ON THE OPTIMAL CONTROL OF SINGLE-STAGE HYBRID MANUFACTURING SYSTEMS VIA NOVEL AND DIFFERENT VARIANTS OF PARTICLE SWARM OPTIMIZATION ALGORITHM

M. SENTHIL ARUMUGAM AND M. V. C. RAO

Received 1 March 2005

This paper presents several novel approaches of particle swarm optimization (PSO) algorithm with new particle velocity equations and three variants of inertia weight to solve the optimal control problem of a class of hybrid systems, which are motivated by the structure of manufacturing environments that integrate process and optimal control. In the proposed PSO algorithm, the particle velocities are conceptualized with the local best (or *pbest*) and global best (or *gbest*) of the swarm, which makes a quick decision to direct the search towards the optimal (fitness) solution. The inertia weight of the proposed methods is also described as a function of pbest and gbest, which allows the PSO to converge faster with accuracy. A typical numerical example of the optimal control problem is included to analyse the efficacy and validity of the proposed algorithms. Several statistical analyses including hypothesis test are done to compare the validity of the proposed algorithms with the existing PSO technique, which adopts linearly decreasing inertia weight. The results clearly demonstrate that the proposed PSO approaches not only improve the quality but also are more efficient in converging to the optimal value faster.

1. Introduction

The hybrid systems combine two different types of categories, subsystems with continuous dynamics and subsystems with discrete dynamics that interact with each other. Such hybrid systems arise in varied contexts in manufacturing, communication networks, automotive engine design, computer synchronization, and chemical processes, among others. In hybrid manufacturing systems, which is considered in this paper, the manufacturing process is composed of the event-driven dynamics of the parts moving among different machines and the time-driven dynamics of the processes within particular machines. Frequently in hybrid systems, the event-driven dynamics are studied separately from the time-driven dynamics, the former via timed state automata or Petri net models, PLC etc., and the latter via differential or difference equations [6].

The hybrid framework can be modeled either by extending the event-driven models with time-driven dynamics; or by extending the traditional time-driven models with event- driven dynamics. The hybrid system-modeling framework, which is motivated by

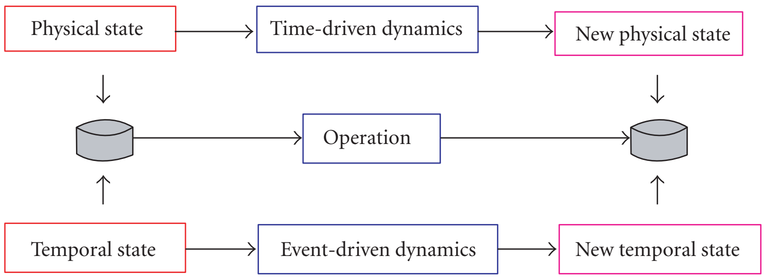


Figure 2.1. Hybrid framework with time-driven and event-driven dynamics.

the structure of many manufacturing system, considered in this research adopts the first category. To represent the hybrid nature of the model, each job is characterized by a *physical state* and a *temporal state*. The physical state represents the physical characteristics of interest and evolves according to the time-driven dynamics (e.g., difference or differential equations) while the job is being processed by a server. The temporal state represents processing arrival and completion times and evolves according to the discrete-event dynamics (e.g., queuing dynamics). The interaction of time-driven with event-driven dynamics leads to a natural tradeoff between temporal requirements on job completion times and physical requirements on the quality of the completed jobs (see Figure 2.1). Such modeling frameworks and optimal control problems have been considered in [1, 8].

A number of algorithms were developed so far to solve such optimal control problems. Particle swarm optimization (PSO) is one of the modern heuristic algorithms under the evolutionary algorithms (EA) and gained lots of attention in solving optimal control problems. Several variants of the PSO technique have been proposed so far, following Eberhart and Kennedy [3, 4]. In this paper, different global versions of PSO with modified velocity equations and inertia weights are investigated. The parameter selections in the PSO equations are carefully anlaysed in terms of pbest and gbest. Three different inertia weight (one standard and two new) variants are adopted with four versions (one existing and three proposed) of velocity equations are investigated in this paper. Among such 12 methods, the best methods are identified and their validity is verified through a number of statistical analyses and approaches.

The remaining of this paper is organized as follows: In Section 2, the optimal control problem of a single-stage hybrid manufacturing system is studied and formulated. The functional procedure and behavior of standard PSO are briefed in Section 3. Section 4 depicts the design of new inertia weight variants, modified velocity equations and the parameter selections for PSO algorithms. The numerical example, the simulation results and the statistical analyses are given in Section 5 and finally the discussions and conclusions are drawn in Section 6.

2. Problem formulation of single-stage hybrid manufacturing system

The general hybrid system framework with event-driven and time-driven dynamics is given in Figure 2.1.

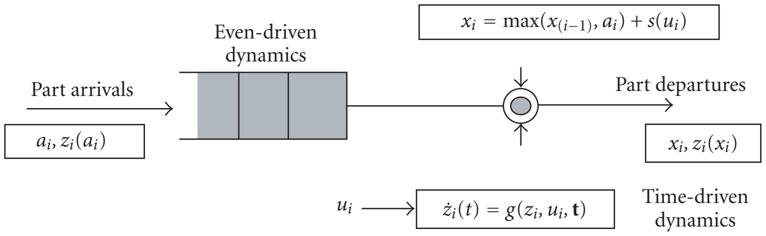


FIGURE 2.2. A single-stage hybrid manufacturing system.

Consider the hybrid model of a single-stage manufacturing hybrid system model is shown in Figure 2.2. A sequence of N jobs $(C_1, C_2, C_3, \ldots, C_N)$ is assigned by an external source to arrive for processing at known times $0 \le a_1 \le a_2 \le \cdots \le a_N$. The jobs are processed first-come first-serve (FCFS) basis by a work-conserving and nonpreemptive server. The processing time is $s(u_i)$, which is a function of a control variable u_i, and $s(u_i) \ge 0$.

The time-driven dynamics of the hybrid system is defined by the equation which evolved the job C_i which is initially at some physical state ξ_i at time x_0.

$$\dot{z}_i(t) = g(z_i, u_i, t), \qquad z_i(x_0) = \xi_i. \tag{2.1}$$

The event-driven dynamics is described by recursive non-linear equations (Max-plus equations) involving a maximum or a minimum operation, which is typically found in models of discrete event systems (DES). For the fist-come first-serve (FCFS), nonpreemptive, single server example in Figure 2.2, these dynamics are given by the "max-plus" recursive equation

$$x_i = \max\left(x_{(i-1)}, a_i\right) + s(u_i), \quad i = 1, \ldots, N, \tag{2.2}$$

where x_i is the departure or completion time of ith job and $x_0 = -\infty$. The recursive relationship given in (2.2) is known in queuing theory as the Lindley equation [8].

From (2.1) and (2.2), it is clear that the choice of control u_i affects both the physical state z_i and next temporal state x_i, and thus time-driven dynamics (2.1) and event-driven dynamics (2.2), justifying the hybrid nature of the system. According to [6], there are two alternative ways to view this hybrid system. The first is as a discrete event queuing system with time-driven dynamics evolving during processing in the server as shown in Figure 2.3.

The second viewpoint interprets the model as a switched system. In this framework, each job must be processed until it reaches a certain "quality level" denoted by Γ_i (e.g., a threshold above which z_i satisfies a desired quality level). That is, the processing time for

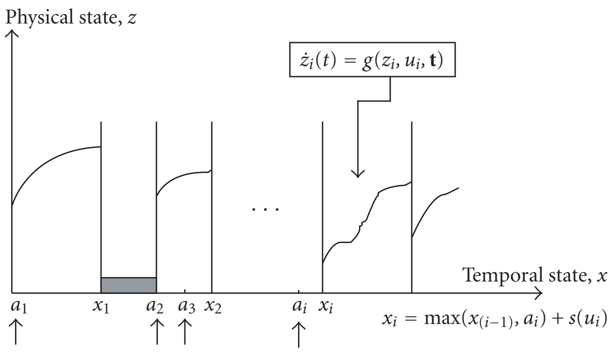

Physical state, z
$\dot{z}_i(t) = g(z_i, u_i, \mathbf{t})$
Temporal state, x
a_1
x_1
a_2
a_3
x_2
a_i
x_i
$x_i = \max(x_{(i-1)}, a_i) + s(u_i)$

Figure 2.3. Typical trajectory.

each job is chosen to satisfy the stopping rule

$$s_i(u_i) = \min\left[t \geq 0;\ z_i(t_0) = \int_{t_0}^{t_0+t} g_i(\tau, u_i, t)\, d\tau + z(t_0) \in \Gamma_i\right]. \tag{2.3}$$

Figure 2.3 shows the evolution of the physical state as a function of time t. It is shown in the figure that the dynamics of the physical state experiences a "switch" when certain events occur. These events may classify into two categories: uncontrolled (or exogenous) arrival events and controlled departure events. In Figure 2.2, the first event is an exogenous arrival event at time a_1. When the first job arrives at a_1, the physical state starts to evaluate the time-driven dynamics until it reaches the departure time x_1. It is clearly observed that the first job completes before the second job arrives and hence there is an idle period, in which the server has no jobs to process. The physical state again begins evolving the time-driven dynamics at time a_2 (arrival of second job) until the second job completes at x_2. However, that the third job arrived before the second job was completed. So the third job is forced to wait in the queue until time x_2. After the second job completes at x_2 the physcal state begins to process the third job. As indicated in Figure 2.3, not only do the arrival time and departure time cause switching in the time-driven dynamics according to (2.1), but also the sequence in which these events occur is governed by the event-driven dynamics given in (2.2).

