Recently, Artificial Intelligence (AI) technology has been applied to many applications. As an extension of Genetic Algorithm (GA) and Genetic Programming (GP), Genetic Network Programming (GNP) has been proposed, whose gene is constructed by directed graphs. GNP can perform a global searching, but its evolving speed is not so high and its optimal solution is hard to obtain in some cases because of the lack of the exploitation ability of it. To alleviate this difficulty, we developed a hybrid algorithm that combines Genetic Network Programming (GNP) with Ant Colony Optimization (ACO) with Evaporation. Our goal is to introduce more exploitation mechanism into GNP. In this paper, we applied the proposed hybrid algorithm to a complicated real world problem, that is, Elevator Group Supervisory Control System (EGSCS). The simulation results showed the effectiveness of the proposed algorithm.

Keywords: elevator group supervisory control system, genetic network programming, ant colony optimization, hybrid algorithm

1. Introduction

Evolutionary optimization technique is a subfield of artificial intelligence, whose search procedures are based on the mechanism of natural selection and genetic operations. Genetic Algorithm (GA) and Genetic Programming (GP) are known as typical evolutionary optimization methods. GA evolves strings and has been applied to many kinds of optimization problems. GP was devised later in order to expand the expression ability of GA by using tree structures. Also, a new graph-based evolutionary algorithm named Genetic Network Programming (GNP) [1, 2] has been proposed as an extension of GA and GP. Although GNP has been applied to many applications, one of the main obstacles in applying GNP to complex problems is high computational cost due to their slow convergence rate. The reason is that GNP doesn’t use much obtained information in the former generations to determine the most promising search direction. Consequently, GNP explores a wider frontier in the search space.

The behavior of GNP is characterized by a balance between exploitation and exploration in the search space. The balance is strongly affected by strategy parameters such as population size, crossover ratio, and mutation ratio, etc. Therefore, a common strategy in the literatures for dealing with the GNP’s slow convergence problem is to combine GNP with other techniques to balance exploitation and exploration.

The Ant Colony Optimization (ACO) is a recent population based optimization method inspired by the observation of a real ant colony and is based upon their information collective behavior [3–5]. The characteristics of ACO include positive feedback and the use of a constructive greedy heuristic. Positive feedback enables the rapid search of a global solution and constructive greedy heuristic helps find acceptable solutions as soon as possible [3]. ACO is extensible, robust, and easy to hybridize. But its defect is easily to trap into a local optimum [6].

In this paper, we developed a hybrid algorithm that combines GNP with ACO. Our motivation is to introduce positive feedback exploitation mechanism into GNP, whose rationale is to combine the advantages of the GNP with ones of ACO. Because of the evolutionary features of GNP, a hybrid algorithm is less likely to be trapped in a local optimum than ACO only. While due to ACO, a hybrid algorithm often converges faster than simple GNP. Generally speaking, the hybrid algorithm usually can explore a better tradeoff between computational cost and global optimality of the solution found.

The paper is organized as follows. Section 2 contains the description of the hybrid algorithm. Section 3 describes the application of the proposed algorithm to Elevator Group Supervisory Control System (EGSCS). Section 4 shows the simulation conditions and results. Finally, some conclusions are devoted in Section 5.
2. Extended Algorithm GNP with ACO

2.1. Basic Structure of GNP

Figure 1 shows the basic structure of GNP. As an extension of GA and GP, GNP has been proposed to have a network structure where functional nodes are connected by directed branches. GNP program is composed of one start node and plural number of judgment nodes and processing nodes. The start node has no functions and no conditional branches. Judgment nodes have decision functions with conditional branches. Each judgment node returns a judgment result and determines the next node to be executed. Processing nodes work as action functions. After the start node, the current node is transferred according to the node connections and judgment results. In processing nodes, actions are conducted to environments. The node transition begins from a start node, and there is no terminal node.

