Available online at www.sciencedirect.com
ScienceDirect
Computers & Industrial Engineering 53 (2007) 559–583
computers & industrial engineering
www.elsevier.com/locate/dsw

PSO-based algorithm for home care worker scheduling in the UK

Chananes Akjiratikarl [a], Pisal Yenradee [a,*], Paul R. Drake [b]

[a] *Industrial Engineering Program, Sirindhorn International Institute of Technology, Thammasat University, Pathum Thani 12121, Thailand*
[b] *E-Business and Operations Management Division, University of Liverpool Management School, Liverpool L69 7ZH, UK*

Received 6 December 2006; received in revised form 11 April 2007; accepted 8 June 2007
Available online 13 June 2007

Abstract

This paper presents the novel application of a collaborative population-based meta-heuristic technique called *Particle Swarm Optimization* (PSO) to the scheduling of home care workers. The technique is applied to a genuine situation arising in the UK, where the provision of community care service is a responsibility of the local authorities. Within this provision, optimization routes for each care worker are determined in order to minimize the distance traveled providing that the capacity and service time window constraints are not violated. The objectives of this paper are twofold; first to exploit a systematic approach to improve the existing schedule of home care workers, second to develop the methodology to enable the continuous PSO algorithm to be efficiently applied to this type of problem and all classes of similar problems. For this problem, a particle is defined as a multi-dimensional point in space which represents the corresponding care activities and assignment priority. The *Heuristic Assignment* scheme is specially designed to transform the continuous PSO algorithm to the discrete job schedule. The *Earliest Start Time Priority with Minimum Distance Assignment* (ESTPMDA) technique is developed for generating an initial solution which guides the search direction of the particle. *Local improvement procedures* (LIP), i.e. *insertion and swap*, are embedded in the PSO algorithm in order to further improve the solution quality. The proposed methodology is implemented, tested, and compared with existing solutions on a variety of real problem instances.

Keywords: Home care; Scheduling; Rostering; Meta-heuristic; Particle Swarm Optimization; Local improvement procedures; Heuristics

1. Introduction

Home care or domiciliary care is the provision of health care and assistance to people in their own homes, according to a formal assessment of their needs. In the UK, it is part of the community care service that is the responsibility of local government authorities. The aim is to provide the care and support needed to assist people, particularly elderly people, people with physical or learning disabilities and people who need assistance

* Corresponding author. Tel.: +66 2986 9009x2107; fax: +66 2986 9112.
E-mail addresses: chananes@hotmail.com (C. Akjiratikarl), pisal@siit.tu.ac.th (P. Yenradee), drake@liverpool.ac.uk (P.R. Drake).

0360-8352/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2007.06.002

due to illness to live as independently as possible in their own homes. Home care is a viable alternative to in-hospital, residential or institutional based nursing care and the underlying driver is that it leads to a higher quality of life for the clients as well as lower costs. Home care is primarily performed by means of personal visitations of care workers to clients in their homes, where they provide care assistance according to a pre-assessment of the clients' needs. The service includes not only simpler low-level tasks such as getting up, dressing, toileting, bathing, preparation of meals, housework, shopping, contact and befriending, but also high-level tasks such as assistance in taking medication and physical therapy, which require higher levels of skills.

In the past, in the UK home care was performed predominantly by care workers employed by the social services departments of local government authorities, i.e. in-house. However, local authorities are now increasingly outsourcing home care from the independent sector, with some outsourcing as much as 100%. This is being driven in the first instance by central government seeking to achieve *best value* where value is seen as the ratio of quality to cost. For an introduction to the drivers of the outsourcing of home care in the UK see (Drake & Davies, 2006). The World Health Organization stresses that strategies should be drawn up for providing support to patients and carers at the community level in order to avoid costly institutional care. This paper aims to make a contribution to this by developing an algorithm to optimize care-worker schedules (or rosters). The paper also demonstrates that the wealth of research into scheduling within the field of Industrial Engineering has the potential for wider application.

Care workers travel from their own homes to deliver care to their allocated clients at a specified time or within a specified time-window, and then return home after finishing their visits. The capacity of not more than 7.5 h of work per day per carer is imposed. The time-windows for each activity are determined during the formal process of assessing the client's needs. Generally, the time-windows of high-level tasks are tighter than those of low-level tasks, because high-level tasks are more critical.

This paper presents an efficient algorithm to schedule the dispatch of care workers to clients in an efficient manner under time and capacity constraints. Optimization techniques are developed to schedule the care workers on a daily basis to minimize the total distance traveled. The potential benefits of efficient scheduling are:

i. to reduce the traveling distance and hence traveling costs of the care workers;
ii. to improve worker utilization by reducing the 'waste' of travel and consequently reducing the number of workers required;
iii. to increase customer service by satisfying all service requirements within their specified time windows;
iv. to free-up care managers' time, so that they can undertake a more regulatory and strategic role.

This research has been conducted in collaboration with the *The Welsh Systems Consortium*; a partnership between seven local government authorities in Wales. They would welcome a scheduler as part of their fully integrated health and social care information system. Currently, local authorities typically use a manual approach to develop feasible and sound rosters. The Advanced Internet & Emergent Systems (AiMES) Centre at the University of Liverpool (www.aimes.net) has been providing help to the Consortium and identified the potential for improvement in the scheduling of care workers. AiMES applied the proprietary software ILOG™ Dispatcher (http://www.ilog.com) and utilized its embedding features to develop a scheduling engine. AiMES has experimented with the different 'solver' methods within ILOG™ to obtain the best results it can. By using a pre-defined first solution heuristic called *savings heuristic* to construct the initial feasible route and the neighborhoods heuristics (i.e. *2-opt*, *Or-opt*, *Relocate*, *Cross*, *Exchange*), the cost of the routes has been further improved, producing a considerable saving in distances traveled. The work presented here seeks to improve on the results obtained.

Home care worker scheduling is an extension of the vehicle routing problem with time windows (VRPTW) with limited route time, even though some specific characteristics are different. In general, the vehicle routing problem involves finding efficient routes for vehicles along a network to minimize or maximize a pre-specified objective function. VRPTW is a combinatorial optimization problem involving extremely large search spaces with correspondingly large numbers of potential solutions. Since the complexity class of the problem is NP-hard (non-deterministic polynomial-time hard), using an exact method is computationally inefficient. Instead,

a solution based on the *Particle Swarm Optimization* (PSO) meta-heuristic is proposed and adapted to solve the problem.

PSO is a population-based searching technique. Its development was based on observations of the social behavior of animals such as bird flocking, fish schooling, and swarm theory. Recently, PSO has been applied successfully in many areas such as continuous function optimization (Eberhart & Kennedy, 1995; Kennedy & Eberhart, 1995), flowshop scheduling (Tasgetiren, Liang, Sevkli, & Gencyilmaz, 2007), job shop scheduling (Lian, Gu, & Jiao, 2006), resource constrained project scheduling (Zhang, Li, & Tam, 2006), part-machine grouping (Andrés & Lozano, 2006), task allocation (Yin, Yu, Wang, & Wang, 2007), and neural network training (Shen, Shi, Yang, & Ye, 2006). PSO has many desirable characteristics:

i. the concept of PSO is very simple and it can be implemented easily for many applications;
ii. it is versatile, robust and general purpose in that it can be applied to similar versions of a problem with minor modification;
iii. it is computationally efficient in the sense that reasonable solutions can be obtained in short computational time;
iv. unlike genetic algorithms it involves only a few parameters so that it is easier to find the best combination of parameter values (Lian et al., 2006);
v. it is easily integrated with many local search techniques to improve the solution quality;
vi. the algorithm is well suited to parallelization by implementation on a cluster of workstations.

However, a drawback of PSO is fast convergence that can cause it to become trapped in local optima. This can be overcome by the application of a *local improvement procedure* (LIP). In PSO, the solution of each particle generally spreads around the search space. The local optima occur when each particle is trapped in its own area and better solution cannot be found. LIP enables each particle to fine-tuned search in its own local area and encourage particle to find a better solution, thus, escape from local optima.

Due to the continuous nature of PSO, a specially designed heuristic is necessary to apply PSO to the VRPTW. One of the contributions of this paper is the development of the problem mapping assignment technique, and solution generation to establish the care worker schedule. To the authors' knowledge, PSO is applied to solve this type of problem for the first time in the literature. Nevertheless, some heuristic techniques are known to apply to similar types of problems.

The remaining sections of this article are organized as follows. The next section gives some background on PSO and VRPTW techniques as well as a review of staff planning and scheduling, the application of PSO and previous solution techniques for related problems. Section 3 explains descriptions of the problem including the objective function and constraints. Section 4 explains the application of PSO to home care worker scheduling, followed by the computational experiments in Section 5. Finally, the concluding remarks and a direction for future research are presented in Section 6.

2. Literature review

2.1. Staff planning and scheduling problem

The research on staff planning and scheduling reported in the literature commonly relates to the determination of the number of staff with particular skills and allocation of staff according to demand. The objectives are, for example, to minimize costs, meet customers' demands, satisfy employee preferences or distribute work equally. Applications include airline crew scheduling (Goumopoulos & Housos, 2004; Yan & Tu, 2002), bus or truck driver scheduling (Bianco, Bielli, Mingozzi, Ricciardelli, & Spadoni, 1992), nurse scheduling (Cheang, Li, Lim, & Rodrigues, 2003), call-centre scheduling (Lin, Lai, & Hung, 2000). For further details of these problems, refer to Ernst, Jiang, Krishnamoorthy, and Sier (2004).

Research into staff planning for home care in Sweden has been reported by Everborn, Flisberg, and Rönnqvist (2006). The local authorities in Sweden are facing increasing costs due to the rise in the number of older people and those who require support. A decision support system called LAPS CARE has been developed to aid the staff-planning task. The *repeated matching* approach is used as an optimization routine to obtain a

solution for the routing problem. Considerable savings are reported in the planning time, the quality of routes and the quality of service.