This system is hybrid is the sense that it combines the time-driven dynamics (2.1) with the event-driven dynamics (2.2), the two being coupled through the choice of control sequence $\{u_1,\dots,u_N\}$. Hence the optimal control problem considered in this paper is to minimize an objective function of the form

$$J = \sum_{i=1}^{N} \{\theta_i(u_i) + \phi_i(x_i)\}^{\eta}. \tag{2.4}$$

Although, in general, the state variables $z_i,\dots,z_N$ evolve continuously with time, minimizing (2.4) is an optimization problem in which the values of the state variables are considered only at the job completion times $x_1,\dots,x_N$. Since the stopping criterion in (2.3) is used to obtain the service times, a cost on the physical state $z_i(x_i)$ is unnecessary because the physical state of each completed job satisfies the quality objectives, that is, $z_i(x_i) \in \Gamma_i$.

Obviously, the control variable u_i is affecting the processing time through $s_i = s(u_i)$ for extensions to cases with time-varying controls $u_i(t)$ over a service time. By assuming $s_i(\cdot)$ is either monotone increasing or monotone decreasing, given a control u_i, service time s_i can be determined from $s_i = s(u_i)$ and *vice versa*. For simplicity, let $s_i = u_i$ and the rest of the analysis is carried out with the notation u_i. Hence the optimal control problem, with $\eta = 1.5$, denoted by P is of the following form:

$$P: \min_{u_1,\dots,u_N} \left\{ J = \sum_{i=1}^{N} \{\theta_i(u_i) + \phi_i(x_i)\}^{1.5} : u \geq 0,\ i = 1,\dots,N \right\} \tag{2.5}$$

subject to

$$x_i = \max\left(x_{(i-1)}, a_i\right) + s(u_i), \quad i = 1,\dots,N. \tag{2.6}$$

The optimal solution of P is denoted by u_i^* for $i = 1,\dots,N$, and the corresponding departure time in (2.6) is denoted by x_i^* for $i = 1,\dots,N$.

3. Review of standard particle swarm optimization techniques

The particle swarm optimization (PSO) is a parallel evolutionary computation technique developed by Kennedy and Eberhart based on the social behavior of metaphor. The PSO technique has ever since turned out to be a competitor in the fields of numerical optimization. The evolutionary algorithms, EAs, like genetic algorithm (GA) and evolutionary programming (EP) are search algorithms based on the simulated evolutionary process of natural selection, variation, and genetics. Both GA and EP can provide a near global solution [4]. PSO is similar to the other evolutionary algorithms in that the system is initialized with a population of random solutions, conceptualized as particle. Each particle is assigned a randomized velocity and is iteratively moved through the problem space. It is attracted towards the location of the best fitness achieved so far by the particle itself, called as personal best (pbest) and the location of the best fitness achieved so far across the whole population, known as global best (gbest). The PSO algorithm includes some tuning parameters which are clearly influence the performance of the algorithm, often referred to as exploration-exploitation tradeoff. Exploration is the ability to test various regions in the problem space in order to locate a good optimum, hopefully a global solution. Exploitation is the ability to concentrate the search around a promising candidate solution in order to locate the optimum precisely. A complete theoretical analysis of PSO has been described by Clerc and Kennedy in [2].

Also, PSO will not follow survival of the fittest, the principle of other EAs. PSO when compared to EP has very fast converging characteristics; however it has a slow fine-tuning ability of the solution. Also PSO has a more global searching ability at the beginning of the run and a local search near the end of the run. Therefore, while solving problems with more local optima, there are more possibilities for the PSO to explore local optima at the end of the run [5, 7].

PSO is basically through simulation of bird flocking in two-dimension space. The position of each particle is represented by XY axis position and the velocity is expressed by vx (the velocity of X axis) and vy (the velocity of Y axis). Modification of the particle position is realized by position and velocity information. Each particle knows its best value so far (*pbest*) and its XY position. The information corresponds to personal experiences of each particle in the concept of individual learning and cultural transmission (ILCT). Moreover, each particle knows the best value so far in the group (*gbest*) among *pbests*. The information corresponds with the knowledge of how the other particles around them have performed in the concept of ILCT [5]. Namely, each particle tries to modify its position using the following information:

(i) the distance between the current position and *pbest*;

(ii) the distance between the current position and *gbest*.

This modification can be represented by the concept of velocity. Velocity of each particle can be modified by the following equation:

$$v_i^{k+1} = wv_i^k + c_1 rand_1 \times (pbest_i - X_i^k) + c_2 rand_2 \times (gbest - X_i^k), \tag{3.1}$$

where

(i) v_i^{k+1}: velocity of particle i at iteration k;

(ii) w: weighting function;

(iii) c_1 *and* c_2: two positive constants named as cognitive and social parameter respectively (normally $c_1 = c_2 = 2$);

(iv) *rand*: random number between 0 and 1;

(v) X_i^k: current position of particle i at iteration k;

(vi) $pbest_i$: *pbest* of particle i;

(vii) *gbest*: *gbest* of the group.

And the current position can be modified by the following equation:

$$X_i^{k+1} = X_i^k + v_i^{k+1}. \tag{3.2}$$

In the standard PSO, the inertia weight is introduced as a decreasing function which is set to a higher value ($w_{\max}$) at initial stage and is decreased linearly with the iteration, k to a lower value ($w_{\min}$) and it is represented by the equation

$$w = w_{\max} - \left(\frac{w_{\max} - w_{\min}}{k_{\max}}\right) \times k, \tag{3.3}$$

where $k_{\max}$ is the maximum iteration number.

From (3.1), three terms are taken into consideration: the first term is wv_i^k, is the particle's previous velocity weighted by the inertia weight w. The second term, $(pbest_i - X_i^k)$,

Table 4.1

Method	PSO parameter	Velocity equation
W	Standard PSO	$v_i^{k+1} = wv_i^k + c_1 rand_1 \times (pbest_i - X_i^k) + c_2 rand_2 \times (gbest - X_i^k)$
X	$c_1 = c_2 = \left(1 + \frac{pbest_i}{gbest}\right)$ $rand_1 = rand_2 = rand$	$v_i^{k+1} = wv_i^k + \left(1 + \frac{pbest_i}{gbest}\right) \times rand \times (pbest_i + gbest - 2X_i^k)$
Y	$c_1 = c_2 = 2$ $rand_1 = rand_2$	$v_i^{k+1} = wv_i^k + [2 \times rand \times (pbest_i + gbest - 2X_i^k)]$
Z	$c_1 = c_2 = \left(1 + \frac{gbest}{pbest_i}\right)$ $rand_1 = rand_2 = rand$	$v_i^{k+1} = wv_i^k + \left(1 + \frac{gbest}{pbest_i}\right) \times rand \times (pbest_i + gbest - 2X_i^k)$

Table 4.2

Method	Inertia weight
A	$w = w_{\max} - \left(\frac{w_{\max} - w_{\min}}{k_{\max}}\right) \times k$
B	$w_i = \left(1 - \frac{gbest_i}{pbest_i}\right)$
C	$w_i = \left(1 - \frac{gbest_i}{(pbest_i)_{average}}\right)$

is the distance between the particle's best previous position, and its current position. And the third term, $(gbest - X_i^k)$, is the distance between the swarm's best experience, and the ith particle's current position. Equation (3.2) provides the new position of ith particle, adding its new velocity, to its current position. In general, the performance of each particle is measured according to a fitness function, which is problem-dependent. In optimization problems, the fitness function is usually the objective function under consideration.

