PSO-Greedy Search Algorithm for Helicopter Mission Assignment in Disaster Relief Operations

Adriana Andreeva-Mori[1], Keiji Kobayashi[2] and Masato Shindo[3]
Japan Aerospace Exploration Agency, Mitaka, Tokyo, 181-0015 Japan

In the immediate aftermath of a large-scale disaster, optimal helicopter rescue mission assignment is critical to saving many lives. The current practice in the field is mostly human-centered, however. Japan Aerospace Exploration Agency has been developing a decision-support system for aircraft operation in order to promptly plan and execute rescue missions. The current research focuses on evacuation missions in particular and investigates the potential of particle swarm optimization (PSO) with integrated greedy search in aircraft resource management. We propose a robust particle model which can reflect various helicopter properties as well as evacuation mission characteristics. PSO parameters are modified and set based on numerical simulations and the values determined in this way are used in an optimization of disaster relief mission assignments based on real data obtained during the Great East Japan Earthquake and Tsunami in 2011. We initialize the swarm with a greedy search algorithm to increase the calculation speed and improve the quality of the results. It is shown that a hybrid PSO/ greedy search algorithm can be successfully adapted to disaster relief support systems and provide valuable analysis and decision-making information to the authorities in charge.

Nomenclature

a_m = mission assignment coefficient, 1 if mission *m* is assigned to a helicopter in the fleet; 0 otherwise

a_m^h = mission assignment variable, 1 if mission *m* is assigned to helicopter *h*; 0 otherwise

$c_{1,2,3}$ = acceleration coefficient

c^h = capacity of helicopter *h*

[1] Researcher, Operation Systems and Aviation Safety Technology Research Group, andreevamori.adriana@jaxa.jp
[2] Section Leader, Operation Systems and Aviation Safety Technology Research Group, kkoba@chofu.jaxa.jp
[3] Senior Researcher, Operation Systems and Aviation Safety Technology Research Group, shindo.masato@jaxa.jp

d_m = distance from the helicopter base to mission *m* demand location

$F^h_{cycle,i}$= set of missions assigned to vehicle *h* in cycle *i*

H = total number of rescue helicopters, with set K $= \{1, \dots, H\}$

M = total number of evacuation missions, with set I $= \{1, \dots, M\}$

n^h_{cycle} = number of cycles for helicopter *h*, h $= \{1, \dots, H\}$

n^h_{refuel}= total number of refuels for helicopter *h*

R_m = number of people to be transported (evacuees) in mission *m*

p_{bi} = global best found at current iteration

p_g = global previous best found by the swarm so far

p_i = individual previous best found so far

$r_{1,2,3}$ = random functions with values uniformly distributed in [0,1]

$\sigma(V_i)$ = sigmoid function

$t^h_{cyclei,}$= total time of missions in cycle *i*

t_{lim} = hard deadline to bring evacuees to the base (complete the rescue missions selected)

t^h_{load} = time necessary for one evacuee to board vehicle *h*

t^h_m = time required by helicopter *h* to complete rescue mission mt^h_{tl} = total time necessary for take-off and landing of helicopter *h*

t^h_{range} = maximum flight time available between two refuels, helicopter specific

t^h_{refuel}= time necessary for a helicopter to refuel, helicopter specific

V^h_{cruise}= Cruising speed of helicopter *h*

V_i = particle velocity at iteration *i*

V_{max} = maximum particle velocity

x_i = particle position at iteration *i*

x_{ij} = mission assignment model particle position, j^{th} mission assigned to i^{th} helicopter

ω = inertia weight

I. Introduction

Disaster relief operations include a large number of ground and air vehicles, personnel and authorities. In the aftermath of large-scale disasters, especially, timely information sharing and prompt planning and execution of rescue missions is critical for saving many lives. In this view, Japan Aerospace Exploration Agency (JAXA) has been developing an integrated aircraft operation system for disaster relief (D-NET 2), which overview is shown in Figure 1. D-NET 2 will combine satellites, manned and unmanned aircraft in direct search and rescue, an unprecedented endeavor not just in Japan, but also in the world. A key component of the system is going to be the optimal resource management block, shown in red in Figure 1, which will assign and schedule operations reconnaissance and rescue operations.

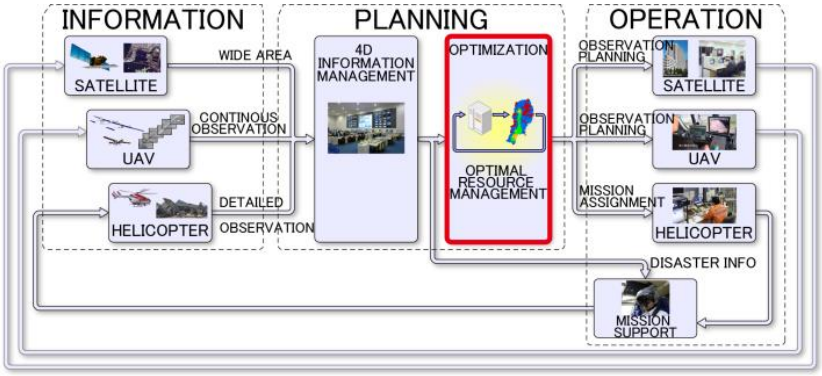

Figure 1. An overview of JAXA's integrated aircraft operation system for disaster relief.

In the preliminary design of the system, two blocks of the optimal management subsystem have been defined – information gathering (i.e. reconnaissance) optimization and rescue mission optimization. Details on reconnaissance optimization and satellite imagery application can be found in [1]. The general search and rescue flow is shown in Figure 2. It is assumed that right after the disaster strikes no information about the damage and rescue needs is available. Therefore, reconnaissance is necessary in the whole stricken area. Here, suppose this disaster area is divided into cells for easy understanding. Say information gathering is performed by helicopters which fly continuously over the disaster area without landing at the site. At this initial stage, cells with rescue needs are detected. At cell *i*, this needs location happened at time t_{loc}^{i}. At this point, the fact that there are people who need rescue is known, but their status and exact number is unknown. To obtain all this information, a helicopter has to land at the site, timing defined as t_{def}^{i}. This completes the information gathering stage. Next is the rescue stage, shown in blue in Figure 2. Here, only the evacuation process is considered. Assume that t_{res}^{i} is the time when rescue was completed at cell *i*. It is often said, however, that there is some critical time within which rescue should be completed before the number of survivors

drops drastically. Let us define some time t_{lim} as a hard rescue "deadline". When disaster relief resources available are sufficient to provide rescue to all people within t_{lim}, mission assignment optimization for all missions is enough. A common practice then is to assign resources in such a way as to minimize the time necessary to bring all survivors to safe places. For large-scale disasters, however, constraints on the resources become active and no feasible solution over all missions within t_{lim} is available.

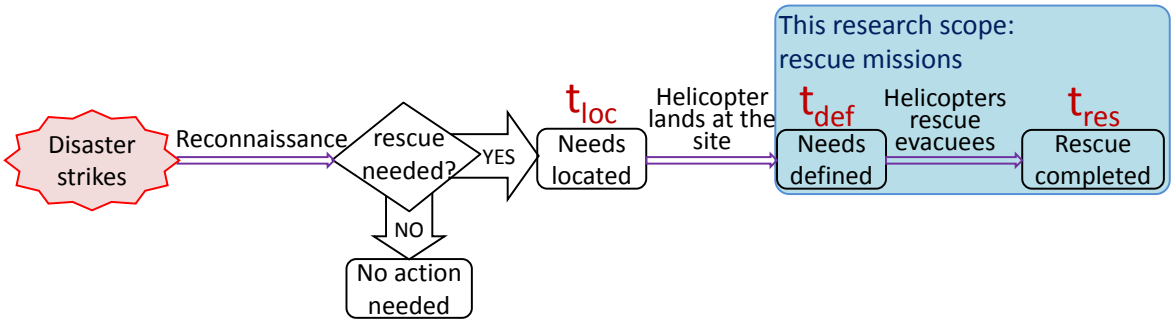

Figure 2. Search and rescue flow model implemented in JAXA's integrated aircraft operation system for disaster relief. The model has been generated based on interviews with professionals with disaster relief operation experience.

The purpose of this research is to develop and apply an optimization algorithm for helicopter mission assignment which provides optimal or sub-optimal results in near real-time environment. Here, we explore the potential of particle swarm optimization (PSO) algorithm combined with a greedy search algorithm as part of the optimization resource management subsystem preliminary design. The core of the algorithm proposed consists of PSO, with a greedy search being implemented in order to reduce the search space by introducing reasonably good initial swarm parameters.

Surprisingly, despite the obvious need for disaster relief management systems, research and development in this area is relatively scarce. A good overview of available technologies and design requirement to such a management system has been presented in NATO technical report published in 2008 [2]. In the report, however, no specific planning tool has been chosen as the most promising candidate for such a system. Researchers at universities and academic institutions, on the other hand, have analyzed particular helicopter planning strategies and tested them on various scenarios. Vehicle routing, in particular, has been a point of interest and, as such, several optimization techniques have been proposed. Dynamic vehicle routing approach has been reported as efficient [3], but the instances tested are limited to 10 vehicles and a hundred nodes. Problems of similar sizes are seen in many other research papers [4], [5], [6]. The main issue there lies in the representation of aircraft performance constraints, such as helicopter types

[7], transport capacity [8], flight range and refueling/maintenance times. In the case of a disaster, various vehicles gather in the stricken area, so the model should be able to easily reflect properties of a heterogeneous fleet. In the recent years Ozdamar et al. proposed a novel hierarchical clustering and routing for disaster relief [9] and provided a method to efficiently solve large scale problems with a solution deviating from the optimal one by less than 12% within a reasonable calculation time. The more complicated the algorithm is, the more parameters need to be set for sufficient results. Optimal ambulance assignment has also been investigated [10], but in this case the number of ambulances is scalable to the number of patient notes, whereas in the case of a large-scale disaster the rescue missions outnumber greatly the available vehicles.