As shown in Fig. 1, GNP can be illustrated by its “Phenotype” and “Genotype.” Phenotype GNP shows the directed graph structure where nodes are connected by directed branches, and Genotype GNP provides the chromosomes encoded into bit-strings. The structure of the gene of node \( i \) is set as shown in Fig. 1. There are node genes and connection genes in the genes of the nodes. \( NT_i \) is the allele of node type (0: start node, 1: judgment node, 2: processing node). \( NF_i \) indicates the function label which is defined in the library. \( d_i \) is the time delay of node \( i \). \( C_{ik} \) denotes the \( k \)-th connecting node number from the current node \( i \) and \( d_{ik} \) shows the time delay of its transition.

In evolutionary computation, each individual is evaluated in the problem environment. Then the offspring who can survive at the next generation is determined by fitness. Crossover, Mutation, Tournament Selection and Elite Preservation are used as the genetic operators of GNP. The outline of evolution is described as follows:

1. Generate initial population and calculate the fitness of initial population;
2. Execute tournament selection, genetic operations and generate new individuals for the next generation;
3. Calculate the fitness of the new individuals;
4. Repeat 2-3 until the terminal condition meets.

2.2. Basic Concepts of ACO

The ACO imitates the behaviors of ants. Consider an ants’ nest and a source of food. The ants manage to find the shortest path between the food and nest with the aid of the pheromone. The pheromone is the material deposited by the ants, which is used to mark the ants’ trails as they walk, thereby guiding the next movement.

The ants are guided by the amount of the pheromone. Initially, a group of ants explores the surface without a predetermined direction. After food is found, the ants go back to the nest and lay down the pheromone. After a short time period, the differences among the pheromone deposits in the routes are enough to influence the movement of the new ants. New ants tend to choose a path having the high intensity of the pheromone. Also due to evaporation, the old trails, which are not reinforced by the new ants for a long time, will eventually vanish. This produces a feedback to the system, which contributes and promotes the use of the best path. Any trail that is rich of the pheromone will thus become the target path.

Basically ACO uses two functions to guide the search toward the optimal solution when it is applied to the traveling salesman problem [4]. Let \( H^n(r,a) \) be the intensity of the pheromone on edge \((r,a)\) of the trail at time \( n \). After all ants have generated the tours, the pheromone intensity is updated at time \( n \)

\[
H^n(r,a) = (1 - \rho)H^{n-1}(r,a) + \sum_{m \in M} h^a_m(r,a) .
\]

where \( \rho \in (0, 1) \) is a parameter of evaporation. \( M \) is the set of suffixes of ants in the colony and \( h^a_m(r,a) \) is the intensity of the pheromone laid by the ant \( m \) at time \( n \).

Let \( \eta(r,a) \) be the visibility between the vertex \( r \) and \( a \). \( \eta(r,a) \) is the inverse of the distance in the traveling salesman problem. The probability that the ant \( m \) chooses \( a \) as the next vertex when it is at the vertex \( r \) at time \( n \) is given by

\[
p^a_m(r,a) = \begin{cases} 
H^n(r,a)^\alpha \eta(r,a)^\beta & \text{if } a \in M_m \\
0 & \text{otherwise}
\end{cases}
\]

where \( M_m \) is the set of vertices not visited yet by the ant \( m \), \( \alpha \) and \( \beta \) are two parameters that control the relative importance of the trail versus visibility.
### Table 1. Relationship between GNP and ACO

<table>
<thead>
<tr>
<th>GNP</th>
<th>Ant Colony Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individuals</td>
<td>Ants</td>
</tr>
<tr>
<td>Branch between nodes</td>
<td>Edge</td>
</tr>
<tr>
<td>Transition</td>
<td>Tour</td>
</tr>
<tr>
<td>Fitness</td>
<td>Total distance</td>
</tr>
</tbody>
</table>

The procedures of GNP with ACO can be summarized as follows:

- **Initialization phase**: We first determine the number of each kind of nodes, which is identical in all individuals. Then, the connection genes $C_{i1}, C_{i2}, \ldots, C_{ik}, \ldots$ are set randomly.

- **Fitness evaluation phase**: After all individuals run, the fitness value of each individual is calculated. Fitness functions are used to evaluate the performance of the individuals and also used to update the pheromone intensity. In GNP with ACO, tournament selection and elite selection are used.

- **Pheromone updating phase**: The pheromone on each branch of GNP is calculated by the accumulation and vaporization using a pheromone update function.