4. New variants of inertia weight and velocity equation

The role of inertia weight is very crucial on PSO's performance and convergence behavior. The inertia weight is employed to control the impact of the history of velocities on the current velocity. In this way, the inertia weight regulates the trade-off between global and local exploration abilities of the swarm. A suitable value for the inertia weight provides a balance between the global and local exploration ability of the swarm which results in better convergence rates. Similarly, the velocity equation of PSO also plays an important role for a quality solution and faster convergence. Since the new position of the particle depends on the velocity, it is very important to design the parameters in the velocity equation carefully in order to move closer to the optimal solution faster. Such a careful design of PSO parameters is considered in this paper. In this paper, new variants to the inertia weight and velocity equations are considered and they are classified into two groups. Group 1 (Table 4.1) includes the new proposed velocity equations and Group 2 (Table 4.2) consists of inertia weight variants.

```
Step 1. Initialize
            Set index of Global best (Gbest index) = 1.
Step 2. Create population
            Randomize the positions and velocities for entire population
            Set the reference value of best position PB (i). Fitness
            Update velocity vector.
Step 3. Calculate P (i). Fitness
            If P (i). Fitness < reference value of best position PB (i). Fitness Then
            Set new reference value of best position as P (i). Fitness
            If PB (i). Fitness < PB (GBestIndex). Fitness
            Then Set new GBestIndex = i.
Step 4. Calculate the particle velocity using the swarm equations given in group 1 & 2
            Update the particle positions using (3.1).
Step 5. Repeat until Max number of generation or best solution.
```

Algorithm 4.1

Each method described in group 1 adopts all the three inertia weights given in group 2 individually and hence 12 methods are formed namely methods WA, WB, WC, XA, XB, XC, Y, YB, YC, ZA, ZB, and ZC. All these 12 methods are implemented in the PSO algorithm (Algorithm 4.1), to solve the optimal control problem of the single-stage hybrid manufacturing systems.

5. Experimental results and statistical analyses

In order to compare the validity and usefulness of the proposed PSO methods, the optimal control problem from (2.5) and (2.6) with the following functions are considered:

$$\theta_i(u_i) = \frac{2}{u_i}, \qquad \phi(x_i) = 3^* x_i, \qquad \eta = 1.5. \tag{5.1}$$

Now (2.5) becomes

$$\min_{u_1,\dots,u_N} \left\{ J = \sum_{i=1}^{N} \left(\frac{2}{u_i} + 3x_i \right)^{1.5} \right\} \tag{5.2}$$

subject to

$$x_i = \max\left(x_{(i-1)}, a_i\right) + u_i. \tag{5.3}$$

The optimal controls (u_i), the departure time (x_i) and cost or fitness (J) for the objective function given in (5.2) are computed with the following parameter settings.

(i) The maximum number of generations is set as 2000.

(ii) The population size = 20.

```
ab(1) = 0.3; ab(2) = 0.5; ab(3) = 0.7; ab(4) = 1
        For bb = 1 To N/4
              For aa = 1 To 4
                      ab(aa + 4*bb) = ab(aa) + 1*bb
              Next aa
        Next bb
        For i = 1 To N
                      a(i) = ab(i)
Next i
```

Algorithm 5.1. Arrival sequence for hybrid systems.

Table 5.1. Statistical analyses of various methods at the 500th generation.

Method number	Method	Average	Best	Worst	SD	CV	Avedev
1	WA	15379.9102	13907.6580	19786.4941	170.9924	0.0111	62.1193
2	WB	7891.9087	7795.7406	8190.4794	25.5744	0.0032	14.0143
3	WC	7901.4545	7777.6306	8292.2425	20.4858	0.0026	10.1641
4	XA	17376.4738	14081.6692	20189.4607	180.6685	0.0104	91.4140
5	XB	9004.9407	8055.2561	10690.0855	24.8136	0.0028	37.1037
6	XC	9398.6967	8174.7296	11432.7873	62.7363	0.0067	32.0660
7	YA	17221.8462	14097.6442	19386.2370	177.1593	0.0103	131.8242
8	YB	7815.1086	7769.5156	7933.5245	2.4994	0.0003	1.2602
9	YC	7817.5676	7763.1663	7895.4521	3.5350	0.0005	1.9573
10	ZA	15750.0587	12465.2807	17613.1775	58.2452	0.0037	26.0689
11	ZB	7858.3669	7795.8160	8026.0831	6.9454	0.0009	3.4780
12	ZC	7861.0846	7779.7099	8011.0604	14.6579	0.0019	9.4978

(iii) Number of jobs = 50, and
(iv) Total number of run (simulation) = 1000.

The arrival sequence (a_i for $i = 1$ to N) for $N = 50$ is obtained from Algorithm 5.1.

The PSO algorithms associated with the 12 methods are simulated 1000 times at different intervals of time. The optimal control variable (u_i), the departure time (x_i for $i = 1, 2, \ldots, 50$) and the fitness objective function (J) are computed for all 12 methods. The average value of the optimal control variable (u_i), and the corresponding departure time (x_i) are presented in Tables 5.5 and 5.6. The cost or fitness value of the objective function which represents the class of single-stage hybrid system is recorded for every 500 generations and their statistical analyses are compared in Tables 5.1–5.4.

All the 12 methods are executed through visual basic program and the fitness values for all the methods are taken by running the simulation 1000 times at different times.

Table 5.2. Statistical analyses of various methods at the 1000th generation.

Method number	Method	Average	Best	Worst	SD	CV	Avedev
1	WA	11983.6000	10168.3405	13881.3154	170.9860	0.0143	122.9995
2	WB	7814.2846	7755.6382	8033.7556	20.1515	0.0026	11.3079
3	WC	7820.6383	7764.4469	8097.1594	12.0986	0.0015	2.8512
4	XA	13441.9106	10168.3405	16386.9300	78.3690	0.0058	46.8587
5	XB	7921.0846	7772.2090	9475.1826	70.5600	0.0089	10.7831
6	XC	8083.1708	7781.7568	9239.8054	63.8962	0.0079	30.9782
7	YA	12383.4070	10567.3418	13685.8476	74.8552	0.0060	44.3305
8	YB	7765.2765	7744.6986	7806.4233	1.8247	0.0002	1.2557
9	YC	7768.6854	7742.2869	7801.9017	3.0599	0.0004	1.8542
10	ZA	11172.4901	9643.2855	13158.6611	108.8048	0.0097	104.0414
11	ZB	7778.0332	7744.3391	7825.2758	7.2364	0.0009	1.6913
12	ZC	7778.1319	7744.8737	7837.4551	2.7583	0.0004	1.8686

Table 5.3. Statistical analyses of various methods at the 1500th generation.

Method number	Method	Average	Best	Worst	SD	CV	Avedev
1	WA	8608.4865	8091.6661	9669.7744	45.5096	0.0053	27.0239
2	WB	7789.7420	7748.8893	7862.9554	5.4448	0.0007	3.5318
3	WC	7797.7273	7749.4070	7884.1366	3.7937	0.0005	1.9491
4	XA	10545.2723	8091.6661	13129.3045	99.6716	0.0095	50.2086
5	XB	7778.7645	7744.7194	7988.4480	11.3561	0.0015	2.4609
6	XC	7792.4137	7748.8792	8085.4559	19.0309	0.0024	7.5050
7	YA	8749.2009	8201.6420	9651.5879	58.3183	0.0067	38.0115
8	YB	7753.3627	7740.4907	7782.9877	1.7317	0.0002	0.8079
9	YC	7754.8549	7738.6329	7774.2684	1.7599	0.0002	1.1162
10	ZA	8859.8511	8170.7338	9804.0423	14.0901	0.0016	15.3803
11	ZB	7759.7919	7739.6439	7792.9689	0.7206	0.0001	0.7065
12	ZC	7759.8744	7740.0971	7783.7389	1.6642	0.0002	0.7627

The *average values* (Mean), best and worst fitness values among 1000 simulated results and the *standard deviations* (SD) of the fitness values for each method are calculated. In order to strengthen the comparison, few more statistics tests are conducted, the *coefficient variance*, which is calculated from the ratio of standard deviation to the mean and *the average deviation*, which will, give the average of the absolute deviation of the

TABLE 5.4. Statistical analyses of various methods at the 2000th generation.