2.2. Application of PSO to scheduling

PSO is an evolutionary computational algorithm developed originally by Eberhart and Kennedy (1995), see also Kennedy and Eberhart (1995). It is a population-based searching technique that simulates the social behavior of birds flocking or fish schooling. Each individual, called a particle, represents a point in the search space. The population, called a swarm, represents the set of points that are potential solutions. PSO is based on the interaction and the social communication of the group of particles. The particle flies iteratively through the search space by using the velocity function, which is constantly updated according to its own previous experience and the group's experience. Each particle tends to adjust the position toward its own previous best position and the group's previous best position. Tracking and memorizing the best positions encountered builds upon the particle's experience. PSO possesses a memory as every particle remembers the best position it has reached. PSO combines local search (through self-experience) with global search (through neighboring experience), attempting to balance exploration and exploitation.

Tasgetiren, Liang, Sevkli, and Gencyilmaz (2004) applied PSO to the single machine total weighted tardiness problem. A heuristic called the *smallest position value* (SPV) rule was developed to translate the continuous position value of PSO to a discrete job sequence. A local search procedure, called *variable neighborhood search* (VNS) further improved the performance of PSO. The proposed algorithm was tested against ant colony optimization (ACO) and iterative local search (ILS) and gave the best results for the benchmark problems considered. They concluded that PSO is as good as ACO and ILS.

For flowshop scheduling, Tasgetiren et al. (2007) used the same heuristic as for single machine scheduling (SPV and VNS). They showed that 57 out of 90 and 195 out of 800 best-known solutions were improved for total flowtime and makespan, respectively. Lian et al. (2006) embedded new crossover operators from a genetic algorithm in the original PSO and reported superior results over the standard genetic algorithm for the makespan objective. Liao, Tseng, and Luarn (2007) extended the discrete version of PSO for flowshop scheduling. The computational results showed advantages over the continuous-based PSO presented by Tasgetiren et al. (2007) and over genetic algorithms. Allahverdi and Al-Anzi (2006) applied PSO to assembly flowshop scheduling. They compared the performance of PSO with tabu search and the EST heuristic. Their computational analysis indicates that tabu search outperforms the others when the due-dates range is relatively wide, but PSO significantly outperforms the others for difficult problems, i.e. tight due-date ranges.

For job shop scheduling, Xia and Wu (2005) implemented the Hybrid PSO and Simulated Annealing algorithm. The results show that the proposed algorithm is a viable and effective approach for this type of problem, especially for large problem size.

For the Traveling Salesman Problem (TSP), a sub-problem of the vehicle routing problem, many versions of the PSO have been reported to generate satisfactory results. Pang, Wang, Zhou, and Dong (2004a) used fuzzy matrices in representing the position and velocity of the particle and Pang et al. (2004b) introduced a second modified PSO for TSP. They implemented the space transformation to map Cartesian continuous space onto permutation space and applied local search and chaotic operations to improve the solution quality. Lopes and Coelho (2005) presented a hybrid model based on PSO and fast local search for the blind traveling salesman problem. The order crossover concept from genetic algorithms (GAs) was used to move the particles across the search space.

For flexible manufacturing system scheduling, Jerald, Asokan, Prabaharan, and Saravanan (2005) compared a GA, simulated annealing, a memetic algorithm, and PSO for scheduling a flexible manufacturing system with multiple objective functions. They found PSO to be superior.

Zhang et al. (2006) applied PSO to resource-constrained project scheduling to minimize total project duration. The performance of PSO is compared with the three simple heuristics (minimum total float, shortest activity duration and minimum late finish time) and an evolutionary heuristic in the form of GA. The computation results indicate that PSO can obtain better results than the simple heuristic methods whilst achieving the same results as the GA, although PSO required fewer search iterations.

2.3. Previous solutions for VRPTW

VRPTW involves the determination of an efficient set of routes, all starting and ending at a central depot, for a fleet of vehicles intended to serve a given set of customers. All customers may be visited only once by only one vehicle. Each customer must be served within a specified time interval or window. A vehicle is not allowed to begin service after the time window's upper bound. A waiting time is incurred if a vehicle reaches the customer before the time window's lower bound. Each customer has a specified service time. The total route time of a vehicle is the sum of its travel times (which are proportional to the distances traveled), its waiting or idle times and the service times. The maximum route time should not exceed the maximum route time of each vehicle. The objective of VRPTW is to minimize the route length, the service cost, the travel time or the number of vehicles or a combination of these depending upon the particular application.

The current VRPTW solution techniques can be categorized as exact algorithms, construction and improvement heuristics or meta-heuristics. An intensive survey of previous articles on VRPTW can be found in Desrochers, Lentra, Savelsbergh, and Soumis (1988) and Solomon and Desrosiers (1988). Recently, Bräysy, Dullaert, and Gendreau (2004) provided a comprehensive survey and compared previous VRPTW applications based on evolutionary algorithms.

'Exact' algorithms for VRPTW have been investigated by many researchers. Kolen, Kan, and Trienekens (1987) introduced a branch and bound method for VRPTW. Desrochers, Desrosiers, and Solomon (1992) used the column generation method to solve linear programming relaxation of the set partitioning formulation of the VRPTW. Fisher, Jornsten, and Madsen (1997) presented an optimization algorithm based on *K*-tree relaxation and Lagrangian deposition methods.

Due to the massive computational requirement of exact algorithms, constructive heuristics have been introduced to 'build' the vehicle route. Solomon (1987) was the first to introduce a variety of route construction heuristics. His results show that a sequential time-space based insertion algorithm outperforms other techniques. While Potvin and Rousseau (1993) reported that the use of parallel construction philosophy can substantially improve Solomon's results. Solomon's test problems have been used by several researchers as standard benchmark problems for VRPTW. For the improvement heuristics, the algorithm based on edge exchange is suggested by Lin and Kernighan (1973). Potvin and Rousseau (1995) proposed a new 2-*opt* exchange heuristic. Savelsbergh (1990) introduced the local search technique based on the *k*-exchange concept.

Meta-heuristics have been applied by many researchers. The search schemes of meta-heuristics are mainly based on simulating nature and on artificial intelligence. The strategies explore the search space more thoroughly in order to avoid local optima. Meta-heuristics include GAs (Potvin & Bengio, 1996), simulated annealing (Chiang & Russell, 1996), tabu search (Potvin, Kervahut, Garcia, & Rousseau, 1996; Taillard, Badeau, Gendreau, Guertin, & Potvin, 1997) and parallel tabu search (Garcia, Potvin, & Rousseau, 1994). Some authors have reported the hybridization of meta-heuristics. Thangiah, Osman, and Sun (1994) introduced GenSAT which is the hybrid combination of GAs, simulated annealing, tabu search and local search techniques. The Route-Neighborhood-Based Two-Stage (RNETS) meta-heuristic was proposed by Hwa, Liu, and Shen (1999). The concepts of nested parallel route construction and end handling are introduced. Gambardella, Taillard, and Agazzi (1999) applied a multiple ant colony system in which the first colony minimizes the number of vehicles while the second colony minimizes the distance traveled. Tan, Lee, Zhu, and Ou (2001) also developed and enhanced various meta-heuristics including simulated annealing (with updated cooling scheme), a variant of tabu search ('strict' tabu) and a GA (with new crossover operations, hybrid hill-climbing and adaptive mutation). Homberger and Gehring (2005) presented a hybridization of two-phase meta-heuristics which combines (μ, λ) evolution strategy and tabu search heuristics.

In summary, the complexity of the VRPTW problem is NP-hard (Bouthillier & Crainic, 2005; Liu & Shen, 1999; Tan et al., 2001). The number of possible solutions for VRPTW grows exponentially with the problem size. Using an exact algorithm will lead to an excessive computational requirement. As an alternative, heuristic techniques that are fast and yield good quality solutions should be applied. PSO is such a technique that has many desirable characteristics as stated above and it performs well for scheduling when compared to various

other heuristics. This indicates that the PSO-based algorithm is potentially suitable to efficiently solve the home care worker scheduling problem.

3. Home-care worker scheduling

The problem is concerned with the dispatching of care workers on a daily basis to minimize the total distance traveled whilst satisfying all constraints. The problem description given below is derived from discussions with Ceredigion local authority, one of the members of the *Welsh Systems Consortium*, and an analysis of its home care data.

i. Care workers are scheduled according to the requirements of the clients, who may require more than one activity/visit per day. An example abstract of care requirements is given in Appendix A note the postcodes have been masked, as this data is confidential.
ii. Demand has substantial peaks during the morning and early evening.
iii. Each activity must be delivered within a specified time window and location.
iv. Each activity can involve only one visit by one care worker, i.e. no activity splitting is allowed. The maximum number of routes is equal to the number of care workers.
v. Care workers start from their homes and return after finishing all their assigned activities. The total traveling distance is the sum of the distance from the care worker's home to the first client, the distances between the successive clients and the distance from the last client back to the worker's home.
vi. For critical, medical activities the time window is a target time ± 5 min. For non-critical activities the window is ± 15 min. If a care worker arrives before a time window, the service cannot begin and a waiting-time is incurred. If a care worker arrives after time window (late), the solution is infeasible.
vii. The maximum capacity of each worker is 7.5 h per day, including travel time.
viii. In the model used here, each worker is assumed to be available 24 h per day, but can be used for only 7.5 h in that period.
ix. The travel speed of a care worker is assumed to be 30 miles per hour and traffic conditions are ignored (note, 30 miles per hour is the default urban speed limit in the UK).
x. Locations are represented by easting and northing coordinates for each postcode. The Euclidean or straight-line distance is assumed between locations.
xi. Some issues have been neglected in this first study, such as client-carer familiarity, skill-matching requirements, shift patterns and male/female preference for care workers.

