2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Phase Transition Particle Swarm Optimization

Ji Ma, Junqi Zhang†, Wei Wang and Jing Yao
Department of Computer Science and Technology
Key Laboratory of Embedded System and Service Computing, Ministry of Education
Tongji University, Shanghai, 200092, China
† Corresponding author (*E-mail*: zhangjunqi@tongji.edu.cn)

***Abstract*—In nature, a phase transition is the transformation of a thermodynamic system from one phase to another. Different phases of a thermodynamic system have distinctive physical properties. Inspired by this natural phenomenon, this paper presents a Particle Swarm Optimization (PSO) based on the Phase Transitions model which consists of *solid*, *liquid* and *gas* phases. Each phase represents a distinctive behavior of the swarm. Transitions of condensation, solidification and deposition can enhance the exploitation capability of the swarm. While the transitions of fusion, vaporization and sublimation from the other direction improve the exploration capability of the swarm. The proposed model directs the swarm to transform among phases dynamically and automatically according to the evolutional states to balance between exploration and exploitation adaptively. Especially, it uses a new modified PSO algorithm called Simple Fast Particle Swarm Optimization (SFPSO) in the *solid* phase, which modifies the original PSO by adding new parameters simply to make the algorithm convergence more quickly. The proposed algorithm is validated by extensive simulations on the 28 real-parameter optimization benchmark functions from CEC 2013 compared with other three representative variants of PSO.**

I. Introduction

Particle swarm optimization (PSO) is a stochastic global optimization technique inspired by the social behavior of bird flocking or fish schooling [1], [2]. In the classical PSO, each particle in a swarm population adjusts its position in the search space by the best position it has found so far ($\boldsymbol{pbest}$) and also the overall best position found so far by the whole swarm ($\boldsymbol{gbest}$). The PSO algorithm is easy to implement and has been empirically shown to perform well on many optimization problems. However, it may easily get trapped in a local optimum when solving complex multimodal problems.

This paper applies the phase transitions model to PSO and presents a new algorithm, namely Phase Transition Particle Swarm Optimization (PTPSO). In PTPSO, the swarm is endowed with one of three different phases (*solid*, *liquid* and *gas* phase) during the evolution process, and it could transform to another one according to the evolutional states dynamically and automatically.

We organize the paper as follows. Section II reviews related work in PSO and phase transitions. In Section III, we analyze the PSO from the aspect of force attraction and introduce a simple fast PSO for the particles to fly in the *solid* phase. In Section IV , the Phase Transition Particle Swarm Optimization (PTPSO) and its characteristics are described. Section V experimentally validates the PTPSO and compares it with three PSO variants on 28 benchmark functions. Conclusions are drawn in Section VI.

II. Background

A. Particle Swarm Optimization (PSO)

The original PSO algorithm is discovered through simplified social model simulation. It is related to the bird flocking, fishing schooling and swarm theory. The PSO was first designed to simulate birds seeking food which is defined as a cornfield vector. The bird will find food through social cooperation with other birds around it. It was then expanded to multidimensional search. The original PSO algorithm is described as below [1], [2]:

$$v_i^d = v_i^d + c_1 r_1^d (pbest_i^d - x_i^d) + c_2 r_2^d (gbest^d - x_i^d) \quad (1)$$

$$x_i^d = x_i^d + v_i^d \quad (2)$$

where D is the dimension of solution space and $d = 1, 2, .., D$; $\boldsymbol{X_i} = (x_i^1, x_i^2, ..., x_i^D)$ represents the position of the ith particle; $\boldsymbol{V_i} = (v_i^1, v_i^2, ..., v_i^D)$ is velocity of the ith particle; $\boldsymbol{pbest_i} = (pbest_i^1, pbest_i^2, ..., pbest_i^D)$ is the best previous position yielding the best fitness value for the ith particle; and $\boldsymbol{gbest} = (gbest^1, gbest^2, ..., gbest^D)$ is the best position discovered by the whole swarm. c_1 and c_2 are two parameters to weight the relative importance of $\boldsymbol{pbest_i}$ and $\boldsymbol{gbest}$, respectively; r_1^d and r_2^d are two random numbers in the range [0,1].

Since the original PSO was introduced, many researchers have worked on improving its performance in various ways. One of the variants [3] introduces a parameter called inertia weight (w) into the original PSO algorithms as follows:

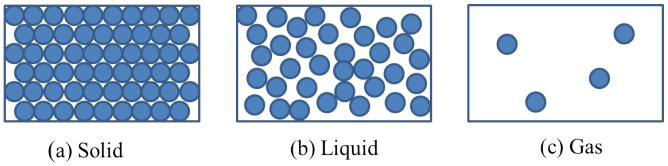
$$v_i^d = w v_i^d + c_1 r_1^d (pbest_i^d - x_i^d) + c_2 r_2^d (gbest^d - x_i^d) \quad (3)$$

the inertia weight is used to balance the global and local search abilities. A large inertia weight facilitates global search, and a small inertia weight is more appropriate for local search. A linearly decreasing inertia weight over the course of search was proposed by Shi and Eberhart [3]. Since then, the inertia weight has been used in almost all PSO variants. To balance the ability of local search and global search, Shi and Eberhart [3] proposed a scheme to decrease w linearly from 0.9 to 0.4 over the course of search process.