Method number	Method	Average	Best	Worst	SD	CV	Avedev
1	WA	7846.1401	7746.8580	8271.0692	18.9162	0.0024	6.4859
2	WB	7778.5150	7745.7858	7851.7101	3.2332	0.0004	2.1306
3	WC	7785.0885	7743.6033	7878.1134	3.2447	0.0004	0.8094
4	XA	8775.0653	7760.2905	9929.4221	185.4796	0.0211	100.8503
5	XB	7756.7710	7737.9135	7789.3040	1.9710	0.0003	0.9622
6	XC	7760.4903	7739.8483	7892.4369	7.9065	0.0010	1.8714
7	YA	7814.6565	7763.5000	7937.2679	8.9888	0.0012	5.1735
8	YB	7747.9780	7734.8516	7765.4052	0.7423	0.0001	0.3795
9	YC	7749.1308	7736.5328	7766.8408	1.6399	0.0002	1.0236
10	ZA	7935.2318	7802.2804	8476.1164	20.9959	0.0026	4.8424
11	ZB	7750.9974	7735.9609	7770.2229	0.3444	0.0000	0.2394
12	ZC	7751.6891	7738.2488	7771.5835	1.3291	0.0002	0.5233

fitness values from their mean, which are taken in 1000 simulation runs. Added to these analyses, *hypothesis t-test: analysis of variance* (ANOVA) also carried out to validate the efficacy among the proposed algorithms. These statistics analysis are presented in Tables 5.1–5.4. The graphical analysis is done through Box plot, which is shown in Figure 5.3.

In order to ease the analysis, all the 12 methods are compared with respect to group 1 and group 2 classification. That is, all the methods in group 1 are compared with each of the inertia weight given in group 2 and vice versa. The comparisons of fitness values between each method are done in 3 dimensional plots using MATLAB. And they are shown in Figures 5.1 and 5.2.

The optimal control variables $(u_1, u_2, \dots, u_{50})$ in (5.2) and the departure time of each job $(x_1, x_2, \dots, x_{50})$ in (5.3) are computed for all 12 methods and tabulated in Tables 5.5 and 5.6, respectively. From the departure time of each job, the queue lengths of the server (of the single-stage hybrid system) at the arrival times $(a_1, a_2, \dots, a_{50})$ are computed and plotted in Figure 5.4 for all the 12 methods individually.

The dynamic behavior of each particle in the search space for each method with $N = 50$ is taken over 2000 generations and the particles positions are recorded and presented in Table 5.7 and Figure 5.5. The particle positions for the methods which are with W and A are moving away form the equilibrium point (the position where the optimal solution is obtained) often and takes a lot of generations to settle whereas in methods comprising of Y, Z and B (sometimes C), the particle positions are always in a closer range of the equilibrium point and converge faster.

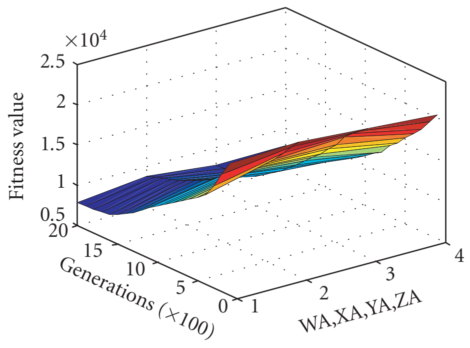
The execution times for each method are calculated for every simulation and hence the average execution time is calculated and presented in Table 5.8 and Figure 5.6, from which it is understood that methods comprises of Y and Z are yield the optimal solution faster with less execution time.

Table 5.5. Comparison of optimal control variable (u_i) for all 12 methods.

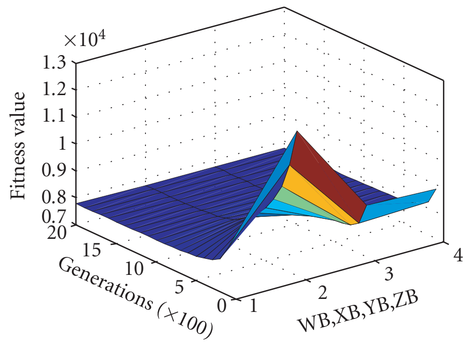
	Arrival		W			X			Y			Z	
	a(i)	A	B	C	A	B	C	A	B	C	A	B	C
1	0.3	0.241	0.233	0.213	0.234	0.259	0.230	0.221	0.249	0.211	0.271	0.232	0.213
2	0.5	0.215	0.246	0.207	0.272	0.240	0.367	0.257	0.290	0.227	0.243	0.216	0.200
3	0.7	0.385	0.282	0.287	0.215	0.249	0.213	0.234	0.215	0.265	0.252	0.257	0.347
4	1	0.201	0.277	0.296	0.247	0.253	0.191	0.293	0.246	0.298	0.276	0.295	0.240
5	1.3	0.225	0.177	0.249	0.271	0.204	0.244	0.240	0.251	0.259	0.348	0.258	0.248
6	1.5	0.269	0.246	0.230	0.184	0.241	0.207	0.234	0.247	0.265	0.147	0.243	0.254
7	1.7	0.216	0.260	0.231	0.344	0.273	0.291	0.286	0.269	0.210	0.249	0.265	0.269
8	2	0.251	0.272	0.287	0.299	0.295	0.271	0.235	0.234	0.266	0.243	0.247	0.230
9	2.3	0.252	0.221	0.303	0.224	0.286	0.202	0.241	0.237	0.289	0.265	0.255	0.219
10	2.5	0.226	0.224	0.217	0.296	0.230	0.305	0.319	0.272	0.249	0.259	0.214	0.230
11	2.7	0.286	0.281	0.203	0.143	0.236	0.256	0.227	0.266	0.234	0.190	0.300	0.284
12	3	0.245	0.279	0.275	0.258	0.233	0.224	0.232	0.225	0.227	0.262	0.218	0.269
13	3.3	0.192	0.230	0.243	0.190	0.211	0.203	0.236	0.257	0.211	0.302	0.256	0.228
14	3.5	0.244	0.234	0.218	0.189	0.261	0.226	0.260	0.215	0.269	0.179	0.208	0.205
15	3.7	0.252	0.230	0.325	0.365	0.246	0.362	0.239	0.243	0.255	0.221	0.319	0.268
16	4	0.300	0.301	0.220	0.257	0.284	0.209	0.237	0.285	0.265	0.340	0.217	0.299
17	4.3	0.212	0.267	0.217	0.393	0.240	0.262	0.271	0.219	0.210	0.245	0.242	0.237
18	4.5	0.227	0.243	0.267	0.295	0.236	0.260	0.253	0.312	0.230	0.132	0.225	0.230
19	4.7	0.297	0.261	0.249	0.188	0.227	0.243	0.272	0.238	0.305	0.316	0.247	0.249
20	5	0.263	0.230	0.261	0.188	0.295	0.233	0.200	0.231	0.255	0.264	0.285	0.285
21	5.3	0.249	0.204	0.235	0.223	0.234	0.206	0.271	0.230	0.236	0.215	0.228	0.213
22	5.5	0.242	0.211	0.246	0.199	0.261	0.197	0.271	0.240	0.243	0.231	0.228	0.271
23	5.7	0.223	0.334	0.264	0.210	0.236	0.324	0.203	0.262	0.247	0.253	0.274	0.268
24	6	0.292	0.246	0.254	0.425	0.270	0.272	0.255	0.268	0.274	0.317	0.270	0.247
25	6.3	0.229	0.285	0.232	0.154	0.272	0.228	0.203	0.205	0.230	0.229	0.276	0.202
26	6.5	0.287	0.274	0.199	0.145	0.231	0.206	0.261	0.249	0.231	0.213	0.221	0.207
27	6.7	0.250	0.230	0.314	0.662	0.256	0.301	0.252	0.247	0.256	0.259	0.260	0.331
28	7	0.231	0.209	0.261	0.158	0.239	0.282	0.285	0.299	0.284	0.296	0.244	0.261
29	7.3	0.244	0.225	0.258	0.220	0.219	0.235	0.230	0.245	0.218	0.245	0.245	0.206
30	7.5	0.246	0.256	0.253	0.186	0.241	0.242	0.257	0.260	0.201	0.217	0.212	0.272
31	7.7	0.263	0.258	0.224	0.181	0.240	0.252	0.212	0.259	0.295	0.338	0.296	0.272
32	8	0.249	0.263	0.261	0.191	0.301	0.254	0.331	0.236	0.285	0.193	0.247	0.251
33	8.3	0.201	0.202	0.232	0.507	0.226	0.262	0.200	0.248	0.243	0.218	0.232	0.212
34	8.5	0.308	0.266	0.234	0.217	0.243	0.255	0.251	0.219	0.255	0.231	0.237	0.233
35	8.7	0.233	0.322	0.300	0.202	0.300	0.255	0.250	0.260	0.257	0.290	0.251	0.275
36	9	0.253	0.209	0.231	0.173	0.234	0.240	0.275	0.272	0.246	0.265	0.280	0.280
37	9.3	0.260	0.241	0.200	0.210	0.222	0.222	0.246	0.235	0.236	0.220	0.221	0.227
38	9.5	0.229	0.224	0.385	0.409	0.224	0.288	0.241	0.240	0.231	0.237	0.235	0.234
39	9.7	0.281	0.240	0.235	0.224	0.297	0.231	0.249	0.256	0.281	0.234	0.244	0.244
40	10	0.233	0.298	0.248	0.347	0.253	0.247	0.258	0.269	0.251	0.311	0.300	0.295
41	10.3	0.252	0.253	0.253	0.268	0.266	0.251	0.258	0.255	0.253	0.213	0.249	0.254
42	10.5	0.272	0.264	0.260	0.344	0.270	0.273	0.264	0.269	0.266	0.321	0.272	0.271
43	10.7	0.256	0.275	0.281	0.294	0.284	0.282	0.307	0.284	0.304	0.275	0.298	0.290
44	11	0.304	0.299	0.306	0.285	0.311	0.308	0.425	0.304	0.309	0.279	0.299	0.301
45	11.3	0.337	0.326	0.360	0.355	0.328	0.342	0.336	0.325	0.327	0.331	0.332	0.330
46	11.5	0.376	0.373	0.363	0.338	0.365	0.342	0.350	0.363	0.368	0.503	0.366	0.380
47	11.7	0.385	0.405	0.406	0.458	0.426	0.405	0.365	0.405	0.403	0.418	0.408	0.398
48	12	0.448	0.466	0.489	0.395	0.450	0.467	0.431	0.470	0.469	0.533	0.468	0.460
49	12.3	0.564	0.567	0.566	0.785	0.550	0.566	0.631	0.576	0.584	0.628	0.558	0.567
50	12.5	0.833	0.929	0.741	0.900	0.770	0.773	0.733	0.819	0.795	0.597	0.819	0.795