In this research we propose helicopter mission assignment based on particle swarm optimization which requires relatively few parameter settings and allows for practically any vehicle performance constraints and a large number of missions and vehicles. To further improve the performance of the algorithm, we initialize the positions of the particle in the swarm based on a greedy search executed prior to the PSO itself. To the best of our knowledge, there has been no reported research on helicopter mission planning with PSO application.

This paper is organized as follows. Section II describes the problem and presents a mathematical model. It also gives and overview of particle swarm optimization and the particle representation, fitness function and parameter settings proposed for our problem. The results obtained here are used for the numerical test runs of a scenario which is based on data obtained during real disaster relief operations in the immediate aftermath of the Great East Japan Earthquake and Tsunami in 2011. Here, the greedy algorithm is also presented and merged into the PSO. The test scenario, aircraft assumptions and test results are shown in Section III. Section IV summarizes this work and outlines areas for future research.

II. Problem Setting

In this section we first describe the optimization problem and its properties and provide its mathematical formulization. We then consider particle swarm optimization which is the core of the algorithm for helicopter mission assignment, define the necessary parameters and investigate convergence and solution quality. After that, we introduce a greedy algorithm and discuss the advantages of its implementation in the problem.

A. Problem Description

Consider H rescue helicopters available and a total number of M evacuation missions which need to be completed. Generally speaking, this is an NP-hard problem. As JAXA's integrated aircraft operation system for disaster relief is meant mainly for large-scale disasters, we assume that the total number of missions exceeds the number of helicopters available ($H > M$). Each rescue mission m is characterized by its distance from a helicopter base (d_m) and the number of evacuees waiting to be transported (R_m) to a safe location. Not all missions can be processed by all helicopters, i.e. there are missions which require special equipment such as hoist or medical equipment. The time needed to complete a mission is helicopter specific. This assumption accounts for properties such as helicopter cruising speed, take-off and landing times, time necessary to board an evacuee, etc. Based on interviews with people involved in direct search and rescue, we impose a "one stop per trip" constraint. Theoretically speaking, a large helicopter can collect evacuees from several spots before returning to its base and thus make rescue more efficient. In practice, however, out of consideration for the safety of evacuees who are already onboard, several stops per trip are almost never allowed. Releasing this constraint will possibly improve the efficiency of relief operations, but at present we focus on developing a practical decision support system, so we opt for "one stop per trip" constraint. Besides, here we assume that each rescue mission is assigned to no more than one vehicle. There is only one helicopter base available, i.e. a single depot problem. Each helicopter has a certain capacity which cannot be exceeded. The flight range for each vehicle is also defined in terms of flight time. The time for a single refueling is also fixed. As noted by Wex et al. [11], such a problem is similar to the multiple Traveling Salesman Problem with salesman-specific travel times or the parallel-machine scheduling problem with unrelated machines non-batch sequence-dependent setup times. Apart from the work of Wex et al, according to Weng et al. [12] only one research paper on the above problem has been published, and it does not provide specific definition of the operational constraints. A major difference between all of the above work and this research is the hard constraint we impose on operation time. This constraint is analogous to the 72 hours limit for survival often stated as critical in immediate post-disaster management. Therefore, our system tries to propose a schedule which transports as many survivors as possible to a safe place (the helicopter base in this case) under a survival-time-based operational constraint, i.e. we do not necessarily complete all rescue missions and some nodes remain unvisited.

B. Mathematical Formulization

Below are the notation and the mathematical model of the problem described above. A cycle indicates the group of missions between two refuels.

M	Total number of rescue missions, with set I $= \{1, \dots, M\}$
H	Total number of rescue helicopters, with set K $= \{1, \dots, H\}$
R_m	Number of people to be transported (evacuees) in mission *m*
t_{lim}	Hard deadline to bring evacuees to the base (complete the rescue missions selected)
t_{refuel}^h	Time necessary for a helicopter to refuel, helicopter specific
t_{range}^h	Maximum flight time available between two refuels, helicopter specific
$t_{cycle,i}^h$	Total time of missions in cycle *i*
a_m^h	Mission assignment variable, 1 if mission *m* is assigned to helicopter *h*; 0 otherwise
t_m^h	Time required by helicopter *h* to complete rescue mission *m*
c^h	Capacity of helicopter *h* (maximum number of evacuees which can be transported by *h*)
n_{refuel}^h	Total number of refuels for helicopter *h*
n_{cycle}^h	Number of cycles for helicopter *h*, h $= \{1, \dots, H\}$. One denotes a block between two refuels.
$F_{cycle,i}^h$	Set of missions assigned to vehicle *h* in cycle *i*

$$\text{Maximize} \sum_{h=1}^{H} \sum_{m=1}^{M} a_m^h R_m \quad \text{(Fit)}$$

Subject to:

$$\sum_{i=1}^{n_{cycle}^h} t_{cycle,i}^h + t_{refuel}^h n_{refuel}^h \le t_{lim}, \qquad h = 1, \dots, H \quad \text{(Con1)}$$

$$a_m^h \in \{0,1\}, m = 1, \dots M; h = 1, \dots, H \quad \text{(Con2)}$$

$$\sum_{i=1}^{H} a_m^i \le 1, \quad m = 1, \dots, M \quad \text{(Con3)}$$

$$a_m^h R_m \le c^h, m = 1, \dots M; h = 1, \dots, H \quad \text{(Con4)}$$

$$\sum_{m \in F_{cycle,i}^h} a_m^h t_m^h = t_{cycle,i}^h, \qquad i = 1, \dots, n_{cycle}^h ; h = 1, \dots, H \quad \text{(Con5)}$$

$$t_{cycle,i}^h \le t_{range}^h , i = 1, \dots, n_{cycle}^h \ ; h = 1, \dots, H \quad \text{(Con6)}$$

$$n^h_{cycle} = n^h_{refuel} - 1,\ h = 1, \dots, H \qquad \text{(Con7)}$$

The fitness function (Fit) maximizes the number of people transported from each incident location to the base. Constraint (Con1) ensures that the total time of each helicopter does not exceed the hard deadline t_{lim}. Constraint (Con2) indicates that each mission can be assigned to one helicopter ($a^h_m = 1$) or not to be assigned at all ($a^h_m = 0$). Constraint (Con3) ensures that each rescue mission is assigned to no more than one helicopter. Constraint (Con4) imposes the helicopter capacity constraint and ensures that the number of evacuees transported by each vehicle does not exceed its capacity. Constraint (Con5) defines the flight time between two refuels as the sum of completion times of the missions assigned for the cycle. Constraint (Con6) makes sure that the fuel constraint for each helicopter is met. Constraint (Con6) shows that there is exactly one refuel between each cycle. We assume that all helicopters are fully-prepared for operation at the start of the rescue, so (Con7) is feasible.

C. Particle Swarm Optimization Overview

The particle swarm optimization is an evolutionary computation paradigm originally proposed by Kennedy and Eberhart in 1995 [13], [14]. The main idea behind the algorithm is that sharing simple information can result in preforming complicated optimization. The search process adopted in the original algorithm is analogues to the way bird flocks search for food. The basic flow is shown in Figure 3. Agents, called particles here, move in the hyper search space evaluating a fitness function with respect to their position. Each particle has a memory, too, and remembers its best position (individual previous best, p_i) found so far. Besides, particles share the information among themselves so that each particle is aware of the best position (global previous best, p_g) found in the swarm so far. Each particle's movement from one position to another is decided by its velocity, which is in turn determined based on its previous velocity and the best positions found by the particle itself and the swarm as a whole. As particles move in the search space, they evaluate their position on each iteration and when necessary, update p_i and p_g. The search continues until a stopping criterion is met, which can be either a sufficiently good fitness or a maximum number of iterations.

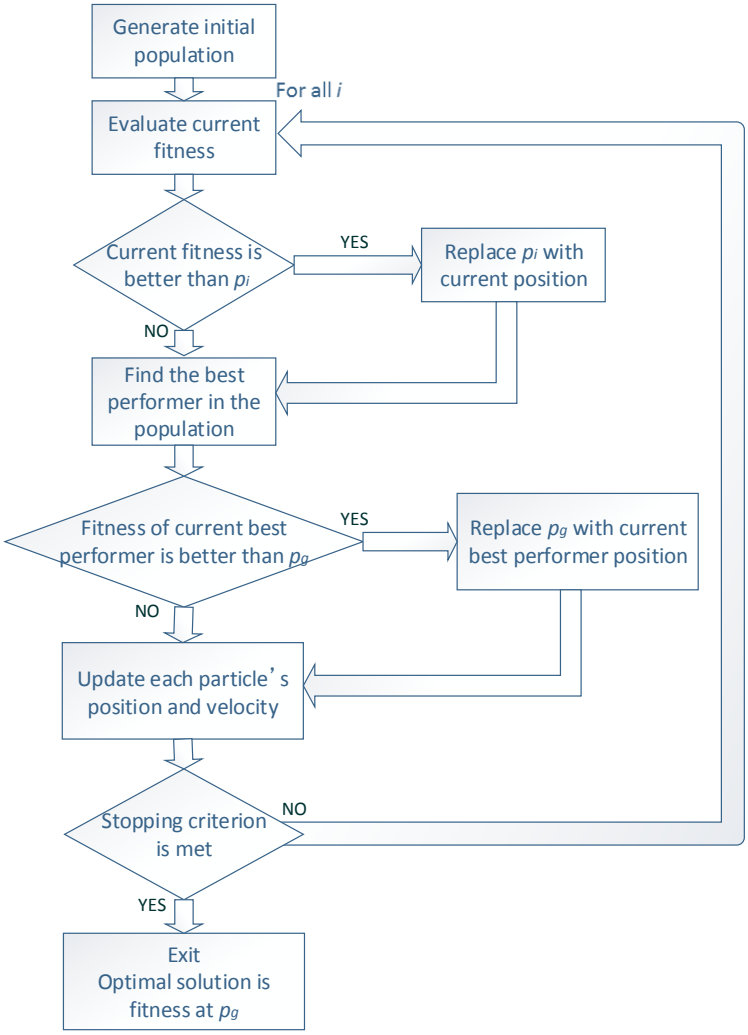

Figure 3. Basic PSO flow.