This problem is an extension of the *Vehicle Routing Problem with Time Windows* (VRPTW). It differs from the standard vehicle routing problem in that all care workers (in other words, vehicles) start and end at their own home rather than a central, shared depot. The problem is equivalent to multiple depot vehicle routing in which the number of depots is equal to the number of vehicles. Additionally, there may be some important social issues involved in care worker scheduling, such as the shift pattern (minimize idle time) and worker-client familiarity. For a detailed description of the VRPTW notation and mathematical formulation used see Tan et al. (2001).

4. The PSO methodology

4.1. Particle Swarm Optimization (PSO) for care worker scheduling problem

PSO is a collaborative population-based search technique inspired by the simulation of the social behaviour of particles such as bird flocking or fish schooling in searching for their food. The pseudo-code of the algorithm used here is given below.

Pseudo code1: PSO-based algorithm

Start
Apply Initial Solution Heuristic ESTPMDA % see *Pseudo code3*
Initialize PSO parameters: Random x^0_{ijk} and translate to X^0_k matrix according to *ESTPMDA* heuristic
: *popsize*, *maxiter*, χ, w^0, c_1, c_2, v_{max}, V^0_k, $pbest^0_k$, $gbest^0$ for all k
LIP parameters: *nselect*, *numinsert*, *probaccept*
Do {
For{ *k = 1 to popsize*
Solution representation using Heuristic assignment % see *Pseudo code2*
Fitness value evaluation % calculate $f(Assign(X^t_k))$ using Eq. 3
Apply Local Improvement Procedures % see *Pseudo code4*
Update objective value after LIP, *pbest* and *gbest*
if $f(Assign(X^t_k)) < f(Assign(pbest^{t-1}_k))$
$pbest^t_k = x^t_k$,
if $f(Assign(pbest^t_k)) < f(Assign(\ pbest^{t-1}_{gbest^{t-1}}\ ))$
$gbest^t = k$.
Calculate velocity % using Eq. 4
Update position value % using Eq. 6
}End for
} while (termination)
End

Step 1. Initialization. PSO is initialized with a population of random particles (solutions) and velocities. Each particle is represented by a (*numcw***numj*) matrix, where *numcw* and *numj* are the number of care workers and the number of care activities or jobs, respectively. The columns denote care activities ranked by their start time whereas the rows refer to the care worker IDs. The particle is represented as $X^t_k = [x^t_{ijk}]_{numcw*numj*k}$, where x^t_{ijk} refers to the position value of care worker i for care activity j of particle k at iteration t and X^t_k is a position value matrix of particle k at iteration t. The population is the set of all particles. The set of particles at iteration t is represented by $X^t = \{X^t_1, X^t_2, X^t_3, \ldots, X^t_K\}$, where K is the population size. To aid understanding of the representation, Fig. 1 shows the matrix representation of particle X^t_k.

PSO starts by the generation of the continuous position values of each dimension using the following formula:

$$x^0_{ijk} = X_{min} + (X_{max} - X_{min}) \times U(0,1) \tag{1}$$

where, X_{min} and X_{max} are the pre-defined range of position values and $U(0,1)$ is a uniform random number in the range 0–1. The position value of each particle will be translated into the schedule using the *Heuristic*

$X^t_k =$

Care worker ID	Earliest Start Time (*EST*) Care Activities: 1	2	……	……	*numj*
1	$x^t_{1,1,k}$	$x^t_{1,2,k}$			$x^t_{1,numj,k}$
2	$x^t_{2,1,k}$	$x^t_{2,2,k}$			$x^t_{2,numj,k}$
:					:
numcw	$x^t_{numcw,1,k}$	$x^t_{numcw,2,k}$	…..	…..	$x^t_{numcw,numj,k}$

Fig. 1. The matrix representation of individual particle X^t_k.

$X^t_k =$

Care worker ID	*EST* care activities: J3	J5	J4	J1	J2	J6
Cw1	1.8	0.8	5.1	3.5	1.9	2.4
Cw2	4.5	3.2	2.8	4.6	3.7	1.5
Cw3	3.6	2.9	1.3	0.2	2.6	5.1

Fig. 2. Position value matrix of particle kth at iteration t (X^t_k).

$AP^t_k =$

Priority	*EST* care activities: J3	J5	J4	J1	J2	J6
1st	Cw1	Cw1	Cw3	Cw3	Cw1	Cw2
2nd	Cw3	Cw3	Cw2	Cw1	Cw3	Cw1
3rd	Cw2	Cw2	Cw1	Cw2	Cw2	Cw3

Fig. 3. Assignment priority matrix of particle X^t_k (AP^t_k).

Assignment technique, which is explained in the next section. The initial solution heuristic (*ESTPMDA*) is applied to all particles in establishing the initial solutions. The random position values obtained from Eq. (1) are mapped according to the solution from the *ESTPMDA* heuristic in the first iteration.

Particles fly towards the new search space by using the velocity function. The initial velocities are generated randomly according to the following formula:

$$v^0_{ijk} = V_{min} + (V_{max} - V_{min}) \times U(0,1) \tag{2}$$

where, v^0_{ijk} is the corresponding velocity value of particle x^0_{ijk}, which is the uniform random number from $U[V_{min}, V_{max}]$. V_{min} and V_{max} are the pre-defined range of the velocity values, which are fixed for the whole population and all iterations and equal to $[-(X_{max} - X_{min})/4, (X_{max} - X_{min})/4]$.

Step 2. Heuristic Assignment Technique. In this application, *Heuristic Assignment* is tailored to decode the continuous nature of PSO into discrete scheduling. *Pseudo code2* along with an example illustrate the procedure of *Heuristic Assignment*. The position values of each particle (Fig. 2) are sorted into ascending order along the column matrix. In each column, the smaller the position value, the more attractive the care worker is for assignment to the care activity in that column. The position value matrix is translated into priority matrix (AP^t_k) in which the smaller value has the higher priority, as in Fig. 3.

	EST care activities: J3	J5	J4	J1	J2	J6
Care worker ID	Cw1	Cw1	Cw2	Cw3	Cw1	Cw2

Fig. 4. Actual assignment after feasibility checking.

$X^t_k =$

Care worker ID	*EST* care activities: J3	J5	J4	J1	J2	J6
Cw1	1.8	0.8	5.1	3.5	1.9	2.4
Cw2	4.5	3.2	1.3	4.6	3.7	1.5
Cw3	3.6	2.9	2.8	0.2	2.6	5.1

Fig. 5. Position value matrix of particle X^t_k after *repairing*.

Pseudo code2: Heuristic Assignment Technique

```
k= 0, u = 0
Do{ k = k+1
    Sort X^t_k and obtain priority matrix AP^t_k
    j=0
    Do {j = j+1
        priority = 0
        Do{     priority = priority + 1
                activity = EST(j)                          % from priority matrix AP^t_k:
                candidate care worker = AP^t_k (priority, activity)
                Assign candidate activity to candidate care worker
                    If (capacity >= capacity limit)        % feasibility Checking
                        then feasible = false
                    Else
                        If (time windows conflict with already assigned activity)
                            then move candidate activity itself (within time windows) or
                                move itself and other assigned activities
                            If (no time conflict) feasible = true
                            If (time conflict) feasible = false
                        Else
                            feasible = true
                        End If
                    End If
        }while (priority < numcw and feasible = false)
        If (feasible = true) then
            Assign activity to candidate care worker
            Update occupied time and care worker capacity
            If (priority ≠ 1)                  % apply Repairing Algorithm
              then
                a = X^t_k(1, activity)         % get the position value of the highest priority care worker
                b = X^t_k(priority, activity)  % get the position value of the actual assigned care worker
                X^t_k(1, activity) = b
                X^t_k(priority, activity) = a
            End If
        Else                                   % feasible = false
            unassignlist(u) = activity,  u = u+1
        End If
        }while(j <= numjob)
}while(k <= popsize)
```

The assignment of care activities to care workers is performed on a one-by-one basis. Each care activity is assigned to the care worker with the highest priority at its ideal start time. The feasibility checking of time windows and capacity is performed during the assignment. During the feasibility checking, the candidate activity must be able to start within its time windows, the service time of the assigned care worker must not overlap with the previous assignment, and the total service time must not exceed the capacity

limit (7.5 h). If the total service time exceeds the capacity limit, the candidate activity is assigned to the care worker with the next highest assignment priority. When infeasibility occurs due to time windows, the problem is dealt with as follows. Firstly, the new candidate care activity is moved within the earliest and latest start time to avoid overlapping, and then re-checked for feasibility. In case that the candidate activity overlaps with the preceding activity, it will move itself backward to *tmove* time units (which is equal to the amount of overlapping time), or move forward otherwise. If moving remains infeasible, i.e. moving causes new overlapping with other job or the allowable time for moving is less than the overlapping time, the previously assigned care activities alone or both candidate and assigned care activities are moved within their time windows. In order to do so, the amount of required time unit ($tmove_{new}$) to overcome new overlapping will be calculated. The preceding or following care activities are moved according to $tmove_{new}$. Once again, if moving causes new overlapping time, $tmove_{new}$ are recalculated and the next preceding or following activity are moved according to the recalculated $tmove_{new}$. The moving procedure will terminate immediately when the schedule is feasible. Otherwise, the moving procedure will be continued until the first or last assigned activity is reached. Finally, if moving still violates the time windows constraint, the candidate activity is assigned to the next care worker. Once the care activity is assigned, the service time of the care worker is occupied by the service duration of that care activity. On the other hand, if the candidate care activity can not be assigned to any care worker routes, the activity will be placed in the unassigned activity list. The rest of the care activities are assigned sequentially in the same manner. The procedure is repeated for the whole population. An example of particle assignment is given in Fig. 4.