TABLE 5.6. Comparison of departure time (x_i) of each job.

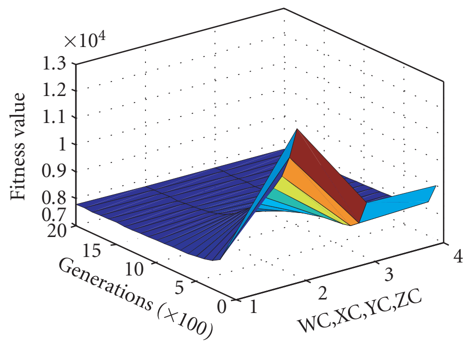
	Arrival	W			X			Y			Z		
	a(i)	A	B	C	A	B	C	A	B	C	A	B	C
1	0.3	0.541	0.533	0.513	0.534	0.559	0.530	0.521	0.549	0.511	0.571	0.532	0.513
2	0.5	0.756	0.779	0.720	0.806	0.799	0.897	0.778	0.840	0.737	0.814	0.748	0.713
3	0.7	1.141	1.061	1.007	1.021	1.048	1.110	1.012	1.054	1.002	1.066	1.005	1.060
4	1	1.342	1.338	1.304	1.268	1.302	1.301	1.306	1.300	1.300	1.341	1.300	1.300
5	1.3	1.567	1.515	1.553	1.571	1.506	1.545	1.545	1.551	1.559	1.689	1.558	1.548
6	1.5	1.836	1.761	1.783	1.754	1.747	1.751	1.780	1.798	1.824	1.836	1.801	1.801
7	1.7	2.053	2.022	2.014	2.098	2.020	2.042	2.066	2.067	2.034	2.085	2.066	2.070
8	2	2.304	2.293	2.301	2.397	2.315	2.313	2.301	2.301	2.300	2.328	2.313	2.300
9	2.3	2.556	2.521	2.605	2.621	2.601	2.515	2.541	2.538	2.589	2.593	2.569	2.519
10	2.5	2.781	2.745	2.821	2.916	2.831	2.821	2.860	2.810	2.838	2.852	2.783	2.748
11	2.7	3.068	3.026	3.024	3.060	3.066	3.076	3.088	3.075	3.073	3.042	3.082	3.032
12	3	3.312	3.305	3.299	3.318	3.300	3.300	3.320	3.300	3.300	3.304	3.300	3.301
13	3.3	3.504	3.535	3.543	3.508	3.511	3.504	3.556	3.557	3.511	3.606	3.556	3.529
14	3.5	3.748	3.769	3.761	3.698	3.771	3.730	3.816	3.772	3.780	3.785	3.764	3.733
15	3.7	4.000	3.999	4.086	4.065	4.018	4.092	4.055	4.015	4.035	4.005	4.083	4.001
16	4	4.300	4.301	4.306	4.322	4.301	4.301	4.291	4.300	4.300	4.345	4.300	4.300
17	4.3	4.513	4.568	4.523	4.715	4.542	4.564	4.571	4.519	4.510	4.590	4.543	4.538
18	4.5	4.740	4.811	4.789	5.010	4.778	4.824	4.824	4.831	4.740	4.722	4.768	4.767
19	4.7	5.037	5.072	5.038	5.198	5.005	5.067	5.096	5.069	5.045	5.038	5.015	5.017
20	5	5.299	5.302	5.299	5.386	5.300	5.300	5.296	5.300	5.300	5.302	5.300	5.302
21	5.3	5.549	5.506	5.535	5.609	5.534	5.506	5.571	5.531	5.536	5.517	5.528	5.514
22	5.5	5.791	5.718	5.782	5.808	5.795	5.703	5.842	5.771	5.778	5.748	5.756	5.785
23	5.7	6.014	6.052	6.046	6.018	6.031	6.028	6.045	6.032	6.026	6.002	6.031	6.053
24	6	6.306	6.297	6.300	6.442	6.301	6.300	6.300	6.300	6.300	6.318	6.300	6.300
25	6.3	6.535	6.585	6.532	6.596	6.573	6.528	6.503	6.505	6.530	6.547	6.576	6.502
26	6.5	6.822	6.859	6.732	6.741	6.804	6.734	6.765	6.754	6.761	6.760	6.796	6.708
27	6.7	7.071	7.089	7.045	7.403	7.060	7.035	7.016	7.001	7.018	7.019	7.056	7.039
28	7	7.302	7.299	7.307	7.561	7.299	7.317	7.301	7.300	7.301	7.315	7.300	7.300
29	7.3	7.546	7.525	7.564	7.781	7.519	7.552	7.531	7.545	7.520	7.561	7.545	7.506
30	7.5	7.792	7.781	7.817	7.967	7.760	7.793	7.788	7.805	7.720	7.778	7.757	7.778
31	7.7	8.056	8.039	8.042	8.148	8.000	8.046	8.001	8.064	8.015	8.116	8.053	8.049
32	8	8.305	8.302	8.303	8.338	8.301	8.300	8.332	8.300	8.300	8.309	8.300	8.300
33	8.3	8.506	8.503	8.535	8.845	8.527	8.562	8.531	8.548	8.543	8.527	8.532	8.512
34	8.5	8.814	8.770	8.769	9.062	8.770	8.817	8.782	8.768	8.798	8.758	8.769	8.745
35	8.7	9.047	9.092	9.069	9.264	9.070	9.072	9.033	9.028	9.055	9.047	9.020	9.020
36	9	9.300	9.301	9.300	9.437	9.305	9.312	9.308	9.300	9.301	9.312	9.300	9.300
37	9.3	9.560	9.542	9.500	9.647	9.526	9.534	9.553	9.535	9.536	9.532	9.521	9.527
38	9.5	9.789	9.767	9.885	10.056	9.751	9.821	9.794	9.775	9.768	9.769	9.756	9.761
39	9.7	10.071	10.007	10.120	10.280	10.047	10.053	10.044	10.031	10.049	10.003	10.000	10.005
40	10	10.303	10.305	10.368	10.627	10.300	10.300	10.301	10.300	10.300	10.314	10.300	10.300
41	10.3	10.555	10.558	10.620	10.895	10.566	10.551	10.559	10.555	10.553	10.527	10.549	10.554
42	10.5	10.827	10.822	10.881	11.239	10.836	10.824	10.823	10.823	10.819	10.848	10.821	10.825
43	10.7	11.083	11.097	11.162	11.533	11.120	11.106	11.130	11.108	11.122	11.123	11.119	11.115
44	11	11.387	11.396	11.468	11.819	11.431	11.413	11.556	11.412	11.431	11.402	11.419	11.415
45	11.3	11.724	11.722	11.829	12.174	11.759	11.755	11.892	11.737	11.758	11.733	11.751	11.745
46	11.5	12.100	12.095	12.191	12.512	12.124	12.097	12.242	12.100	12.126	12.236	12.116	12.124
47	11.7	12.484	12.500	12.598	12.971	12.550	12.501	12.606	12.506	12.529	12.654	12.524	12.522
48	12	12.933	12.966	13.086	13.366	12.999	12.969	13.037	12.976	12.998	13.187	12.992	12.982
49	12.3	13.497	13.533	13.652	14.151	13.549	13.534	13.668	13.552	13.583	13.815	13.550	13.549
50	12.5	14.330	14.462	14.393	15.050	14.319	14.307	14.401	14.371	14.378	14.412	14.369	14.344



(a)



(b)



(c)

FIGURE 5.1. Three-dimensional plot of the comparison between methods. (a) WA, XA, YA, and ZA; (b) WB, XB, YB, and ZB; (c) WC, XC, YC, and ZC.