- **Genetic operation phase**: During this phase, the generations are divided into common generations and special generations. We set a special generation every 10 common generations. In each generation, tournament selection and crossover are carried out conventionally, where the pheromone is considered as one of the attributes of branches of GNP. Mutation is done as usual in common generations. In special generations, the new individuals are produced using the information of the pheromone instead of mutation. The more pheromone on the branch is, the higher probability of the connection of the branch would appear in the new GNP individual.

- **Convergence determination phase**: The whole process is completed when the terminal condition meets.

#### 2.3.1. Basic Structure of GNP with ACO

Optimization techniques can be classified into two categories: global search methods and local search methods. A global search method explores the global search space without using local information about promising search directions. Consequently, they are less likely to be trapped in local optima, but their computational cost is higher. The global search methods often focus on “exploration,” but less attention to “exploitation.” Like most global search methods, GNP is not easily trapped in local minima. However, it converges slowly. Combining GNP with ACO, guarantees the rapid search of GNP by using the information obtained in the former generations. GNP with ACO converges on an optimal solution through the accumulated and vaporized pheromone information. The hybrid algorithm, GNP with ACO explores a better trade-off between computational cost and global optimality of the solution found.

There are many similarities between simple GNP and ant colony. Table 1 shows their similarities. Each individual corresponds to each ant. The transition of GNP is the tour made by the ants. Branch between nodes of GNP represents the edge of the trail. GNP is evaluated by the fitness function, which quantifies the optimality of a solution. The solution of GNP whose fitness is the best is equivalent to the shortest path the ant can find. Fig. 2 shows the flow chart of the proposed GNP with ACO.

#### 2.3.2. Calculation of Pheromone

The pheromone on $k$-th branch of node $i$ connecting to node $a$ of individual $m$ in the $n$-th generation $h^m_{n}(i,k,a)$ is calculated as follows using fitness function,

$$h^m_{n}(i,k,a) = \begin{cases} \frac{F^n - f^n_m}{F^n} , & x(i,k) = a \\ 0 , & x(i,k) \neq a \end{cases}$$

where

- $h^m_{n}(i,k,a)$: pheromone on the $k$-th branch of node $i$ connecting to node $a$ of individual $m$ in the $n$-th generation
- $f^n_m$: fitness of individual $m$ in $n$-th generation
- $F^n$: the worst fitness of the individuals in the $n$-th generation
- $x(i,k)$: node number connecting from the $k$-th branch of node $i$.

Then, a pheromone update function, that is, the function for calculating the total pheromone on the $k$-th branch of node $i$ connecting to node $a$ in the $n$-th generation is obtained as follows:

$$H^n(i,k,a) = (1-\rho)H^{n-1}(i,k,a) + \sum_{m\in M} h^m_{n}(i,k,a)$$

Vol.11 No.9, 2007

Journal of Advanced Computational Intelligence and Intelligent Informatics

1151
where

\[ H^n(i,k,a): \text{pheromone on the } k\text{-th branch of node } i \text{ connecting to node } a \text{ in the } n\text{-th generation} \]

\[ \rho: \text{parameter of evaporation} \]

\[ M: \text{set of suffixes of individuals.} \]

Finally, the probability of connecting the \( k \)-th branch of node \( i \) to node \( a \) in the \( n \)-th generation \( P^n(i,k,a) \) is obtained as follows using the pheromone \( H^n(i,k,a) \).

\[ P^n(i,k,a) = \frac{H^n(i,k,a)}{\sum_{a \in A(i,k)} H^n(i,k,a)} \]  \hspace{1cm} (5)

where

\[ P^n(i,k,a): \text{probability of connecting the } k\text{-th branch of node } i \text{ to node } a \text{ in the } n\text{-th generation} \]

\[ A(i,k): \text{set of node numbers connecting from the } k\text{-th branch of node } i. \]

At a special generation, the offspring is produced by \( P^n(i,k,a) \).

