@mails.tsinghua.edu.cn

S. Wang
e-mail: sywang@amss.ac.cn

B. Liu · L. Wang
Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Automation, Tsinghua University, Beijing 100084, China

B. Liu
e-mail: bliu@amss.ac.cn

L. Wang
e-mail: wangling@mail.tsinghua.edu.cn

B. Liu · Y. Liu
School of Economics and Management, Beihang University, Beijing, 100191, China

Y. Liu
e-mail: liuying@buaa.edu.cn

analyze the asymptotic convergence properties of PBMH described by the UF and MAs by the UFmeme.

Keywords Metaheuristics · Unified framework · Population-based metaheuristics · Memetic Algorithms · Convergence analysis · Evolutionary computing

1 Introduction

Due to the complexities of optimization problems, optimization techniques have attracted much attention in a variety of fields. So far, researchers have proposed various kinds of optimization techniques to tackle the difficulties. For instance, due to the complexity of shop scheduling in manufacturing systems, exact solution techniques, such as branch and bound and mathematical programming (Lageweg et al. 1978), are hard to apply to large-scale problems. Although the constructive heuristics (Johnson 1954; Nawaz et al. 1983) can quickly construct feasible schedules from scratch, it is hard to obtain satisfactory solutions (Ruiz and Maroto 2005). Starting from one or a set of solutions and gradually improving the solution(s) by applying some specific problem knowledge, the single solution-based metaheuristics, e.g., simulated annealing (SA) (Ishibuchi et al. 1995), tabu search (TS) (Nowicki and Smutnicki 1996; Grabowski and Pempera 2001, 2007), and variable neighborhood search (VNS) (Hansen and Mladenovic 2001), or the population-based metaheuristics (PBMH), e.g., genetic algorithms (GAs) (Holland 1975; Wang 2003; Smutnicki and Tyński 2006), have been applicable to a wide range of problems without being tailored and can find near- or global-optimal solutions with acceptable computational costs (Glover and Kochenberger 2003; Hart et al. 2005). Population-based metaheuristics (PBMH) are characterized by iterative, random, population-based and often biologically or socially inspired features, which represent a range of universal problem solvers. As special incarnations of PBMH, Evolutionary Algorithms (EAs) that are inspired by biological evolution, e.g., genetic algorithms (GAs) (Wang 2003; Reeves and Yamada 1998; Holland 1975), evolutionary programming (EP) (Fogel et al. 1966), evolution strategy (ES) (Schwefel 1995), and genetic programming (GP) (Koza 1992), have shown great potential in tackling some hard scheduling problems and have been one of the main subjects of research in the community of Operations Research (OR) (Dimopoulos and Zalzala 2000; Ruiz and Maroto 2005).

Especially important, besides the above EAs which are all based on genetics and (natural/artificial) principles (Bäck 1996), so far, a large amount of PBMH have been proposed, which are not based on such evolutionary principles, though their main loop shares some similarities with EAs. Among this kind of category are swarm intelligence (e.g., ant colony optimization and particle swarm optimization) (Bonabeau et al. 1999; Kennedy et al. 2001; Dorigo and Gambardella 2002), differential evolution (DE) (Price et al. 2005), cultural algorithms (CA) (Reynolds 1994), scatter search (SS) (Glover 1977, 1998; Laguna and Martí 2003; Martí 2006; Martí et al. 2006.), and artificial immune systems (AIS) (Dasgupta 1998), etc. Most of these techniques are similar in nature, but differ in the details of implementations as well as the nature of the problems to solve. Currently, particle swarm optimization (PSO) (Sha and Hsu 2006; Tasgetiren et al. 2007; Liu et al. 2007, 2008; Li et al. 2008; Wang and Liu 2008), DE (Nearchou and Omirou 2006; Onwubolu and Davendra 2006), SS (Nowicki and Smutnicki 2005), and ant colony optimization (ACO) (Gravel et al. 2002; T'kindt et al. 2002; Rajendran and Ziegler 2004), have been investigated and stimulated the interests of research in OR.

During the past two decades, population-based optimization has attracted great attention from both academia and industry in many fields not limited in OR. This is motivated, firstly by the increasing requirements to improve searching efficiency, to improve searching quality and to reduce computational costs; secondly by the increasing difficulties arising from engineering optimization problems, such as NP-hardness, multiple objectives (Fonseca and Fleming 1995; Hoogeveen 2005; Coello Coello 2006), strong and many constraints (Michalewicz and Schoenauer 1996; Coello Coello 2002), co-existence of discrete and continuous variables (Hart et al. 2005; Bonissone et al. 2006), dynamic and non-deterministic environments (Jin and Branke 2005). As the most popular type of PBMH, *Genetic algorithm* seeks the solution of a problem in the form of a string of numbers and always uses recombination in addition to selection and mutation operations. With solutions in the form of computer programs, *Genetic programming* determines the fitness of solution by the ability to solve a computational problem. With a fixed structure of the program, *Evolutionary programming* evolves numerical parameters. *Evolution strategy* works with a vector of real numbers and usually uses self-adaptive mutation rates. *Differential evolution* and *Particle swarm optimization* are based on vector differences and the idea of animal flocking behavior respectively, and they are both primarily suited for numerical optimization problems. Based on the idea of ant foraging by pheromone communication to form path, *Ant system* (AS) is mainly suited for combinatorial optimization problems. So far, many improvements and generalizations of PBMH are available. The application fields of each PBMH have been extended, and hybridization (sometimes called Memetic Algorithms, MAs) of PBMH with local search and domain knowledge has also been a hot topic (Moscato 1989; Ong 2002; Hart et al. 2004; Ong and Keane 2004; Krasnogor and Smith 2005; Bonissone et al. 2006).

Compared with a lot of literature on the application of PBMH on specific problems, the systematical research on PBMH is also very important but rather scarce at present, such as the unified framework and the systematical convergence theory (De Jong 2006). Particularly, most of the theoretical research efforts on unified framework and convergence theory have been only focused on the EAs, not on PBMH. For instance, Eiben et al. (1995) presented a General Search Procedure (GSP) to provide a unifying framework for describing both constructive and iterative search algorithms, including GA, SA, threshold accepting, hill-climbing, depth-first search, breadth-first search, and best-first search. In order to specify such a general algorithm, seven elements were defined, i.e., search space, operations on set space, initial solution(s), selection, production, reduction and evaluation function. The unifying framework provided a ground to study the theoretical properties and served as a basis to implement the search-based problem solver described by the authors. By examining the roles of the most important components used in EAs, Eiben and Smith (2003) presented a general scheme consisting of six crucial elements, i.e., *representation* (definition of individuals), *evaluation function* (fitness function), *population*, *parent selection mechanism*, v*ariation operators* (recombination and mutation), and *survivor selection mechanism* (replacement). And a particular EA could be defined by specifying the details of the six components. Hertz and Kobler (2000) and Calégari et al. (1999) respectively proposed general frameworks for EAs, where seven main features were used to characterize a particular EA, namely *individuals*, *evolution process*, *neighborhood*, *information sources*, *infeasibility*, *intensification*, and *diversification*. And the proposed framework was used to explain the underlying philosophy of several well-known techniques, such as GA, SS, AS and adaptive memory algorithms. Smith (1998) formalized the GAs within a "generate-and-test" framework and proposed a framework for classifying adaptive GAs based on the scope of the adaptation as well as the nature of the transition function guiding the search. By analyzing a number of memory-based meta-heuristics, such as TS, SS, GA and AS, Taillard et al.

(2001) found that the implementations of these algorithms are very similar and proposed a unified presentation named adaptive memory programming, which is characterized by the exploitation of a memory to construct a new solution, an improvement procedure to find a even better solution, and a memory updating procedure based on the knowledge brought by the improved solution. Bäck (1996) unified different EA such as GP, GAs and ES, all based on genetics and (natural/artificial) selection. Meanwhile, the research on unifying taxonomic framework for hybrid or memetic algorithms has also caught attention. Talbi (2002) provided a general classification scheme for hybrid algorithms based on the design space and implementation space. Krasnogor and Smith (2005) designed a syntactical model for MAs to provide a better understanding about the roles/interrelations of/between different components of MAs. With the syntactical model, taxonomy of MAs was constructed to give directions of innovation in designing and developing MAs. Since 1990s, many novel algorithms following the general idea of PBMH have emerged, such as AS, PSO and DE. These population-based algorithms are not biologically but socially inspired, and they also cannot be simply regarded as instances of the above proposed unified framework which are dedicated to depicting EAs. Obviously, it is a challenge and meaningful to provide a unified framework as general as possible, which not only includes those algorithms the researches previously focused on but also the newly emerging computing techniques. Thus, we could understand the essential philosophy of more generally PBMH from a systematical standpoint.

After several decades of research in the field of PBMH, the scientific community observes that the number of applications based on PBMH has grown rapidly, whereas the advancement of PBMH theory has been much slower, and there is a noticeable gap between the application of PBMH and the related theory. For instance, in the field of Operational Research, the heuristic methods including EC based methods, commonly as a preferred alternative to exact methods have greater flexibility and power to take advantage of special properties of the search space and could obtain satisfying optimization results, but they do not provide a convergence guarantee (Kochenberger et al. 2004). So, it is particularly important to carry out systematically theoretical research on PBMH, where the convergence analysis is one of the most challenging issues. A few of research works have intended to fill up the gap, while, most of the theoretical research efforts on convergence theory have been only focused on the EAs, little on PBMH. Apart from the schemata theorem of simple GA (Holland 1975), Markov chain theory has been used to model the behaviors of GAs (Nix and Vose 1992). Based on Markov chain theory, Eiben et al. (1991) and Fogel (1992) proved GAs with elitist selection converge to the global optimum with probability 1. Rudolph (1994) proved that the canonical GA cannot converge to the global optimum but elitist strategy can help it converge with probability 1. Generalizing the theory (Rudolph 1994) from binary Euclidean search space to general search space, Rudolph (1996) provided conditions for EAs with elitist strategy to converge to the global optimum with probability 1. By evaluating the eigenvalues of the transition matrix of the Markov chain, Suzuki (1995) estimated the convergence rate of the GA with elitist strategy in terms of a mutation probability. Liu et al. (2006) proved the convergence property of their multi-agent GA for constraint satisfaction problems. By using non-stationary Markov model, Cao and Wu (1997) investigated the asymptotic convergence property of the GA with adaptive crossover and mutation. Based on the theory about the convergence rate of Markov chain (Rosenthal 1995), He and Kang (1999) discussed the convergence rate of GA by using the minorization condition and obtained a bound on the convergence rate for the algorithm with time-invariant operators on a general state space as well as a bound on the convergence rate for the algorithm with time-variant operators on a finite state space. Based on Davis's work (1991), Suzuki (1998)

discussed the theoretical property on the Markov chain of GA, which is applicable to SA-like strategies. Rudolph (1997) analyzed the effect of Cauchy mutation distribution on the local convergence behavior of EAs. In addition, Fogel (1992) analyzed the convergence rate of EP based on Markov chain, and François (1998) theoretically investigated a simple ES to provide the relationship between convergence property and the parameters of the strategy. As for PSO, based on the theoretical analysis of parameter selection (Clerc and Kennedy 2002), Trelea (2003) analyzed the dynamic behavior and the convergence property using discrete-time dynamic system theory and provided qualitative guidelines for parameter selection. A good survey about convergence conditions and Markov chain analysis in EA field could be found in Rudolph (1998). As for hybrid EAs, Li and Jiang (2000) provided the proof of globally asymptotical convergence of the hybrid algorithm combining GA, SA and chemotaxis algorithm. Based on the convergence analysis for GAs, Ong et al. (2006) presented classification of memes adaptation in adaptive MAs on the basis of the mechanism used and the level of historical knowledge on the memes employed, and analyzed the asymptotic convergence property of the adaptive MAs by means of finite Markov chain. Based on the model of evolutionary Turing machine, Eberbach (2005) studied the convergence property and rate of EAs, and the sufficient conditions of the completeness and optimality of EA were also investigated.