An important parameter for each particle is its velocity, as it determines the direction of the search. At iteration i, the velocity V_i and position x_i of the particle is updated as shown below.

$$\mathrm{V}_i = V_i + c_1 r_1 (p_i - x_i) + c_2 r_2 (p_g - x_i) \tag{1}$$

$$\mathrm{x}_i = x_i + V_i \tag{2}$$

Here, r_1 and r_2 are random functions with values uniformly distributed in [0,1] randomly generated at each iteration and for each particle; c_1 and c_2 are the parameters which determine how strong the particle will be pulled towards its best position and towards the swarm's best position, respectively. These two parameters are often called acceleration coefficients [15] of the cognitive and social part. V_i is kept within a certain range $[-V_{max}, +V_{max}]$ depending on the problem search space.

As seen from the description above, PSO has very few parameters which need to be set, which is the main reason why this method is adopted in the current research. Unlike most scheduling problems, the scale of disaster relief operations, mission distribution and resource availability are difficult to predict. Therefore, the optimization algorithm needs to be robust and with few specific settings.

Even though the original version of the PSO was developed for continuous problems, soon a discrete version was proposed [16]. The main differences between continuous and discrete PSO consist in the particle representation and parameter tuning. Since the mission assignment problem investigated in this research is based on the discrete PSO, its basic characteristics and alterations implemented here are discussed in detail below.

D. Particle Representation

Most researchers dealing with PSO applications to discrete problems agree that a key to successfully solving the problem is to find an appropriate representation of its solution by a PSO particle [17], [18]. In the standard version of the discrete binary PSO proposed by Kennedy and Eberhard [16], a particle is a sequence of 0 and 1, i.e. a bit string. This mapping is still widely used ([19], [20]) as it allows a relatively straightforward representation of the particle's position and velocity. A similar approach is adopted by Shi et al. [21], who consider an ordered integer sequence to represent the order of cities which define the route in a traveling salesman problem, and by Salman et al., who use define a particle as an M-dimensional vector to represent an M-task schedule [18]. In both of the above models, the particle is again seen as a vector. Chen et al. [22] use similar particle representation and propose a hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem. Constraints there pose a problem, though, and the obtained solution has to be checked for feasibility. The main limitation of these approaches is the difficulty to model assignments of multiple tasks (jobs) to the same task-performer. Liao et al. [17] solve this issue by designing a "job-to-position" representation for a particle. They consider a two-dimensional array with binary elements (0 or 1). The first dimension indicates the position of the job and the second one the job itself. When the element (i,j) is 1, the i^{th} job is assigned to the j^{th} position. Otherwise (i,j) is zero. In Liao et al.'s model, each job is assigned a position in the sequence. In this research, a similar modeling is adopted. For illustration purposes, consider 2 rescue helicopters and 6 rescue missions. When no constraints on the mission completion time are present or active, each mission is assigned to exactly one helicopter, so the sum of all elements in any row is exactly 1. The particle representing a sample mission assignment [1 2 3 4 5 6]→[1 2 1 1 2 2 1] is shown in Table 1. Therefore, the particle's position is defined as $X = [x_{ij}]$, for $i = 1$: number of helicopters, $j = 1$: number of missions, where $x_{ij} = 1$ if and

only if j[th] mission is assigned to the i[th] helicopter and $x_{ij} = 0$ otherwise. In the sample assignment shown in Table 1, $x_{11} = x_{22} = x_{31} = x_{42} = x_{52} = x_{61} = 1$ and the remaining elements$x_{12} = x_{21} = x_{32} = x_{41} = x_{51} = x_{62} = 0$. For reasons related to particle velocity definition and calculation, it is important that each mission is assigned to exactly one vehicle, i.e. no tasks are left uncompleted. In this research, however, an important operational constraint is the time available for rescue missions t_{lim}. This constraint leaves some tasks uncompleted, i.e. some evacuees are left behind. One way to reflect this property is to introduce another variable which shows whether a certain mission was included in final assignment. This will lead to zero rows in the original matrix. We have chosen another approach, however. We keep the form of the matrix shown in Table 1 and introduce a dummy vehicle. When a mission is assigned to the dummy vehicle, it will not be included in the rescue plan, i.e. all missions assigned to the dummy vehicle are the leftover from the assignment to the real fleet. By including a dummy vehicle in the array, the basic properties remain unchanged- the particle position representation is $X = [x_{ij}]$, for $i = 1: \text{number of helicopters}$, $j = 1: \text{number of missions}$, where $x_{ij} = 1$ if and only if j[th] mission is assigned to the i[th] helicopter and $x_{ij} = 0$ otherwise, and the sum of the elements in each row remains 1. In the example in

Table **2**, missions 1, 3 and 5 are assigned to helicopter 1, mission 6- to helicopter 2 and missions 2 and 4 are left out of the plan.

Table 1.Helicopter-mission assignment model when all missions are to be completed.

	Helicopter 1	Helicopter 2
Mission 1	1	0
Mission 2	0	1
Mission 3	1	0
Mission 4	0	1
Mission 5	0	1
Mission 6	1	0

Table 2. Helicopter-mission assignment with a dummy vehicle.

	Helicopter 1	Helicopter 2	Dummy
Mission 1	1	0	0
Mission 2	0	0	1
Mission 3	1	0	0
Mission 4	0	0	1
Mission 5	1	0	0
Mission 6	0	1	0

E. Particle Velocity

The particle's velocity should measure the probability for a certain mission to be assigned to a certain helicopter. The original equation (1) for continuous PSO results in velocity values such as the ones shown in Table 3. For discrete PSO, it is useful to convert these values by the sigmoid function from equation (3) as to have them in the range of [0,1], as suggested by Kennedy et al. [16], [14] and adopted in the scheduling problem optimization by Liao et al. [17].

$$\sigma(V_i) = \frac{1}{1 + exp(-V_i)} \tag{3}$$

Table 3. Original particle velocity as determined by Equation (3).

	Helicopter 1	Helicopter 2	Dummy
Mission 1	1.214	-1.624	0.410
Mission 2	-3.074	1.565	1.509
Mission 3	4.482	-2.516	-1.967
Mission 4	-3.393	0.655	2.738
Mission 5	1.158	1.516	-2.675
Mission 6	0.697	1.074	-1.771

Keeping in mind that by introducing the dummy vehicle, the sum of all probabilities in each row should equal 1, the particle velocity trail becomes as the one shown in **Table 4**. Higher values of V_i indicate higher probability of the particle choosing the value 1. For example, according to **Table 4**, at this point it is most likely that mission 3 will be assigned to helicopter 1, as the probability there is 0.834.

Table 4. Particle velocity considering sigmoid conversion and 1 mission-1 vehicle constraint.

	Helicopter 1	Helicopter 2	Dummy
Mission 1	0.502	0.107	0.391
Mission 2	0.026	0.489	0.485
Mission 3	0.834	0.063	0.103
Mission 4	0.020	0.404	0.576
Mission 5	0.462	0.498	0.039
Mission 6	0.428	0.478	0.093

F. Fitness Function and Constraints

An array representation of mission/helicopter has another main advantage, namely that it allows for a very straightforward definition of costs and rewards associated with each particular assignment. In the helicopter mission assignment problem, mission cost is equivalent to mission time. It includes the travel time from the base to the incident location and back, landing and take-off time, as well as time required to load/unload evacuees. Each mission time depends on the helicopter to which it is assigned. Some missions require special equipment such as hoist, for example,

not available on all aircraft. To reflect this constraint, we add penalty to the mission time of helicopters not capable of performing a certain mission. The order of the penalty exceeds feasible mission times, so in practice we assign infinite mission time to these aircraft. This adjustment is done to keep the convergence of the numerical simulation. Furthermore, at this stage of the research it is assumed that a rescue mission cannot be assigned to more than one vehicle, i.e. if the number of evacuees exceeds the vehicle's capacity, it will have to make multiple rounds in order to complete the mission, which is reflected in the mission time. On the other hand, the reward associated with the completion of each mission is the number of evacuees who have been transported from incident sites to the helicopter base.

In this research, for any assignment *P*, the fitness function is defined as the sum of all mission rewards which are assigned to a non-dummy vehicle. The goal is to maximize the fitness (p), i.e. the total rewards or evacuees saved.

$$\text{fitness(p)} = \sum_{i} Reward_i \tag{4}$$

for all missions *i* assigned to a non-dummy vehicle. This is equivalent to

$$\text{fitness(p)} = \sum_{m=1}^{M} a_m R_m \tag{5}$$

where

a_m $= \begin{cases} 1, & \text{if mission } i \text{ is assigned to a non} - \text{dummy helicopter} \\ 0, & \text{if mission } i \text{ is assigned to the dummy helicopter} \end{cases}$

R_m : Reward of completing mission *i*

M : Total number of evacuation missions

Let us consider the sample mission assignment shown in Table 2. Assume that the cost/reward table is as shown in Table 5. Here, we assume that helicopter 1 cannot perform mission 6, so its mission time is set to 1000. According to Table 2, missions 1, 3 and 5 are assigned to helicopter 1, mission 6- to helicopter 2 and missions 2 and 4 are assigned to the dummy vehicle. This optimization is done under the time constraint $t_{lim} = 30$. The total mission time of helicopter 1 is 25, of helicopter 2 - 25, and the reward contribution of each vehicle is 31 and 50 respectively, adding up to a total reward of 81. Assigning mission 3 to helicopter 2 would have let to the same total reward, but both solutions are equally well-acceptable in terms of the fitness function unless further constraints are induced.