Repairing Algorithm. After all care activities have been assigned, the position values matrix is revised to be compatible with the new assignment. The interchange of position values is performed at this stage according to the rule that the assigned care worker must have highest priority or smallest position value. If the activity is not assigned to the care worker with the highest priority, position values are interchanged between the highest priority care worker and the actual assigned care worker. For the example in Figs. 3 and 4, care activity J4 is revised as highlighted in Fig. 5.

Step 3. Fitness value evaluation. The fitness value of each particle is calculated after the activity schedules have been constructed. Let $Assign(X_k^t)$ be the corresponding sequence at iteration t of particle X_k^t. The objective function $f(Assign(X_k^t))$ of all particles is calculated according to Eq. (3). This is the sum of the distances from the care workers' homes to their first clients, the distances between the successive clients, the distances from the last clients back to the workers' homes, and the penalty cost due to infeasibility assignment.

$$\text{Total distance} = \sum_{o=1}^{numcw}\sum_{j=1}^{numj} p_{cw(o),j} d_{cw(o),j} + \sum_{i=1}^{numj}\sum_{j=1_{j\neq i}}^{numj} p_{ij} d_{ij} + \sum_{o=1}^{numcw}\sum_{j=1}^{numj} p_{j,cw(o)} d_{j.cw(o)} + M \cdot numu \tag{3}$$

where: d_{ij} is the distance from client i's home to client j's;
$p_{ij} = 1$ if a journey is made from i to j and 0, otherwise;
$cw(o)$ is a care worker o;
numu is a number of activities in *unassignlist*;
$M =$ large number.

Step 4. Previous best particle and global best particle. With a flock of birds searching for food, each bird tries to follow the ones closest to the food source. Similarly, in PSO the past experience of particles is used to direct the search direction for the next iteration. Each particle memorizes its own personal best solution and the global best solution encountered is also memorized. In every iteration the position giving the best fitness value of the kth particle is recorded as the previous personal best solution and represented as $pbest_k^t = (pbest_{ijk}^t)_{numcw*numj*k}$. The best value obtained so far by the whole swarm is tracked and memorized as global best or $gbest^t$, where $gbest^t$ represents the index of the best particle at iteration t. Initially, the previous best position value $pbest_k^0$ is set equal to the initial position value X_k^0. The initial global best particle is the particle with the minimum fitness value among $pbest_k^t$, which is set equal to $pbest_{gbest^o}^0$. At

each iteration t, the current fitness value of each particle $f(Assign(X_k^t))$ is compared with the previous best fitness value $f(Assign(pbest_k^{t-1}))$. If $f(Assign(X_k^t)) < f(Assign(pbest_k^{t-1}))$, $pbest_k^t$ is set equal to X_k^t, which is the set of current position values of particle k. Then, $gbest^t$ is also updated by comparing the new personal best fitness value with the global best fitness value; if $f(Assign(pbest_k^t)) < f(Assign(pbest_{gbest^{t-1}}^{t-1}))$, then set $gbest^t$ equal to k.

Step 5. Calculation of velocity. PSO searches for optima by updating its position using velocity function. The particle flies toward a new position by calculating a rate of position change or velocity. $V_k^t = [v_{ijk}^t]_{numcw^* numj^* k}$ represents matrix of velocity components associated with each dimension of particle k at iteration t. The historical information is used to calculate the velocity and the particle changes position according to the following equation:

$$v_{ijk}^{t+1} = \chi[w^t \cdot v_{ijk}^t + c_1 \cdot r_1 \cdot (pbest_{ijk}^t - x_{ijk}^t) + c_2 \cdot r_2 \cdot (pbest_{ij,gbest^t}^t - x_{ijk}^t)] \tag{4}$$

The velocity equation (Kennedy & Eberhart, 1995) consists of three main parts: the momentum; the cognitive part, which takes into account the personal experience of each particle; the social part which represents the inter-sharing of experience among particles. For the momentum, the inertia weight (w^t) is employed to control the impact of the previous record of velocity on the current velocity. If w^t approaches 0, the velocity function will consider only the previous best and global best parts, so that the particle changes position more quickly to the best position. If w^t is larger, the rate of change is higher. A large inertia weight facilitates global exploration, while a small value facilitates local exploitation. A 'good' value of inertia weight provides balance between the exploration and exploitation ability of the swarm. The variables r_1 and r_2 are random real numbers from $U(0,1)$. The acceleration constants c_1 and c_2 represent the weights of the stochastic acceleration terms that pull each particle toward *pbest* and *gbest*. A high value allows the particle to make a rapid movement toward *pbest* and *gbest*, while a low value allows the particle to move far from the target region before coming back. Commonly, authors who apply PSO use the parameter values recommended by Shi and Eberhart (1998) or Trelea (2003). Shi and Eberhart (1998) suggested that the inertia weight should be set initially to 0.9 and gradually decreased to 0.4 with the decrement factor of 0.975 and acceleration constants set to 2. Trelea (2003) recommended a very precise constant inertia weight of 0.7968 and acceleration constant of 1.4962. Trelea conducted convergence analysis using graphical parameter selection on the benchmark functions. The 'best' set of parameters is, of course, problem specific.

The application of the constriction factor (χ) is suggested by Clerc (1999) to ensure the convergence of PSO. The constriction factor is defined as

$$\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, \quad \text{where} \quad \varphi = c_1 + c_2, \ \ \varphi > 4 \tag{5}$$

φ is normally set to 4.1, which makes χ equal to 0.729. Moreover, the travel distance of the particle is controlled by the maximum velocity (v_{max}) in order to avoid moving past a good solution. v_{max} serves as a constraint to control the maximum global exploration of PSO. If v_{max} is too high, the particle may 'fly past' a good solution. If it is too low, the particle may not explore sufficiently, which may result in being trapped in local optima. An intensive review and guidance on parameters of PSO can be found in Eberhart and Shi (2001).

Step 6. Update position value. At every iteration, the velocity obtained from *Step 5* is added to the current position values (Eq. (6)) to move the particle toward a new position and generate a new potential solution.

$$x_{ijk}^{t+1} = x_{ijk}^t + v_{ijk}^{t+1} \tag{6}$$

Step 7. Termination. After the particle moves into a new position, *Heuristic Assignment* is performed to generate the new feasible schedule, the performance of each particle is evaluated, and so on. The same procedures from *Steps 2* to *6* are repeated until the maximum number of iterations is reached.

4.2. Initial solution method, Earliest Start Time Priority with Minimum Distance Assignment (*ESTPMDA*)

An initial solution is obtained from *ESTPMDA* heuristics as in *Pseudo code3*. In the first stage, the care activities are arranged according to the earliest start time (*EST*) rule. EST is earliest time that an activity can begin in the schedule, i.e. care activities are arranged in ascending order of their ideal start time. The procedure tries to insert activities, individually into all care-worker routes and calculates the cost of insertion (additional distance) for each route. Then, the priority matrix is created. Let *INSERT* (π, a) refer to the insertion of a into sequence π. The activity is assigned to the highest possible priority route provided that the time windows and capacity constraints are not violated. The solutions to the time infeasibility are referring to Step 2 Heuristic Assignment Technique. If the infeasibility occurs, the activity will be assigned to the next highest priority route. Once the assignment is accomplished, the priority value is revised according to the assignment and is kept in the matrix table for future reference. In case that the activity has tried to assign to all care workers (*priority* = *numcw*) but if the infeasibility still remains, that activity will be kept in the unassigned activity list. The procedure is repeated until all activities are assigned.

```
……
Pseudo code3: Initial Solution Method, ESTPMDA

Sort care activities according to EST rule
    k=0, u = 0
Do{ k = k+1
    If  k>1 then apply Pairwise Interchange to EST activity list        % no. of times for PI = pchange * numj
    End If
        j=0
    Do{ j = j+1
            activity = EST(j)
            Do{ i = 1
                INSERT(route(i), activity)        % insert activity to every route
                Calculate cost of insertion       % additional traveling distance
            }while (i <= numcw)
            Create priority matrix AP^0_k         % by ranking cost of insertion (higher cost lower priority)
                priority = 0
            Do{ priority = priority+1
                    π1= AP^r_k (priority, activity)  % route of candidate care worker
                    π2 = INSERT(π1, activity)
                    If (feasible) then π1= π2, feasible = true; repair(AP^r_k)  % repair where necessary
                    Else feasible = false
                    End If
            }while (priority < numcw or feasible = false)
                If (priority = numcw and feasible = false)
                    unassignlist(u) = activity,  u = u+1
                End If
        }while(j <= numj)
}while(k <= popsize)
……
```

The *ESTPMDA* heuristic can be used to obtain one feasible solution. Since PSO is a population-based search technique, variants of feasible solutions must be obtained for each of the particles in the population. In order to accomplish this, pair-wise interchange (*PI*) is applied to the *EST* activity list. By randomly selected two activities and then interchange their position. The parameter *pchange* (the percentage of activities selected for *PI*) controls the number of times *PI* is performed, i.e. the amount

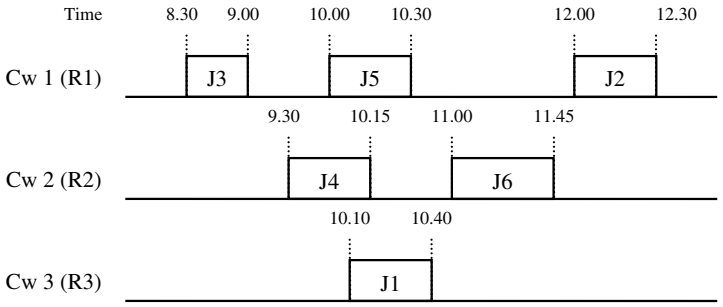

Fig. 6. Care worker schedule before application of LIP.