Another variant is the comprehensive learning PSO (CLPSO) proposed by Liang *et al*. [4], where particles are able to learn from distinctive $\boldsymbol{pbest}$ positions on different dimensions. The velocity updating as follow:

$$v_i^d = w v_i^d + c \cdot r_i^d (pbest_{f_i(d)}^d - x_i^d) \quad (4)$$

where $\boldsymbol{f_i} = [f_i(1), f_i(2), ..., f_i(D)]$ defines which particles' $pbests$ the particle i should follow. $pbest_{f_i(d)}^d$ can be the



(a) Solid (b) Liquid (c) Gas

Fig. 1. Microscopic view of the three states of matter.

corresponding dimension of any particle's $pbest$ including its own $pbest$, and the decision depends on probability Pc, referred to as the learning probability, which can take different values for different particles.

B. States of Matter

In nature, a state of matter is one of the distinct forms that different phases of matter take on [5]. Three states of matter are observable in everyday life: *solid*, *liquid*, and *gas* phase.

1) solid phase:

- When matter is in a solid phase, it holds its shape. As it is illustrated in Fig.1 (a), although the tiny atom particles that make up the molecules of a solid are in motion all the time, the molecules are not free to go anywhere.
- Unlike a liquid, a solid object does not flow to take on the shape of its container, nor does it expand to fill the entire volume available to it like a gas does.

2) liquid phase:

- Liquid phase is considered to be a transition state. That means a liquid lies between the two extremes of ordered arrangement of solids and disorder in gases. As it is shown in Fig.1 (b), the molecules in a liquid are not as close together as those in a solid but they are not as spread out as the molecules in a gas.
- Liquid phase has some extent of freedom of motion, because it has a definite volume but no definite shape. The intermolecular force attraction is only temporary in liquid phase which allows it to move freely resulting in a limited degree of particle mobility.

3) gas phase:

- Gas phase is usually difficult to see and sometimes called vapor. In a gas, the molecules are very far apart, the microscopic view of gas is given in Fig.1 (c). In fact, gas will fill a container of any shape by spreading out as much as possible.
- The main difference between gas phase and other phases is that gas molecules move around much more. The random movement of constituent particles in gas phase is due to negligible intermolecular force attraction such that intermolecular distance between molecules is very high.

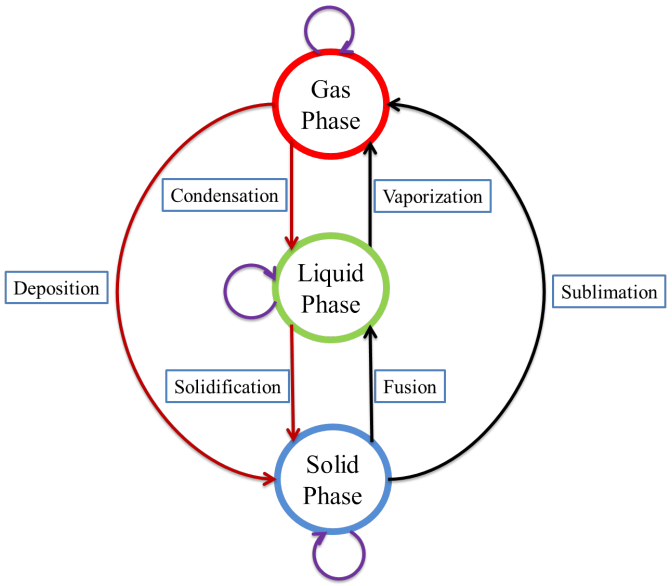


Fig. 2. Phase transitions.

C. Phase transitions

A phase transition is an abrupt, discontinuous change in the properties of a system [6] which is defined as the transformation of a thermodynamic system from one phase to another. There are the key terms in the movement from one phase to another, which is shown in Fig. 2:

- Fusion: the transition by which a solid becomes a liquid.
- Vaporization: the transition by which a liquid becomes a gas.
- Sublimation: an unusual process by which a solid goes directly to the gas phase without turning into a liquid first.
- Solidification: the transition by which a liquid becomes a solid.
- Condensation: the transition by which a gas becomes a liquid.
- Deposition: the transition by which a gas goes directly into the solid phase without becoming a liquid first. This process is the opposite of sublimation.

III. A Simple Fast Particle Swarm Optimization

In this section, we briefly analyze the Particle Swarm Optimization from the perspective of the force attraction and the distance between particle and $gbest$ or $pbest$ in each dimension, and modify the original PSO by adding new parameters simply to make the algorithm convergence more quickly. We called this variant Simple Fast Particle Swarm Optimization (SFPSO), which will be used to derive the PTPSO algorithm in the later section. Since the dynamics of each dimension of particles is in independent of others, we assume that $D = 1$ without loss of generality in the following. When the $pbest_i \neq x_i$ and $gbest \neq x_i$, then equation (3) can be modified as:

$$v_i = wv_i + c_1 r_1 R_i^p \vec{e_i^p} + c_2 r_2 R_i^g \vec{e_i^g} \tag{5}$$

with

$$R_i^p = \mid pbest_i - x_i \mid$$
$$R_i^g = \mid gbest - x_i \mid$$
$$\vec{e_i^p} = \frac{pbest_i - x_i}{\mid pbest_i - x_i \mid}$$
$$\vec{e_i^g} = \frac{gbest - x_i}{\mid gbest - x_i \mid}$$

where R_i^p is the distance between the ith particle and $pbest_i$; R_i^g is the distance between the ith particle and $gbest$; $\vec{e_i^p}$is a unit vector, which points from x_i to $pbest_i$; $\vec{e_i^g}$ is a unit vector, which point from x_i to $gbest$. According Newtons second law [7], we assume that each particle in the swarm has a mass m_i and $m_i = 1$, so we have

$$v_i = wv_i + r_1 \frac{\vec{F_i^p}}{m_i} + r_2 \frac{\vec{F_i^g}}{m_i} \tag{6}$$