Based on the above investigation and analysis, we would like to remark that the term EAs itself already resulted from a unification of the field that comprised, for historical reasons, different branches EP, ES, GA and GP. The unified EA as presented for example by Bäck (1996) was summarizing EA in a unified framework. Also convergence conditions were stated for this framework, based on the transition matrix of the Markov chain, and even for the more general multi-objective case such results are well established right now. However, the extension of this unification to PBMH is relatively new, which is also our main motivation for writing this paper. And the convergence rate and finite-time performance analysis of PBMH based on a more general unified framework is also necessary, and would be more useful for a practical user.

In this paper, we describe the main components of PBMH with functional relationships and propose a unified framework (UF) for PBMH to provide a comprehensive way of describing population-based metaheuristics and to help understand the essential philosophy of PBMH from a systematic standpoint. Then, we illustrate the use of the proposed UF to describe some typical PBMH, e.g., PSO, DE, SS, AS, GA, EP and ES, which can be regarded as instances of the UF. In addition, as a platform to develop new PBMH, the UF is further explained to provide some guidelines for designing effective/efficient algorithms. Subsequently, the UF is extended, namely UFmeme to describe the memetic algorithms (MAs) by adding local search (memetic component) to the framework as an extra-feature. Finally, we theoretically analyze the asymptotic convergence properties of PBMH described by the UF and memetic algorithm by the UFmeme.

The remaining contents are organized as follows. In Sect. 2, we provide the general scheme of PBMH and propose the UF. In Sect. 3, the proposed UF is illustrated by seven well-known PBMH. In Sect. 4, the UF is further explained to provide some guidelines for designing effective/efficient PBMH algorithms. In Sect. 5, the UF is generalized to UFmeme. In Sect. 6, the asymptotic convergent properties of the UF-based PBMH and the UFmeme-based MAs are theoretically analyzed. Finally, in Sect. 7, we end the paper with some conclusions.

INITIALIZE population
While (*TERMINATION CONDITION* is not satisfied) **do**
    S*OCIAL-COOPERATION;*
    SELF-ADAPTATION;
    COMPETITION;
End

Fig. 1 The general framework for PBMH

2 Unified framework for PBMH

2.1 Basic framework of PBMH

Population-based metaheuristics (PBMH) are the generic concepts to represent a range of metaheuristics with iterative process, population-based, guided random search, parallel processing, and inspired by biological, social or other kinds of features. The common underlying philosophy behind PBMH is similar, and their main loop shares some similarities with each other, while the PBMH differ in the details of implementations as well as the nature of the problems to solve. As for any a particular PBMH, its population-based evolution mainly depends on its parameters and operations inspired from biology or social behavior. Based on the analysis of the basic schemes of PBMH, at each loop of PBMH, the main components of PBMH can be described with three main components, namely *social-cooperation*, *self-adaptation*, and *competition* work consecutively. *Social-cooperation* represents the collaborative effect of all individuals in finding the global optima by exchanging information among them and by learning from each other. *Self-adaptation* represents the each individual's personal active/passive thinking of itself, which encourages the individual to be adapted to the environment. *Competition* denotes the updating of population and the better individual, especially the fittest one, will survive in the environment. Based on these main components, a general framework for basic PBMH is shown in Fig. 1.

The most important feature of the framework is its generality, by which any a particular PBMH can be described. In GA, for instance, the crossover operation can be viewed as social-cooperation, the mutation can be considered as self-adaptation, according to which a passive action is performed on an individual to avoiding premature convergence, and the selection can be regarded as competition. In PSO, "adjusting the trajectory of each individual toward the best particle of the swarm" can be considered as social-cooperation, "adjusting the trajectory of each individual toward its own best location" can be viewed as self-adaptation, and the swarm updating strategy corresponds to the competition element. In this work, we focus on studying the unified framework for the above three components and presenting a unified framework for PBMH. Since the initialization and termination condition are usually not specified by a variant of PBMH but implemented under general consideration, they are not discussed in this paper.

2.2 Unified framework (UF)

Based on the above general framework, we describe the PBMH with the following UF, where the social-cooperation, self-adaptation, competition and their related parameters are included.

$$\text{PBMH} = (Pop, S, A, C, \alpha, \beta, \gamma, t). \tag{1}$$

where

$Pop = (Individual_1, Individual_2, \ldots, Individual_\mu)$: Population consisting of μ individuals, where μ denotes the size of population.

S: Component corresponds to social-cooperation.

α: Information needed by the social-cooperation procedure.

A: Component corresponds to self-adaptation.

β: Information needed by the self-adaptation procedure.

C: Component corresponds to competition.

γ: Information needed by the competition procedure.

t: Time or generation.

Next, we will explain the above framework and its related components.

2.2.1 Social-cooperation

The social-cooperation represents the collaborative effect of all individuals in finding the global optima by exchanging information among them and learning from each other. Based on the above definition and analysis of the basic components in social-cooperation, the main information needed to define a special social-cooperation procedure includes: the choice strategy on which individual(s) to cooperate with or learn from for one individual (denoted by s_c), the number of individuals involved (denoted by s_n), the collaborative way of individuals to create new individual(s) (denoted by s_w), and the way of utilizing the history information of population (denoted by s_h). Thus, the social-cooperation is formulized with the following expression:

$$Pop^t_S = S(Pop^t, \alpha^t) \tag{2}$$

where Pop^t and Pop^t_S denote the population at time t before and after social-cooperation, respectively, and $\alpha^t = [s_c, s_n, s_w, s_h]$ denotes the information needed by the social-cooperation to specify a particular cooperative strategy.

2.2.2 Self-adaptation

The self-adaptation represents each individual's personal active/passive thinking of itself without resorting to the information from other individuals, which encourages the individual to evolve independently and to be adapted to the environment. According to its personal adaptation, the individual may respond to its surroundings in two kinds of attitudes, i.e., intensification or diversification searches. The former focuses on improving the quality of individuals by intensive exploitation of the regions around the individual, while the latter concentrates on avoiding premature convergence by extensive exploration of a larger search space. Thus, the self-adaptation can be viewed as a balance element between local exploitation and global exploration.

Thus, the formulization of self-adaptation can be represented by the following expression:

$$Pop^t_A = A(Pop^t, \beta^t) \tag{3}$$

where Pop^t_A denotes the population at time t after individual-adaptation, β^t denotes the information needed by the self-adaptation procedure, which is used to specify a particular self-adaptation strategy.

The above two components of social-cooperation and self-adaptation are used to generate new individuals from old ones, which can be represented by the following expression:

$$O^t = A(S(Pop^t, \alpha^t), \beta^t) \tag{4}$$

where O^t denotes the set of offspring at time t.

2.2.3 Competition (population updating)

The component of competition, that is, the population updating strategy, means how to select the population surviving for the next new generation from the candidate pool mixed by μ parents and λ offspring. The competition procedure often occurs after generating the new candidate solutions by the social-cooperation and self-adaptation processes. Three principal factors to specify a particular competition strategy are the population size c_p (fixed or unfixed), replacement strategy c_r (generational or steady-state), and elitist strategy c_e. Most PBMH work with a population with constant size c_p, whereas there are a few algorithms with variable population size during search process. As for the replacement strategy, generational replacement (denoted by (μ, λ)) means that the λ new individuals replace the μ old individuals between two consecutive iterations, while steady-state replacement (denoted by $(\mu + \lambda)$) means that in the pool of mixture of the μ old individuals and the λ new individuals, some individual are selected. Besides, some algorithms use elitism strategy to reserve good solutions, while others do not apply elitism strategy. Thus, the formulization for competition to generate a new population can be represented by the following expression:

$$Pop^{t+1} = C(Pop^t, O^t, \gamma^t) \tag{5}$$

where Pop^{t+1} denotes the new generated population for time $t+1$, $\gamma^t = [c_p, c_r, c_e]$ denotes the information needed by the competition procedure to specify a particular competition strategy.

2.2.4 UF for PBMH

According to the above formulae for the three main components used in PBMH, an overall unified framework (UF) for PBMH is given as follows:

$$Pop^{t+1} = C(Pop^t, A(S(Pop^t, \alpha^t), \beta^t), \gamma^t) \tag{6}$$

From the above UF, it can be seen that the general framework of PBMH could be well described by the UF. And in the next section, we will specify how to implement the components of the UF to form some particular PBMH.

2.3 Comparison with prevailing framework

In this section, we make an attempt to clarify the main difference between our proposed unified framework and other unified frameworks in the literature, especially the well-known Adaptive Memory Programming (Taillard et al. 2001), which was investigated in the previous introduction section.

The major differences could be summarized as follows: Firstly, most of the research efforts on unified framework have been only focused on the biologically inspired, not on PBMH. The extension of unified framework to PBMH is relatively new, which is also our

main motivation for writing this paper. Since 1990s, many novel algorithms following the general idea of PBMH have emerged, such as ACO, PSO and DE. These population based algorithms are not biologically but socially inspired, and they also cannot be simply regarded as instances of the proposed unified framework in literature which are dedicated to depicting EAs. The proposed UF in this paper provides a unified view as general as possible, which not only includes those algorithms the researches previously focused on but also the newly emerging computing techniques.

Secondly, both Adaptive Memory Programming (AMP) and UF proposed in our paper are similar in terms of the capacity of capturing the common/general underlying philosophy behind the algorithms. However, they differ in the capacity of revealing the details for each component composing the unified viewpoint. Consequently, this difference leads to different capacities in describing algorithms as well as designing novel algorithms. For instance, AMP is characterized by the exploitation of a memory to construct a new solution, an improvement procedure, and a memory updating procedure. As a general framework, AMP does not provide specific features for each component, which depresses its ability to implement the AMP to some particular metaheuristics, and to act as platform to develop new versions of algorithms. In UF for PBMH, the main components of PBMH are described with functional relationships, which define parameters and operation strategies in detail. In other words, UF for PBMH does not overlook the implementation details of algorithms. This enables the user to quickly develop new and effective PBMH by choosing suitable parameter/strategy settings.

Thirdly, the UF could describe all of the population based algorithms (SS, ACO, and GA) in AMP paper, while AMP cannot describe the EP in our paper, which works without memory.

Finally, the components of each PBMH could be explicitly recognized, while the components for AMP are not so explicit. For instance, as an elementary component in AMP, the definition of memory is changing for different algorithms. It is pointed out that "*TS is the only one that has been explicitly developed with a memory*" and "*But a number of other metaheuristics use mechanisms that can be considered as memories. For GA and SS, the memory is constituted by a population of solutions; for AS, the pheromone trail is also a form of memory*". In other words, the rest do not work with explicit memory, which makes it hard to recognize the memory component for other methods which are not listed in AMP paper. To sum up, compared with AMP, UF with clearer definition of components, could achieve a better balance between breadth of covering the algorithms and depth in describing the algorithm's details.

3 Illustrations

Based on the analysis of the basic schemes of a variety of population-based metaheuristics (PBMH), a unified framework (UF) was proposed in last section to provide a systematic view of PBMH. In this section, the relevance of UF and some typical PBMH algorithms is illustrated, including particle swarm optimization, differential evolution, scatter search, ant colony optimization, genetic algorithm, evolutionary programming, and evolution strategies. Compared with researches on frameworks in the literature, this study shows more incarnations of PBMH, and the list of PBMH methods is exhaustive. We make an attempt to demonstrate that all these methods differ a lot with each other in their implementation details which could not be overlooked. And the UF for PBMH can easily reveal the implementation details and make clear distinction between each PBMH algorithms, even if the difference between them is subtle.