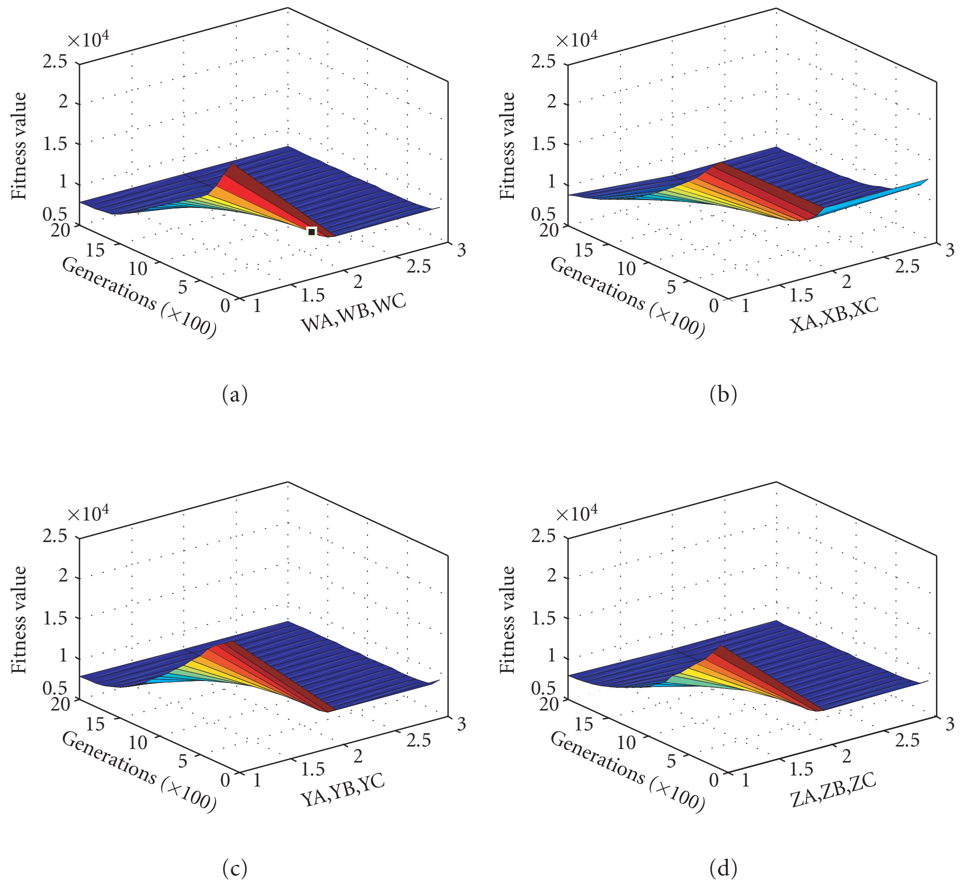


FIGURE 5.2. Three-dimensional plot of the comparison between methods. (a) WA, WB, and WC; (b) XA, XB, and XC; (c) YA, YB, and YC; (d) ZA, ZB, and ZC.

6. Discussions and conclusion

In this paper, new variants to the inertia weight and velocity equations are considered and they are classified into 12 methods. In order to compare the validity and usefulness of the proposed velocity equations in PSO methods with the existing standard PSO (method WA), all the methods are simulated 1000 times at different periods of time, and 1000 simulated results for each method are taken at different timings. The performance of different algorithms is compared with respect to the solution accuracy in the fitness, the standard deviations, co-efficient variance, average deviation, ANOVA t-test, and the percentage of deviation in the fitness from the proposed best method.

From the results stated in Table 5.1–5.4, it is obvious that the method 8 (*YB*) is the best followed by method 9 (*YC*). This clearly establishes the fact that method Y with B and C yields better solutions. This is the most significant outcome of the experiments

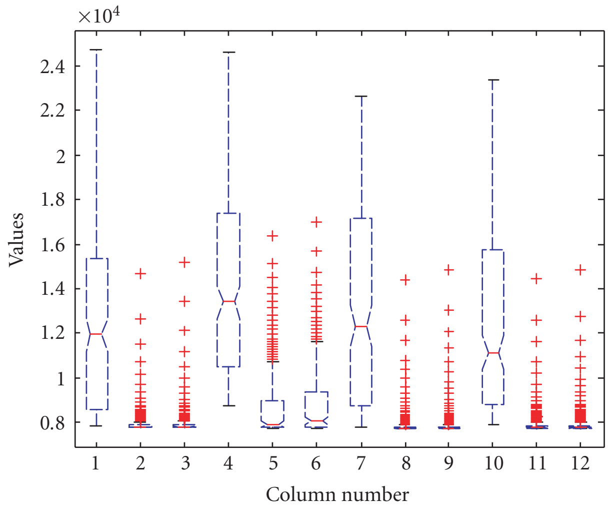


FIGURE 5.3. ANOVA t-test analysis of fitness values over 2000 generations for all 12 methods.

performed. These combinations have been shown to work efficiently with regard to an optimal control problem here but it is believed that these might be equally efficient with regard to all other problems where PSO can be used. All the 12 methods are sorted on the average fitness value, and their rankings are as follows: YB, YC, ZB, ZC, XB, XC, WB, WC, YA, WA, ZA, and XA. From the above rankings, it is very obvious that the methods comprise of Y and B always provide better results compared to other methods. It is also observed that except methods ZA and XA all other methods are better than the standard PSO, which is known here as method WA.

By analyzing the PSO parameters, it is observed that normally the cognitive (c_1) and social (c_2) parameters are set to 2 for a better convergence. This is proved in the proposed methods. In method X, these two constants (c_1 and c_2) are always greater than or equal to 2 whereas they (c_1 and c_2) are always less than or equal to 2 in method Z, since $\text{pbest}_i \geq \text{gbest}$, for $i = 1,2,\dots,50$. In method Y, they (c_1 and c_2) are always equal to 2. PSO algorithm with method Y always yields better result followed by those comprise of method Z with respect to the optimal fitness solution. It is also observed from the Table 5.1–5.4, method Z is always yielding the optimal solution with less standard deviation, which proves this method's accuracy and consistency. So from consistency point of view, PSO algorithm with method Z is a better choice than the one with method Y.

From the hypothesis ANOVA t-test that is shown in Figure 5.3, it can be easily concluded that method YB (refers to method 8 in the figure) is better than any other methods considered in this paper. This box plot representation provides an excellent visual summary of many important aspects of a distribution of fitness value over 2000 generations. The graphical view clearly shows the effectiveness of the proposed algorithms and their fitness distribution.