with

$$\vec{F_i^p} = c_1 R_i^p \vec{e_i^p}$$
$$\vec{F_i^g} = c_2 R_i^g \vec{e_i^g}$$

where $\vec{F_i^p}$ is considered as the force attraction between x_i and $pbest_i$; $\vec{F_i^g}$ is considered as the force attraction between x_i and $gbest$, Fig. 3 shows the schematic of them. According to equation (6), we can find that it can be linear relation between force attraction ($\vec{F_i^p}$, $\vec{F_i^g}$) and the distance (R_i^g, R_i^g) if without random perturbation, and it turns to be uncertain with random perturbation. But the random area of force attraction is in proportion to the distance due to the range of r_1 and r_2 is [0,1]. Fig.4 (a) illustrates the relationship between the force attraction F and the distance R in original PSO.

Actually, it can influence the convergence rate if the particles which are very far away from $gbest$ and $pbest$ have a very small force attraction. As shown in Fig.4 (a), the point A with a smaller distance than the point B ($R_A < R_B$), but the force attraction of A is larger than B ($F_A > F_B$). In order to keep a high convergence rate, we add two new parameters $k_1, k_2 (0 \le k_1 < c_1, 0 \le k_2 < c_2)$ simply, and the equation (5) can be rewritten as follow:

$$v_i = wv_i + \frac{\vec{F_{1i}^p}}{m_i} + r_1 \frac{\vec{F_{2i}^p}}{m_i} + \frac{\vec{F_{1i}^g}}{m_i} + r_2 \frac{\vec{F_{2i}^g}}{m_i} \tag{7}$$

with

$$\vec{F_{1i}^p} = k_1 R_i^p \vec{e_i^p}$$
$$\vec{F_{1i}^g} = k_2 R_i^g \vec{e_i^g}$$
$$\vec{F_{2i}^p} = (c_1 - k_1) R_i^p \vec{e_i^p}$$
$$\vec{F_{2i}^g} = (c_2 - k_2) R_i^g \vec{e_i^g}$$

As shown in Fig.4 (b), the point A with a smaller distance than the point B ($R_A < R_B$), and the force attraction of

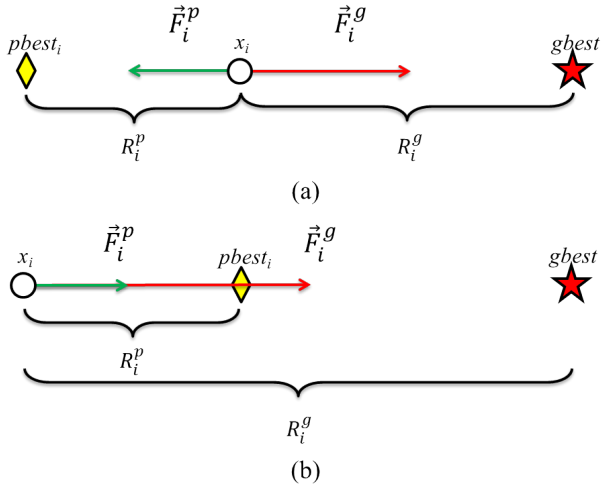


Fig. 3. Two cases of the force attraction in original PSO with $D = 1$: (a) x_i is between $gbest$ and $pbest_i$; (b) x_i is on the side of $gbest$ and $pbest_i$.

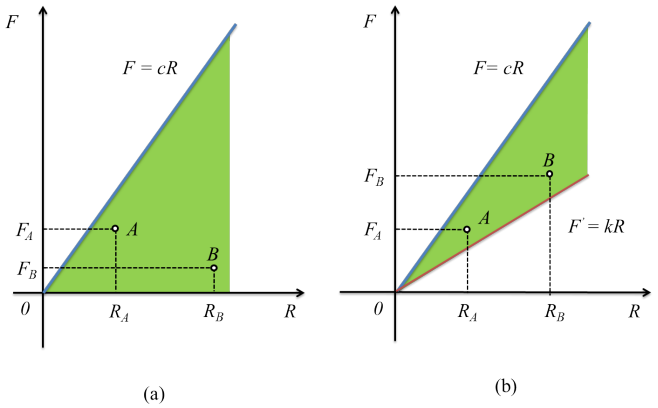
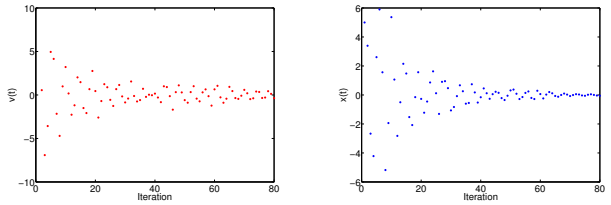


Fig. 4. The comparison of original PSO and PSO with new parameters k: (a) represents the original PSO; (b) represents SFPSO. (R could represent the distance R_i^p or R_i^g, c could represent c_1 or c_2, and k could represent the new parameter k_1 or k_2)

A is smaller than B ($F_A < F_B$). The particles which are very far away from $gbest$ and $pbest$ could have a bigger force attraction, it leads to the swarm convergence fast. According to equation (7), we rewrite the equation (3):

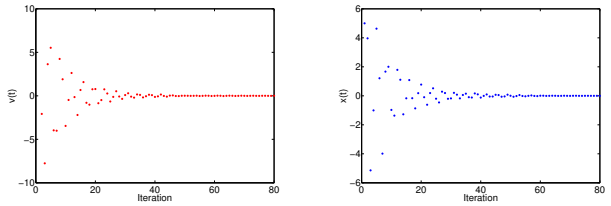
$$\begin{aligned} v_i^d = wv_i^d + (k_1 + (c_1 - k_1) r_1^d)(pbest_i^d - x_i^d) \\ + (k_2 + (c_2 - k_2) r_2^d)(gbest^d - x_i^d) \end{aligned} \tag{8}$$