BEGIN

Initialize: Generate initial swarm of PSO by randomizing particles' positions and velocities
Evaluating all particles in the swarm.
Initialize *personal best* P_i of particles, and *global best* P_g of the swarm.

While (Stopping conditions are not satisfied)
Updating the velocity and position of each particle using the following equations:

$$V_i(t+1) = wV_i(t) + c_1r_1(P_i(t) - X_i(t)) + c_2r_2(P_g(t) - X_i(t))$$

$$X_i(t+1) = X_i(t) + V_i(t+1)$$

Evaluating all particles in the swarm, and updating personal/global best.

End While

Output *global best* P_g and its objective value.

END

Fig. 2 The procedure of PSO

3.1 Particle swarm optimization

3.1.1 Introduction to PSO

As the first example, we consider the particle swarm optimization (PSO) (Kennedy et al. 2001), whose development is based on observation of the social behaviors of animals such as bird flocking and swarm theory. PSO has memory, so knowledge of good solutions is retained by all particles. Additionally, it has constructive cooperation between particles in the swarm, which share information among one another.

In PSO, it starts with the random initialization of a population (*swarm*) of individuals (*particles*) in the search space and works on the social behavior of the particles in the swarm. Therefore, it finds the global best solution by simply adjusting the trajectory of each individual towards its own best location and towards the best particle of the swarm at each time step (*generation*). However, the trajectory of each individual in the search space is adjusted by dynamically altering the velocity of each particle, according to its own flying experience and the flying experience of the other particles in the search space. The procedure of standard PSO is illustrated in Fig. 2.

3.1.2 UF for PSO

As for the social-cooperation strategy, each individual in PSO chooses the *global best* P_g to cooperate with or learn from. The way that individuals collaborate with P_g to create new individual is shown in the third part of velocity updating equation in Fig. 2, which is known as the "social" component, representing the collaborative effect of the particles in finding the global optimal solution. The social component always pulls the particles toward the global best particle found so far. And the history information of population is P_g, which is the best particle found so far at time t in the entire swarm. Thus, we get $\alpha^t = [s_c, s_n, s_w, s_h]$, where, $s_c = choose P_g$, $s_n = 1$, $s_w = c_2r_2(P_g(t) - X_i(t))$, and $s_h = P_g$.

As for the self-adaptation procedure used in PSO, an updated velocity is added to a position for each particle, which represents the personal thinking of each particle shown in the velocity updating equation in Fig. 2, i.e., $\beta^t = [V_i(t+1)] = [wV_i(t) + c_1 r_1(P_i(t) - X_i(t)) + c_2 r_2(P_g(t) - X_i(t))]$.

The parameters of competition procedure in PSO are denoted by $\gamma^t = [c_p, c_r, c_e]$, where, $c_p = \textit{fixed}$, $c_r = (\mu, \lambda)$, and $c_e = P_g/P_i$ *achived and updated*, which means that PSO works with a population with constant size ($\mu = \lambda$), a greedy competition strategy, that is, new $X_i(t+1)$ will always replace $X_i(t)$, and an elitism strategy, that is, updating the *personal best* P_i and *global best* P_g if better solutions are found.

3.2 Differential evolution

3.2.1 Introduction to DE

As the second example, we consider differential evolution (DE) which uses simple differential operator to create new candidate solutions and one-to-one competition scheme to greedily select new candidate, and works with real numbers in natural manner to avoid complicated generic searching operators in GA. Although the original objective in the development of DE was for solving the Chebychev polynomial problem, it has been found to be an efficient and effective solution technique for complex functional optimization problems.

In DE, a population of solutions is initialized randomly, which is evolved to find optimal solutions through the mutation, crossover, and selecting operation procedures. It finds the global best solution by utilizing the distance and direction information according to the differentiations among population. However, the searching behavior of each individual in the search space is adjusted by dynamically altering the differentiation's direction and step length in which this differentiation performs. At each generation, the *mutation* and *crossover* operators are applied on the individuals, and a new population arises. Then, *selection* takes place, and the corresponding individuals from both populations compete to comprise the next generation.

DE also has memory, so knowledge of good solutions is retained in current population, whereas in GA previous knowledge of the problem is destroyed once the population changes and in PSO a secondary archive is needed. DE also has constructive cooperation between individuals, individuals in the population share information between them. Nowadays, DE has attracted much attention and gained wide applications in various fields (Price et al. 2005).

Several variants of DE have been proposed, depending on the selection of the base vector to be perturbed, the number and selection of the differentiation vectors and the type of crossover operators (Price et al. 2005). The procedure of standard DE, denoted as DE/best/1/bin, is summarized in Fig. 3, where $r1, r2 \in \{1, 2, \ldots, N\}$ are randomly chosen and mutually different and also different from the current index i. $F \in (0, 2]$ is a constant number, called scaling factor, which controls amplification of the differential variation $X_{r1}(t) - X_{r2}(t)$. $X_{best}(t)$, the base vector to be perturbed, is the best member of the current population, so that the best information could be shared among the population, $rand(j)$ is the j-th independent random number uniformly distributed in the range of [0, 1]. $randn(i)$ is an index randomly chosen from the set $\{1, 2, \ldots, d\}$. $CR \in [0, 1]$ is a constant number, called crossover parameter, which controls the diversity of the population.

BEGIN

Initialize: Randomly initialize the population of individual for DE.
Evaluating all individuals.
Determine X_{best} which has the best objective value.

While (Stopping conditions are not satisfied)

Perform ***mutation*** for each individual using the following equation in order to obtain each individual's mutant counterpart.

$$V_i(t+1) = X_{best}(t) + F(X_{r1}(t) - X_{r2}(t))$$

Perform ***crossover*** between each individual and its corresponding mutant counterpart using the following equation in order to obtain each individual's trial individual.

$$u_{i,j}(t+1) = \begin{cases} v_{i,j}(t+1), & \text{if } (rand(j) \le CR) \text{ or } j = randn(i), \\ & \quad j = 1, 2, \ldots, d \\ x_{i,j}(t), & \text{otherwise.} \end{cases}$$

Evaluate the objective values of the trial individuals.
Perform ***selection*** between each individual and its corresponding trial counterpart using the following equation so as to generate the new individual for the next generation.

$$X_i(t+1) = \begin{cases} U_i(t+1), & \text{if } F(U_i(t+1)) < F(X_i(t)), \\ X_i(t), & \text{otherwise.} \end{cases}$$

Updating X_{best} and its objective value if possible.

End While

Output X_{best} and its objective value.

END

Fig. 3 The procedure of DE

3.2.2 UF for DE

DE works with three operators (i.e., mutation, crossover, and selection), which are iteratively used in order to make the population evolve. Mutation occurs at the social-cooperation phase, crossover happens at the stage of self-adaptation, and the selection is used in the competition phase. As for the social-cooperation strategy of DE/best/1/bin, each individual is generated by adding the weighted difference between a defined number of individuals randomly selected from the previous population ($X_{r1}(t)$ and $X_{r2}(t)$) to another individual $X_{best}(t)$, as shown in mutation equation in Fig. 3. Thus, we get $\alpha^t = [s_c, s_n, s_w, s_h]$, where, $s_c = chooseX_{best}(t), X_{r1}(t),$ and $X_{r2}(t), s_n = 3, s_w = X_{best}(t) + F(X_{r1}(t) - X_{r2}(t))$, and $s_h = X_{best}$.

As for the self-adaptation procedure used in DE, the *crossover* operator is applied to increase the diversity of the population, which acts as self-adaptation shown in the crossover equation in Fig. 3, i.e., $\beta^t = [crossover]$.

The parameters for competition procedure in DE is denoted by $\gamma^t = [c_p, c_r, c_e]$, where, $c_p = fixed$, $c_r = (\mu + \lambda)$, and $c_e = X_{best}$, which means that DE works with a population with constant size ($\mu = \lambda$), a one to one based greedy selection criterion is used shown in the selection equation in Fig. 3, and updated the X_{best} if possible.

BEGIN
Initialize: Use *Diversification Generation Method* to generate a population of trial solutions
Use an arbitrary trial solution (or seed solution) as an input
Apply the *Improvement Method* to transform a trial solution into one or more enhanced trial solutions.
Use the A *Reference Set Update Method* to build and maintain a *reference set* consisting of the b "best" solutions found
While (Stopping conditions are not satisfied)
Generate *new subsets* with the *Subset Generation Method* by operating on the reference set, and a subset of its solutions is produced as a basis for creating combined solutions
Apply *Solution Combination Method* to a selected subset of solutions produced by the above step to obtain one or more new trial solutions.
Apply *Improvement Method* to the trial solutions produced in last step.
Apply *Reference Set Update Method.*
End While
Output *Reference Set*.
END

Fig. 4 The procedure of SS

3.3 Scatter search

3.3.1 Introduction to SS

As the third example, we consider the scatter search (SS) (Glover 1977; Laguna and Martí 2003) which was first introduced by Glover as a heuristic for integer programming. SS was motivated by the idea of using a systematic process to enable solution combinations to meet the desired characteristics or restrictions. Although the original objective in the development of SS was for solving the integer programming: creating composite decision rules and surrogate constraints, nowadays it has been successfully applied to a wide range of hard optimization problems (Martí 2006). In contrast to other PBMH like PSO, SS and Path Relinking (PR) **systematically** orient their exploration and exploitation relative to a set of reference points that typically consist of good solutions with some degree of diversity (Martí 2006). SS utilizes systematic designs for creating new solutions and provides unifying principles for joining solutions based on generalized path constructions in Euclidean space while other approaches resort to randomization. It maintains a relatively small reference set of solutions and chooses two or more elements of the reference set in a systematic way with the purpose of creating new solutions, whereas in GAs, the population size is usually large which facilitates thoroughly sampling the search space to create offspring by the crossover and mutation.

In SS (Glover 1998), it generates a starting set of solution vectors to guarantee a critical level of diversity and applies heuristic processes designed for the problem considered as an attempt to improve these solutions. Then, a subset of the best vectors is designated to be reference solutions in terms of quality and diversity. New solutions are created by means of structured combinations of subsets of the current reference solutions and the heuristic

processes applied above are used again to improve the new solutions. Finally, a collection of the "best" improved solutions is added to the reference set. Glover (1998) identified a well known template for implementation of SS/PR, consisting of five subroutines, listed as (1) diversification generation, (2) reference set update, (3) subset generation, (4) solution combination, and (5) improvement. These steps are repeated until the reference set does not change. Through strategic exploration of the template, effective and efficient SS/PR implementation can be achieved. Since each of its subroutines in the template can be implemented in a variety of ways, SS/PR could be flexible and the basic procedure of SS/PR template is summarized in Fig. 4.

3.3.2 UF for SS

A SS works with five operators (i.e., diversification generation, improvement, reference set update, subset generation, and solution combination) that are iteratively used in order to make the reference set evolve. Diversification generation occurs at the initialization phase, improvement happens at the stage of self-adaptation, reference set update is used in the competition phase, and both subset generation and solution combination are at the social-cooperation phase. As for the social-cooperation strategy of SS, $\alpha^t = [s_c, s_n, s_w, s_h]$, where, $s_c =$ *subset generation strategy*, the value of s_n is not more than the size of reference set, $s_w =$ *sulution combination strategy*, and $s_h =$ *reference set*.

As for the self-adaptation procedure used in SS, the improvement operation is applied to increase the quality of the trial solutions, i.e., $\beta^t =$ *improvement strategy*.

The parameters for competition procedure in SS is denoted by $\gamma^t = [c_p, c_r, c_e]$, where, $c_p =$ *fixed*, $c_r = (\mu + \lambda)$, and $c_e =$ *reference set*, which means that SS works with a reference set with constant size ($\mu = \lambda = b$), a collection of the "best" improved solutions is added to the reference set, Reference Set Update method is to build Reference Set with the "best" b solutions, and then order the solutions in Reference Set according to their qualities.