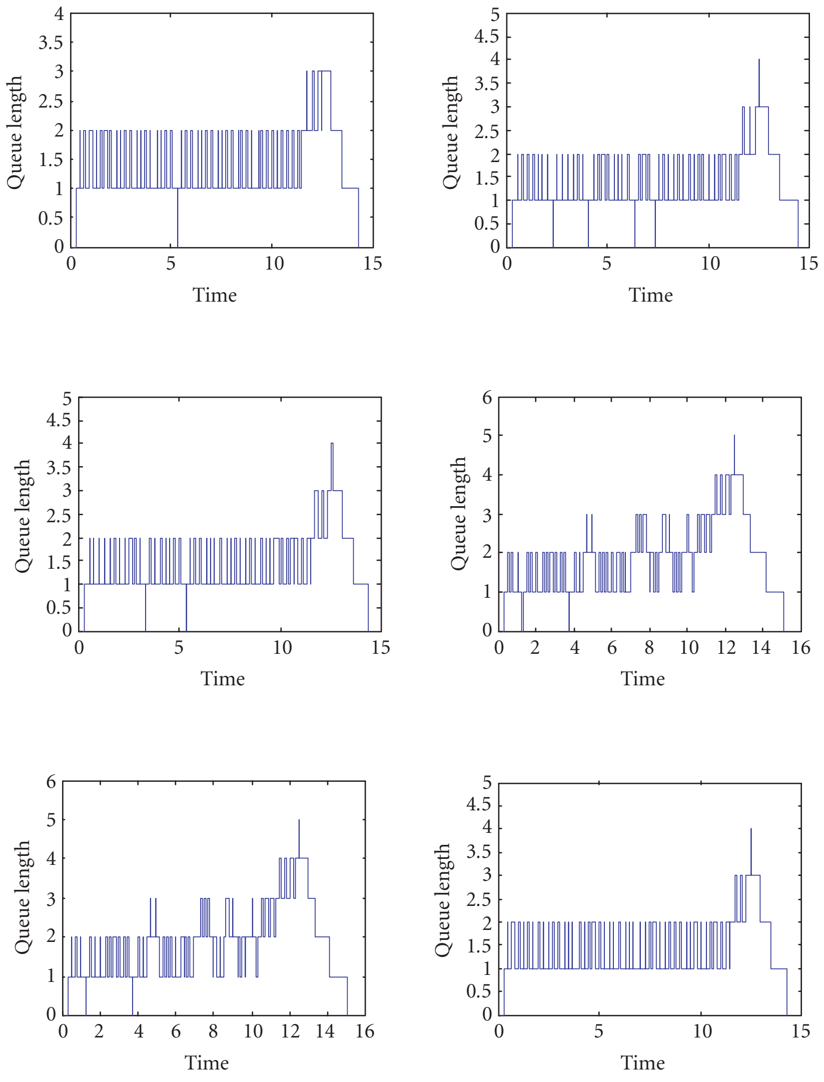


FIGURE 5.4. Queue length versus arrival time for methods WA, WB, WC, XA, XB, XC, YA, YB, YC, ZA, ZB, and ZC, respectively.

Figure 5.4 provides the information about the queue length in the server at the arrival time of each job. The queue lengths obtained through methods YB, YC, ZB, and ZC are more or less the same and proves their superiority among the other methods.

Figure 5.5 presents the dynamic behavior of each particle in the search space for each method, which is taken over 2000 generations. The particle positions for the methods with W and A are moving away form the equilibrium point (the position where the

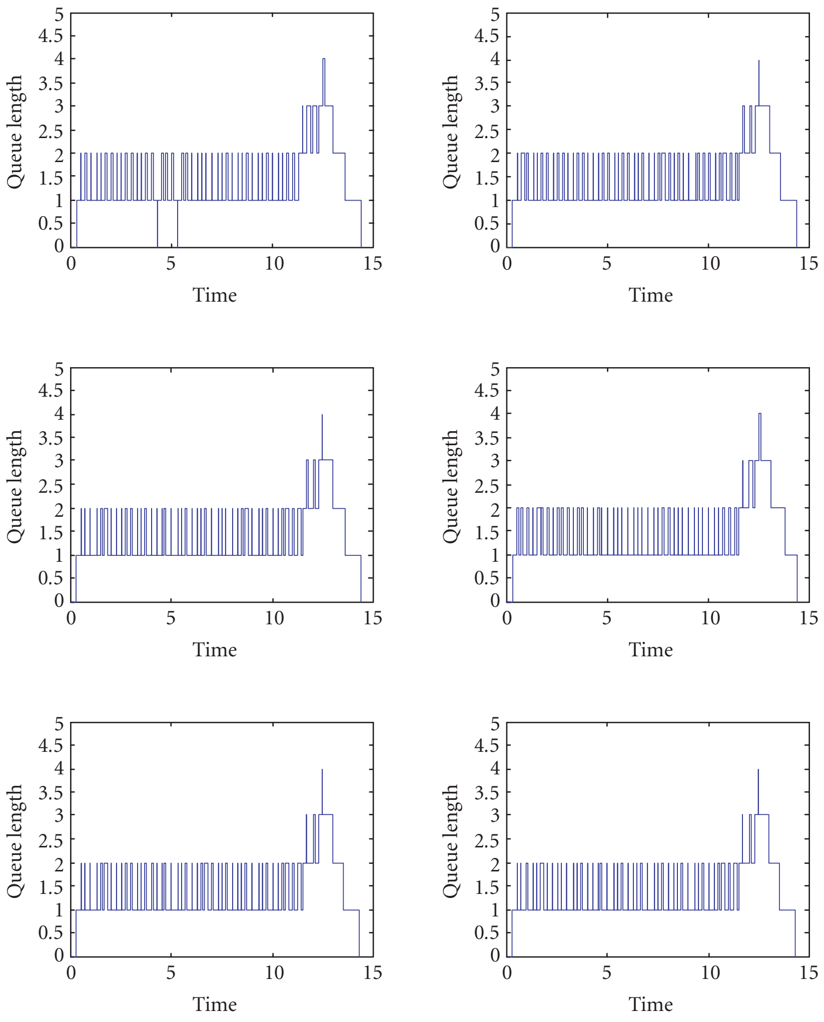
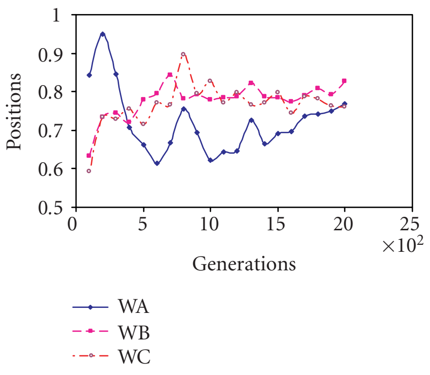


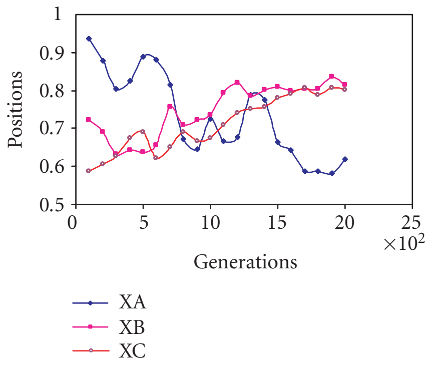
Figure 5.4. Continued.

optimal solution is obtained) often and takes a lot of generations to settle whereas in methods with Y, Z, and B (sometimes C), the particle positions are always in a closer range of the equilibrium point and converge faster. It is very obvious that the particle positions in methods YB, YC, ZB, and ZC are in a very narrow range, which implies how fast they are reaching the equilibrium point to obtain the optimal solution. This again proves the effectiveness and faster convergence capability of the methods YB, YC, ZB, and ZC.

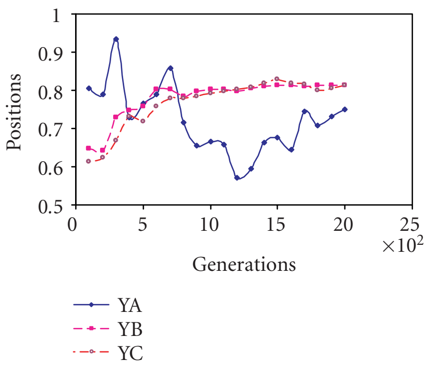
The most important aspect of simulation program is its execution time. Any algorithm, which runs at less execution time compared to other algorithms, is considered as best method. From Table 5.8 and Figure 5.6, it is clearly identified that among the 12



(a)

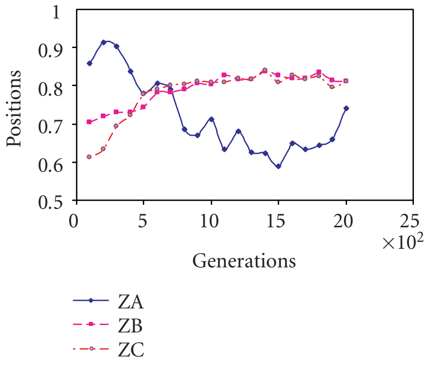


(b)

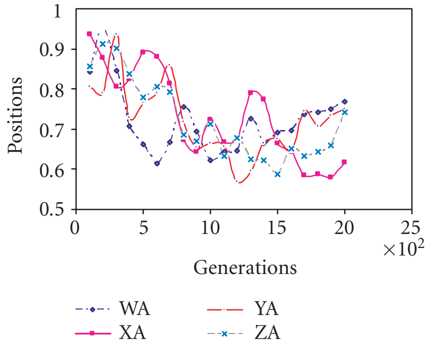


(c)

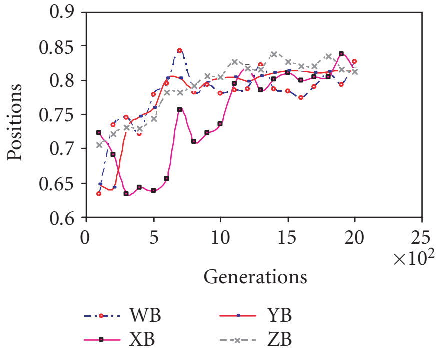
FIGURE 5.5. Comparison of the particle's best position for methods. (a) WA, WB, and WC; (b) XA, XB, and XC; (c) YA, YB, and YC; (d) ZA, ZB, and ZC; (e) WA, XA, YA, and ZA; (f) WB, XB, YB, and ZB; (g) WC, XC, YC, and ZC.