Now, we discuss the influence of value of k_1 and k_2 to the convergence rate. In order to illustrate the convergence of the position and velocity for equation (2) and equation (8) respectively, we set $D = 1$ and $pbest_i = gbest = 0$. The corresponding results can be see in Fig. 5, Fig. 6 and Fig. 7. From Fig.5 (b) and Fig.6 (b), we can see that the position change for the original PSO becomes more instable than SFPSO, and the convergence rate with different values of k_1 and k_2 is be illustrated in Fig. 7. It is clear that the larger k_1 and k_2 are, the faster convergence rate is. In the next section, this SFPSO is used in the *solid* phase of the Phase Transitions



(a) Velocity of the particle (b) Position of the particle

Fig. 5. The convergence curves of the original PSO ($w = 0.8, x_i(0) = 10, v_i(0) = 10, c_1 = c_2 = 1.49445$ and $k_1 = k_2 = 0$).



(a) Velocity of the particle (b) Position of the particle

Fig. 6. The convergence curves of SFPSO ($w = 0.8, x_i(0) = 10, v_i(0) = 10, c_1 = c_2 = 1.49445$ and $k_1 = k_2 = 0.5$).

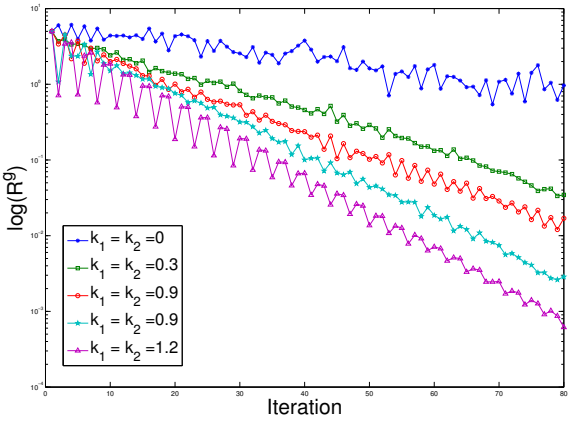


Fig. 7. The convergence curves of SFPSO with different k_1 and k_2 ($w = 0.8, x_i(0) = 10, v_i(0) = 10, c_1 = c_2 = 1.49445$).

Particle Swarm Optimization (PTPSO), and characteristics of PTPSO is given.

IV. Phase Transitions Particle Swarm Optimization

We model PSO as a thermodynamic system of three phases, the *solid* phase Ω_s, *liquid* phase Ω_l and *gas* phase Ω_g, and we present a Particle Swarm Optimizer based on the Phase Transition. In PTPSO, the swarm in each phase have a distinctive behavior:

$$v_i^d = wv_i^d + \begin{cases} a_s^d & , \rho \in \Omega_s \\ a_l^d & , \rho \in \Omega_l \\ a_g^d & , \rho \in \Omega_g \end{cases} \tag{9}$$

where ρ denotes the phase of swarm during the evolution process; a_s^d, a_l^d and a_g^d are different accelerated velocities of particles when ρ in different phase.

A. swarm in solid phase

In *solid* phase, the particles of swarm could convergence fast in a smaller region in the whole search space, just as the solid in which the molecules stay close together. And the particles could have small velocities, which is in favour of searching the region more efficiently. The SFPSO, which is explained in detail in Section III, have a good convergence performance, so we use the SFPSO in *solid* phase, and we get

$$\begin{aligned} a_s^d = (k_1 + (c_1 - k_1)r_1^d)(pbest_i^d - x_i^d) \\ +(k_2 + (c_2 - k_2)r_2^d)(gbest^d - x_i^d) \end{aligned} \tag{10}$$

B. swarm in liquid phase

When swarm is in *liquid* phase, we need the particles to explor more space than it do in *solid* phase, which makes for extending the search space to avoid the trapping in a local optimum. We define updating equation of velocity in liquid phase using original PSO as follow:

$$a_l^d = c_1 r_1^d (pbest_i^d - x_i^d) + c_2 r_2^d (gbest^d - x_i^d) \tag{11}$$

C. swarm in gas phase

In *gas* phase, a particle could fly to different directions in different dimensions to search more promising regions to find the global optimum, which increases the swarms' diversity to find the global optimum. CLPSO proposed by Liang [4] is a fine choice, since each dimension of a particle in general can learn from different $pbest$s for different dimensions for a few generation, which makes CLPSO have a larger search spaces than original PSO. So, a_g^d is defined as follow:

$$a_g^d = c \cdot r^d (pbest_{f_i(d)}^d - x_i^d) \tag{12}$$

where $\boldsymbol{f_i} = [f_i(1), f_i(2), ..., f_i(D)]$ defines which particles' $pbests$ the particle i should follow.

