Chapter 5

PSO-based Algorithm for Home Care Worker Scheduling

This chapter 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 are two fold; 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.

5.1 Problem Description

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 of care requirements is given in Appendix C; 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 minutes. For non-critical activities the window is ± 15 minutes. 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 hours per day, including travel time.
viii. In the model used here, each worker is assumed to be available 24 hours per day, but can be used for only 7.5 hours 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). For a detailed description of the VRPTW notation and mathematical formulation used see Tan *et al*. (2001). 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. It should be noted that the travel speed of 30 miles per hour and the Euclidean distance are used in order to aid the comparison with previous technique (by AiMES).

5.2 The PSO Methodology

PSO is a collaborative population-based search technique inspired by the simulation of the social behavior 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 code 5.1: 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: *Heuristic assignment* (see *Pseudo code2*)
And apply *Repairing algorithm* (where necessary)
Fitness value evaluation: calculate $f(Assign(x^t_k))$ from Eq. 3

Apply Local Improvement Procedure: (see *Pseudo code 4*) to some selected previous best (*pbest*) particles and global best (*gbest*) particle

a. Swap Procedure
b. Insertion Procedure

and apply *Repairing algorithm* (where necessary)

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, Figure 5.1 shows the matrix representation of particle X^t_k.

$X^t_k =$

	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	$x^t_{numcw,1,k}$	$x^t_{numcw,2,k}$	…..	…..	$x^t_{numcw,numj,k}$

Figure 5.1 The matrix representation of individual particle X^t_k

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

$$X^0_{ijk} = X_{\min} + (X_{\max} - X_{\min}) \times U(0,1) \tag{5.1}$$

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 to 1. The position value of each particle will be translated into the schedule using the *Heuristic 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. 5.1 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:

$$V^{0}_{ijk} = V_{\min} + (V_{\max} - V_{\min}) \times U(0,1) \tag{5.2}$$

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 code 5.2* along with an example illustrates the procedure of *Heuristic Assignment*. The position values of each particle (Figure 5.2) 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) in which the smaller value has the higher priority, as in Figure 5.3.