Table 5. Sample mission time/ reward table. For a two-vehicle scenario, the first two columns indicate helicopter-specific mission times, the third column shows dummy vehicle mission times (in practice, any value would do), and the forth column indicates the number of people waiting to be rescued in each mission.

	Helicopter 1 (small)	Helicopter 2 (large)	Dummy (random)	Reward (evacuees)
Mission 1	3	3	3	1
Mission 2	15	15	11	5
Mission 3	2	1	17	10
Mission 4	18	6	1	15
Mission 5	20	5	16	20
Mission 6	1000	25	14	50

G. Particle Swarm Optimization Parameter Setting

As mentioned earlier, one of the great advantages of PSO is the small number of parameters which need to be adjusted. In order to determine optimal parameter settings, we use a random mission time/reward table for 5 aircraft (excluding the dummy one) and 100 missions. The initial solution was given not completely random. We used a random subset of all missions and generated a feasible solution based on mission times for each helicopter. A main property of the problem is that even though we do not know the optimal solution, we know multiple feasible solutions by assigning any missions which meet the constraints to our fleet as long as the operational time is less than t_{lim}. Further improvement in the initial solution is done by a greedy search algorithm which is going to be discussed in Section H.

General Parameters

The first parameter that needs to be set is the size of the population. Considering computational cost and based on our numerical experiments, we have decided on the value of 30 particles. Further increase does not necessarily improve the performance and increases computational time.

The second parameter is V_{max}. Even though early research showed than V_{max} is necessary to provide convergence, our experiments show that in this particular formulation of discrete PSO small values of V_{max} chop off unnecessarily the search space whereas sufficiently large values do not influence the performance significantly. Therefore, we opt for leaving out V_{max} altogether.

The third parameter is the so-called inertia weight ω, proposed by Shi and Eberhart [23]. Using inertia weight, the velocity originally defined in equation (1) is modified as:

$$V_i = \omega V_i + c_1 r_1 (p_i - x_i) + c_2 r_2 (p_g - x_i) \quad (6)$$

There have been reports that with proper inertia weight, the influence of V_{max} can be reduced and the convergence of the algorithm as a whole can be improved [23], [24], [25]. We have conducted various tests with both constant and variable inertia weight, but we could see no obvious improvement in the solution and convergence, so for simplicity no inertia weight is used here (i.e., $\omega = 1$).

Neighbor Topology Parameters

As for the neighbor topology, an interesting observation is made. As noted also by Liao et al. [17], whereas in the original discrete PSO the social part is obtained based on all particles' history so far, better results are obtained when only the current iteration best position is used. Liao et al, however, do not analyze this phenomenon or go into any detail about the possible reason. In order to investigate the most advantageous parameter setting, we introduce another part in the velocity trail previously shown in equation (1). Here, just like r_1 and r_2, r_3 is a random function defined in [0,1], and c_3 is the acceleration coefficient determining how strong the particle will be pulled towards the best position p_{bi} found at this iteration.

$$\mathrm{V}_i = V_i + c_1 r_1 (p_i - x_i) + c_2 r_2 (p_g - x_i) + c_3 r_3 (p_{bi} - x_i) \tag{7}$$

In particular, the values of c_1, c_2 and c_3 are of great interest. To investigate their optimal setting, we perform a series of numerical tests varying their values between 0 and 5, but always using the same initial particle population, in order to eliminate the influence of the initialization parameters.

[All Particles, All Iterations] Best Position Acceleration Coefficient c_2

As reported by some researchers already [17], standard acceleration coefficient values are around 2. Here, we first consider coefficients equal to 2, as to investigate the influence of each acceleration coefficient on the convergence and fitness value obtained. All computational tests were run 10 times for 2000 iterations with a swarm size of 30 particles. From the results shown in **Figure 4** it is seen that the best fitness value is obtained when the best position in the current iteration only is used instead of using all particles' history ($c_1 = 2, c_2 = 0, c_3 = 2$). The convergence improves slightly when $c_1 = 2, c_2 = 2, c_3 = 2$, but as seen from **Figure 4**, for all cases when $c_2 = 2$, the fitness value is worse than the simulations with $c_2 = 0$. A possible reason for this is convergence to a local minimum instead of global one.

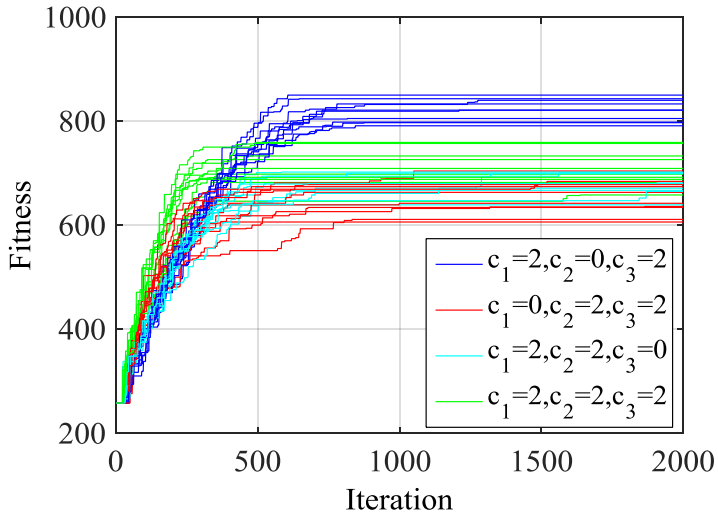

Figure 4. Influence of iteration best, global best and individual best.

Keeping sufficient variety in the particle population is a key to avoid premature convergence. Some researchers have proposed methods which measure the Hamming distances among the particles at each iteration and based on their values create "modulations" when needed. These techniques require more computational time, though, so for the purposes of our system should be avoided whenever possible. Examining the diversity of the swarm under the current problem formulation is not easy, however. Here, we use the Hamming distance between the missions assigned to the dummy vehicle to evaluate the swarm diversity. The Hamming distance is the number of positions with different values in the binary string. For example, in our case, when all 100 missions are assigned to the dummy vehicle, the dummy vehicle mission representation is

$$dummy_1 = [1\ 1\ 1\ \ldots 1]$$

When no missions are assigned to the dummy vehicle,

$$dummy_2 = [0\ 0\ 0\ \ldots 0]$$

and the Hamming distance between $dummy_1$ and $dummy_2$ is 100. **Figure 5** shows the Hamming distances for each iteration between each particle's the dummy-assigned missions and the best particle's dummy-assigned missions. The diversity of the swarm is expressed by the maximum difference in the swarm's dummy Hamming distances at each iteration, which is easily visualized by the width of the colored line in **Figure 5**.

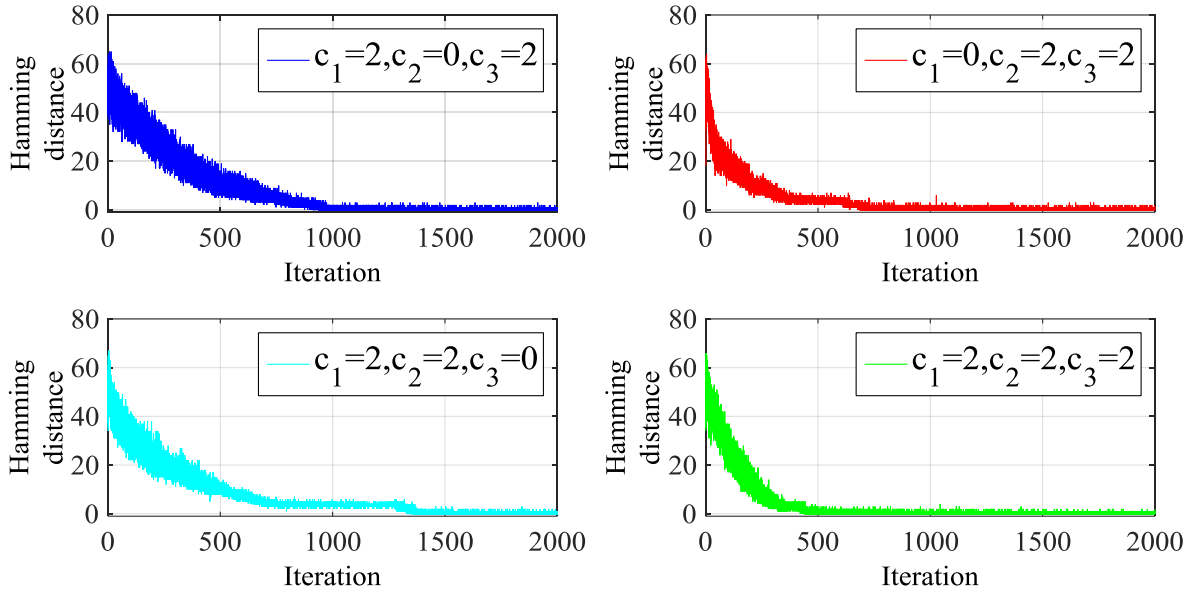

Figure 5. Hamming distances at each iteration. Large Hamming distance indicates swarm diversity, small Hamming distance shows convergence.