D. Condition of the Phase Transitions

In nature, the phase transitions among the *solid*, *liquid*, and *gas* phases due to the effects of temperature and pressure [8]. However, in PTPSO, we use the $\boldsymbol{pbest}$ and $\boldsymbol{gbest}$ in the swarm to directs the swarm to transform among the phases dynamically and automatically, and we use a time interval θ between two phase transitions to avoid the swarm to transform among phases frequently. The conditions of the phase transitions in PTPSO are defined as follow:

- Fusion: $\boldsymbol{gbest}$ is not updated, but the numbers of $\boldsymbol{pbest}$ updated is larger than the lower limit of $\boldsymbol{pbest}$ updated during the time interval θ;
- Vaporization: $\boldsymbol{gbest}$ is not updated, and the numbers of $\boldsymbol{pbest}$ updated is less than the lower limit of $\boldsymbol{pbest}$ updated during the time interval θ;

- Sublimation: $\boldsymbol{gbest}$ is not updated, and the numbers of $\boldsymbol{pbest}$ updated is larger than the lower limit of $\boldsymbol{pbest}$ during the time interval θ;
- Solidification: $\boldsymbol{gbest}$ is updated;
- Condensation: $\boldsymbol{gbest}$ is not updated, but the numbers of $\boldsymbol{pbest}$ updated larger than the lower limit of $\boldsymbol{pbest}$ during the time interval θ;
- Deposition: $\boldsymbol{gbest}$ is updated;

Remark 1: In PTPSO, if the $\boldsymbol{gbest}$ is updated, the whole swarm will transform to *solid* phase in next generation immediately, no matter what phase the swarm is in. So the time interval is just helpful in the case that the $\boldsymbol{gbest}$ is not updated.

Considering the velocity of particles in the swarm reduces quickly when the swarm transform into *solid* phase, and the position of particles is very close, we record the velocity of particles when the solidification takes place, and it will be recover when the Fusion happens. As for the sublimation, we use restarting strategy to initialize whole swarm when it happens. And the pseudocode of PTPSO is shown in Algorithm 1.

V. Experiments

In this section, we will illustrate the performance of the proposed PTPSO algorithm through the benchmark functions that were submitted for CEC 2013 Special Session on Real-Parameter Optimization [9]. All these functions are minimization problems, the search ranges of all test functions are defined as: $[-100, 100]^D$.

The proposed PTPSO algorithm has been implemented using MATLAB 7.6.0 in a PC which has a CPU of Intel(R) Core(TM) i3 3.20GHz and a memory of 4G. Experiments are conducted on 28 functions with 30 dimensions. The details of these test functions are given in TABLE I.

TABLE I. Details of Unimodal, Basic multimodal and Composition test functions

Type	No.
Unimodal Function	$f_1 - f_5$
Basic Multimodal Function	$f_6 - f_{20}$
Composition Function	$f_{21} - f_{28}$

TABLE II. Parameters Setting for Three variants of PSO

Algorithm	Parameters
PSO1	$\omega = 0.9 \rightarrow 0.4, c1 = c2 = 2$
PSO2	$\omega = 0.9 \rightarrow 0.4, c1 = c2 = 1.49445$
CLPSO	$\omega = 0.9 \rightarrow 0.4, c = 1.49445$

TABLE III. Parameters Setting for PTPSO

Parameters	Explain
$c_1 = c_2 = c = 1.49449$	Same as PSO
$\omega = 0.9 \rightarrow 0.4$	Same as PSO
$k_1 = k_2 = 0.5$	They are used to speed up the convergence rate
$\theta = 20$	Time interval two phase transitions
$\xi = 40$	The lower limit of $\boldsymbol{pbest}$ updated during the time interval θ

We compare our proposed PTPSO algorithm with three different variants of PSO. Those algorithms and their parameters setting are shown in Table II and Table III. The population size is set to 40 in all the experiment and the $MaxFES$ is set to 300000 according to [9].

Algorithm 1 PTPSO

BEGIN
Initialize the particle swarm with $\boldsymbol{X}$ and $\boldsymbol{V}$;
Update the $\boldsymbol{pbest}$, $\boldsymbol{gbest}$;
Set the $\rho \in \Omega_s$, $\boldsymbol{V_l} = \boldsymbol{V}$;
Set the $run\_count = 0$ and θ;
while $FEs < MaxFEs$ **do**
  for each particle in the swarm **do**
    if $\rho \in \Omega_s$ **then**
      Update particle velocity according to (9);
    end if
    if $\rho \in \Omega_l$ **then**
      Update particle velocity according to (10);
    end if
    if $\rho \in \Omega_g$ **then**
      Update particle velocity according to (11);
    end if
    Update particle position according to (2);
    Calculate particles fitness;
  end for
  $run\_count = run\_count + 1$;
  Update the $\boldsymbol{pbest}$, $\boldsymbol{gbest}$;
  if $\boldsymbol{gbest}$ is updated **then**
    if $\rho \in \Omega_l$ **then**
      $\boldsymbol{V_l} = \boldsymbol{V}$
    end if
    Set the $\rho \in \Omega_s$, $run\_count = 0$;
  else
    if $run\_count >= \theta$ **then**
      if $\boldsymbol{pbest}$ is updated **then**
        if $\rho \in \Omega_s$ **then**
          $\boldsymbol{V} = \boldsymbol{V_l}$
        end if
        Set the $\rho \in \Omega_l$, $run\_count = 0$
      else
        if $\rho \in \Omega_s$ **then**
          Restarting the swarm with $\boldsymbol{X}$ and $\boldsymbol{V}$
        end if
        Set the $\rho \in \Omega_g$, $run\_count = 0$;
      end if
    end if
  end if
  $GEN = GEN + 1$;
end while
END