3.4 Ant colony optimization

3.4.1 Introduction to ACO

Ant Colony Optimization (ACO) (Bonabeau et al. 1999) is dedicated to discrete optimization problems. Similar to PSO, the development of ACO is based on observations of the social foraging behaviors of ant colonies, and swarm theory. ACO has some attractive characteristics. Ants in a colony exhibit collective activities such as foraging, brood care, and nest building where the mechanisms of self-organization, stigmergic communication, and task partitioning are utilized, i.e., ants cooperate using a form of indirect communication mediated by the pheromone they deposit to build solutions, and the construction of good solutions is a result of the ants' cooperative interaction. Additionally, ACO could perform robust searches in the dynamic optimization environments, and it could continue working even though some ants in the colony fail (Dorigo and Gambardella 2002). Nowadays, ACO has been applied successfully to lots of hard combinatorial optimization problems such as traveling salesman problems, quadratic assignment problems, and scheduling problems.

In ACO, each ant is a constructive procedure that is able to generate feasible solutions by moving on the construction graph corresponding to the considered problem. In each iteration of the construction, each ant decides where to move on the construction graph so as to build a solution incrementally according to the *state transition rule* consisting of two factors, i.e., the pheromone concentration and preference. The former informs the ants of the qualities of

BEGIN
Initialize: Initialize pheromone concentration and parameters.
While (Stopping conditions are not satisfied)
Build a new colony of ants using *state transition rule*
Evaluate ants.
Update pheromone concentration according to *pheromone updating rule.*
End While
Output the best ant.
END

Fig. 5 The procedure of ACO

the solutions that have been generated by making the corresponding move, and guides the ants to take a move that gave good results in earlier constructions, while the latter provides the preference index among available moves for an ant. After all ants in the colony finished their partial construction, the pheromone concentration associated with each possible route is updated according to *pheromone updating rule* consisting of two components: pheromone evaporation and reinforcement. The procedure of standard ACO is summarized in Fig. 5.

3.4.2 UF for ACO

In ACO, the pheromone concentration updating occurs at the social-operation phase while *state transition* happens at the self-adaptation phase. The competition strategy used in ACO is generational replacement, i.e., old ants die, and new ants are built based on the updated pheromone and preference.

As for the social-cooperation strategy, each ant that find a good solution mark their paths on the construction graph by putting some amount of pheromone on the edges of the path they passed. The ants in the next iteration are attracted by the pheromones accumulated on the paths by the ants in the colony. Thus, each ant in ACO cooperates with and learns from all the ants in the colony. The way that ants collaborate is determined by pheromone concentration updating rule. And the extent to which the history information of colony is utilized is based on the pheromone deposited on the paths in the constructive graph. Thus, we get $\alpha^t = [s_c, s_n, s_w, s_h]$, where, $s_c = \textit{ant colony}$, $s_n = \textit{size of ant colony}$, $s_w = \textit{pheromone concentration updating rule}$, and $s_h = \textit{pheromone deposited}$.

As for the self-adaptation procedure used in ACO, each ant decides where to move on the construction graph so as to build a solution incrementally according to the *state transition rule* consisting of two factors, i.e., the pheromone concentration and preference. Thus, we get $\beta^t = [\textit{state transition rule}]$.

The parameters for competition procedure in ACO is denoted by $\gamma^t = [c_p, c_r, c_e]$, where, $c_p = \textit{fixed}$, $c_r = (\mu, \lambda)$, and $c_e = \textit{best ant}$, which means that ACO works with a colony with constant size $(\mu = \lambda)$, a generational replacement based competition strategy is used, that is, new ant will always replace the old ant, and the best ant found so far is reserved.

3.5 Genetic algorithms

3.5.1 Introduction to GA

Genetic algorithms (GAs) (Holland 1975) draw inspiration from biological systems and have been proved useful in a variety of search and optimization problems. Its development is

BEGIN
Initialize: Randomly initialize a population
Calculate the fitness value of each individual.
While (Stopping conditions are not satisfied)
Select pairs of individuals to reproduce based on the selection strategy.
Create new offspring through crossover and mutation.
Calculate the fitness value of the offspring.
Update the population using updating strategy.
End While
Output the best chromosomes.
END

Fig. 6 The procedure of GAs

based on the survival-of-the-fitness principle, which tries to retain more genetic information from generation to generation. In GAs, it starts with the random initialization of a population of solutions (*chromosomes*), consisting of a set of elements (*genes*) in the search space. Traditionally, solutions are represented by strings of bits, but different encodings (e.g., float or permutation) have been also proposed for special optimization problems. In each iteration, after the fitness of the whole population is evaluated, the individuals are stochastically selected from the last generation of population based on their fitness. Then the set of genetic operators such as crossover and mutation are used to the selected individuals in creating the offspring of the next generation. This generational process is repeated until a termination condition has been reached. The procedure of standard GA is summarized in Fig. 6.

3.5.2 UF for GAs

A GA works with three operators (i.e., selection, crossover, mutation) that are iteratively used in order to make the population evolve. The selection and crossover occur at the social-cooperation phase, while the mutation is applied at the self-adaptation phase. The steady state or generational replacement strategy is used in the competition phase.

As for the social-cooperation strategy, the selection operator decides which individuals act as parents to breed offspring. Popular and well-studied selection methods include roulette wheel selection and tournament selection. The number of parents is usually two. The way that individuals collaborate is determined by the crossover operator which combines the two parents in order to create new offspring individuals. Typical crossover methods include PMX, C1 etc. In each iteration, the GA keeps no history information of population. Thus, we get $\alpha^t = [s_c, s_n, s_w, s_h]$, where, $s_c = \textit{selection strategy}$, $s_n \geq 2$, $s_w = \textit{crossover strategy}$, and $s_h = \textit{no history}$.

As for the self-adaptation procedure used in GAs, each individual passively adapts itself to the environment by the mutation operator which introduces some noise in order to prevent premature convergence. Typical mutation strategy includes stochastic mutation, INV, INSERT, SWAP. Thus, we get $\beta^t = [\textit{mutation strategy}]$.

The parameters for competition procedure in GAs are denoted by $\gamma^t = [c_p, c_r, c_e]$, where $c_p = \textit{fixed}$, $c_r = \textit{replacement strategy}$, and $c_e = \textit{elitist strategy}$, which means that GAs work with a population of constant size, and the replacement strategy mainly includes generational and steady state replacement. Usually the best one with the highest fitness will survive, that is called elitist strategy.

BEGIN
Initialize: Initialize population and evaluate all the individuals.
While (Stopping conditions are not satisfied**)**
Perform mutation to generate new solutions and evaluate them.
Perform selection to form new population.
End While
Output the best one.
END

Fig. 7 The procedure of EP

3.6 Evolutionary programming

3.6.1 Introduction to EP

Evolutionary Programming (EP) (Fogel et al. 1966; Chellapilla 1998; Yao et al. 1999) which is inspired by the principles of biological evolution, are used for both functional and combinatorial optimization, although it was originally proposed to evolving finite state automata for machine learning. In EP, each parent breeds one offspring by mutation, $(\mu + \mu)$ replacement strategy is used, and crossover is not used. The procedure of standard EP is summarized in Fig. 7.

3.6.2 UF for EP

In EP, there is no social-cooperation. Thus, we get $\alpha^t = [s_c, s_n, s_w, s_h]$, where, $s_c = N/A, s_n = N/A, s_w = N/A$, and $s_h = N/A$. As for the self-adaptation procedure used in EP, each individual adapts itself to the environment by the mutation operator (e.g., Gaussian mutation in the real-valued genotypes). Thus, we get $\beta^t = [\textit{mutation strategy}]$. The parameters for competition procedure in EP is denoted by $\gamma^t = [c_p, c_r, c_e]$, where $c_p = \textit{fixed}, c_r = (\mu + \mu)$, and $c_e = \textit{elitist strategy}$, which means that EP work with a population of constant size, $(\mu + \mu)$ replacement strategy is used. Usually the best one with the highest fitness will survive, that is called elitist strategy.

3.7 Evolution strategies

3.7.1 Introduction to ES

Evolution strategies (ESs) (Ingo Rechenberg 1994; Schwefel 1995; Beyer and Schwefel 2002) which is inspired by the ideas of adaptation and evolution, are typically applied to functional optimization. In ESs, each individual coded in real-valued vector goes through mutation, recombination, and selection. Mutation is performed by adding a Gaussian distributed random value simultaneously to each vector element. The characteristic feature of ESs lies in the self-adaptation of the standard deviation of the Gaussian distribution used in the mutation. Two basic recombination schemes in ESs are local recombination and global recombination. In the former, two parents are involved to create one child, while in the latter more than two individuals contribute to the offspring.

3.7.2 UF for ESs

As for the social-cooperation strategy, individuals are recombined based on the recombination schemes, such as local or global recombination. The parent selection in ESs is not biased by fitness, while it is drawn randomly with uniform distribution from the population. And the extent to which the history information of population is utilized is based on the self-adaptation of the standard deviation of the Gaussian distribution used in the mutation. Thus, we get $\alpha^t = [s_c, s_n, s_w, s_h]$, where, $s_c = \textit{uniform random}, s_n \geq 2, s_w = \textit{recombination strategy}$, and $s_h = \textit{self-adaptation of the standard deviation}$.

As for the self-adaptation procedure used in ESs, the individual is mutated by adding a Gaussian distributed random value. And at the same time, the standard deviation of the Gaussian distribution is also self-adapted. Thus, we get $\beta^t = [\textit{Gaussian mutation strategy}]$. The parameters for competition procedure in ESs is denoted by $\gamma^t = [c_p, c_r, c_e]$, where $c_p = \textit{fixed}, c_r = \textit{replacement strategy}$, and $c_e = \textit{elitist strategy}$, which means that ESs work with a population of constant size, the replacement strategy mainly includes generational (μ, λ) and steady state $(\mu + \mu)$ replacement. Usually the best one with the highest fitness will survive, that is called elitist strategy.

4 Design issues for the UF-based PBMH

In the previous section, we show that various well-known meta-heuristics can be regarded as instances of the UF by specifying the parameters of UF. In this section, we will show how the UF acts as a platform to develop new variants of PBMH. We now revisit the important design issues in the UF. Some of the principal design issues are as follows:

- How can we make decisions to select which individual(s) to cooperate with or learn from for one individual?
- How many individuals are involved in the social-cooperation?
- How do the individuals collaborate to create new individual(s)?
- What is the extent to which the history information of population is utilized?
- How to make a balance between local exploitation and global exploration by the self-adaptation in the optimization process?
- Which individuals in the population are to be improved by the self-adaptation and how do we choose among them?
- What kind of self-adaptation strategy should be used?
- What kind of population competition strategy is used to select the population surviving for the next new generation?
- How to assign the CPU cost among the social-cooperation, self-adaptation, and the competition procedures?

The first four issues are related to $\alpha^t = [s_c, s_n, s_w, s_h]$ in social-cooperation; the next three issues are related to β^t in self-adaptation; the next one is to $\gamma^t = [c_p, c_r, c_e]$ in competition; and the last one is to the overall framework.

4.1 Choice of strategy in social-cooperation

The social-cooperation represents the collaborative effect of the individuals in finding the global optimal by exchanging information among individuals and learning from each other. The way that the social-cooperation works influences the performance of the optimization

method. Choosing what kind of social-cooperation strategy is problem-specific. Furthermore, time-variant social-cooperation strategies may yield significant improvement of the algorithms.