Table 1
Example of *swap*

Initial routes	Iteration: pair of activities to swap	Before re-arrangement	After re-arrangement	Feasibility checking and evaluation
R1: 3 5 2 R2: 4 6	1st: swap J3 & J4	R1: 4 5 2 R2: 3 6	R1: 4 5 2 R2: 3 6	Infeasible (time conflict between J4 & J5)
R1: 3 5 2 R2: 4 6	2nd: swap J3 & J6	R1: 6 5 2 R2: 4 3	R1: 5 6 2 R2: 3 4	Feasible and suppose it improves objective function
R1: 6 5 2 R2: 4 3	3rd: swap J5 & J4	R1: 6 4 2 R2: 5 3	R1: 4 6 2 R2: 3 5	Feasible but suppose it does not improve fitness function

$AP^t_k =$

	J3	J5	J4	J1	J2	J6
1st	Cw1	Cw1	Cw2	Cw3	Cw1	Cw2
2nd	Cw3	Cw3	Cw3	Cw1	Cw3	Cw1
3rd	Cw2	Cw2	Cw1	Cw2	Cw2	Cw3

Fig. 7. Assignment priority matrix of particle X^t_k (AP^t_k) after *repairing*.

Table 2
Example of *insertion*

Initial routes	Iteration: selected activities and route	Schedule after insertion procedure	Feasibility checking and evaluation
R1: 3 5 2 R2: 4 6 R3: 1	1st: insert J3 to Cw3	R1: 5 2 R2: 4 6 R3: 3 1	Feasible and suppose improve objective function
R1: 5 2 R2: 4 6 R3: 3 1	2nd: insert J5 to Cw3	R1: 2 R2: 4 6 R3: 3 5 1	Infeasible (time conflict between J5 & J1)

of disturbance of the original *EST* sequence. Referring to *Pseudo code3*, the application of PI will results in the different sequence of activities to be selected (*activity*) to be inserted to *route*(i). This will give rise to the differences in occupied time and care worker by each activity. For instance, suppose activity 3 and 2 compete for the same care worker. If activity 3 comes before activity 2 in the *EST* sequence, the activity 3 will be able to assign to that care worker while activity 2 cannot, and vice versa. When applying the same *ESTPMDA* procedure to the new activity sequence after *PI*, as a result, a different initial feasible solution is obtained for each particle.

Table 3
Test parameters for Taguchi design of experiments

Parameters	Descriptions	Setting	
		Level (−1)	Level (1)
PSO's parameters			
A. *maxiter*	Maximum no. of iterations	10	20
B. *popsize*	Population size	10	20
C. $[X_{min}, X_{max}]$	Range of initial position value	$U[0,n]$	$U[-n,n]$
D. v_{max}	Maximum velocity	$(X_{max} - X_{min})/5$	$(X_{max} - X_{min})/2$
E. χ	Constriction factor	0.729	1.0
F. w	Inertia weight	0.8	$0.9 \rightarrow 0.4$ (decrement factor, $\alpha = 0.975$)
G. c_1 and c_2	Acceleration constant	1.4962	2.0
Initial solution (ESTPMDA)'s parameters			
H. *pchange*	Percentage of selecting activities to do PI	5%	20%
Local improvement's parameters			
I. *nselect*	Percentage of selection of particle for LIP	20%	50%
J. *numinsert*	Percentage of care workers to be inserted	50%	100%

Response Table for Signal to Noise Ratios
Smaller is better

Level	maxiter	popsize	xminmax	vmax	x	w	c1c2	pchange	nselect	numinsert
1	-50.72	-50.71	-50.69	-50.70	-50.70	-50.71	-50.70	-50.53	-50.75	-50.70
2	-50.65	-50.65	-50.68	-50.67	-50.67	-50.65	-50.66	-50.83	-50.62	-50.67
Delta	0.07	0.05	0.01	0.03	0.03	0.06	0.04	0.30	0.13	0.03
Rank	3	5	10	9	7	4	6	1	2	8

Response Table for Means

Level	maxiter	popsize	xminmax	vmax	x	w	c1c2	pchange	nselect	numinsert
1	340.7	340.1	339.5	339.8	340.0	340.4	340.0	333.4	341.8	339.7
2	337.9	338.5	339.1	338.8	338.6	338.2	338.6	345.3	336.8	338.9
Delta	2.8	1.6	0.4	1.0	1.3	2.1	1.4	11.9	5.0	0.8
Rank	3	5	10	8	7	4	6	1	2	9

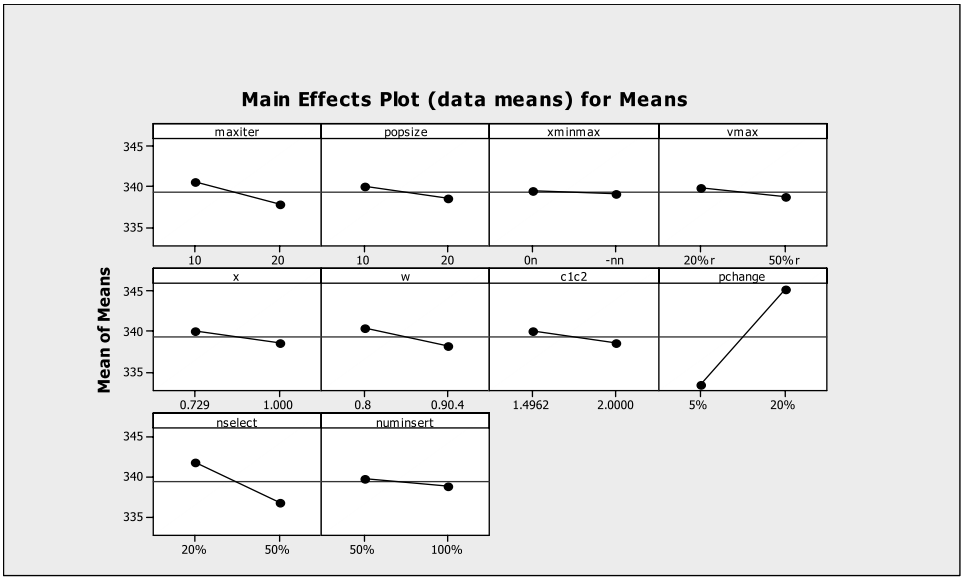

Fig. 8. Response table for S/N ratio and means and main effects plots (data means) for the Taguchi experiment.

4.3. Local improvement procedures (LIP)

In order to overcome the fast convergence of standard PSO, a *local improvement procedure* (LIP) is applied during the searching phase. LIP is a simple and effective search algorithm, which can be applied easily to standard PSO. It allows the particles to explore more of the search area, escaping from local minima, and improving the solution quality. After the fitness evaluation step, LIP is applied to the global best solution and some randomly selected previous best solutions according to the specified number of selection (*nselect*). *Pseudo code4* gives the *swap* and *insertion* procedures.

```
……
Pseudo code4: Local Improvement Procedure
a.  Swap
Randomly select nselect previous best particles and select gbest particle   %Apply LIP for each selected route
    u = 0
Do{ u = u+1
        π1 = route(u)                                   % route of care worker u
        v = u
        Do{ v=v+1
            π2 = route(v)                               % route of care worker v
                a=0
            Do{ a = a+1
                    b=0
                Do{ b= b+1
                    π3 = SWAP(π1, pos(a), pos(b) )      % pos(a) = activity in position a of route π1
                    π4 = SWAP(π2, pos(b), pos(a) )      % pos(b) = activity in position b of route π2
                    Rearrange activities according to start time
                    If (feasible and improve objective function or pass acceptance probability test)
                       then  π1 =π3 , π2 =π4 ; repair   % repair where necessary
                    End If
                }while (b <= no. of activities in π2)
            }while (a <= no. of activities in π1)
        }while (v <= numcw)
}while(u <= numcw-1)
……
b.  Insertion
   j=0
Do{ j=j+1
     priority = 0
          i=0
       Do {i = i+1
            priority = priority + 1
            activity = EST(j)                           % from priority matrix AP'k:
            π1 = route(activity)                        % current route of  selected activity
            π2= route(AP'k (priority, activity) )       % route of candidate care worker
            π3 = REMOVE(π1, activity)
            π4 = INSERT(π2, activity)
            If (feasible and improve objective function or pass acceptance probability test)
               then  π1 =π3 , π2=π4 ; repair            % repair where necessary
                  feasible = true
            Else   feasible = false
            End If
       }while(i<= numinsert*numcw or feasible = true)
}while (j <= numj)
……
```

4.3.1. Swap procedure

Swap interchanges activities between care workers' routes. Given a set of care workers' routes $S = \{R_1, R_2, \ldots, R_p, \ldots, R_q, \ldots, R_m\}$, each route consists of the sequence of service activities on that route. The procedure starts by selecting a pair of routes R_p and R_q. Let $SWAP\ (\pi, a, b)$ be the operation to remove activity in position a of sequence π and replace it with the activity in position b of another sequence. The exchange of care activities is done sequentially along the selected pair of routes. The number of ways of selecting pair of care workers' routes is equal to $m\ (m-1)/2$, where m is the total number of care workers' routes. Suppose nR_1 and nR_2 are the numbers of activities in R_1 and R_2, respectively. The total number of swaps for each selected pair of routes (neighborhood size) is equal to $nR_1 \cdot nR_2$. In swap procedure, the selected activity is placed at a position that is consistent with its time window. Only feasible moves are accepted. After interchanging a pair of activities, the objective value is re-computed. The strategy accepts the first improving move and adopts it for the new starting route. In case that moving results in the same objective function, the new route is accepted with a pre-defined probability (*probaccept*).

Fig. 6 and Table 1 illustrate an example of *swap*. The selected initial routes are care-worker-route-1 (R1) and care-worker-route-2 (R2). *Swap* begins by interchanging activities J3 and J4, followed by the re-arrangement according to the activities' start times and feasibility checking. In the next iteration, activities J3 and J6 are interchanged. The schedule after re-arrangement may be different but is used for the objective function evaluation only. If the improvement is found, the starting route for the next iteration is the sequence after improvement but before re-arrangement to avoid swapping back to the previous solution. The same procedure is continued until all pairs of activities are interchanged and all pairs of routes are considered.