3. Application of GNP with ACO to EGSCS

3.1. Review of EGSCS

Elevator Group Supervisory Control Systems (EGSCS) [7] are control systems that manage multiple elevators in a building in order to efficiently transport the passengers. The systems assign a service car for a new passenger in a hall. The assignment is a kind of real-time scheduling problem for transportation systems. The performance of EGCS is measured by several criteria such as the average waiting time of passengers, the percentage of passenger’s waiting more than 60 seconds, and power consumption [8], and EGSCS manages elevators to minimize the above criteria; it is, however, difficult to satisfy all criteria at the same time.

The passenger traffic pattern in modern buildings with EGCS varies considerably throughout a typical business day. Different traffic patterns have very different effects, and each pattern requires its own analysis. So, we study the systems with three types of traffic patterns, i.e., “Regular Traffic,” “Up-peak Traffic” and “Down-peak Traffic.”

Destination Floor Guidance System (DFGS):

In order to obtain more accurate information on passenger’s destination, Destination Floor Guidance System (DFGS) [9,10] has been developed so that passengers can input their destinations at elevator halls. At each floor there is a keypad where the passenger selects which floor they wish to go to. The system then guides the passenger to an elevator that will be stopping at their destination floor. There are no floor buttons inside the cage.

Double-Deck Elevator Systems (DDES):

Recently, for improving the capability of EGCS, the Double-Deck elevator [11], where two cages are connected with each other, is expected as the next generation elevator. It allows that the passengers at two consecutive floors could be serviced simultaneously. In DDES, a passenger can in principle board either the lower or upper cage. Here, instead of “upper cage” and “lower cage,” we also use the terms “self cage” and “other cage” in a more general sense. Obviously, Double-Deck Elevator Systems (DDES) become more complex in their behaviors than conventional Single-Deck Elevator Systems (SDES).

DDES has specific features as shown below, and their careful consideration is expected to improve the performances of the group supervisory control.

One Cage Service: Self cage stops without any service while the other cage serves passengers at the floor. This situation causes not only the deterioration of transportation capability but also psychological stress to passengers.

Coincident Service: Both cages serve passengers at a stop. Coincident service can contribute to improve both transportation capability and comfortable riding.

Separate Riding for Identical Destination: Passengers for the identical destination ride on both cages. Therefore, the transportation capability deteriorates by two stops at the same floor.

3.2. DDES with DFGS Using GNP with ACO

Double-Deck Elevator Systems with Destination Floor Guidance Systems are so complex in that the assignment of the optimal cage to each new hall call is fairly difficult due to the enormous amount of information obtained. GNP with ACO is expected to be appropriate for the assignment problem in elevator systems. The reason is that: GNP with ACO can realize a rule based Elevator Group Supervisory Control System (EGCS) due to its directed graph structure with judgment nodes and processing nodes, which makes EGCS more flexible in different traffics. And also, EGCS can be built by an evolutionary method with mutation, crossover and selection, which could develop new efficient and effective rules that elevator experts can not imagine as well as saving the time for designing EGCS.

The structure of Double-Deck Elevator System (DDES) with Destination Floor Guidance System

![Fig. 3. Structure of DDES with DFGS using GNP with ACO.](image-url)
3.2.1. Evaluation Items

In our proposed method the following 12 evaluation items are defined to construct GNP considering the features of DDES with DFGS.

The first 6 evaluation items are common in SDES and DDES, which are indicated by the lower suffix \(sd\), and the remaining 6 ones are defined for DDES according to its specific features, indicated by the lower suffix \(d\).

- \(AT_{sd}\): Predicted arrival time of the assigned hall call to the self cage including the incremental arriving time of the already registered hall calls to the self cage
- \(AET\_{sd}\): Maximum of the arrival time plus elapsed time since the assignment of the hall calls to the self cage
- \(NP\_{sd}\): Number of passengers in the self cage
- \(NC\_{sd}\): Number of assigned hall calls to the self cage
- \(RR\_{sd}\): Predicted riding rate (passenger number/cage capacity) of the self cage when the self cage arrives at the assigned hall call including the incremented riding rate of already registered hall calls to the self cage
- \(CHC\_{sd}\): Check whether the emerged hall call coincides with the cage calls of the self cage
- \(AT\_i\): Sum of the incremental predicted arrival time of the already assigned hall calls to the other cage
- \(AET\_i\): Maximum of the arrival time plus elapsed time since the assignment of the hall calls to the other cage
- \(DNP\_d\): Difference of the number of passengers between the self and other cage
- \(DNC\_d\): Difference of the number of assigned hall calls between the self and other cage
- \(CCS\_d\): Check the coincident service
- \(CSR\_d\): Check the separate riding for identical destination