As for the choice of strategy in social-cooperation, many selecting strategies have been used. For instance, in PSO there are two basic versions of PSO algorithms: global and local versions (Kennedy et al. 2001). In the global version each particle keeps track of the overall best particle obtained so far by any particle in the swarm, while in the local version each particle keeps track of the best particle obtained within a local topological neighborhood of particles. The experiment results based on the well-known functional benchmark problems have shown that the global PSO has a better ability in exploring the search space, whereas the local one better in exploiting the search space. In DE, the individual cooperates with the best individual in the current population and two or more randomly selected individuals, in SS particular subset generation and solution combination methods are applied to the reference set to exchange information, in ACO each ant cooperates with and learns from all the ants in the colony according to the pheromone concentration updating rule, in GAs many selection strategies have been proposed (e.g., fitness proportional selection, ranking selection, roulette wheel selection, tournament selection) where usually two parents are combined to breed on offspring, while in EP there is no cooperation strategy is used.

From the above analysis, it could be seen that almost all the strategies are deterministic and determined by a special rule. In order to design an effective and efficient social-cooperation mechanism, from our point of view, it is crucial that the most beneficial information that facilitates seeking the optima should be shared and propagated among the population. For example, the best individual who usually carries the most beneficial information should be used in the collaboration learning. The number of individuals involved in the social-cooperation is problem-dependent. In the situation of multi-model, to avoid to be trapped in local optima, the number of information sources should be increased so that more information could be provided to the individual. For instance, in one version of DE, five individuals including the best one provide information for each individual in the social-cooperation phase. The aim of the individuals' collaboration is to make the better information (e.g., genes in GAs, position information in PSO, and pheromone in ACO) to be effectively inherited by the offspring. The intuitional explanation for this is in that the individual who inherits the good characteristics from its parents could perform better in the next generation. Many collaboration strategies have been presented in the existing EAs. Also the way that the individuals collaborate is solution representation-dependent. For instance, in GA one-point crossover, n-point crossover, and uniform crossover is for binary representation, arithmetic recombination for floating-point representation, and partially mapped crossover for permutation representation. The history information of population is also crucial, whether the past experiences by individuals are good or bad. For instance, in PSO, the global best individual found so far is achieved and updated, which is used in social-cooperation to guide the rest particles towards the high-fitness domain. While in GAs with no elitist strategy, the information of the current population is discarded in the case of the generational replacement strategy.

4.2 Choice of strategy in self-adaptation

The self-adaptation represents each individual's personal active/passive thinking of itself without resorting to the information from other individuals, which evolves the individual independently and encourages it to be adapted to the environment. Typical personal responses to the environment are intensification or diversification searches. The balance between intensification (mainly local exploitation) and diversification (mainly global exploration) is

crucial for the good performance. For instance, there are two basic versions of PSO algorithms: global and local versions (Kennedy et al. 2001). In both of the two versions, each particle learns from its previous best experiences in the self-adaptation process. Many studies have been carried out to prevent premature convergence and to balance the exploration and exploitation abilities. Usually, many variants of PBMH are short of local improvement ability. It has been observed that the PBMH can significantly be improved by using personal improvement strategy during the self-adaptation phase. The use of personal improvement aims to intensify the search in some regions of the search space, which has proven to be successful in many applications.

Usually the majority of PBMH in the literature apply self-adaptation process to every individual in every generation. In order to make the self-adaptation process more efficient, studies may be carried out on the designing the mechanism to determine which individuals are chosen for evolving by self-adaptation and the probability of performing the self-adaptation. Statistics of the population distribution, landscape information of the optimization problems may be helpful to make the decision.

The self-adaptation strategy is also important for the search process. It may be helpful for the individual-adaptation strategy to be capable of changing the individuals' searching behaviors accordingly to the convergence state of the evolutionary search. It is maybe helpful to use multiple self-adaptation strategies simultaneously in the population to avoid getting trapped in local optima.

4.3 Choice of competition strategy

The competition determines how to select the population surviving for the next new generation from the candidate pool mixed by parents and offspring. Usually the majority of PBMH in the literature use a population with constant size, and the competition strategy chooses which individuals will be survived in the next generation. The competition decision is often based on the fitness (or objective) values of the individuals. The higher the quality of the individual is, the higher the chosen probability of the individual is. For instance, in the steady state replacement strategy of GA, the pool of the parents and offspring are ranked based on their fitness, and top individuals are selected for the next generation. This strategy could make the population quickly converge if the selection pressure is too high. Thus it is needed to be further investigated how to maintain a population with good quality and diversity by the competition strategy so that the exploitation and exploration abilities could be well balanced.

4.4 Tradeoff among three components in UF

During each iteration of PBMH variants, periods of social-cooperation, self-adaptation, and competition work consecutively. How to effectively allocate the CPU budget in a dynamic way among the above three components is crucial for the overall optimization performances of PBMH.

5 UFmeme for MAs

In this section, we will generalize the UF for PBMH to UFmeme by adding local search (memetic component) to the framework as an extra-feature which could describe the memetic algorithms (MAs).

INITIALIZE population
While (*TERMINATION CONDITION* is not satisfied) **do**
MEMETIC LEARNING;
S*OCIAL-COOPERATION;*
SELF-ADAPTATION;
COMPETITION;
End

Fig. 8 The general framework of adaptive MAs

5.1 Brief introduction to MAs

Recently, hybrid heuristics have been a hot topic in the fields of both Computer Science and Operational Research. It assumes that combining the features of different methods in a complementary fashion may result in more robust and effective optimization tools. Particularly, it is well known that the performances of PBMH can be improved by combining problem-dependent local searches. Memetic Algorithms (MAs) (Moscato 1989; Hart et al. 2004) may be considered as a union of a population-based global search and local improvements. On one hand, MAs are inspired by Darwinian principles of natural evolution, according to which the evolutionary adaptation of a population is combined with individual learning within the lifetimes of its members. On the other hand, MAs are inspired by Dawkins' notion of a meme defined as a unit of cultural evolution that is capable of local refinements. In the case of MAs, "memes" refer to the strategies (e.g., local refinement, perturbation, or constructive methods, etc.) that are employed to improve individuals. In MAs, several studies (Ishibuchi et al. 2003; Hart et al. 2004) have been focused on how to achieve a reasonable combination of global search and local search, and how to make a good balance between exploration and exploitation. Traditionally, most of the MAs rely on the use of one single PBMH algorithm for globally rough exploration and one single local search for locally fine improvements. Some recent studies (Krasnogor 2002; Ishibuchi et al. 2003; Hart et al. 2004; Ong and Keane 2004; Krasnogor and Smith 2005; Liu et al. 2005, 2010; Ong et al. 2006) on the choice of local searches have shown that the choice significantly affects the searching efficiency.

5.2 UF for MAs (UFmeme)

5.2.1 Basic framework of MAs

In this paper, we mainly consider the adaptive MAs (Ong et al. 2006), which promote both cooperation and competition among various problem-specific memes and favor neighborhood structures containing high quality solutions that may be arrived at low computational efforts. In adaptive MAs, at each generation of evolution, four main components, namely *social-cooperation*, *self-adaptation*, *competition*, and *memetic learning* work consecutively. The former three components are the same with those of UF. *Memetic learning* means the use of multiple memes in the search and the decision on which meme to apply on an individual is made dynamically. In particular, for each individual in the population, a meme is selected from a pool of memes considered in the search to conduct the local improvements. After local improvement, the individuals in the original population are replaced with the improved solution depending on the learning mechanism, i.e., Lamarckian or Baldwinian learning. Based on these main components, the general framework of adaptive MAs is outlined in Fig. 8.

The most important feature of the framework is its generality, by which any a particular adaptive MA can be described. For instance, in order to avoid the negative effects of incorrect local search, Ong and Keane (2004) coined the term "Meta-Lamarckian learning" to introduce the idea of adaptively choosing multiple memes during a MA search in the spirit of Lamarckian learning and successfully solved continuous optimization problems by the proposed MA with multiple local searches. To choose a suitable meme at each decision point, the strategy gathers knowledge about the ability of the memes to search on a particular region of the search space from a database of past experiences archived during the initial search. The memes identified then form the candidate memes that will compete, based on their rewards, to decide on which meme will proceed with the local improvement. In this manner, it was shown that the strategy proposed creates opportunities for joint operations between different memes in solving the problem as a whole, because the diverse memes help to improve the overall population based on their areas of specialization.

In this general framework of adaptive MAs, the memetic learning stage can happen before or after social-cooperation, self-adaptation, and competition, or in any imaginable combination.

5.2.2 UFmeme

In this section, we generalize the UF for PBMH to UFmeme by adding local search (named *memetic learning*) as an extra-feature to describe the memetic algorithms (MAs). The component *memetic learning* means the use of multiple memes in the search and the decision on which meme to apply on an individual is made dynamically. The information needed by the memetic learning procedure includes: individual learning frequency (denoted by m_f), individual subset selection scheme (denoted by m_s), individual learning intensity (denoted by m_i), and individual learning models or meme (denoted by m_c), the individuals in the original population are replaced with the improved solution according to what learning mechanism (denoted by m_l). Thus, the formulization for memetic learning procedure could be represented by the following functional expression:

$$Pop^{t+1} = M(Pop^t, \zeta^t) \tag{7}$$

where $\zeta^t = [m_f, m_s, m_i, m_c, m_l]$ denotes the information needed by the memetic learning procedure which is used to specify a particular adaptive memetic strategy.

According to the previous function expression for the three main components used in PBMH and the above function expression for memetic learning, an overall UF for MAs (UFmeme) is given in the following mathematical way:

$$Pop^{t+1} = M(C(Pop^t, A(S(Pop^t, \alpha^t), \beta^t), \gamma^t), \zeta^t) \tag{8}$$

We have noticed that both the self-adaptation and memetic learning intend to improve the performance of individuals, but they are different in the intelligence used. As a natural ingredient in PBMH, self-adaptation represents each individual's personal active/passive thinking of itself without resorting to the information from other individuals, which does not involve social knowledge/population statistical information. However, memetic algorithms are defined as a combination of a population-based evolution computation and local improvements on selected individuals. Being not a natural ingredient of PBMH, the memetic learning is a contrived addictive to PBMH, and requires an outside intelligence in designing four issues, including individual learning frequency, individual subset selection scheme, individual learning intensity, individual learning models or meme, and learning mechanism. Some of the above five items require current generation/population's information, immediate preceding or neighboring culture evolution, which is not the case for self-adaption.

6 Convergence analysis for the UF-based algorithms

In this section, we analyze the convergence properties of the UF-based PBMH based on the theory of Markov chain and previous efforts on convergence analysis of particular EAs (Eiben et al. 1991; Rudolph 1994, 1996; Suzuki 1995; Ong et al. 2006). Furthermore, we theoretically analyze the asymptotic convergence of MAs based on UFmeme.

6.1 Basic knowledge of finite Markov chain (Häggström 2002)

Definition 1 Let $\mathbf{P}$ be a $k \times k$ transition matrix with transition probabilities $\{p_{i,j} : i, j = 1, \ldots, k\}$. A stochastic process $(X_0, X_1, \ldots)$ with finite state space $S = \{s_1, s_2, \ldots, s_k\}$ is defined to be a homogeneous Markov chain with transition matrix $\mathbf{P}$, if for all n, all $i, j \in \{1, 2, \ldots, k\}$ and all $i_0, i_1, \ldots, i_{n-1} \in \{1, \ldots, k\}$ we have

$$\begin{aligned} &\mathrm{P}(X_{n+1} = s_j | X_0 = s_{i_0}, X_1 = s_{i_1}, \ldots, X_{n-1} = s_{i_{n-1}}, X_n = s_i) \\ &\quad = \mathrm{P}(X_{n+1} = s_j | X_n = s_i) \\ &\quad = p_{i,j} \end{aligned} \tag{9}$$

Definition 2 A Markov chain $(X_0, X_1, \ldots)$ with state space $S = \{s_1, s_2, \ldots, s_k\}$ and transition matrix $\mathbf{P}$ is said to be irreducible if for all $s_i, s_j \in S$ we have

$$\mathrm{P}(X_{m+n} = s_j | X_m = s_i) > 0 \tag{10}$$

Furthermore, if the Markov chain $(X_0, X_1, \ldots)$ is homogeneous, then

$$\mathrm{P}(X_{m+n} = s_j | X_m = s_i) = (\mathbf{P}^n)_{i,j} > 0 \tag{11}$$

Definition 3 A Markov chain is said to be aperiodic if for all its states $s_i \in S$, we have

$$d(s_i) = \gcd\{n \geq 1 : (\mathbf{P}^n)_{i,i} > 0\} = 1 \tag{12}$$

where $\gcd\{a_1, a_2, \ldots\}$ represents the greatest common division of $a_1, a_2, \ldots$. The period $d(s_i)$ of a state $s_i \in S$ is the greatest common divisor of the set of times that the chains can return (i.e., has positive probability of returning) to s_i.