4.3.2. Insertion procedure

A novel, efficient insertion procedure, which utilizes the priority matrix, is introduced here. It utilizes the priority matrix (AP_k^t) in selecting the new care-worker-route to be inserted. The priority matrix represents the degree of attractiveness of each care worker's route to the activity. Let $REMOVE\ (\pi, a)$ refer to remove activity a from route π. Following the priority matrix, care activities are removed one-by-one from the current route and inserted into the new route that has the next highest priority. Similarly, the selected activity is inserted into the new route at the position that is compatible with its time windows. Then, feasibility checking and fitness value evaluation are performed. The procedure accepts the improving move or the equivalent move with the probability *probaccept*. If a better move is not found, the selected activity seeks the route with the next highest priority. *Insertion* does not try to insert into every care worker's route since this would require a high computational effort. The number of care workers to be inserted is controlled by the parameter *numinsert*, which is set to 50% of the total number of care workers' routes. The neighborhood size for insertion procedure is equal to $50\% \cdot m \cdot n$, where m is the total number of care workers' routes and n is the total number of care activities. *Insertion* allows the particle to explore the solution space more thoroughly than applies only swap procedure since it allows care activities to be added or removed from care workers' routes which resulting in changing the load level and traveling direction of care workers. For both *swap* and *insertion*, *probaccept* is set to 0.5. After the LIP is completed, the position value matrix is revised to be compatible with the new sequence.

Recall the priority matrix (Fig. 3) from Step 2 of Section 4.1. The example insertion procedure is shown in Fig. 7 and Table 2. According to the priority matrix, activity J3 is removed from the current route and inserted into the new route with the next highest priority. In this case the route with the next highest priority is R3. The activity is inserted at its time windows, then feasibility checking and fitness evaluation are performed. The insertion is accepted and used as the initial route for the next iteration. Next, activity J5 is considered and the insertion into R3 is tried first. If infeasibility occurs, J5 tries the next priority route, R2. The trial is carried out in the same manner for all activities until a better move is achieved or the allowable limit *numinsert* is reached.

5. Numerical experiment

The algorithm has been tested on a sample of 'real' home care service data, having been coded in MATLAB® (www.mathworks.com) and run on a Pentium M processor with 1.6 GHz CPU speed and 512MB RAM.

Two experiments are presented. The first experiment is to analyze the performance of PSO with different parameter settings on different problem instances using Taguchi design of experiments. The experiment is to determine the 'best' combination of the parameters that gives robust performance. The second experiment compares the solutions obtained by PSO with those obtained by the corresponding local government authority using its existing manual processes and those obtained by the AiMES Centre when using the proprietary software ILOG™ Dispatcher 4.0 (http://www.ilog.com).

5.1. Experiment 1: PSO parameters setting using Taguchi design of experiments

The parameters of PSO control the balance between the global exploration and local exploitation ability and also the ability of the algorithm to find the optimum. The parameter settings are problem specific since different combinations work well for different problems. Experiment 1 determines the best combination of parameter values for the care worker scheduling. The performance with different combinations of parameters values is tested according to Taguchi design of experiments. The Taguchi approach uses a series of experiments to find parameter settings that yield the 'best' results and the least possible sensitivity to noise, where noise is any uncontrolled variation that could affect the performance of the algorithm. The application of Taguchi design of experiments for finding robust scheduling heuristics and optimal parameter combination for the scheduling problem can be found in Dooley and Mahmoodi (1992) and Cheng and Chang (2007).

The design of experiment is based on an $L12$ orthogonal array (Appendix B). The 10 different 'real' problem instances are selected as the outer array (uncontrollable factors). The data are taken from the care requirements of two different service zones from Monday to Friday. The inner array of control variables contains the 10 parameters in Table 3. Each parameter is set at two levels. The values of inertia weight and acceleration constant include those recommended by Shi and Eberhart (1998) and Trelea (2003). Trials are run with and without the constriction factor suggested by Clerc (1999).

The $L12$ orthogonal array gives 12 combinations of parameter settings. Each combination is tested on 10 different problems, giving a total of 120 experimental settings. In order to avoid random error, each setting was run for three replications and the average result is used in the analysis. The parameter settings for the experiment are summarized in Table 3.

MINITAB® Release 14 is used in the analysis of results. Fig. 8 shows the response table for signal-to-noise (S/N) ratio and means and the main effects plots obtained. The objective of robust parameter design is to set factors, that have significant effects on the response, at levels that maximize the S/N ratio. For this problem, the factor levels should be set to minimize the mean objective function as well.

The main effect plots for means show the means of the objective function for each factor level of each parameter. The rank values reveal which factors have the greatest effect, which in this case are *pchange*, *nselect*, and *maxiter*. In terms of maximizing S/N ratio and minimizing mean response, all factors except *pchange* should be set at level 1.

Parameters *popsize* and *numinsert* should be set at level 1 since they do not affect both S/N and means response significantly (rank 5 and 8, respectively) but they affect the computational time considerably. In order to save the computational effort while still maintaining good results, they are set at level -1.

Using the best factor levels obtained, the refining experiment is further conducted for the top three ranking parameters, i.e. *pchange*, *nselect*, *maxiter*. Each parameter is set at three levels covering the best factor levels from the previous experiment. *pchange* is set at 2%, 5%, 10%, *nselect* is set at 35%, 50%, 65%, and *maxiter* is set at 10, 20, 30. In this experiment with three parameters each at three levels the $L9$ orthogonal array (Appendix C) is applied. The results are presented in Fig. 9. It can be seen that *pchange* has a significant effect on the means response. The results show factor $maxiter = 20$ gives slightly better results than $maxiter = 30$. This may be because of random error or the probabilistic characteristic of PSO. However, it can be concluded that $maxiter = 20$ is sufficient for PSO to converge. According to the response table for signal-to-noise (S/N) ratio and means and the main effects plots, the best factor levels are $pchange = 2\%$, $maxiter = 20$, and $nselect = 35\%$.

Response Table for Signal to Noise Ratios

Smaller is better

Level	pchange	nselect	maxiter
1	-50.43	-50.54	-50.68
2	-50.60	-50.60	-50.49
3	-50.65	-50.55	-50.52
Delta	0.22	0.06	0.19
Rank	1	3	2

Response Table for Means

Level	pchange	nselect	maxiter
1	331.6	335.5	341.2
2	338.0	338.0	334.0
3	340.6	336.6	334.8
Delta	9.0	2.5	7.2
Rank	1	3	2
Rank	1	3	2

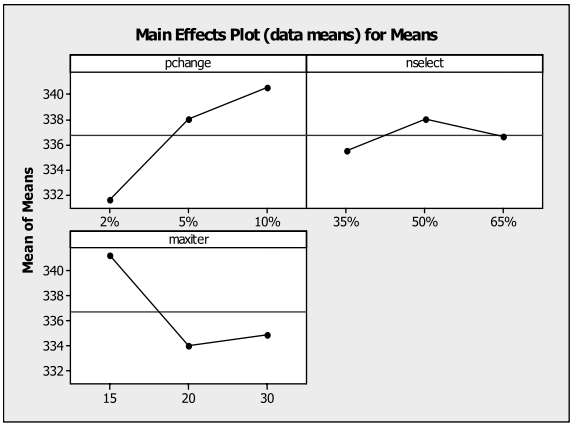

Fig. 9. Response table for S/N ratio and means and main effects plots (data means) for the refining experiment.

5.2. Experiment 2: comparison of different variants of algorithm

The experiment is to demonstrate the performance of PSO with the inclusion or exclusion of Local Improvement Procedure (LIP) and initial solution heuristic (ESTPMDA). It is also to provide the evidence that both heuristic techniques contribute to the improvement of the algorithm's performance. The experiment is performed on four variants of PSO using the same set of real test data. The experiment is executed for 30 replications. For each case, the same sets of applicable parameters as concluded from Experiment 1 are employed. The following types of PSO are tested and the average results are shown in Table 4.

$PSO_{ESTPMDA}$: applying PSO algorithm to each particle in a population using ESTPMDA to create initial solution for each particle.
PSO_{LIP}: a PSO algorithm with LIP using randomly generated initial solutions.
$LIP_{ESTPMDA}$: an application of LIP to set of solutions in which the starting solutions are obtained by ESTPMDA (exclude PSO).
$PSO_{ESTPMDA,LIP}$: a complete version of the algorithm, PSO with ESTPMDA and LIP.

The results has shown that $PSO_{ESTPMDA,LIP}$ gives best results of all test cases, followed by $LIP_{ESTPMDA}$, PSO_{LIP}, and $PSO_{ESTPMDA}$ accordingly. For all test cases, the results of all different variants of algorithm differ from others significantly according to *ANOVA* results at 95% confidence level. The existence of all heuristics influences the effectiveness of the algorithm. It can be seen that using PSO ($PSO_{ESTPMDA}$) or LIP alone ($LIP_{ESTPMDA}$), the algorithm gives not so good solution. With the inclusion of initial solution heuristic, $LIP_{ESTPMDA}$ can perform better than $PSO_{ESTPMDA}$ since LIP has better local search ability than PSO. The initial solution heuristic (ESTPMDA) also presents a significant part of the algorithm. Without ESTPMDA (PSO_{LIP}), there is no guideline for a searching direction. As a result, particles may be directed to a wrong direction at the beginning which may be difficult to be back to the right direction. It can be seen that all

Table 4
Comparison of results (total miles traveled) of different variants of algorithms

Test problem	$PSO_{ESTPMDA}$	PSO_{LIP}	$LIP_{ESTPMDA}$	$PSO_{ESTPMDA,LIP}$
Test 1	408	386	373	328
Test 2	411	403	379	351
Test 3	445	433	404	361
Test 4	510	474	430	405
Test 5	459	452	394	374

components help each other. ESTPMDA direct particles to the right track, PSO encourage the information sharing among particles and utilize its own searching experience, while LIP has ability to fine-tuned search around the local area. When the solution of each particle spreads around solution space, LIP will encourage each particle to fine-tuned search in that area to find a better solution. Once the better solution is found, other particles will update themselves based on better solution found so far. The new searching area might be established, hence, help them to escape from local optimum. In conclusion, the algorithm will perform the best when all mechanisms are included and all parts contribute to the improving of solution quality.