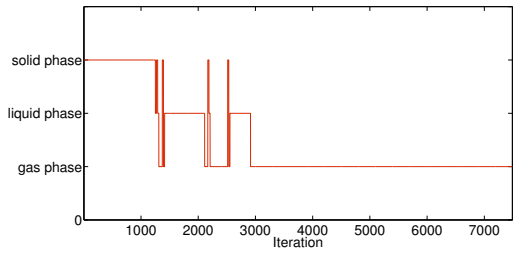
Table IV lists the mean and standard deviation of the fitness values over 51 independent runs for each problem, and the best results are shown in bold. It can be seen from Table IV that PTPSO performs better than the other three algorithms on functions f_2-f_3, f_7, f_{10}, f_{12}-f_{13}, f_{18}, f_{20} and f_{23}-f_{27}, int terms of the average fitness values; PTPSO has the same mean value as other three algorithm on functions f_1 and f_5. The comparsion of convergence rate between and other three algorithms is also carried out on twenty functions, selected from unimodal, basic multimodal and composition functions as the representatives. The convergence rates of the

TABLE IV. MEAN FUNCTION VALUES AND STANDARD DEVIATIONS OVER 51 RUNS OF THE 28 TESTED BENCHMARK PROBLEMS

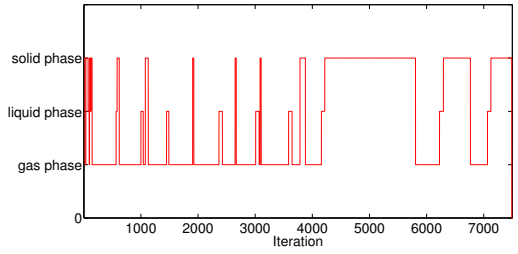
Func. No.		PSO1	PSO2	CLPSO	PTPSO	Func. No.		PSO1	PSO2	CLPSO	PTPSO
f_1	Mean	**-1.400000e+003**	**-1.400000e+003**	**-1.400000e+003**	**-1.400000e+003**	f_{15}	Mean	4.524437e+003	6.837346e+003	**4.085293e+003**	4.294271e+003
f_1	Std.Dev.	1.716400e-025	6.927638e-026	**0.000000e+000**	6.824240e-026	f_{15}	Std.Dev.	1.352138e+006	3.764788e+005	**2.217639e+005**	1.382437e+006
f_2	Mean	1.565201e+007	1.496858e+007	1.676468e+007	**6.399401e+006**	f_{16}	Mean	2.022035e+002	2.022701e+002	**2.010632e+002**	2.014839e+002
f_2	Std.Dev.	1.594956e+014	9.457975e+013	2.492442e+013	**2.267129e+013**	f_{16}	Std.Dev.	1.052762e-001	1.102307e-001	5.685439e-002	**4.978751e-002**
f_3	Mean	1.225886e+008	1.702660e+008	2.884749e+008	**7.756010e+007**	f_{17}	Mean	3.707515e+002	**3.580572e+002**	4.019380e+002	3.787626e+002
f_3	Std.Dev.	1.928585e+017	3.893948e+016	1.831421e+016	**7.918914e+015**	f_{17}	Std.Dev.	**3.722442e+002**	5.284239e+002	7.692356e+002	3.971244e+002
f_4	Mean	**-4.740129e+002**	3.335609e+003	2.924986e+004	5.542473e+003	f_{18}	Mean	5.283302e+002	6.297546e+002	5.272407e+002	**4.917042e+002**
f_4	Std.Dev.	**8.819068e+004**	1.765599e+006	4.644325e+007	8.974962e+006	f_{18}	Std.Dev.	1.689051e+003	1.651949e+003	6.565912e+002	**4.242977e+002**
f_5	Mean	**-1.000000e+003**	**-1.000000e+003**	**-1.000000e+003**	**-1.000000e+003**	f_{19}	Mean	5.038064e+002	**5.034338e+002**	5.065026e+002	5.039981e+002
f_5	Std.Dev.	2.652148e-025	**1.225261e-025**	1.626444e-024	2.605619e-025	f_{19}	Std.Dev.	1.224889e+000	9.910063e-001	9.869410e-001	**8.712785e-001**
f_6	Mean	-8.202179e+002	-8.181115e+002	**-8.473913e+002**	-8.418077e+002	f_{20}	Mean	6.144618e+002	6.147076e+002	6.117062e+002	**6.110683e+002**
f_6	Std.Dev.	1.508893e+003	1.357348e+003	**4.032285e+002**	1.116671e+003	f_{20}	Std.Dev.	1.682599e+000	8.265051e-001	**1.199178e-001**	4.219935e-001
f_7	Mean	-7.621494e+002	-7.635811e+002	-7.448292e+002	**-7.674035e+002**	f_{21}	Mean	9.872226e+002	**9.846612e+002**	9.881722e+002	1.007431e+003
f_7	Std.Dev.	2.231002e+002	1.755740e+002	**5.349714e+001**	1.233318e+002	f_{21}	Std.Dev.	6.730260e+003	4.820507e+003	**7.028904e+002**	7.276757e+003
f_8	Mean	-6.790653e+002	**-6.790717e+002**	-6.790645e+002	-6.790579e+002	f_{22}	Mean	2.000436e+003	**1.674767e+003**	3.648873e+003	2.637943e+003
f_8	Std.Dev.	3.008995e-003	4.311543e-003	3.379074e-003	**2.603645e-003**	f_{22}	Std.Dev.	1.229145e+005	**7.492182e+004**	1.076281e+005	4.045513e+005
f_9	Mean	-5.777055e+002	**-5.778745e+002**	-5.736514e+002	-5.761390e+002	f_{23}	Mean	5.460152e+003	7.730306e+003	5.175000e+003	**4.775897e+003**
f_9	Std.Dev.	8.186197e+000	1.875486e+001	**3.139575e+000**	1.114881e+001	f_{23}	Std.Dev.	1.267321e+006	**2.973347e+005**	1.643995e+005	4.591426e+005
f_{10}	Mean	-4.998659e+002	-4.996356e+002	-4.955316e+002	**-4.998815e+002**	f_{24}	Mean	1.268526e+003	1.266974e+003	1.276332e+003	**1.262146e+003**
f_{10}	Std.Dev.	**4.581160e-003**	1.342845e-001	1.650447e+000	6.609069e-003	f_{24}	Std.Dev.	6.324321e+001	1.081747e+002	**2.875218e+001**	8.669359e+001
f_{11}	Mean	-3.706390e+002	**-3.798276e+002**	-3.536425e+002	-3.712422e+002	f_{25}	Mean	1.385365e+003	1.385583e+003	1.392171e+003	**1.378876e+003**
f_{11}	Std.Dev.	5.952988e+001	**2.597947e+001**	3.964299e+001	8.738399e+001	f_{25}	Std.Dev.	6.304907e+001	1.115803e+002	**2.642875e+001**	1.042958e+002
f_{12}	Mean	-2.317906e+002	-1.753202e+002	-2.277909e+002	**-2.369836e+002**	f_{26}	Mean	1.513336e+003	1.512772e+003	1.400969e+003	**1.400178e+003**
f_{12}	Std.Dev.	4.605180e+002	3.703400e+003	**1.408667e+002**	4.916068e+002	f_{26}	Std.Dev.	4.964082e+003	4.904673e+003	1.129429e-001	**4.666844e-002**
f_{13}	Mean	-5.975738e+001	-2.970247e+001	-7.427091e+001	**-9.286941e+001**	f_{27}	Mean	2.203200e+003	2.188762e+003	2.303465e+003	**2.165434e+003**
f_{13}	Std.Dev.	1.022219e+003	1.362917e+003	**3.522784e+002**	6.506441e+002	f_{27}	Std.Dev.	9.538437e+003	6.470107e+003	**2.228614e+003**	6.408788e+003
f_{14}	Mean	9.947394e+002	**7.203441e+002**	2.224728e+003	1.601774e+003	f_{28}	Mean	1.836658e+003	1.819361e+003	**1.700000e+003**	1.878331e+003
f_{14}	Std.Dev.	1.115697e+005	**6.249524e+004**	7.820955e+004	3.033001e+005	f_{28}	Std.Dev.	1.535924e+005	1.337880e+005	**5.365406e-017**	1.745473e+005