In all cases, the swarm tends to converge, thus reducing the average Hamming distance. In the cases of $c_2 = 2$, though, the width of the line decreases much sooner than in the case of $c_2 = 0$, i.e. the diversity of the swarm is significantly reduced at an earlier stage of the optimization. Therefore, it is safe to say that $c_2 \neq 0$ limits the exploration abilities of the swarm prematurely.

Based on the above analysis, in the rest of the research we adopt a zero value for c_2, using non-zero values for c_3 instead.

[Single Particle, All Iterations] Best Position Acceleration Coefficient c_1

To explore the dependence of the fitness and convergence on c_1 and c_3 values, two series of computational tests are performed- fixed value for c_3 ($c_3 = 2$) with varying c_1 (**Figure 6**), and fixed value for c_1 ($c_1 = 2$) with varying c_3 (**Figure 7**). The results from the first test series do not show a clear dependence of either the fitness value or the convergence properties of PSO. The right graph in **Figure 6** shows the minimum, maximum and average fitness values for 10 tests for each value of c_1. As mentioned earlier, higher fitness values means more evacuation missions accomplished within the time limit. The disaster relief mission assignment problem has to be solved in real time, so multiple runs are not a feasible option. In other words, the algorithm has to be robust. The results show, however, that

no clear "best value" of c_1 is available, which, put in another way, is equivalent to choosing any value of c_1 within reasonable range which suits the design needs. Here, we set $c_1 = 2$.

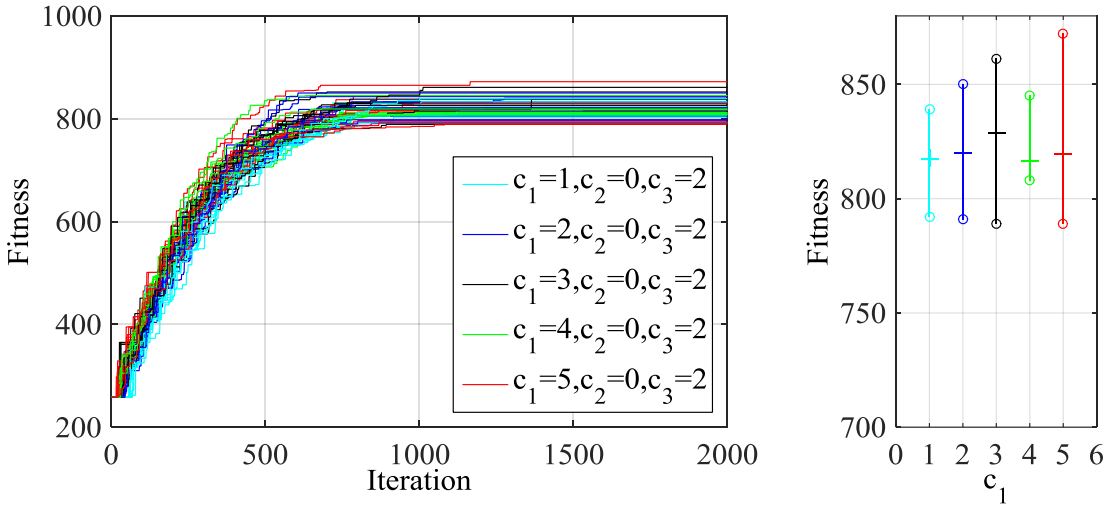

Figure 6. Dependence of the fitness and convergence properties on c_1.

[All Particles, Current Iteration Only] Best Position Acceleration Coefficient c_3

In the second test series, the effects of varying c_3 value are investigated. Here, unlike in the previous test series, a relatively straightforward dependence of the best fitness on the acceleration parameter is seen, as shown in **Figure 7**. Generally speaking, PSO's convergence improves for higher values of c_3, but this worsens the average best fitness value. PSO's performance for $c_3 = 0$ is significantly worse than that for any value, so it is concluded that non-zero values for c_1 and c_3 are appropriate. From this perspective, a smaller value is c_3 is chosen for the rest of the tests, i.e. $c_3 = 1$.

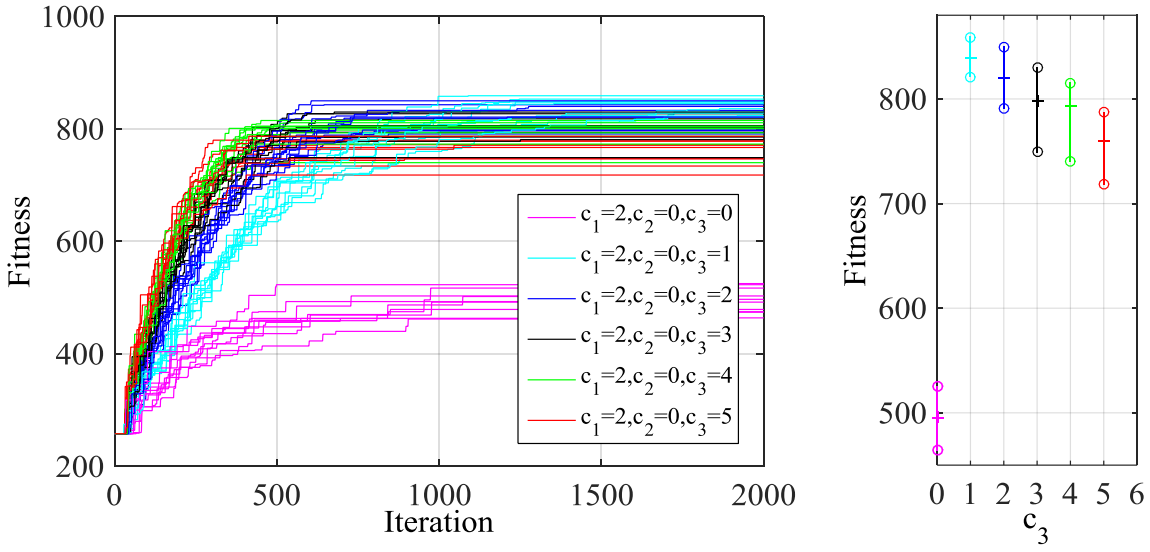

Figure 7. Dependence of the fitness and convergence properties on c_3.

H. Greedy Search Algorithm Implementation

The first step of the PSO algorithm requires initial solution generation. In many problems, no feasible solution is known at the start of the optimization. In this case, however, not just a feasible, but a sub-optimal solution can be generated based on a greedy algorithm. In order to efficiently use all information available for the disaster relief missions and at the same time minimize the amount of calculation, a new parameter called "profitability" is introduced. Profitability is defined for each helicopter-mission pair and is determined as the mission time/reward rate in each case. Using the values from Table 5, the profitability values become as shown in Table 6. Very small values such as the profitability of mission 6 when assigned to helicopter 1 indicate that this vehicle is incapable of performing the mission, either due to capacity of equipment constraints. The profitability defined here can easily by enhanced or substituted by incident's "severity", as proposed by [11]. A weighted sum of the profitability and severity of rescue missions might further bring the simulation closer to practical rescue operations. We believe that such an alteration will not change dramatically the performance of our algorithm, so this paper focuses on profitability only. We consider severity constraint implementation as a subject of future work.

Table 6. Mission time /reward table with profitability. Generally speaking, high profitability indicates high number of evacuees in a relatively close location, whereas low profitability shows few evacuees far from the base.

	Helicopter 1 (small)	Helicopter 2 (large)	Dummy (random)	Reward (evacuees)	Profitability (helicopter 1)	Profitability (helicopter 2)
Mission 1	3	3	3	1	0.33	0.33
Mission 2	15	15	11	5	0.33	0.33
Mission 3	2	1	17	10	5.00	10.00
Mission 4	18	6	1	15	0.83	2.50
Mission 5	20	5	16	20	1.00	4.00
Mission 6	1000	25	14	50	0.05	2.00

An overview of the algorithm is briefly shown below.

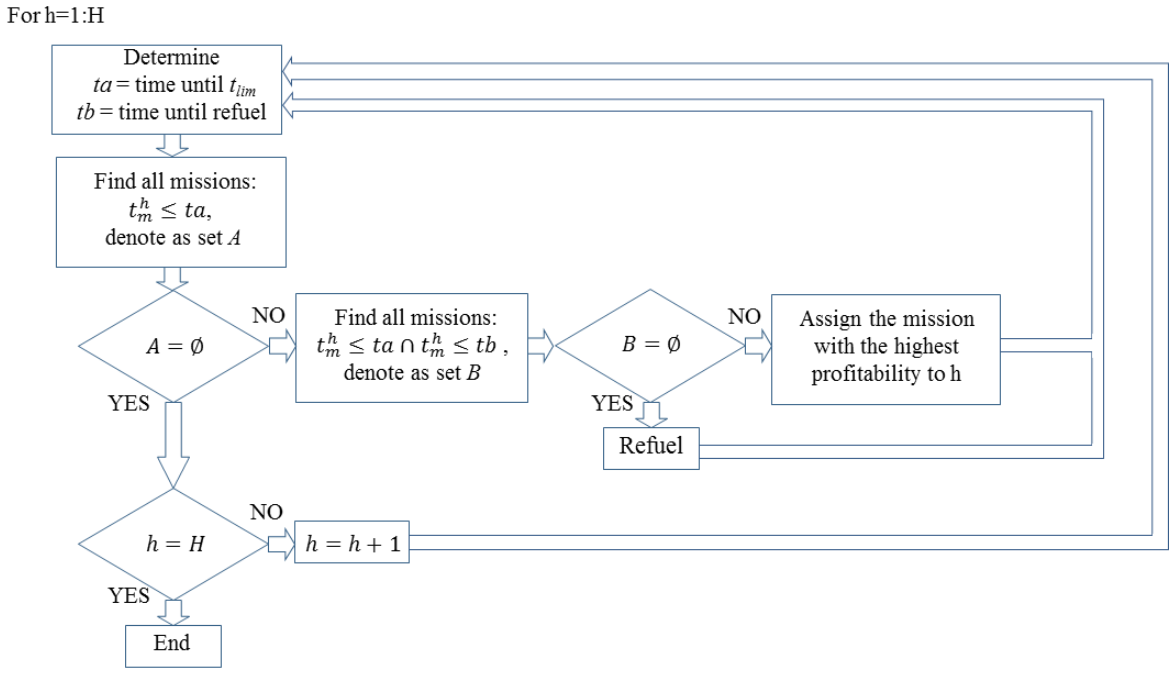

Figure 8. Overview on the greedy algorithm used to generate the initial population for the PSO mission assignment problem.