Theorem 1 *Let transition matrix* $\mathbf{P}$ *be a reducible stochastic matrix which could be transformed to* $\left(\begin{smallmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{R} & \mathbf{T} \end{smallmatrix}\right)$ *by applying the same permutations to rows and columns, where* $\mathbf{C}$ *is a* $m \times m$ *primitive stochastic matrix and* $\mathbf{R}, \mathbf{T} \neq \mathbf{0}$. *Then*

$$\mathbf{P}^\infty = \lim_{n\to\infty} \mathbf{P}^n = \lim_{n\to\infty} \begin{pmatrix} \mathbf{C}^n & \mathbf{0} \\ \sum_{i=0}^{n-1} \mathbf{T}^i \mathbf{R} \mathbf{C}^{n-i} & \mathrm{T}^n \end{pmatrix} = \begin{pmatrix} \mathbf{C}^\infty & \mathbf{0} \\ \mathbf{R}^\infty & \mathbf{0} \end{pmatrix} \tag{13}$$

is a stable stochastic matrix with $\mathbf{P}^\infty = 1' p^\infty$, *where* $p^\infty = p^0 \mathbf{P}^\infty$ *is unique regardless of the initial distribution, and* p^∞ *satisfies*: $p_i^\infty > 0$ *for* $1 \leq i \leq m$ *and* $p_i^\infty = 0$ *for* $m \leq i \leq k$.

6.2 Markov chain analysis of the UF based PBMH

In this study, we suppose that the search spaces of the optimization procedures are finite. The state number of population is finite (k), and the probability that the population will be in any given state Pop^{t+1} in the next generation is only determined by its last state Pop^{t}, regardless of the previous states.

First, we provide some explanations on why self-adaptation procedure can be modeled as stochastic processes. The reason that the self-adaption component is modeled as stochastic process lies in that based on analysis of the self-adaption scheme of a variety of PBMH, we observed self-adaption operates on individuals in a nondeterministic way. For instance, as for the self-adaption procedure, in PSO, as a kind of random perturbation, an updated velocity which is computed by multiplication between random coefficient and distance difference, is added to a position for each particle; in DE, the self-adaption operates independently on each element of the individual vector by probabilistically perturbing each element; in ACO, each ant decides where to move on the construction graph so as to build a solution incrementally according to the probabilistic state transition rule consisting of two factors, namely, pheromone concentration and preference; in binary GA, mutation operates independently on each element of individual vector by probabilistically flipping each bit; and in real-valued EP, each individual adapts itself to the environment by Gaussian mutation.

Rather than following the strategy of relegating decisions to random choices, as often is popular in the above mentioned PBMH, SS/PR puts an emphasis on relying on systematic rules (Martí et al. 2006). However, the subroutines in SS/PR show some randomness and have stochastic elements within them. The improvement procedure of SS/PR is to transform trial solutions created in solution combination procedure into one or more enhanced trial solutions (Glover 1998). SS/PR is open and flexible for improvement procedure, which is considered as self-adaption component in UF can be stochastic. For example, in the work of Laguna et al. (1999), neighborhood structures based on add, drop, and node swap moves were used in improvement procedure. The improvement phase was iterated by randomly selecting node(s) in graph with probability being proportional to its weight to carry out exploration of neighborhoods. Besides, it was stated that the neighborhood in improvement procedure is based on the collection of move operation (e.g., "flip" moves for 0–1 variables, insert or swap moves for permutation). And a random selection of such a collection is possible, in the interest of simplification (Glover 1998).

PR generates new solutions by exploring trajectories that connect high quality solutions, by starting from initial solutions, and generating a path in the neighborhood space that leads toward the guiding solutions. Both the initial and guiding solutions are chosen from the reference set. Sometimes, three crucial elements in PR could be stochastic, e.g., selection of initial solutions, selection of guiding solutions, and decision on successive moves to generate path between and initial and guiding solutions. In the research of Ghamlouche et al. (2002), guiding and initial solutions are chosen randomly from the reference set. Another example demonstrated that only guiding solutions are randomly chosen from the reference set (Glover et al. 2003). Regarding to the moves which construct path between and initial and guiding solutions in the neighborhood space, PR generates different paths guiding initial solutions to finial guiding solutions. Thus, decision has to be made to choose among the different paths, and a random selection of such a collection of moves is possible (Martí et al. 2006).

However, we have to admit that typical improvement procedures in SS/PR could be deterministic. For instance, based on graph model of flow shop scheduling problem, Nowicki (1999) developed a local search strategy based on generalization of the block elimination properties with the insertion neighborhood structure, and the moves are deterministic. Furthermore, the authors of this paper incorporated block based local search into PSO for flow

shop scheduling with limited buffer (Liu et al. 2008), and the moves in the neighborhood are deterministic.

Based on the above literature analysis on PBMH including SS/PR, it seems that improvement procedure in SS/PR has twofold characteristics, one is stochastic and another is deterministic. Considering the two facts that usually the deterministic rules could be regarded as a special case of stochastic ones, and that SS/PR is quiet flexible and open in the improvement procedure, we think the deterministic self-adaption procedure is a special form of the general procedure with randomness. We carefully conclude that systematic improvement procedure (i.e., self-adaption component) in SS/PR could be considered as a kind of systematic improvement with randomness.

Thus, the search process of any variants of PBMH characterized with randomness could be treated as finite Markov chains $\{Pop^t, t \geq 0\}$, with a finite state space $S = \{s_1, s_2, \ldots, s_k\}$. The states of finite Markov chains represent the different possible population.

The probabilistic changes in the population of a PBMH which is caused by the three basic evolving operators (i.e., social-cooperation, self-adaptation, and competition) may be modeled using stochastic matrices $\mathbf{P}_S, \mathbf{P}_A, \mathbf{P}_C$ respectively. Consequently, the process of the UF-based PBMH could be modeled as the following $k \times k$ transition matrix $\mathbf{P}$

$$\mathbf{P} = \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A) \tag{14}$$

To simplify the proof, in this study we suppose that the transition matrices $\mathbf{P}_S, \mathbf{P}_A, \mathbf{P}_C$ are independent of time, which lead to a homogeneous Markov chain. While, in some situations (e.g., the parameters α, β, and γ in unified parameters is time-variant), inhomogeneous Markov chains may be appropriate.

Theorem 2 *The Markov chain of UF-based PBMH is homogeneous.*

Proof For arbitrary states $i, j \in k$, the transition probability $\mathbf{P} = (p_{i,j})$ is independent from time. That is,

$$\mathbf{P}_{i,j}(m, m+n) = \mathrm{P}\{Pop^{m+n} = j | Pop^m = i\} \tag{15}$$

Namely, the Markov chain of UF-based PBMH is homogeneous. The transition probability could be written as follows:

$$\mathbf{P}_{i,j}(m, m+n) = (\mathbf{P}^n)_{i,j} \tag{16}$$

□

Theorem 3 *The Markov chain of UF-based PBMH is irreducible and aperiodic.*

Proof Through the social-cooperation procedure, each state of S could be probabilistically transferred to another state in the S. Thus, $\mathbf{P}_S$ is stochastic. The same holds for the self-adaptation and competition and their related transition matrices $\mathbf{P}_A$ and $\mathbf{P}_C$. Due to the fact that each individual in the population could probabilistically change from state i to state j through the self-adaptation procedure, $\mathbf{P}_A$ is positive. $\mathbf{P}_C$ denotes the state transition matrix of each individual that undergoes competition procedure, which indicates that $\mathbf{P}_C$ has at least one positive entry in each row.

Since $\mathbf{P}_S$ is stochastic, there exists at least one positive entry in each row of $\mathbf{P}_S$. Thus, $\mathbf{P}_A$ is positive. Therefore, by matrix multiplication $\mathbf{P}_S\mathbf{P}_A$ is positive. Since $\mathbf{P}_C$ has at least one positive entry in each row and $\mathbf{P}_S\mathbf{P}_A$ is positive, $\mathbf{P} = \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)$ is also strictly positive. Hence it is irreducible and aperiodic. □

Theorem 4 *There are only positive recurrent states in the Markov chain of UF-based PBMH.*

Proof The whole state space of UF-based PBMH is a closed set because its Markov chain is irreducible. Therefore, the Markov chain is composed of only positive recurrent states. □

Theorem 5 *The Markov chain of UF-based PBMH is ergodic.*

Proof According to definition of ergodicity, if the state of Markov chain is irreducible, aperiodic, and has positive recurrent, then state j is ergodic.

According to the definition of ergodicity, the initial distribution p^0 does not influence the limit behavior of the Markov chain. Therefore, the initialization of the algorithm can be done arbitrarily. □

Theorem 6 *The UF-based PBMH does not converge to the global optima with probability* 1.

Proof Let $Pop^t = \{Individual_1^t, \ldots, Individual_N^t\}$ be the population at the tth generation, $Z^t = \max\{f(Individual_i^t), i = 1, 2, \ldots, N\}$ be the best fitness of Pop^t, and $S_{\text{opt}} \in S$, S_{opt} is the set of global optima. From Theorems 4 and 5 we know that the probability that UF-based PBMH can transit to any state in state space, and the UF-based PBMH is in state i converges to $\lim_{t\to\infty} p_i^t = p_i^\infty > 0$. Consequently,

$$\lim_{t\to\infty} \mathrm{P}\{Z^t = S_{\text{opt}}\} \leq 1 - p_i^\infty < 1 \tag{17}$$

So, the UF-based PBMH does not converge to the global optima with probability 1. □

Theorem 7 *The UF-based PBMH with the elitist strategy has the property of global asymptotic convergence as time tends to infinity.*

Proof Due to the elitist strategy, the Markov chain should be adapted by placing the elitist individual at the leftmost position of the current population, which does not involved in the evolutionary process. The transition probabilities of those states containing the same elitist individual are assumed to be listed one below the other in the transition matrix, and the better the elitist individual's fitness the higher the position of the corresponding state in the matrix.

Since the elitist individual is not affected by the operation, the extended transition matrices for social-cooperation $\mathbf{P}_S^+$, self-adaptation $\mathbf{P}_A^+$, and competition $\mathbf{P}_C^+$ can be written as block diagonal matrices as follows: $\mathbf{P}_S^+ = diag(\mathbf{P}_S, \mathbf{P}_S, \ldots, \mathbf{P}_S)$, $\mathbf{P}_A^+ = diag(\mathbf{P}_A, \mathbf{P}_A, \ldots, \mathbf{P}_A)$, and $\mathbf{P}_C^+ = diag(\mathbf{P}_C, \mathbf{P}_C, \ldots, \mathbf{P}_C)$.