four algorithms are presented in Fig. 8. respectively, where FEs denotes the number of function evaluations. For functions f_1-f_3, f_5-f_6, f_{10}, f_{20}-f_{21} and f_{27}, it is obvious that PTPSO converges much faster than the other three algorithms.

The effect of the phase transitions in PTPSO is also investigated using the unimodal function f_1 and basic multimodal function f_{20}. The search behavior of PTPSO on these two functions are illustrated in Fig. 9. The figure is plotted based on the data of a typical run that can represent the search behavior of the algorithm in most cases. According to the figure, on the unimodal function f_1, we can see that the swarm stay in $solid$ phase at the beginning stage of the optimization process, until at about $Iteration = 1000$, because it is relatively easy to locate the global optimum and make the $\boldsymbol{gbest}$ updated, the swarm have a long time to stay in *solid* phase, which can reduce the search space by the newly added parameter k_1 and k_2 and speed up the convergence. In contrast, on the multimodal function f_{20}, the global optimum is more difficult to locate. As it seen in Fig.9 (b), the phase transitions happened frequently during the evolution process in PTPSO, those phase transitions help the swarm avoid to trapped in a local optimum. Fusion, vaporization, and sublimation bring in diversity, which makes PTPSO jump out of the local optimum and continue improving the swarm; Condensation, solidification and deposition enhance the exploitation capability of the swarm, which makes PTPSO converge fast.



(a) The phase transitions in f_1



(b) The phase transitions in f_{20}

Fig. 9. The illustration of the search behaviour of PTPSO in f_1 and f_{20} with 30 dimensions during the 7500 iteration at once running.

VI. CONCLUSION

In this paper we have proposed a new algorithm called Phase Transitions Particle Swarm Optimization (PTPSO) inspired by the behaviour of different phases and the phase transitions model. In this approach, the swarm is endowed with one of three different phases (*solid*, *liquid* and *gas* phase) during the evolution process, and the phase of the swarm can transform among those three phases dynamically and automatically according to the evolutional states to balance between exploration and exploitation adaptively. Transitions of condensation, solidification and deposition can enhance the exploitation capability of the swarm. While the transitions of fusion, vaporization and sublimation from the other direction improve the exploration capability of the swarm. Especially, it uses a new modified PSO algorithm called Simple Fast Particle