A greedy solution determined like this might not be optimal. Here, the greedy-search-based assignment is only used to initialize the positions of each particle in the PSO, i.e. to narrow-down the search space for better computational cost and improved PSO performance. As discussed in Section E, however, maintaining sufficient differences in the initial hamming distances is critical to avoid premature convergence to local optimum. To ensure the diversity of the swarm, the greedy-search is performed not over the whole mission set, but on a randomly chosen subset of the whole.

Once the particles initial positions are determined by the above greedy algorithms, the PSO described earlier is applied for finalizing the mission assignment optimization.

III. Test Scenario

A. Disaster Area and Evacuation Mission Distribution

The optimization method described above was tested for a disaster relief scenario based on data from the Great East Japan Earthquake and Tsunami. One of the main issues related with disaster relief mission planning is the insufficient amount of data and/or its unavailability. Because of privacy issues, information on the exact location and number of evacuees, for example is rarely released. Therefore, building a complete database from sporadic information is extremely difficult. Local authorities might have data about the missions performed during disaster relief, but such information is only released in summary reports such as [26]. To deal with this issue, JAXA has been developing scenarios about rescue mission location and number of evacuees based on general data released, interviews with local rescue authorities, as well as some probabilistic assumptions. JAXA conducted research on the activities of helicopters of local fire departments, medical assistance teams, self-defense forces, police departments and coast guard. Flight reports were obtained and used to build up the rescue mission data base which includes location and number of evacuees at each site. Where possible, interviews with pilots and rescue personnel involved in the relief missions were conducted to verify our assumptions. As a result, the scenario considered here has information on the location of the helicopter base (a single helicopter base in this case), each rescue mission, and number of evacuees which need to be transported to a safe place (here, the helicopter base). The modeled disaster area is Iwate Prefecture, with Hanamaki Airport being the helicopter base where all aircraft take-off and land. Altogether, 160 missions are considered. Each one involves 1 to 100 evacuees. It should be noted here that 100 evacuees cannot be transported at once by any of the available helicopters, so the biggest missions have been split into blocks of 25. The distribution by location is shown in Figure 11 and a histogram of the number of evacuees at each incident spot is shown in Figure 10. The missions with 95-100 evacuees describe transportation of people from facilities such as hospitals or schools. As for the mission time distribution, 180 helicopter-mission allocations have been penalized with mission time 1000 to show that the vehicles in question are incapable of conducting certain missions. A detailed histogram of the remaining mission times is shown in Figure 9.

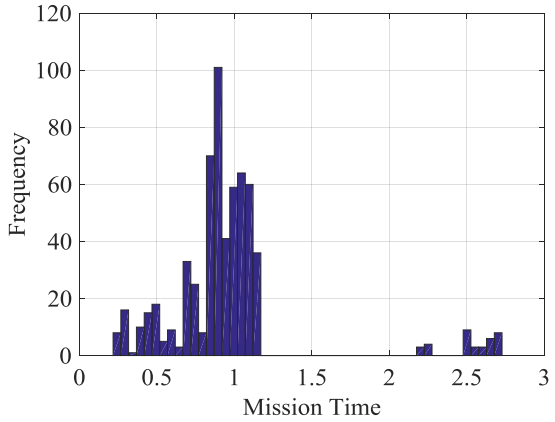

Figure 9. Histogram of the mission times potentially assigned to each of the five helicopters available in the fleet, excluding the penalized missions.

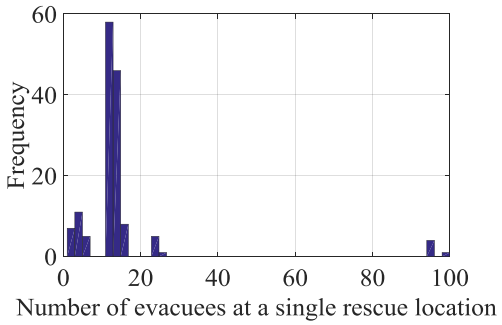

Figure 10. Histogram of the evacuees included in rescue missions in Iwate Prefecture

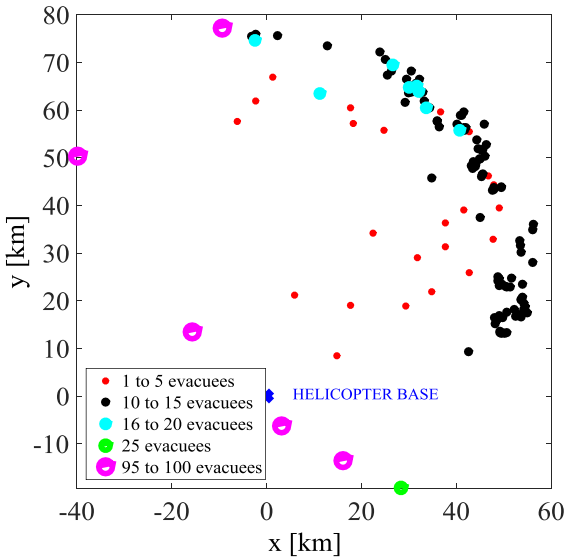

Figure 11. Mission distribution in Iwate Prefecture

The total number of evacuees who need to be transported to the base is 2153, i.e. if no time limit constraint is imposed, the best fitness will be 2153.

B. Aircraft Assumptions

We consider three helicopter types: small, medium and large with max flight time between two refuels 3h, 2h 45 min and 2h 30min, respectively. Maximum flight time can be set to reflect the aircraft performance and other extra requirements imposed by the operators. Each aircraft has a cruising speed of 100 kt and needs 30 min to refuel. Small

aircraft can transport up to 5 people, medium- up to 14 and large- up to 25 at a time. We assume a basic rescue fleet of five vehicles- 1 large, 3 medium and 1 small. We assume that each organization which is immediately involved in direct search and rescue can contribute 1 or 2 helicopters to the fleet of this prefecture.

Once the mission locations, number of evacuees and aircraft properties are defined, the cost/reward table needed for the optimization can be determined.

For each possible mission *m* being assigned to helicopter *h*, the cost index is determined as follows:

$$t_m^h = 2\frac{d_m}{V_{cruise}^h} + t_{tl}^h + R_m t_{load}^h \tag{8}$$

where

t_m^h : Time required by helicopter *h* to complete mission *m*(time [hours])

d_m : Distance from the helicopter base to mission *m* demand location

V_{cruise}^h : Cruising speed of helicopter *h*

t_{tl}^h : Total time necessary for take-off and landing of helicopter *h*

R_m : Number of evacuees in mission *m* who need to be transported

t_{load}^h : Time necessary for one evacuee to board vehicle *h*

Under the above assumptions, the shortest time necessary for a rescue fleet of 1 small, 3 medium and 1 large vehicles to complete all missions was estimated at 39 h 12 min. To do so, we alter the fitness function of our optimization problem. Instead of the fitness function presented in Section II (B) and later modified to the one shown in Section II (E), we minimize the sum of completion times of all rescue missions. The mathematical representation is shown below. The notation is the same as the one defined in Section II (B). Here, constraint (Con2') guarantees that all missions are going to be assigned to some vehicle.

$$\text{Minimize max}\left(\sum_{i=1}^{n_{cycle}^h} t_{cycle,i}^h + t_{refuel}^h n_{refuel}^h,\ \ h = 1, \dots, H\right) \tag{Fit'}$$

Subject to:

$$a_m^h \in \{0,1\}, m = 1, \dots M; h = 1, \dots, H \tag{Con1`}$$

$$\sum_{i=1}^{H} a_m^i = 1,\ \ m = 1, \dots, M \tag{Con2`}$$

$$a_m^h R_m \le c^h, m = 1, \dots M; h = 1, \dots, H \quad \text{(Con3')}$$

$$\sum_{m \in F_{cycle,i}^h} a_m^h t_m^h = t_{cycle,i}^h, \qquad i = 1, \dots, n_{cycle}^h;\ h = 1, \dots, H \quad \text{(Con4')}$$

$$t_{cycle,i}^h \le t_{range}^h\ , i = 1, \dots, n_{cycle}^h\ ; h = 1, \dots, H \quad \text{(Con5`)}$$

$$n_{cycle}^h = n_{refuel}^h - 1,\ \ h = 1, \dots, H \quad \text{(Con6`)}$$

It should be noted that 39 h 12 min is continuous operational time of the vehicles, i.e. night conditions and bad weather, as well as maintenance other that refueling, are not considered. Such detailed operational constraints are going to be implemented in the future.