5.3. Experiment 3: comparative performance of PSO with previous solution techniques

The results from Experiment 1 and Experiment 2 identify the parameter levels and the PSO procedure that minimize the objective function and its variability. For this experiment, $PSO_{ESTPMDA,LIP}$ with the best combination of parameter settings is used to test the performance of PSO. The test results are compared with those obtained by *The Welsh Systems Consortium* using the existing manual approach and those obtained by AiMES using ILOG™. The experiment was performed on five selected sets of 'real' home care data. The data consists of over 100 activities per day requested by 50 clients to be carried out by 12 care workers. For each test case, PSO was run for 30 replications.

The convergence or stopping criterion used here for PSO is that the algorithm terminates when no improvement in the objective function is seen in 20 consecutive iterations. This normally allows PSO to run for approximately 3.5 min. The ILOG application also runs until no more convergence is seen and this typically takes a similar amount of time.

From the comparison of results in Table 5 and Fig. 10, it can be seen that the PSO-based algorithm yields consistently and substantially better results, with total mileage savings from 11% to 31%. The statistical *t*-test was applied to compare the results obtained by PSO and AiMES. Since the *p*-values of all test cases are less than 0.01, it can be concluded that the results obtained from PSO are consistently and significantly better than those obtained by AiMES.

6. Conclusions

The home care service is growing in the UK and in other countries around the world. Home care can be provided directly by the state or an independent provider with the aim of achieving *best value*, in terms of quality and cost. The drive to maximize quality and minimize costs creates a need for care-worker scheduling algorithms to support the planning process by reducing costs, improving customer service and reducing the cost of planning, etc. The novel application of a PSO-based scheduling algorithm to care-worker scheduling has been presented here.

The algorithm combines local improvement techniques to effectively and efficiently schedule care-workers. The objective is the minimization of the total distance traveled by all care workers, while satisfying the capacity and delivery time window constraints. The algorithm utilizes the population-based evolutionary searching characteristic of PSO to explore the solution space globally, and also exploit the local search ability of *local improvement procedures* (*swap* and *insertion*) to fine-tune the neighborhood area more thoroughly. The

Table 5
Comparison of results (total miles traveled) of PSO and previous solution techniques

Test problem	Welsh systems consortium	AiMES	$PSO_{ESTPMDA,LIP}$ mean results	$BPSO_{ESTPMDA,LIP}$ best results[a]	Saving ($BPSO_{ESTPMDA,LIP}$ compared to AiMES)	% Saving
Test 1	592	349	328	307	42	12
Test 2	643	384	351	325	59	15
Test 3	658	417	361	329	88	21
Test 4	1388	535	405	370	165	31
Test 5	667	388	374	347	32	11

[a] Best results over 30 replications.

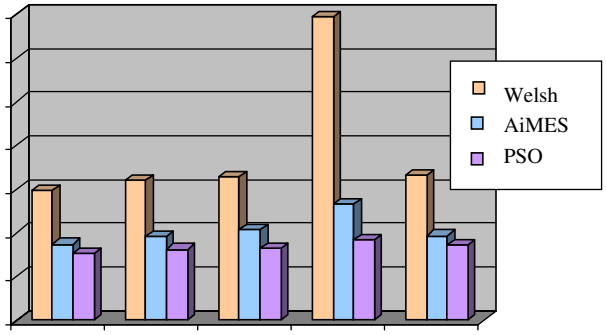

Fig. 10. Graphical representation of results of PSO and previous solution techniques.

specially designed *Heuristic Assignment* scheme is applied to enable the transformation of the continuous PSO into a discrete schedule. The *Earliest Start Time Priority with Minimum Distance Assignment* (ESTPMDA) technique (initial solution method) is employed to guide the search direction of the particles.

A parameter study has been performed using Taguchi design of experiments in order to find the 'best' combination of parameter values for this specific application. In order to assess the solution qualities and computational performance, the methodology has been tested on 'real' demand data and the results compared with those obtained with the existing manual approach and those obtained by the AiMES Centre at the University of Liverpool using ILOG™. The PSO-based algorithm produced significantly and consistently better results across all the test problems.

This work contributes to the development of an efficient methodology to improve the scheduling of care workers and also introduces the application of PSO-based algorithm to solve this type of problem and all classes of similar problems. Future research will extend to multiple objectives, i.e. minimize the number of care workers together with the distance traveled, and include some social issues such as work-shift pattern, care worker and client familiarity and skill matching requirements.

Acknowledgements

This research is supported by the Royal Golden Jubilee Ph.D. Program of Thailand Research Fund, Grant No. PHD/0084/2544 and the University of Liverpool Management School.

Appendix A. Example of care requirements

Client No.	Postcode	Day	Start time	End time	Duration minutes	Home care activities			
1	*	4	8:45	10:15	90	Meal preparation	Continence care-pad change	Prepare drinks-hot drink	Bed bath
2	*	1	18:01	18:31	30	Personal hygiene-simple wash	Prepare for bed	Assist to bed – equipment	Continence care-pad change
3	*	5	8:00	8:30	30	Transfers-assist from bed	Dressing and undressing	Make bed	Deliver meals

Appendix A (*continued*)

Client No.	Postcode	Day	Start time	End time	Duration minutes	Home care activities			
4	*	3	20:30	21:15	45	Prepare drinks-cold drink	Update-write	Assist to bed – equipment	Prepare hot water bottle
5	*	1	9:05	9:50	45	Joint care social services	Continence care-pad change	Check bed/ change	Update-write
6	*	1	19:56	20:26	30	Prepare for bed	Assist to bed – Equipment	Joint care social services	Continence care-pad change
7	*	1	8:11	8:56	45	Dressing and undressing	Continence care-pad change	Assist from bed	Shower
5	*	7	9:05	9:50	45	Continence care-pad change	Check skin condition	Dressing and undressing	Assist from bed
6	*	7	10:00	11:00	60	Assist from bed	Joint care social services	Make bed	Continence care-pad change
6	*	6	10:00	11:00	60	Assist from bed	Joint care social services	Make bed	Continence care-pad change
6	*	1	10:00	11:00	60	Bed bath	Joint care social services	Make bed	Continence care-pad change
8	*	1	11:05	11:35	30	Dressing and undressing	Make bed	Drinks-hot drink	Bed bath
5	*	7	18:30	19:00	30	Check skin condition	Continence care-pad change	Simple wash	Joint care social services
7	*	6	8:11	8:56	45	Dressing and undressing	Continence care-pad change	Make bed	Shower
6	*	7	19:56	20:26	30	Prepare for bed	Assist to bed – equipment	Joint care social services	Shower
8	*	6	11:05	11:35	30	Dressing and undressing	Make bed	Prepare drinks-hot drink	Bed bath
8	*	7	11:05	11:35	30	Dressing and undressing	Make bed	Prepare drinks-hot drink	Bed bath
7	*	7	8:11	8:56	45	Dressing and undressing	Continence care-pad change	Shower	Assist from bed
7	*	1	17:45	18:15	30	Continence care-pad change	Update-write	Empty commode	Joint care social services

(*continued on next page*)

Appendix A (*continued*)

Client No.	Postcode	Day	Start time	End time	Duration minutes	Home care activities			
5	*	6	9:05	9:50	45	Continence care-pad change	Check skin condition	Dressing and undressing	Assist from bed
7	*	6	17:45	18:15	30	Continence care-pad change	Update-write	Empty commode	Joint care social services
7	*	7	17:45	18:15	30	Continence care-pad change	Update-write	Empty commode	Joint care social services
6	*	6	19:56	20:26	30	Continence care-pad change	Assist to bed – equipment	Joint care social services	Continence care-pad change
5	*	6	18:30	19:00	30	Check skin condition	Continence care-pad change	Simple wash	Joint care social services
9	*	1	12:10	12:40	30	Continence care-pad change	Wash dishes	Toilet	Joint care social services
9	*	7	12:10	12:40	30	Continence care-pad change	Wash dishes	Toilet	Joint care social services
9	*	1	17:00	17:30	30	Continence care-pad change	Wash dishes	Meal preparation	Toilet
9	*	6	17:00	17:30	30	Continence care-pad change	Wash dishes	Meal preparation	Toilet

Appendix B. *L*12 Orthogonal array (for 10 factors each at three levels)

Test	*maxiter*	*popsize*	$[X_{min}, X_{max}]$	v_{max}	χ	w	c_1 and c_2	*pchange*	*nselect*	*numinsert*
1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1
2	−1	−1	−1	−1	−1	1	1	1	1	1
3	−1	−1	1	1	1	−1	−1	−1	1	1
4	−1	1	−1	1	1	−1	1	1	−1	−1
5	−1	1	1	−1	1	1	−1	1	−1	1
6	−1	1	1	1	−1	1	1	−1	1	−1
7	1	−1	1	1	−1	−1	1	1	−1	1
8	1	−1	1	−1	1	1	1	−1	−1	−1
9	1	−1	−1	1	1	1	−1	1	1	−1
10	1	1	1	−1	−1	−1	−1	1	1	−1
11	1	1	−1	1	−1	1	−1	−1	−1	1
12	1	1	−1	−1	1	−1	1	−1	1	1

Appendix C. *L*9 Orthogonal array (for three factors each at three levels)

Test	*pchange*	*nselect*	*maxiter*
1	1	1	1
2	1	2	2
3	1	3	3
4	2	1	2
5	2	2	3
6	2	3	1
7	3	1	3
8	3	2	1
9	3	3	2

References

Allahverdi, A., & Al-Anzi, F. S. (2006). A PSO and a tabu search heuristics for the assembly scheduling problem of the two-stage distributed database application. *Computers and Operations Research, 33*, 1056–1080.