……

Pseudo code 5.2: Heuristic Assignment Technique

```
    k= 0
Do{ k = k+1
    Sorting and Prioritizing: sort X^t_k and obtain priority matrix AP^t_k
    j=0
    Do {j = j+1
        priority = 0
        Do{
                priority = priority + 1
            From AP^t_k:  activity = EST(j)
                candidate care worker = AP^t_k (priority, activity)
            Assignment: assign candidate activity to candidate care worker
            Feasibility Checking:
                If (capacity >= capacity limit)
                    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

End

}while(*feasible* = false)

Assignment: assign activity to candidate care worker

Update information: update occupied time, care worker capacity

If (*priority* ≠ 1)

then **Apply Repairing Algorithm:**

%Get the position value of the highest priority care worker

$a = X^t_k(1, activity)$

%Get the position value of the actual assigned care worker

$b = X^t_k(priority, activity)$

$X^t_k(1, activity) = b$

$X^t_k(priority, activity) = a$

End

}while(*j* <= *numjob*)

}while(*k* <= *popsize*)

……

X^t_k =

	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

Figure 5.2 Position value matrix of particle k^{th} 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

Figure 5.3 Assignment priority matrix of particle X^t_k (AP^t_k)

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 hours). 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 Figure 5.4.

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

Figure 5.4 Actual assignment after feasibility checking.

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.

		EST care activities					
Care worker ID		J3	J5	J4	J1	J2	J6
$X^t_k =$	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

Figure 5.5 Position value matrix of particle X^t_k after *repairing*

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 Figures 5.3 and 5.4, care activity J4 is revised as highlighted in Figure 5.5.

Step 3. Fitness value evaluation

The fitness value of each particle is calculated after the activity schedules have been constructed. Let $Assign(X^t_k)$ be the corresponding sequence at iteration t of particle X^t_k. The objective function $f(Assign(X^t_k))$ 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.

$$\text{Total Distance} = \sum_{o=1}^{numcw}\sum_{j=1}^{numj} p_{cw(o),j} d_{cw(o),j} + \sum_{i=1}^{numj}\sum_{j=1_{j\neq i}}^{numj} p_{ij} d_{ij} + \sum_{o=1}^{numcw}\sum_{j=1}^{numj} p_{j,cw(o)} d_{j.cw(o)} + M \cdot numu \quad (5.3)$$

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 k^{th} particle is recorded as the previous personal best solution and represented as $pbest^t_k = (pbest^t_{ijk})_{numcw*numj*k}$.. 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 $pbest^0_k$ is set equal to the initial position value X^0_k. The initial global best particle is the particle with the minimum fitness value among $pbest^t_k$, which is set equal to $pbest^0_{gbest^0}$. At each iteration t, the current fitness value of each particle $f(Assign(X^t_k))$ is compared with the previous best fitness value $f(Assign(pbest^{t-1}_k))$. If $f(Assign(X^t_k)) < f(Assign(pbest^{t-1}_k))$, $pbest^t_k$ is set equal to X^t_k, 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 $f(Assign(pbest^t_k)) < f(Assign(\ pbest^{t-1}_{gbest^{t-1}}\ ))$, then set $gbest^t$ equal to k.

Step 5. Calculation of velocity

PSO searches for optima by updating generations. The particle flies toward a new position by calculating a rate of position change or velocity. $V^t_k = \{v^t_{1,1,k}\ ,\ v^t_{1,2,k}\ ,\ldots,\ v^t_{1,numj,k}\ ,\ v^t_{2,1,k}\ ,\ v^t_{2,2,k}\ ,..,\ v^t_{numcw,1,k}\ ,\ v^t_{numcw,2,k}\ ,\ldots,\ v^t_{numcw,numj,k}\}$ represents the set 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:

$$v^{t+1}_{ijk} = \chi\ [\ w^t.v^t_{ijk} + c_1.r_1.\left(pbest^t_{ijk} - x^t_{ijk}\right) + c_2.r_2.\left(pbest^t_{ij,gbest^t} - x^t_{ijk}\right)\ ] \quad (5.4)$$

The velocity (Eq. 5.4) 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 (2003) 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:

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \text{ where } \varphi = c_1 + c_2,\ \varphi > 4 \tag{5.5}$$

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. 5.6) to move the particle toward a new position and generate a new potential solution.

$$x^{t+1}{}_{ijk} = x^t{}_{ijk} + v^{t+1}{}_{ijk} \tag{5.6}$$

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.

5.3 Initial Solution Method, ESTPMDA heuristics

An initial solution is obtained from *ESTPMDA* heuristics as in *Pseudo code* 5.3. 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 code 5.3: Initial Solution Method, ESTPMDA

Sorting: sort care activities according to *EST* rule
    k=0
Do{ $k = k$+1
    If *k>1* ***then*** apply Pairwise Interchange (no. of times for PI = *pchange* * *numj*) to *EST* activity list
        j=0
    Do{ $j = j$+1
        activity = *EST*(j)
        Insert: *activity* to every route, *INSERT*(*route*(i), *activity*) for i = 1 to *numcw*
        Calculate: cost of insertion (additional traveling distance)
        Create priority matrix AP^0_k**:** ranking cost of insertion (higher cost lower priority)
            priority = 0
        Do{ *priority* = *priority*+1
            π_1= *route of candidate care worker* = AP^t_k (*priority*, *activity*)
            π_2 = *INSERT*(π_1, *activity*)
        If (feasible) then π_1= π_2, *feasible* = *true*; *repair*(AP^t_k) where necessary
        Else *feasible* = *false*
        End
        }while (*priority* <= *numcw* or *feasible* = *true*)
    }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 of disturbance of the original *EST* sequence. Referring to *Pseudo code* 5.3, 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.

5.4 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 code 5.4* gives the *swap* and *insertion* procedures.

……

Pseudo code 5.4: Local Improvement Procedure

a. Swap

Randomly select *nselect* previous best particles and select *gbest* particle

Apply *LIP* for each selected route:

$u = 0$

Do{ $u = u+1$

$\pi_1 = route(u)$

$v = u$

Do{ $v=v+1$

$\pi_2 = route(v)$

$a=0$

Do{ $a = a+1$

$b=0$

Do{ $b= b+1$

pos(a) = activity in position a of route π_1 and pos(b) = activity in position b of route π_2

$\pi_3 = SWAP(\pi_1, pos(a), pos(b))$ and $\pi_4 = SWAP(\pi_2, pos(b), pos(a))$

Rearrange activities according to start time

If (feasible and improve objective function or pass acceptance probability test)

then $route(u) = \pi_3$, $route(v) = \pi_4$ *; repair*

End

}while (b <= no. of activities in π_2)

}while (a <= no. of activities in π_1)

}while (v <= *numcw*)

}while(u <= *numcw*-1)

……

b. Insertion

$j=0$

Do{ $j=j+1$

priority = 0

i=0

Do {$i = i$+1

$priority = priority + 1$

From AP^t_k: $activity = EST(j)$

$\pi_1 = route\ of\ activity\ EST(j)$

π_2= $route\ of\ candidate\ care\ worker = AP^t_k\ (priority, activity)$

$\pi_3 = REMOVE(\pi_1, activity)$

$\pi_4 = INSERT(\pi_2, activity)$

If (feasible and improve objective function or pass acceptance probability test)

then $route\ of\ activity\ EST(j) = \pi_3$, $route\ of\ candidate\ care\ worker = \pi_4$

$feasible = true$; repair

Else $feasible = false$

End

}while(i<= $numinsert$*$numcw$ or $feasible = true$)

}while (j <= $numj$)

……

5.4.1 Swap Procedure

Swap interchanges activities between care workers' routes. Given a set of care workers' routes S = {R_1 , R_2 ,…, R_p ,..., R_q ,..., 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 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 the case that moving results in the same objective function, the new route is accepted with a pre-defined probability (*probaccept*).

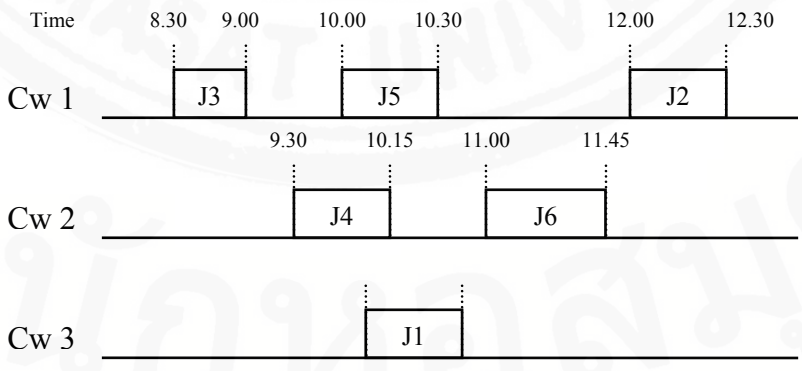

Figure 5.6 Care worker schedule before application of LIP

Table 5.1 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

Figure 5.6 and Table 5.1 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 founded, the starting route for the next iteration is the sequence before re-arrangement. The same procedure is continued until all pairs of activities are interchanged and all pairs of routes are considered.