C. Iwate Prefecture Mission Assignment Simulation

The greedy-search algorithm is applied to Iwate Prefecture scenario under $t_{lim} = 18$ h constraint. The hard time constraint was set to be less than 50% of the time necessary to complete all rescue missions. When a greedy search is done over all 160 missions, the fitness is 1275. To provide diversity of the swarm's particle position, 30 missions at random are excluded on each run (altogether 30 runs to generate initial positions for all particles are done), the flight ranges of all aircraft are set lower by 6 to 12 min, and $t_{lim} = 17.5$ h. The fitness then varies between 1061 and 1175. The positions defined this way are used to initialize the swarm.

Using the PSO setting parameters and the greedy-search-based particle initialization discussed in the previous section, we do 10 PSO runs under $t_{lim} = 18$ h constraint. The calculation results are shown in Figure 12. The average best fitness value is 1305.3, or given only 18 hours operational rescue time, 1305 out of all 2153 evacuees can be transported to a safe place. For the 10 tests run, the worst fitness value is 1301 and the best one is 1313. The total operational time histogram of all vehicles for all 10 runs (altogether $5\ vehicles \times 10\ test\ runs = 50$) is shown in the histogram in Figure 12. As seen from the figure, most vehicles assignment matches the 18 h limit, so it is concluded that the assignment distribution is feasible.

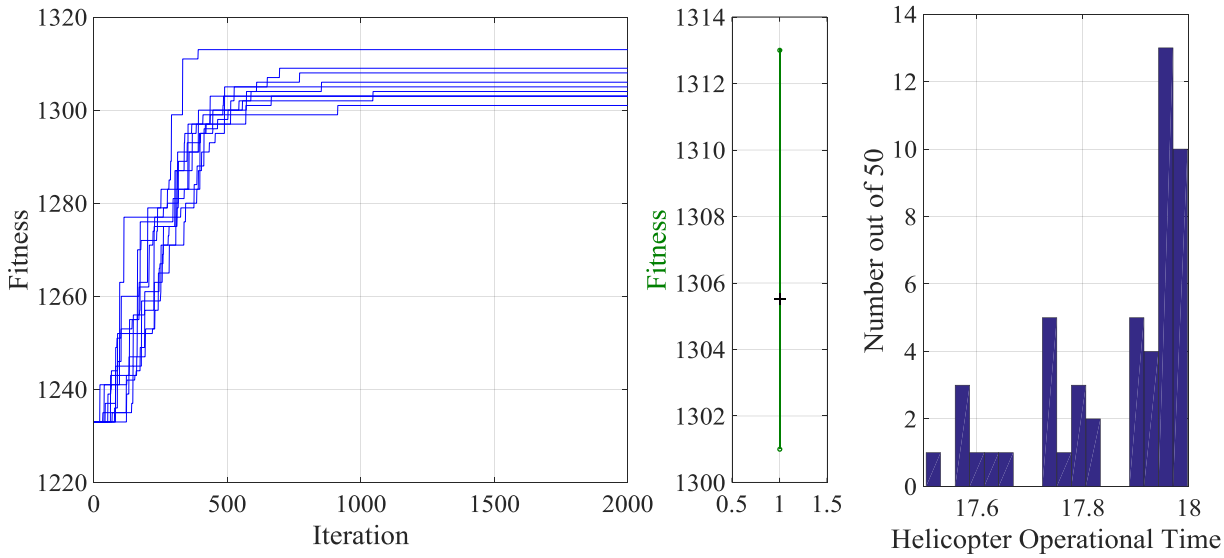

Figure 12. Iwate Pref. evacuation mission assignment simulation for 5 helicopters

The missions chosen to be executed differ slightly among iterations, but overall the solution is robust enough, as shown in Figure 13. We found that 61 missions out of all 160 missions are always assigned to a non-dummy helicopter while 48 missions are never performed, i.e. always assigned to a dummy vehicle. The robustness of the solution shows that the assignment is more or less governed by the greedy search solution, which in turn means that the vehicle assignment proposed by the system follows human operator's logic and can be applied easily in practice. However, because of the heuristic nature of PSO, the mission assignment differs slightly among the runs. Introducing the PSO improves the solution by 38, so the role of the PSO is to refine the results while complying with all constraints. The results also show that the "profitability" property of each mission-helicopter is an important parameter in mission assignment. Any improvement in the mission assignment which increases the fitness is crucial for the system, as it is equivalent to more lives saved in the immediate aftermath of the disaster.

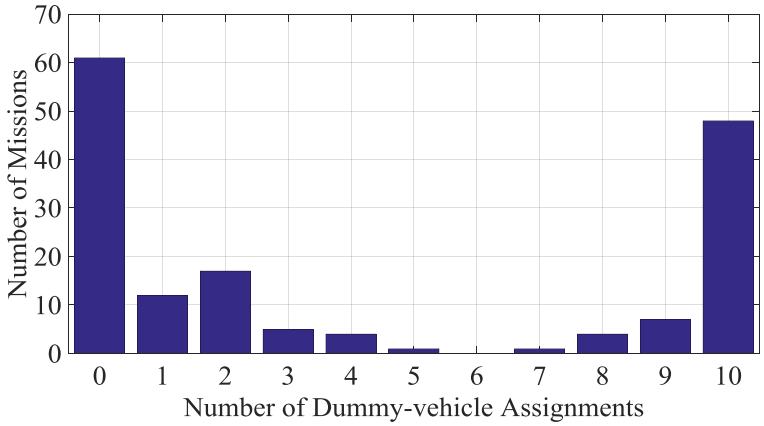

Figure 13. Dummy-vehicle assignments for all simulation runs. Zero indicates that a mission was never assigned to the dummy vehicle, 10 shows that a mission was assigned to the dummy vehicle in all 10 runs.

D. Comparison with Other Algorithms

To validate the performance of the proposed PSO-greedy search algorithm, we compare the obtained results and calculation times to a genetic algorithm solver and a mixed-integer linear programming solver. All numerical tests are done on the Iwate Pref. test scenario. Here, no fuel constraints are implemented. Instead, the total operational time is reduced from 18 hours to 15.5 hours. We set t_{lim} to 15.5 hours because even with absolutely no wasted time, at least 5 refuel sessions are necessary during the 18 hours operation, so the maximum "pure" operational time is 15.5 h for medium and large vehicles, and 15 h for the small vehicle. Therefore, "15.5 hours and no refuel" is a looser constraint then "18 hours and refuel". For reference purposes, we determine also the solution by the greedy algorithm as well. The greedy solution fitness value is 1346. Therefore, we ran three sets of numerical tests using the 1) developed PSO-greedy search algorithm under no refuel/ t_{lim}=15.5 hours constraint, 2) a genetic algorithm and 3) mixed-integer linear programming optimization. All experiments are done on Intel Core i7-3820 CPU@ 3.60 GH with 8.00 GB RAM running on Windows 7, using MATLAB®2015a, so the influence of simulation environment is eliminated.

Genetic Algorithm Implementation

For the problem model we refer to the work of Chu et al [27]. We use the *ga* function provided in the MATLAB® Global Optimization Toolbox, adapted to handle integer parameters. The parameters are set according to Deep at al [28]. The algorithm uses tournament selection, Laplace crossover and Power mutation adapted to handle integer constraints. We set the maximum number of generations to 1000 and the number of individuals that are guaranteed to survive to the next generation to 10% of the population number.

The genetic algorithm by itself fails to find a feasible solution to the problem. We believe that the reason for this outcome lies in the properties of the mission time/reward table. To reflect vehicle capacities and equipment, may of the mission times are penalized. GA fails to find a solution which does not include any penalized helicopter-specific missions. The issue with the way GA handles infeasible solutions has also been outlined by Kennedy et al. [14]. Simulations are done for population ranging between 10 and 300 and the maximum number of iterations and allowed constraint tolerances are varied, but in no case a feasible solution is obtained.

Next, we initialize the GA solution using the greedy algorithm presented in Section II (H). The generation of one set of suboptimal solutions (one population) takes on average 0.10 sec. The initial best fitness is 1330. We have to increase the population to 330 in order to have improvement in the fitness value. We run 10 trials under the same initial conditions, but only in 5 cases we observe an improvement to 1332 from the original value of 1330. This value is worse than that obtained by pure greedy search alone. When the initial solution includes the optimal greedy search solution, no change is observed. The authors believe this is due to the strong dependence of GA on the initial solution and its difficulty to avoid local minima. Many of the mission locations are close to each other, so there are many local minima. This characteristic makes GA inappropriate for mission assignment decision support systems like the one being developed at JAXA.

Mixed-integer Linear Programming Implementation

We solve the same problem as a mixed-integer linear program (MILP) referring to the general capacitated knapsack model presented by Garfinkel in [29]. Here, we apply the *intlinprog* function from MATLAB® Optimization Toolbox. It should be noted that a relaxation to a linear program gives an optimal objective value of -1358.85, which means that the optimal fitness value of our problem is somewhere between 1346 (the value obtained by the greedy search) and 1358. The stopping criteria for the PSO-Greedy algorithm is the number of iterations (set at 2000), while the stopping criteria for the MILP was calculation time (set at 600 sec). The results from 10 PSO-Greedy runs and a MILP run are shown in Figure 14. As seen from the figure, the PSO-Greedy algorithm performs better in terms of both calculation time and fitness value. MILP provides better results than GA discussed above, however.