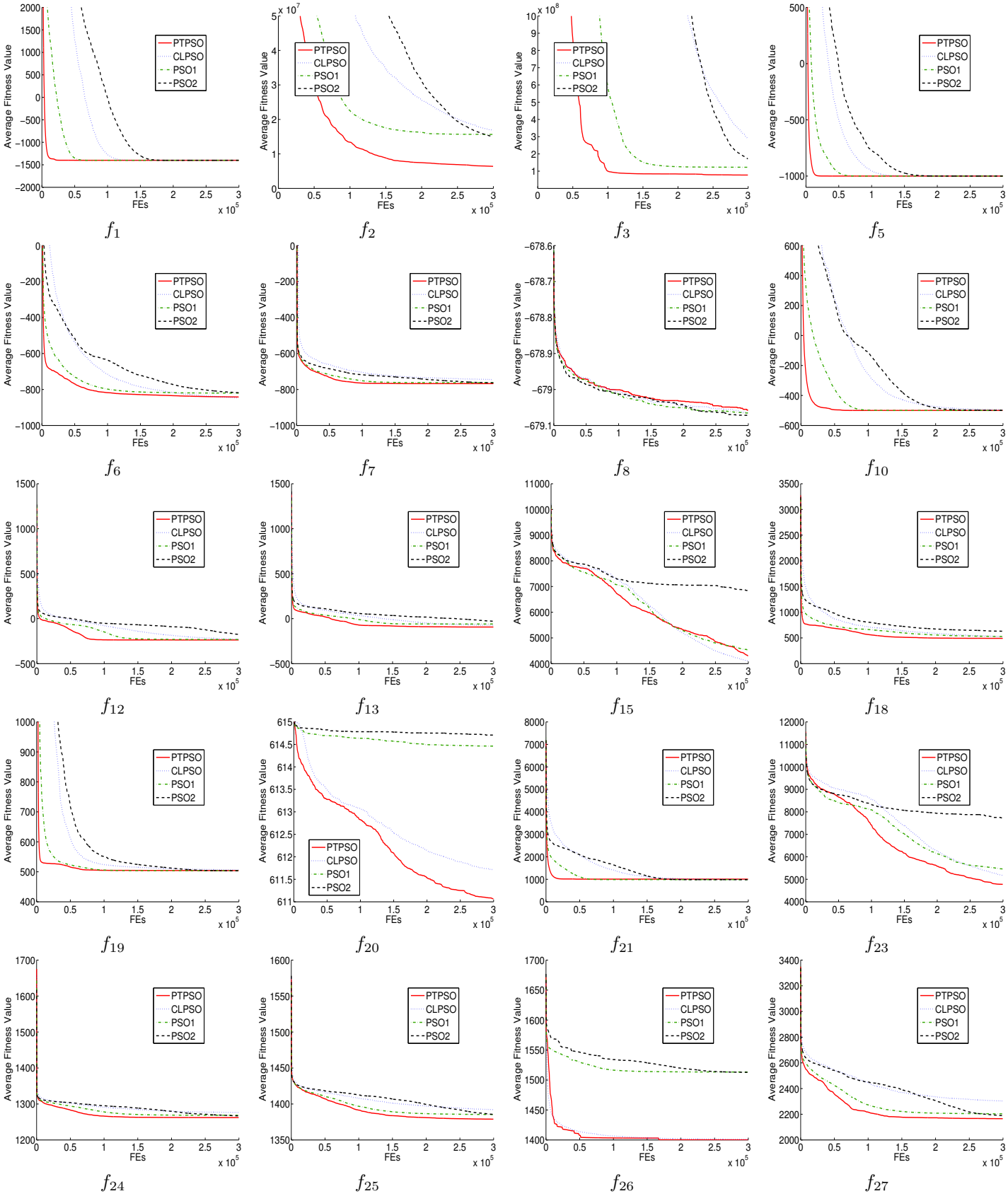


Fig. 8. The comparison of convergence rates among PTPSO, PSO1, PSO2 and CLPSO over 51 runs on twenty functions with the Dimension $D = 30$.

Swarm Optimization (SFPSO) in the *solid* phase, which modifies the original PSO by adding new parameters simply to make the algorithm convergence more quickly. We have performed an extensive experimentation to show the performance of the proposed approach and compared PTPSO with three varaints of PSO. Based on the results of the four algorithms on the 28 benchmark problems of CEC 2013, we conclude that PTPSO has good search capabilities and generates better solutions and manages to prevent permature convergence and keep the fast-converging feature of the original PSO.

For future research, we plan to integrate with other features of the phase transitions modal to strengthen the PTPSO algorithm. On the other hand, in the multiobjective and dynamic optimization problems, more diversity is required. We also plan to test our PTPSO algorithm in those problems.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (NSFC), under Grants No. 61272271, 61332008, 61174158, 61103072, 61103071 and 61103068. This program is also supported by the National Basic Research Program of China (973 Program) under Grant No. 2014CB340404, Natural Science Foundation Program of Shanghai under Grant No.12ZR1434000, Fundamental Research Funds for the Central Universities under Grants No.0800219201, Research Fund for the Doctoral Program of Higher Education of China (20110072120065) and the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry.

REFERENCES

[1] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory", *Proceedings of the 6th International Symposium on Micromachine and Human Science*, Nagoya, Japan, pp. 39-43, 1995.

[2] J. Kennedy and R. Eberhart, "Particle Swarm Optimization", *Proceedings of the 1995 IEEE International Conference on Neural Networks (Perth, Australia)* , IEEE Service Center, Piscataway, NJ, IV: pp. 1942-1948, 1995.

[3] Shi Y and Eberhart R, "A modified particle swarm optimizer", *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence* , IEEE, pp. 69-73, 1998.

[4] J. J. Liang, A. K. Qin, P. N. Suganthan and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions",*Transactions on Evolutionary Computation*, Volume 10, 2006.

[5] Tabor D. "Gases, liquids and solids: and other states of matter", Cambridge University Press, 1991.

[6] Stanley and H Eugene. "Introduction to phase transitions and critical phenomena", Oxford University Press, pp. 336, 1987.

[7] Breithaupt, J."Physics".Nelson Thornes, 2001.

[8] Landau L."The theory of phase transitions". *Nature*, pp. 840-841, 1936.

[9] J. J. Liang, B. Y. Qu, P. N. Suganthan and A. G, Hernandez-Diaz,"Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization", Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China, and Technical Report, Nanyang Technological University, Singapore, Technical Report 201212. 2013