3.2.2. Assigning Algorithm

In the GNP with ACO controller part, firstly, the information on the elevator system is transferred to the System Information Judgment Part. In this part, the new hall call is classified based on three following terms, the degree of the variance of the elevator positions \(VP\_{sd}\), the origin floor and direction of the new hall call \(EF\_{sd}\) and the destination floor of the new hall call \(DF\_{sd}\). \(VP\_{sd}\) is used for the binary judgment whether the degree of the variance of the elevator positions is less than the average one over past 5 minutes or not. \(EF\_{sd}\) is used for the judgment of the origin floor and direction of the new hall call with 5 branches, \{Base, General-Low-Up, General-Low-Down, General-High-Up, General-High-Down\}. \(DF\_{sd}\) is used for the judgment of the destination floor of the new hall call with 3 branches, \{Base, General-Low, General-High\}.

Secondly, a candidate cage with the minimum value of the evaluation function is selected in the Cage Selection Part. A candidate cage is selected in this part by the following equation. First, the cage evaluation function \(e(i)\) of cage \(i\) is calculated by Eq. (6).

\[
e(i) = \sum_{p \in P} w_X(p) \cdot x_p(i) \quad \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots (6)
\]

where

- \(P\): set of suffixes of nodes transited in the cage selection part (\(P\) is determined by node transition)
- \(w_X(p)\): weighting of evaluation item \(X\) at the cage selection node \(p\) (\(w_X(p)\) is optimized during evolutionary process)
- \(x_p(i)\): normalized value of evaluation item \(X\) of cage \(i\) at the cage selection node \(p\).

The normalized value \(x_p(i)\) is calculated by Eq. (7)

\[
x_p(i) = \frac{X_p(i)}{X_{aveMax}} \quad \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots (7)
\]

where

- \(X_p(i)\): value of evaluation item \(X\) of cage \(i\) at the cage selection node \(p\)
- \(X_{aveMax}\): maximum value of averaged evaluation item \(X\) over past 5 minutes among cages.

The reason of using the normalized value of \(x_p(i)\) is that different evaluation items have different scales. As for the evaluation item \(\{CHC\_{sd}, CCS\_d\}\), \(x_p(i) = 0\) if satisfied, and \(x_p(i) = 1\) if not satisfied. It is reversed in the case of \(\{CSR\_d\}\). Finally, the candidate cage \(d\) is selected by Eq. (8)

\[
d = \arg\min_{i \in I} e(i) \quad \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots (8)
\]

where, \(I\): set of cage IDs.

Then, the selected candidate cage \(d\) is evaluated again by individual evaluation items one by one to confirm whether it is the optimal one or not in the Cage Judgment Part. In cage judgment nodes in this part, the binary judgment like Eq. (9) is carried out except for \(\{CHC\_{sd}, CCS\_d, CSR\_d\}\).

\[
y_j(d) \leq r_j^Y \quad j \in J \quad \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots (9)
\]

where

- \(J\): set of suffixes of nodes in the cage judgment part
- \(y_j(d)\): normalized value of evaluation item \(Y\) of cage \(d\) at the cage judgment node \(j\)
- \(r_j^Y\): threshold parameter of evaluation item \(Y\) of the cage judgment node \(j\) (\(r_j^Y\) is optimized during evolutionary process).
- \(y_j(d)\) is also calculated by the following equation similar
to Eq. (7).
\[ y_j(d) = \frac{Y_j(d)}{Y_{AveMax}} \] .......................... (10)

\[ Y_j(d): \text{value of evaluation item } Y \text{ of cage } d \text{ at the cage judgment node } j \]
\[ Y_{AveMax}: \text{maximum value of averaged evaluation item } Y \text{ over past 5 minutes among cages.} \]

As for \( \{CHC_{sd}, CCS_d, CSR_d\} \), the binary judgment (satisfy/not) is done. If Eq. (9) is satisfied and cage judge-
ment node \( j \) is connected to the node in the Hall Call Assignment Part, then the new hall call is assigned to the opti-
mal cage \( d \) in the Hall Call Assignment Part. Otherwise, i.e., the candidate cage \( d \) does not satisfy Eq. (9), which
means the condition of evaluation item \( Y \) is not satisfied, then, the node transition is resumed from the cage selec-
tion part in order to select another candidate cage again.