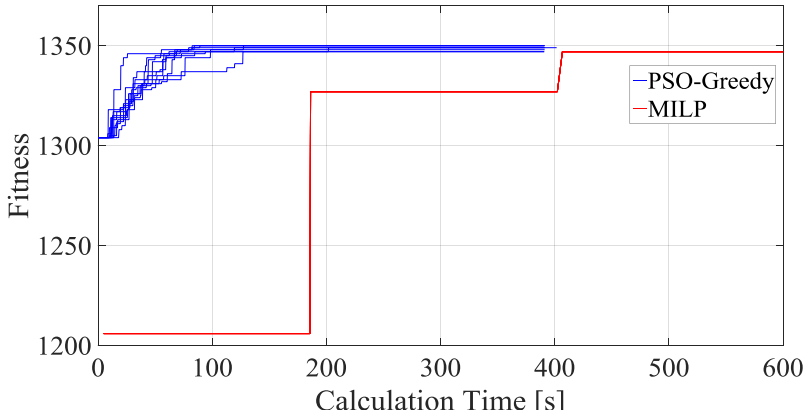

Figure 14. Comparison between the results obtained by PSO-Greedy and mixed-integer linear programming.

It should be noted here that the calculation time is a very important factor in the system design. To the best of our knowledge, we could not find any specific constraint on calculation time feasibility for mission assignment in immediate disaster relief. JAXA has participated in numerous disaster drills and we have measured the time between rescue mission request and aircraft assignment to be approximately 4 min when no decision-support tools are available. As seen from the results here, 240 s after the start of the optimization, PSO-Greedy has already converged to its best value (ranging from 1347 to 1350), while MILP fitness value is still 1327.

The optimization is terminated because the maximum number of iterations is reached, so for a complete analysis we set this constraint to 3600 sec. As the calculation time increases, the best fitness value also increases to 1352 (reached 683 sec after the start of the optimization and unchanged until the end). The comparison between all three algorithms showed that the proposed PSO-Greedy search provides a solution very close to the upper bound (determined by the relaxed LP problem) within the shortest calculation time. On the other hand, MILP can lead to better solutions, but requires more time. Therefore, it is shown that implementing the greedy search to initialize the swarm shortens the calculation time significantly and setting the PSO parameters appropriately allows the algorithm to avoid local minima.

IV. Summary and Future Work

This research showed that particle swarm optimization combined with a greedy search can be successfully applied to helicopter mission assignment in disaster relief planning. A particle model was formulated which could extremely easily reflect all aircraft performance constraints, as well as relief mission location and reward constraints. Using the

very same particle model, further operational considerations such as helicopter available equipment and crew can be implemented. A brief investigation on the appropriate parameter setting was presented and the obtained results were used to run a test scenario based on real disaster relief data from the Great East Japan Earthquake and Tsunami. The greedy search used to determine the positions of the initial swarm assured robust solution and easy-to-understand mission assignment, an important feature to the human-centered disaster management system.

At this point, necessary future work is considered in two main streams- algorithm performance improvement and further practical implementations.

The former includes investigation in possible automatic adjustment of the acceleration parameters discussed in this paper to avoid any dependence on the specific disaster scenario and in the same time assure robust and timely optimization. Further investigation into the initial solution provided by the greedy search and possible random positions to provide extended swarm diversity is also necessary. Testing the overall accuracy on any algorithm improvements should also involve benchmark instances. We are currently working on two major algorithm adjustments. The first one allows for helicopters to join and drop out of the fleet and the second splits up large missions into smaller once to allow for optimization with a large number of small aircraft.

The latter future work stream is related to trying out more scenarios and imposing further constraints on helicopter performance and mission assignment. At present, we consider evacuating injured/isolated people from affected locations only and no goods transportation is planned. Also, in the current version, any number of helicopter are allowed to refuel simultaneously at the base, while it is expected that imposing a constraint on the maximum number of refueling spots at each base would change the helicopter assignment. Further increase in helicopter bases is also a point of interest. Finally, testing all of the above in scenarios based on real disaster relief data and/or prediction is also a key to successful system design. Therefore, we are developing more accurate and complete disaster scenarios for several areas in Japan to verify our optimization.

Even though there are still some issues which need to be clarified and improved, the authors believe that this paper has shown the potential of a greedy-search improved particle swarm optimization to disaster relief helicopter mission assignment, so more work in this line of research is to continue. We will continue our work with professionals involved directly in immediate post-disaster operations to make sure the system under development can be used as a decision support tool in practice.

References

[1] Andreeva-Mori, A., Shindo, M., Kobayashi, K. and Okuno, Y., "Integrated Aircraft Operation System for Disaster Relief-Optimal Aircraft Operation Decision Support Tool (in Japanese)," in *JSASS 51st AIrcraft Symposium* , Takamatsu, Japan, 2013.

[2] NATOTask Group SAS-045, "Computer Based Decision Support Tool for Helicopter Mission Planning in Disaster Relief and Military Operations," RTO/NATO 2008, 2008.

[3] Hsueh, C.F., Chen, H.K., Chou, H.W., "Dynamic Vehicle Routing for Relief Logistics in Natural Disasters," *Vehicle Routing Problem*, T. C. a. H. Gold, I-Tech, Vienna, Austria, 2008, pp. 73-84.

[4] Barbarosoglu, G., Ozdamar, L. and Cevik, A., "An interactive approach for hierarchical analysis of helicopter logistics in disaster relief operations," *European Journal of Operations Research,* vol. 140, pp. 118-133, 2002.

[5] De Angelis, V., Mecoli, M., Nikoi, C. and Storchi, G., "Multiperiod integrated routing and scheduling of World Food Programme cargo planes in Angola," *Computers & Operations Research,* vol. 34, no. 6, pp. 1601-1615, 2007.

[6] Ozdramar, L. and Yi, W., "A dynamic logistics coordination model for evacuation and support in disaster response activities," *European Journal of Operations Research,* no. 179, pp. 1177-1193, 2007.

[7] Ozdramar, L., "Planning helicopter logistics in disaster relief," *OR Spectrum,* no. 33, pp. 655-672, 2011.

[8] Timlin, F., Marie, T. and Pulleyblank, W. R., "Precedence constrained routing and helicopter scheduling: Heuristic design," *Interfaces,* pp. 100-111, 1992.

[9] Ozdamar, L. and Demir, O., "A hierarchical clustering and routing procedure for large scale disaster relief logistics planning," *Transportation Research Part E* , no. 48, pp. 591-602, 2012.

[10] Andersson, T., and Varbrand, P., "Decision support tools for ambulance dispatch," *Journal of the Operational Research Society*, no. 58, p. 195–201, 2007.

[11] Wex, F., Schryen, G., Stefan, F., and Neumann, D., "Emergency response in natural disaster management: Allocation and scheduling of rescue units," *European Journal of Operational Research*, no. 235, pp. 697-708, 2014.

[12] Weng, M.X., Lu,J., Ren, H., "Unrelated parallel machine scheduling with setup consideration," *International Journal of Production Economics*, no.70, pp. 215-226, 2001.

[13] Kennedy, J., Eberhart, R. C., "Particle swarm optimization," *Proceedings of the IEEE international conference on neural networks*, pp. 1942-48, 1995.

[14] Kennedy, J. and Eberhart, R. C., *Swarm Intelligence*, Morgan Kaufmann Publishers, 2001.

[15] Poli, R., Kennedy, J. and Blackwell, T., "Particle swarm optimization: An overview," *Swarm intelligence,* vol. 1, no. 1, pp. 33-57, 2007.

[16] Kennedy, J., Russel, E. H., "A discrete binary version of the particle swarm algorithm," *Proceedings of the conference on systems, man and cybernetics,* pp. 4104-4109, 1997.

[17] Liao, C.J., Tseng, C.T. and Luarn, P., "A discrete version of particle swarm optimization for flowshop scheduling problems," *Computers & Operations Research,* vol. 34, pp. 3099-3111, 2007.

[18] Salman, A., Ahmad, I. and Al-Madani, S., "Particle swarm optimization for task assignment problem," *Microprocessors and Microsystems,* vol. 26, pp. 363-371, 2002.

[19] Agrafiotis, K. D. and Cedeno, W., "Feature Selection for Structure-Activity Correlation Using Binary Particle Swarms," *Journal of Medical Chemistry,* vol. 45, no. 5, pp. 1098-1107, 2002.

[20] Khanesar, M. A., Teshnehlab, M. and Shoorehdeli, M. A. "A novel binary particle swarm optimization," *Mediterranean Conference on Control and Automation,* pp. 1-6, 27-29 June 2007.

[21] Shi, X. H., Liang, Y. C., Lee, H. P., Lu, C. and Wang, Q. X., "Particle swarm optimization-based algorithms for TSP and generalized TSP," *Information Processing Letters,* vol. 103, no. 5, pp. 169-176, 2007.

[22] Chen, A.-l., Yang, G.-k., and Wu, Z.-m., "Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem," *Journal of Zhejiang University SCIENCE A,* vol. 7, no. 4, pp. 607-614, 2006.

[23] Shi, Y. and Eberhart, R. C., "A modified particle swarm optimizer," *Proceedings of the IEEE international conference on evolutionary computation,* pp. 69-73, 1998.

[24] Eberhart, R. C. and Shi, Y., "Particle swarm optimization: developments, applications and resources," *Proceedings of the 2001 Congress on Evolutionary Computation,* vol. 1, pp. 81-86, 2001.

[25] Chatterjee, A. and Siarry, P., "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization," *Computers & Operations Research,* vol. 3, no. 3, pp. 859-871, 2006.

[26] "*Aircraft Operations in Large-scale Disaster Relief* (in Japanese)," 2013.

[27] Chu, P., and Beasley, J., "A Genetic Algorithm for the Generalised Assignment Problem," *Computers Operations Research,* vol. 24, no. 1, pp. 17-23, 1997.

[28] Deep, K., Singh, K. P., Kansal, M. L., Mohan, C., "A real coded genetic algorithm for solving integer and mixed integer optimization problems", *Applied Mathematics and Computation*, vol. 212, no. 2, pp. 505-518, 2009.

[29] Garfinkel, R. S., Nemhauser, G. L., "*Integer programming*", vol. 4, New York, Wiley, 1972.