Andrés, C., & Lozano, S. (2006). A particle swarm optimization algorithm for part-machine grouping. *Robotics and Computer-Integrated Manufacturing, 22*(5–6), 468–474.

Bianco, L., Bielli, M., Mingozzi, A., Ricciardelli, S., & Spadoni, M. (1992). A heuristic procedure for the crew scheduling problem. *European Journal of Operational Research, 58*, 272–283.

Bouthillier, A. L., & Crainic, T. G. (2005). A cooperative parallel meta-heuristic for the vehicle routing problem with time windows. *Computers and Operations Research, 32*, 1685–1708.

Bräysy, O., Dullaert, W., & Gendreau, M. (2004). Evolutionary algorithms for the vehicle routing problem with time windows. *Journal of Heuristics, 10*, 587–611.

Cheang, B., Li, H., Lim, A., & Rodrigues, B. (2003). Nurse scheduling problems – A bibliographic survey. *European Journal of Operational Research, 151*, 447–460.

Cheng, B. W., & Chang, C. L. (2007). A study on flowshop scheduling problem combining Taguchi experimental design and genetic algorithm. *Expert Systems with Application, 32*, 415–421.

Chiang, W., & Russell, R. (1996). Simulated annealing methaheuristics for the vehicle routing problem with time windows. *Annals of Operations Research, 63*, 3–27.

Clerc, M. (1999). The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization. In *Proceeding of the IEEE congress on evolutionary Computation* (pp. 1951–1957). Washington, DC: Piscataway, NJ IEEE service center.

Desrochers, M., Lentra, J., Savelsbergh, M., & Soumis, F. (1988). *Vehicle routing with time windows: Optimization and approximation*. North-Holland, Amsterdam: Vehicle Routing: Methods and Studies.

Desrochers, M., Desrosiers, J., & Solomon, M. (1992). A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research, 40*, 342–354.

Dooley, K. J., & Mahmoodi, F. (1992). Identification of Robust scheduling heuristics: Application of Taguchi methods in simulation studies. *Computers and Industrial Engineering, 22*(4), 359–368.

Drake, P., & Davies, B. M. (2006). Home care outsourcing strategy. *Journal of Health Organization and Management, 20*(3), 175–193.

Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Proceeding of the sixth international symposium on micro machine and human science* (pp. 39–43). Nagoya, Japan: Piscataway, NJ IEEE service center.

Eberhart, R. C., Shi, Y. (2001). Particle swarm optimization: Developments, applications, and resources. In *Proceeding of the IEEE congress on evolutionary computation* (pp. 81–86). Seoul Korea: Piscataway, NJ IEEE service center.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and scheduling: A review of applications, methods and methods. *European Journal of Operational Research, 153*, 3–27.

Everborn, P., Flisberg, P., & Rönnqvist, M. (2006). LAPS CARE – An operational system for staff planning of home care. *European Journal of Operational Research, 171*, 962–976.

Fisher, M. L., Jornsten, K. O., & Madsen, O. B. G. (1997). Vehicle routing with time windows: Two optimization algorithms. *Operations Research, 45*, 488–492.

Gambardella, L. M., Taillard, E., Agazzi, G. (1999). MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows, New Ideas in Optimization, McGraw-Hill, (Also available as Technical Report IDSIA-06-99, IDSIA, Lugano, Switzerland).

Garcia, B. L., Potvin, J. Y., & Rousseau, J. M. (1994). A parallel implementation of the tabu search heuristic for vehicle routing problems with time window constraints. *Computers and Operations Research, 21*, 1025–1033.

Goumopoulos, C., & Housos, E. (2004). Efficient trip generation with a rule modelling system for crew scheduling problems. *Journal of Systems and Software, 69*(1–2), 43–56.

Homberger, J., & Gehring, H. (2005). A two-phase hybrid metaheuristic for routing problems with time windows. *European Journal of Operational Research, 162*, 220–238.

Hwa, F., Liu, F., & Shen, S. Y. (1999). A route-neighborhood-based metaheuristic for vehicle routing problem with time windows. *European Journal of Operational Research, 128*, 485–504.

Jerald, J., Asokan, P., Prabaharan, G., & Saravanan, R. (2005). Scheduling optmisation of flexible manufacturing systems using particle swarm optmisation algorithm. *International Journal of Advanced Manufacturing Technology, 25*, 964–971.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In: *Proceeding of IEEE international conference on neural networks IV* (pp. 1942–1948). Piscataway, NJ IEEE service center.

Kolen, A. W. J., Kan, A. H. G. R., & Trienekens, H. W. J. M. (1987). Vehicle routing with time windows. *Operations Research, 35*, 266–273.

Lian, Z., Gu, X., & Jiao, B. (2006). A similar particle swarm optimization algorithm for job-shop scheduling to minimize makespan. *Applied Mathematics and Computation, 175*(1), 773–785.

Liao, C. J., Tseng, C. T., & Luarn, P. (2007). A discrete version of particle swarm optimization for flowshop scheduling problems. *Computers and Operations Research, 34*, 3099–3111.

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the Traveling Salesman problem. *Operations Research, 21*, 498–516.

Lin, C. K. Y., Lai, K. F., & Hung, S. L. (2000). Development of a workforce management system for a customer hotline service. *Computers and Operations Research, 27*, 987–1004.

Liu, F. H. F., & Shen, S. Y. (1999). An overview of a heuristic for vehicle routing problem with time windows. *Computers and Industrial Engineering, 37*, 331–334.

Lopes, H. S., Coelho, L. S. (2005). Particle swarm optimization with fast local search for the blind traveling salesman problem. In *Proceedings of the fifth international conference on hybrid intelligent systems (HIS'05)* (pp. 245–250). IEEE computer society.

Pang, W., Wang, K., Zhou, C., & Dong, L. (2004a). Fuzzy discrete particle swarm optimization for solving travelling salesman problem. In *Proceedings of the fourth international conference on computer and information technology (CIT'04)* (pp. 796–800). IEEE computer society.

Pang, W., Wang, K., Zhou, C., Dong, L., Liu, M., Zhang, H., & Wang, J. (2004b). Modified particle swarm optimization based on space transformation for solving traveling salesman problem. In *Proceedings of the third international conference on machine learning and cybernetics* (pp. 2342–2346). Shanghai: IEEE.

Potvin, J. Y., & Rousseau, J. M. (1993). A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research, 66*, 331–340.

Potvin, J. Y., & Rousseau, J. M. (1995). Exchange heuristic for routing problems with time windows. *Journal of the Operational Research Society, 46*, 1433–1446.

Potvin, J. Y., & Bengio, S. (1996). The vehicle routing problem with time windows – Part II: Genetic search. *Informs Journal of Computing, 8*, 165–172.

Potvin, J., Kervahut, T., Garcia, B., & Rousseau, J. (1996). The vehicle routing problem with time windows, part I: Tabu search. *Informs Journal of Computing, 8*, 158–164.

Savelsbergh, M. W. P. (1990). An efficient implementation of local search algorithms for constrained routing problems. *European Journal of Operational Research, 47*, 75–85.

Shen, Q., Shi, W., Yang, X., & Ye, B. (2006). Particle swarm algorithm trained neural network for QSAR studies of inhibitors of platelet-derived growth factor receptor phosphorylation. *European Journal of Pharmaceutical Sciences, 28*(5), 369–376.

Shi, Y., & Eberhart, R. C. (1998). Parameter selection in particle swarm optimization. In *Proceedings of the seventh annual conference on evolutionary programming* (pp. 591–600). New York: Springer-Verlag.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research, 35*, 254–265.

Solomon, M., & Desrosiers, J. (1988). Time window constrained routing and scheduling problem. *Transportation Science, 22*, 1–13.

Taillard, E., Badeau, P., Gendreau, M., Guertin, F., & Potvin, J. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science, 31*, 170–186.

Tan, K. C., Lee, L. H., Zhu, Q. L., & Ou, K. (2001). Heuristic methods for vehicle routing problem with time windows. *Artificial Intelligence in Engineering, 15*, 281–295.

Tasgetiren, M. F., Liang, Y. C., Sevkli, M., Gencyilmaz, G. (2004). Particle swarm optimization algorithm for single machine total weighted tardiness problem. In *Proceedings of the 2004 congress on evolutionary computation* (pp. 1412–1419). Piscataway, NJ IEEE service center.

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

Thangiah, S.R., Osman, I.H., & Sun, T. (1994). Hybrid genetic algorithm, simulated annealing and tabu search methods for vehicle routing problems with time windows. Technical report SRU-CpSc-TR-94-27, Computer Science Department, Slippery Rock University.

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

Xia, W., & Wu, Z. (2005). An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. *Computers and Industrial Engineering, 48*, 409–425.

Yan, S., & Tu, Y. (2002). A network model for airline cabin crew scheduling: Case study. *European Journal of Operational Research, 140*, 531–540.

Yin, P., Yu, S., Wang, P., & Wang, Y. (2007). Multi-objective task allocation in distributed computing systems by hybrid particle swarm optimization. *Applied Mathematics and Computation, 184*, 407–420.

Zhang, H., Li, H., & Tam, C. M. (2006). Particle swarm optimization for resource-constrained project scheduling. *International Journal of Project Management, 24*, 83–92.