After the competition procedure, an upgrading operation is needed to compare the best individual in the current population with the preserved elitist individual in the previous population, and update the elitist individual if possible. This procedure is represented by $\mathbf{P}_U$. Then, the transition probabilities from $Pop^t = [Z^{t-1}, Individual_1^t, Individual_2^t, \ldots, Individual_N^t]$ to $Pop^{t+1} = [Z^t, Individual_1^{t+1}, Individual_2^{t+1}, \ldots, Individual_N^{t+1}]$ are

$$\mathrm{P}(Pop^{t+1} = s_j | Pop^t = s_i) = \begin{cases} 1 & \text{if } \max\{f(Individual_i^t), i = 1, 2, \ldots N\} = f(Z^t) \text{ and} \\ & (Individual_1^t, \ldots, Individual_N^t) = (Individual_1^{t+1}, \ldots, \\ & Individual_N^{t+1}) \\ 0 & \text{else} \end{cases} \tag{18}$$

Thus, there is exactly one entry in each row, which is 1. Considering the fact that a state either becomes upgraded or remains unaltered, therefore, the structure of the upgrading procedure can be written as the following lower triangular matrix

$$\mathbf{P}_U = \begin{bmatrix} \mathbf{P}_{U1,1} & & & \\ \mathbf{P}_{U2,1} & \mathbf{P}_{U2,2} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{P}_{Uk,1} & \mathbf{P}_{Uk,2} & \cdots & \mathbf{P}_{Uk,k} \end{bmatrix} \tag{19}$$

where the size of sub-matrices $\mathbf{P}_{Ui,j}$ is $k \times k$, and $\mathbf{P}_{U1,1}$ is a unit matrix. Thus, the transition matrix for the unified formulized representation is

$$\begin{aligned} \mathbf{P}^+ &= \mathbf{P}_C^+(\mathbf{P}_S^+\mathbf{P}_A^+)\mathbf{P}_U = diag(\mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A), \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A), \ldots, \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A))\mathbf{P}_U \\ &= \begin{bmatrix} \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{U1,1} & & & \\ \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{U2,1} & \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{U2,2} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{Uk,1} & \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{Uk,2} & \cdots & \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{Uk,k} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{R} & \mathbf{T} \end{bmatrix} \end{aligned} \tag{20}$$

Obviously,

$$\begin{aligned} &\mathbf{C} = \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{U1,1} = \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A) > 0, \\ &\mathbf{R} = (\mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{U2,1}, \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{U3,1}, \ldots, \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{Uk,1}) \neq \mathbf{0} \end{aligned}$$

and

$$\mathbf{T} = \begin{bmatrix} \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{U2,2} & & \\ \vdots & \ddots & \\ \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{Uk,2} & \cdots & \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{Uk,k} \end{bmatrix} \neq \mathbf{0}$$

$\mathbf{C} = \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A)\mathbf{P}_{U1,1} = \mathbf{P}_C(\mathbf{P}_S\mathbf{P}_A) > 0$ is a stable stochastic matrix which gathers the transition probabilities for states containing globally optimal states. Since $\mathbf{C}$ is a primitive stochastic matrix and $\mathbf{R} \neq \mathbf{0}$, according to Theorem 1, $\mathbf{P}^{+\infty}$ is given by

$$\mathbf{P}^{+\infty} = \lim_{n\to\infty} \mathbf{P}^{+n} = \lim_{n\to\infty} \begin{pmatrix} \mathbf{C}^n & \mathbf{0} \\ \sum_{i=0}^{n-1} \mathbf{T}^i\mathbf{R}\mathbf{C}^{n-i} & \mathbf{T}^n \end{pmatrix} = \begin{pmatrix} \mathbf{C}^\infty & \mathbf{0} \\ \mathbf{R}^\infty & \mathbf{0} \end{pmatrix} \tag{21}$$

which guarantees that the probability of staying in any non-globally optimal state converges to zero. It follows that the probability of being in any globally optimal state converges to one, so that

$$\lim_{t\to\infty} \mathrm{P}(Z^t \in S_{\mathrm{opt}}) = 1 \tag{22}$$

This implies that the UF-based PBMH converges to the global optimum with probability 1 as time tends to infinity. □

6.3 Brief Markov chain analysis of the UFmeme based MAs

In this section, we analyze the global convergence properties of MAs based on the previous discuss on UF-based PBMH, the proposed UFmeme, and previous efforts on convergence analysis of GA-based MAs by Ong et al. (2006). The probabilistic changes in the adaptive MA population due to the operators (i.e., social-cooperation, self-adaptation, and competition) have been modeled in the previous section by the transition matrices $\mathbf{P}_S, \mathbf{P}_A, \mathbf{P}_C$ respectively. Besides the standard operation, the adaptive MA will also refine each individual using different memes at each decision point. We model this process of *memetic learning* as a transition matrix $\mathbf{P}_M$, which represents the state transition matrix of each individual that undergoes memetic learning process. Consequently, the process of the UFmeme could be modeled as the following transition matrix $\mathbf{P}_{MAs}$

$$\mathbf{P}_{MAs} = \mathbf{P}_M \mathbf{P}_C (\mathbf{P}_S \mathbf{P}_A) \tag{23}$$

To simplify the proof, we suppose that the transition matrices $\mathbf{P}_M$ is independent of time, which lead $\mathbf{P}_{MAs}$ to be a homogeneous Markov chain. This is true when the adaptive MAs are global-level (Ong et al. 2006). While, for the local-level adaptive MAs in which the choice of meme is made based on the immediate preceding or neighboring culture evolution, $\mathbf{P}_M$ is dependent of time and varies for each decision point, the inhomogeneous Markov chains may be appropriate (Ong et al. 2006).

Theorem 8 *The Markov chain of UFmeme based global-level adaptive MAs is irreducible and aperiodic.*

Proof As show in Theorem 3, $\mathbf{P} = \mathbf{P}_C(\mathbf{P}_S \mathbf{P}_A)$ is strictly positive. Since $\mathbf{P}_M$ has at least one positive entry in each row and $\mathbf{P} = \mathbf{P}_C(\mathbf{P}_S \mathbf{P}_A)$ is positive, $\mathbf{P}_{MAs} = \mathbf{P}_M \mathbf{P} = \mathbf{P}_M \mathbf{P}_C(\mathbf{P}_S \mathbf{P}_A)$ is also strictly positive. Hence it is irreducible and aperiodic. □

Theorem 9 *There are only positive recurrent states in the Markov chain of UFmeme based global-level adaptive MAs.*

Proof The proof is similar to Theorem 4. □

Theorem 10 *The Markov chain of UFmeme based global-level adaptive MAs is ergodic.*

Proof The proof is similar to Theorem 5. □

Theorem 11 *The Markov chain of UFmeme based global-level adaptive MAs has the property of global asymptotic convergence as time tends to infinity.*

Proof Considering the fact that in the global-level adaptive MAs the probability that the most suitable meme is selected is one as time tends to infinity (Ong et al. 2006), then we could obtain that a state either becomes upgraded or remains unaltered. Therefore, the structure of the memetic learning procedure can be written as a lower triangular matrix. Further, from (20), $\mathbf{P}^+ = \mathbf{P}_C^+(\mathbf{P}_S^+ \mathbf{P}_A^+)\mathbf{P}_U$ is a lower triangular matrix. Thus, the transition matrix $\mathbf{P}_{MAs} = \mathbf{P}_M \mathbf{P}^+$ is also a lower triangular matrix. According to Theorem 1, $\mathbf{P}_{MAs}$ is a reducible stochastic matrix, and the probability of being in any globally optimal state converges to one. This implies that the Markov chain of UFmeme based global-level adaptive MAs converges to the global optimum with probability 1 as time tends to infinity. □

7 Conclusions

In this paper, the main components used in PBMH were described with functional relationships, and a unified framework was proposed for PBMH. With the proposed UF, it provided a unifying way to describe PBMH, and some typical PBMH could be explained as instances of the UF. Then, we discussed the design issues of developing effective/efficient algorithms based on the UF, and we also generalized the UF to UFmeme to describe the memetic algorithms (MAs) by adding local search (memetic component) as an extra-feature. Moreover, we theoretically analyzed the asymptotic convergence properties of the PBMH described by the UF and MAs by UFmeme. In brief, this work provides a unified view of various kinds of PBMH techniques, which helps understand the essential philosophy of PBMH from a systematical standpoint and presents a platform to develop novel population-based algorithms.

Acknowledgements The authors wish to thank the anonymous referee and Editor-in-Chief Prof. Endre Boros for their constructive comments on earlier drafts of this paper. The first author thanks Emeritus Prof. Yihui Jin (Department of Automation, Tsinghua University), Prof. JiKun Huang (Center for Chinese Agricultural Policy, Chinese Academy of Sciences), Prof. M.A. Keyzer (Faculty of Economics and Business Administration, SOW-VU, Vrije Universiteit Amsterdam, The Netherlands) for their support. This research is partially supported by National Science Foundation of China (70871065, 60834004), NCET-2010-0505, Doctoral Program Foundation of Institutions of Higher Education of China (20100002110014), and the Project Sponsored by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry.

References

Bäck, T. (1996). *Evolutionary algorithms in theory and practice*. London: Oxford University Press.

Beyer, H.-G., & Schwefel, H.-P. (2002). Evolution strategies: a comprehensive introduction. *Natural Computing*, *1*, 3–52.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*. London: Oxford University Press.

Bonissone, P. P., Subbu, R., Eklund, N., & Kiehl, T. R. (2006). Evolutionary algorithms + domain knowledge = real-world evolutionary computation. *IEEE Transactions on Evolutionary Computation*, *10*, 256–280.

Calégari, P., Coray, G., Hertz, A., Kobler, D., & Kuonen, P. (1999). A taxonomy of evolutionary algorithms in combinatorial optimization. *Journal of Heuristics*, *5*, 145–158.

Cao, Y. J., & Wu, Q. H. (1997). Convergence analysis of adaptive genetic algorithm. In *Genetic algorithms in engineering systems conf.: innovations and applications*, 1997 (pp. 85–89).

Chellapilla, K. (1998). Combining mutation operators in evolutionary programming. *IEEE Transactions on Evolutionary Computation*, *2*, 91–96.

Clerc, M., & Kennedy, J. (2002). The particle swarm: explosion stability and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, *6*, 58–73.

Coello Coello, C. A. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, *191*, 1245–1287.

Coello Coello, C. A. (2006). Evolutionary multi-objective optimization: a historical view of the field. *IEEE Computational Intelligence Magazine*, *1*, 28–36.

Dasgupta, D. (1998). *Artificial immune systems and their applications*. Berlin: Springer.

Davis, T. E. (1991). *Toward an extrapolation of the simulated annealing convergence theory onto the simple genetic algorithm*. Ph.D. dissertation, Univ. Florida, Gainesville.

De Jong, K. A. (2006). *Evolutionary computation: a unified approach*. Cambridge: MIT Press.

Dimopoulos, C., & Zalzala, A. M. S. (2000). Recent development in evolutionary computation for manufacturing optimization: problems, solutions, and comparisons. *IEEE Transactions on Evolutionary Computation*, *4*, 93–113.

Dorigo, M., & Gambardella, L. M. (2002). Special section on ant colony optimization. *IEEE Transactions on Evolutionary Computation*, *6*, 317–319.

Eberbach, E. (2005). Toward a theory of evolutionary computation. *Biosystems*, *82*, 1–19.