(d)

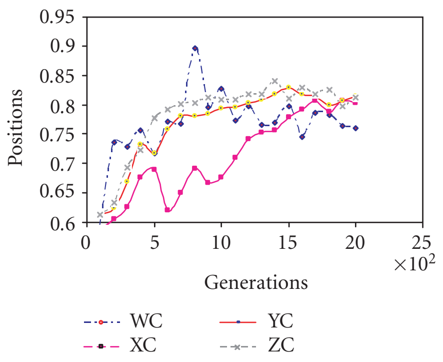


(e)



(f)

FIGURE 5.5. Continued.



(g)

Figure 5.5. Continued.

Table 5.7. Dynamic behavior of particles best position in each generation.

Gener-		W			X			Y			Z	
ation	A	B	C	A	B	C	A	B	C	A	B	C
100	0.8436	0.6335	0.5923	0.9359	0.7232	0.5864	0.8051	0.6480	0.6134	0.8576	0.7049	0.6132
200	0.9486	0.7343	0.7352	0.8787	0.6903	0.6064	0.7900	0.6432	0.6239	0.9131	0.7205	0.6343
300	0.8468	0.7452	0.7288	0.8049	0.6329	0.6262	0.9354	0.7285	0.6689	0.9022	0.7307	0.6931
400	0.7087	0.7218	0.7558	0.8257	0.6425	0.6756	0.7291	0.7470	0.7318	0.8378	0.7296	0.7233
500	0.6632	0.7795	0.7164	0.8899	0.6376	0.6901	0.7647	0.7589	0.7172	0.7805	0.7437	0.7767
600	0.6151	0.7946	0.7706	0.8803	0.6553	0.6207	0.7895	0.8030	0.7573	0.8050	0.7825	0.7918
700	0.6686	0.8427	0.7666	0.8139	0.7558	0.6506	0.8586	0.8021	0.7799	0.7918	0.7822	0.8016
800	0.7559	0.7813	0.8959	0.6729	0.7099	0.6915	0.7169	0.7847	0.7801	0.6869	0.7910	0.8037
900	0.6929	0.7935	0.7956	0.6442	0.7234	0.6673	0.6550	0.7971	0.7837	0.6709	0.8051	0.8120
1000	0.6236	0.7797	0.8266	0.7236	0.7357	0.6758	0.6649	0.8031	0.7932	0.7128	0.8045	0.8086
1100	0.6437	0.7848	0.7723	0.6670	0.7940	0.7102	0.6577	0.8036	0.7964	0.6327	0.8265	0.8077
1200	0.6469	0.7875	0.7971	0.6768	0.8190	0.7418	0.5704	0.7973	0.8023	0.6793	0.8175	0.8183
1300	0.7263	0.8211	0.7648	0.7893	0.7856	0.7521	0.5959	0.8064	0.8089	0.6248	0.8160	0.8173
1400	0.6646	0.7860	0.7702	0.7747	0.8018	0.7555	0.6643	0.8106	0.8175	0.6225	0.8376	0.8406
1500	0.6918	0.7839	0.7967	0.6650	0.8101	0.7791	0.6767	0.8144	0.8289	0.5890	0.8266	0.8096
1600	0.6970	0.7733	0.7459	0.6431	0.8001	0.7922	0.6459	0.8133	0.8176	0.6504	0.8198	0.8284
1700	0.7356	0.7896	0.7868	0.5862	0.8050	0.8067	0.7444	0.8106	0.8145	0.6328	0.8199	0.8171
1800	0.7424	0.8095	0.7819	0.5883	0.8042	0.7887	0.7080	0.8125	0.7998	0.6448	0.8341	0.8258
1900	0.7491	0.7927	0.7636	0.5809	0.8370	0.8066	0.7327	0.8130	0.8066	0.6591	0.8154	0.7967
2000	0.7688	0.8268	0.7605	0.6180	0.8141	0.8025	0.7502	0.8127	0.8137	0.7421	0.8127	0.8128

Table 5.8. Execution time for simulation of various PSO methods.

Method	WA	WB	WC	XA	XB	XC	YA	YB	YC	ZA	ZB	ZC
Execution time (in seconds)	7.35	7.3	7.33	6.89	6.74	6.77	6.52	6.37	6.4	6.71	6.58	6.67

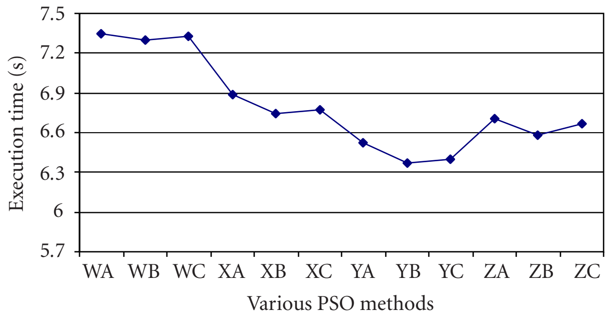


Figure 5.6. The average execution time of various PSO methods.

methods considered, three methods those comprise of method Y are having less execution time. This proves the superior capability of method Y. Algorithms with method Z are in the second rank, which are better than the methods X and W.

In summary, it is proven in so many aspects that the proposed methods, those with Y and also Z are much better than the existing methods and other proposed methods. Hence it can be concluded that the PSO algorithms with method Y is the superior among others and in particular it is more effective with the inertia weight which given in method B in computing the optimal control and fitness solution of the single-stage hybrid system.

References

[1] C. G. Cassandras, D. L. Pepyne, and Y. Wardi, *Optimal control of a class of hybrid systems*, IEEE Trans. Automat. Control **46** (2001), no. 3, 398–415.

[2] M. Clerc and J. Kennedy, *The particle swarm: Explosion, stability, and convergence in a multidimensional complex space*, IEEE Trans. Evol. Comput. **6** (2002), no. 1, 58–73.

[3] J. Kennedy, *The particle swarm: social adaptation of knowledge*, Proc. IEEE International Conference on Evolutionary Computation (Indianapolis, Indiana), IEEE Service Center, New Jersey, 1997, pp. 303–308.

[4] J. Kennedy and R. C. Eberhart, *Particle swarm optimization*, Proc. IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, New Jersey, 1995, pp. 1942–1948.

[5] S. Naka, T. Genji, T. Yura, and Y. Fukuyama, *A hybrid particle swarm optimization for distribution state estimation*, IEEE Trans. on Power Systems **18** (2003), no. 1, 60–68.

[6] D. L. Pepyne and C. G. Cassandras, *Modeling, analysis, and optimal control of a class of hybrid systems*, Discrete Event Dyn. Syst. **8** (1998), no. 2, 175–201.

[7] Y. Shi and R. C. Eberhart, *Empirical study of particle swarm optimization* , Proceedings of the Congress on Evolutionary Computation (Washington D.C., 1999), IEEE Service Center, New Jersey, 1999, pp. 1945–1950.

[8] P. Zhang and C. G. Cassandras, *An improved forward algorithm for optimal control of a class of hybrid systems*, IEEE Trans. Automat. Control **47** (2002), no. 10, 1735–1739.

M. Senthil Arumugam: Faculty of Engineering & Technology (FET), Multimedia University (Melaka Campus), Jalan Ayer Keroh Lama, Bukit Beruang, 75450 Melaka, Malaysia
E-mail address: msenthil.arumugam@mmu.edu.my

M. V. C. Rao: Faculty of Engineering & Technology, Multimedia University (Melaka Campus), Jalan Ayer Keroh Lama, Bukit Beruang, 75450 Melaka, Malaysia
E-mail address: machavaram.venkata@mmu.edu.my