5.4.2 Insertion Procedure

A novel, efficient insertion procedure, which utilizes the priority matrix, is introduced here. It utilizes the priority matrix (AP^t_k) 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 *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.

$AP^t_k =$

	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

Figure 5.7 Assignment priority matrix of particle X^t_k (AP^t_k) after repairing

Table 5.2 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)

5.5 Analysis of Computational Performance

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).

5.5.1 Experiment 1: PSO parameters setting using Taguchi design of experiment

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 design of experiment is based on an $L12$ orthogonal array (see Table 5.3). The ten 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 ten parameters

in Table 5.3. 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).

Table 5.3 $L12$ Orthogonal array (for 10 factors each at 3 levels)

Test	*max iter*	*pop size*	$[X_{min}, X_{max}]$	v_{max}	χ	w	c_1, c_2	*pchange*	*nselect*	*numin sert*
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

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 used in the analysis. The parameter settings for the experiment are summarized in Table 5.4.

Table 5.4 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 → 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%

MINITAB® Release 14 is used in the analysis of results. Figure 5.8 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 3 parameters each at 3 levels the *L*9 orthogonal array (see Table 5.5) is applied. The results are presented in Figure 5.9.

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

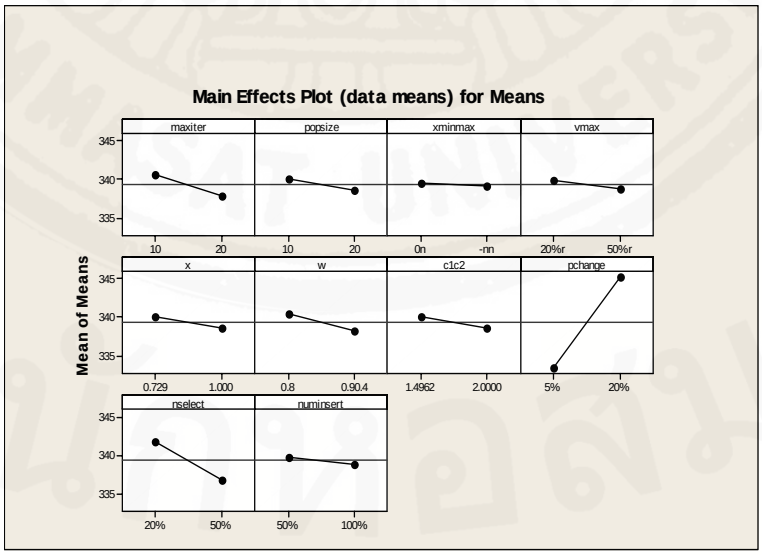

Figure 5.8 Response table for S/N ratio and means and main effects plots (data means) for the Taguchi experiment

Table 5.5 *L*9 Orthogonal array (for 3 factors each at 3 level**)**

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

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

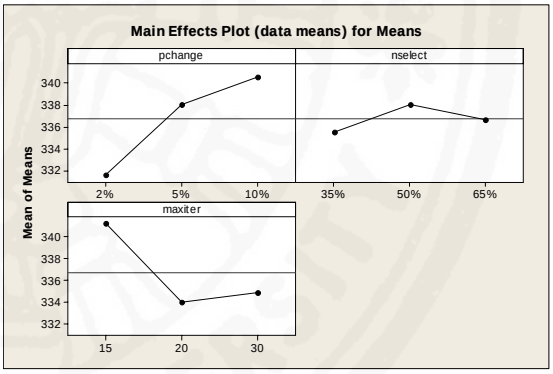

Figure 5.9 Response table for S/N ratio and means and main effects plots (data means) for the refining experiment

5.5.2 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 5.6.

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

Table 5.6 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

The results have 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 quickly converges to local optimum. 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 might move to the wrong direction at the beginning which makes it hard for them to be back to the right direction afterward. It can be seen that all components facilitate one another. ESTPMDA directs particles to the right track, PSO encourages the information sharing among population and utilize its own searching experience, while LIP has an 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 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 presented and all parts contribute to the improving of solution quality.

5.5.3 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 replicationsThe 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 minutes. 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 5.6 and Figure 5.10, it can be seen that the PSO-based algorithm yields consistently and substantially better results, with total mileage savings from 8% 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.

Table 5.7 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**	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

** Best results over 30 replications

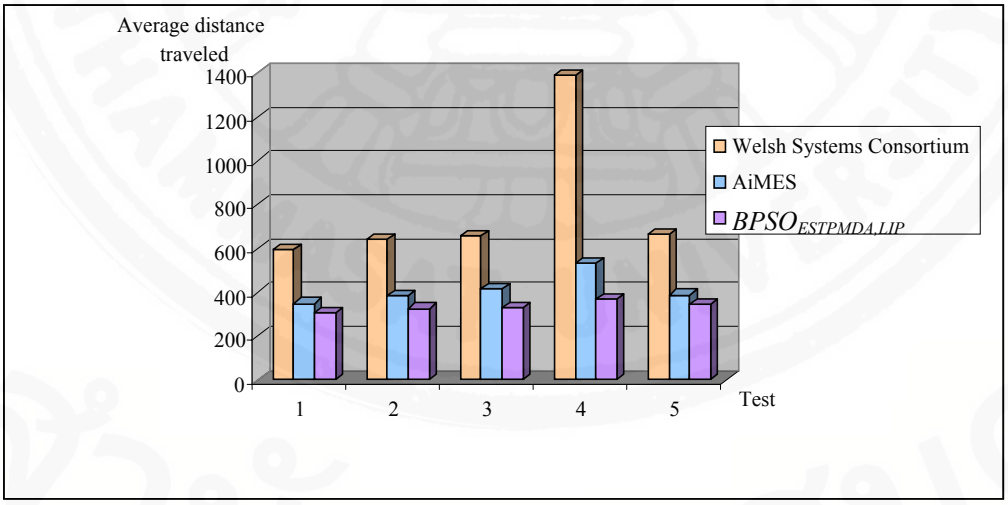

Figure 5.10 Graphical representation of results of PSO and previous solution techniques

5.6 Concluding Remark

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 in this chapter.

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 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 experiment 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.