Eiben, A., & Smith, J. E. (2003). *Introduction to evolutionary computing*. Heidelberg: Springer.
Eiben, A. E., Aarts, E. H. L., & van Hee, K. M. (1991). Global convergence of genetic algorithms: a Markov chain analysis. In *Proc. 1st int. conf. parallel problem solving from nature*, 1991 (pp. 4–12).
Eiben, A., Aarts, E., van Hee, K., & Nuijten, W. (1995). A unifying approach on heuristic search. *Annals of Operation Research*, *55*, 81–99.
Fogel, D. B. (1992). *Evolving artificial intelligence*. Ph.D. dissertation, San Diego: Univ. of California.
Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966). *Artificial intelligence through simulated evolution*. Chichester: Wiley.
Fonseca, C. M., & Fleming, P. J. (1995). An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, *3*, 1–16.
François, O. (1998). An evolutionary strategy for global minimization and its Markov chain analysis. *IEEE Transactions on Evolutionary Computation*, *2*, 77–90.
Ghamlouche, I., Crainic, T. G., & Gendreau, M. (2002). *Path relinking, cycle-based neighbourhoods and capacitated multicommodity network design*. Publication CRT-2002-01, Centre de recherche sur les transports, Université de Montréal.
Glover, F. (1977). Heuristics for integer programming using surrogate constraints. *Decision Sciences*, *8*, 156–166.
Glover, F. (1998). A template for scatter search and path relinking. In *Lecture notes in computer science*: *Vol. 1363* (pp. 13–54). Berlin: Springer.
Glover, F., & Kochenberger, G. (2003). *Handbook of metaheuristics*. Boston: Kluwer Academic.
Glover, F., Laguna, M., & Martí, R. (2003). Scatter search and path relinking: advances and applications. In *Handbook of metaheuristics* (pp. 1–35).
Grabowski, J., & Pempera, J. (2001). New block properties for the permutation flow shop problem with application in tabu search. *The Journal of the Operational Research Society*, *52*, 210–220.
Grabowski, J., & Pempera, J. (2007). The permutation flow shop problem with blocking: a tabu search approach. *Omega*, *35*, 302–311.
Gravel, M., Price, W. L., & Gagne, C. (2002). Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic. *European Journal of Operational Research*, *143*, 218–229.
Häggström, O. (2002). *Finite Markov chains and algorithmic applications*. New York: Cambridge University Press.
Hansen, P., & Mladenovic, N. (2001). Variable neighborhood search: principles and applications. *European Journal of Operational Research*, *130*, 449–467.
Hart, E., Ross, P., & Corne, D. (2005). Evolutionary scheduling: a review. *Genetic Programming and Evolvable Machines*, *6*, 191–220.
Hart, W. E., Krasnogor, N., & Smith, J. E. (2004). *Recent advances in memetic algorithms*. Heidelberg: Springer.
He, J., & Kang, L. (1999). On the convergence rates of genetic algorithms. *Theoretical Computer Science*, *229*, 23–39.
Hertz, A., & Kobler, D. (2000). A framework for the description of evolutionary algorithms. *European Journal of Operational Research*, *126*, 1–12.
Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: University of Michigan Press.
Hoogeveen, H. (2005). Multicriteria scheduling. *European Journal of Operational Research*, *167*, 592–623.
Ingo Rechenberg (1994). *Evolutionsstrategie '94*. Stuttgart: Frommann-Holzboog.
Ishibuchi, H., Misaki, S., & Tanaka, H. (1995). Modified simulated annealing algorithms for the flow shop sequencing problem. *European Journal of Operational Research*, *81*, 388–398.
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*, 204–223.
Jin, Y., & Branke, J. (2005). Evolutionary optimization in uncertain environments—a survey. *IEEE Transactions on Evolutionary Computation*, *9*, 303–317.
Johnson, S. M. (1954). Optimal two- and three- stage production schedules with setup times included. *Naval Research Logistics Quarterly*, *1*, 61–68.
Kennedy, J., Eberhart, R. C., & Shi, Y. (2001). *Swarm intelligence*. San Francisco: Morgan Kaufmann.
Kochenberger, G. A., Glover, F., Alidaee, B., & Rego, C. (2004). A unified modeling and solution framework for combinatorial optimization problems. *OR-Spektrum*, *26*, 237–250.
Koza, J. R. (1992). *Genetic programming*. Cambridge: MIT Press.
Krasnogor, N. (2002). *Studies on the theory and design space of memetic algorithms*. Ph.D. dissertation, Univ. West of England, Bristol, UK.
Krasnogor, N., & Smith, J. (2005). A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *IEEE Transactions on Evolutionary Computation*, *9*, 474–488.

Lageweg, B. J., Lenstra, J. K., & Rinnooy Kan, A. H. G. (1978). A general bounding scheme for the permutation flow-shop problem. *Operations Research*, *26*, 53–67.

Laguna, M., & Martí, R. (2003). *Scatter search: methodology and implementations in C*. Boston: Kluwer Academic.

Laguna, M., Martí, R., & Campos, V. (1999). Intensification and diversification with elite tabu search solutions for the linear ordering problem. *Computers & Operations Research*, *26*, 1217–1230.

Li, B., & Jiang, W. (2000). A novel stochastic optimization algorithm. *IEEE Transactions on Systems, Man, and Cybernetics*, *30*, 193–198.

Li, B. B., Wang, L., & Liu, B. (2008). An effective PSO-based hybrid algorithm for multiobjective permutation flow shop scheduling. *IEEE Transactions on Systems, Man and Cybernetics Part A Systems and Humans*, *38*, 818–831.

Liu, B., Wang, L., Jin, Y. H., Tang, F., & Huang, D. X. (2005). Improved particle swarm optimization combined with chaos. *Chaos, Solitons and Fractals*, *25*, 1261–1271.

Liu, B., Wang, L., & Jin, Y. H. (2007). An effective PSO-based memetic algorithm for flow shop scheduling. *IEEE Transactions on Systems, Man and Cybernetics Part B Cybernetics*, *37*, 18–27.

Liu, B., Wang, L., & Jin, Y. H. (2008). An effective hybrid PSO-based algorithm for flow shop scheduling with limited buffers. *Computers & Operations Research*, *35*, 2791–2806.

Liu, B., Wang, L., Liu, Y., Qian, B., & Jin, Y. H. (2010). An effective hybrid particle swarm optimization for batch scheduling of polypropylene processes. *Computers & Chemical Engineering*, *34*, 518–528.

Liu, J., Zhong, W., & Jiao, L. (2006). A multiagent evolutionary algorithm for constraint satisfaction problems. *IEEE Transactions on Systems, Man and Cybernetics Part B Cybernetics*, *36*, 54–73.

Martí, R. (2006). Scatter search—wellsprings and challenges. *European Journal of Operational Research*, *169*, 351–358.

Martí, R., Laguna, M., & Glover, F. (2006). Principles of scatter search. *European Journal of Operational Research*, *169*, 359–372.

Michalewicz, Z., & Schoenauer, M. (1996). Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, *4*, 1–32.

Moscato, P. (1989). *On evolution, search, optimization, genetic algorithms and martial arts: toward memetic algorithms* (Tech. Rep.). Caltech Concurrent Computation Program, Rep. 826, California Inst. Technol., Pasadena, CA.

Nawaz, M., Enscore, E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, *11*, 91–95.

Nearchou, A. C., & Omirou, S. L. (2006). Differential evolution for sequencing and scheduling optimization. *Journal of Heuristics*, *12*, 395–411.

Nix, A., & Vose, M. D. (1992). Modeling genetic algorithm with Markov chains. *Annals of Mathematics and Artificial Intelligence*, *5*, 27–34.

Nowicki, E. (1999). The permutation flow shop with buffers: a tabu search approach. *European Journal of Operational Research*, *116*, 205–219.

Nowicki, E., & Smutnicki, C. (1996). A fast tabu search algorithm for the permutation flow-shop problem. *European Journal of Operational Research*, *91*, 160–175.

Nowicki, E., & Smutnicki, C. (2005). Some aspects of scatter search in the flow-shop problem. *European Journal of Operational Research*, *169*, 654–666.

Ong, Y. S. (2002). *Artificial intelligence technologies in complex engineering design*. Ph.D. dissertation, Sch. Eng. Sci., Univ. Southampton, Southampton, UK.

Ong, Y. S., & Keane, A. J. (2004). Meta-Lamarckian learning in memetic algorithms. *IEEE Transactions on Evolutionary Computation*, *8*, 99–110.

Ong, Y. S., Lim, M.-H., Zhu, N., & Wong, K.-W. (2006). Classification of adaptive memetic algorithms: a comparative study. *IEEE Transactions on Systems, Man and Cybernetics Part B Cybernetics*, *36*, 141–152.

Onwubolu, G., & Davendra, D. (2006). Scheduling flow shops using differential evolution algorithm. *European Journal of Operational Research*, *171*, 674–692.

Price, K., Storn, R., & Lampinen, J. (2005). *Differential evolution—a practical approach to global optimization*. Berlin: Springer.

Rajendran, C., & Ziegler, H. (2004). Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. *European Journal of Operational Research*, *155*, 426–438.

Reeves, C. R., & Yamada, T. (1998). Genetic algorithms, path relinking and the flowshop sequencing problem. *Evolutionary Computation*, *6*, 45–60.

Reynolds, R. G. (1994). An introduction to cultural algorithms. In *Proc. 3rd annual conference on evolutionary programming*, San Diego, USA (pp. 131–139).

Rosenthal, J. S. (1995). Convergence rates for Markov chains. *SIAM Review*, *37*, 387–405.

Rudolph, G. (1994). Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, *5*, 96–101.

Rudolph, G. (1996). Convergence of evolutionary algorithms in general search spaces. In *Proc. IEEE int. conf. on evolutionary computation*, Nagoya, Japan, 1996 (pp. 50–54).

Rudolph, G. (1997). Local convergence rates of simple evolutionary algorithms with Cauchy mutations. *IEEE Transactions on Evolutionary Computation*, *1*, 249–258.

Rudolph, G. (1998). Finite Markov chains results in evolutionary computation: a Tour d'Horizon. *Fundamenta Informaticae*, *35*, 67–89.

Ruiz, R., & Maroto, C. (2005). A comprehensive review and evaluation of permutation flowshop heuristics. *European Journal of Operational Research*, *165*, 479–494.

Schwefel, H.-P. (1995). *Evolution and optimum seeking*. New York: Wiley.

Sha, D. Y., & Hsu, C. Y. (2006). A hybrid particle swarm optimization for job shop scheduling problem. *Computers & Industrial Engineering*, *51*, 791–808.

Smith, J. (1998). *Self adaptation in evolutionary algorithms*. Ph.D. dissertation, Univ. West of England, England, U.K.

Smutnicki, C., & Tyński, A. (2006). Job-shop scheduling by GA. A new crossover operator. *Operations Research Proceedings*, 715–720.

Suzuki, J. (1995). A Markov chain analysis on simple genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, *25*, 655–659.

Suzuki, J. (1998). A further result on the Markov chain model of genetic algorithms and its application to a simulated annealing-like strategy. *IEEE Transactions on Systems, Man and Cybernetics Part B Cybernetics*, *28*, 95–102.

Taillard, É. D., Gambardella, L. M., Gendreau, M., & Potvin, J.-Y. (2001). Adaptive memory programming: A unified view of metaheuristics. *European Journal of Operational Research*, *135*, 1–16.

Talbi, E. (2002). A taxonomy of hybrid metaheuristics. *Journal of Heuristics*, *8*, 541–564.

Tasgetiren, M. F., Liang, Y. C., Sevkli, M., & Gencyilmaz, G. (2007). A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. *European Journal of Operational Research*, *177*, 1930–1947.

T'kindt, V., Monmarche, N., Tercinet, F., & Laugt, D. (2002). An ant colony optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem. *European Journal of Operational Research*, *142*, 250–257.

Trelea, I. C. (2003). The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters*, *85*, 317–325.

Wang, L. (2003). *Shop scheduling with genetic algorithms*. Beijing: Tsinghua Univ. Press & Springer.

Wang, L., & Liu, B. (2008). *Particle swarm optimization and scheduling algorithms*. Beijing: Tsinghua University Press.

Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation*, *3*, 82–102.