Finally, in the Hall Call Assignment Part, the new call is assigned to the candidate cage by cage assignment nodes.
Node transition returns to the system information judgment part after assignment, and the same procedures are
executed for the next call.

3.2.3. Node Functions

The node functions in each part are defined as follows.
<System Information Judgment Node (3 kinds): Judgment node>

\[ J^{VI}_{dt}: \text{Judge the variance of the elevator position (2 branches)} \]
\[ J^{EF}_{td}: \text{Judge the emerged floor and direction of the new hall call (5 branches)} \]
\[ J^{DF}_{td}: \text{Judge the destination floor of the new hall call (3 branches)} \]

<Cage Selection Node (12 kinds): Processing node>
In the processing nodes, there are totally 12 kinds of nodes.

\[ S(X): \text{select evaluation item } X \text{ from 12 items by the node transition in the Cage Selection Part and calculate Eq. (6)} \]

\[ X \in \{AT_{sd}, AET_{sd}, N_{Psd}, N_{Csd}, RR_{sd}, CHC_{sd}, AT_d, AET_d, DNP_d, DNC_d, CCS_d, CSR_d\} \]

<Cage Judgment Node (12 kinds): Judgment node>

\[ J^Y (d): \text{Judge whether } y_j(d) \leq r_j^Y \text{ is satisfied or not (2 branches)} \]
\[ Y \in \{AT_{sd}, AET_{sd}, N_{Psd}, N_{Csd}, RR_{sd}, AT_d, AET_d, DNP_d, DNC_d\} \]

\[ J^Z (d): \text{Judge whether } Z \text{ of cage } d \text{ is satisfied or not (2 branches)} \]
\[ Z \in \{CHC_{sd}, CCS_d, CSR_d\} \]

<Hall Assignment Node (1 kind): Processing node>

\[ A(d): \text{Assign the cage } d \text{ to the new hall call} \]

### Table 2. Specifications of Elevator Simulator.

<table>
<thead>
<tr>
<th>Items</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Floors</td>
<td>16</td>
</tr>
<tr>
<td>Number of Shafts(Cages)</td>
<td>6(12)</td>
</tr>
<tr>
<td>Floor Distance [m]</td>
<td>4.5</td>
</tr>
<tr>
<td>Max. Velocity [m/s]</td>
<td>2.5</td>
</tr>
<tr>
<td>Max. Acceleration [m/s²]</td>
<td>0.7</td>
</tr>
<tr>
<td>Jerk [m/s³]</td>
<td>0.7</td>
</tr>
<tr>
<td>Cage Capacity [person]</td>
<td>20</td>
</tr>
<tr>
<td>Time spent on</td>
<td></td>
</tr>
<tr>
<td>—Time for Opening Door [s]</td>
<td>2.0</td>
</tr>
<tr>
<td>—Time for Closing Door [s]</td>
<td>2.3</td>
</tr>
<tr>
<td>—Time for Riding [s/person]</td>
<td>1.0</td>
</tr>
<tr>
<td>Passenger Density [person/h]</td>
<td></td>
</tr>
<tr>
<td>—Regular Time</td>
<td>3000</td>
</tr>
<tr>
<td>—Up-peak</td>
<td>2700</td>
</tr>
<tr>
<td>—Down-peak</td>
<td>3300</td>
</tr>
</tbody>
</table>

3.2.4. Fitness Function

The fitness function of GNP individual is calculated by a weighed sum of waiting time, maximum waiting time,
one cage service and loops of GNP as follows.

\[ \text{Fitness} = \frac{1}{N} \sum_{n=1}^{N} t_n^2 + w_1 \cdot \left( t_{\text{max}} \right)^2 + w_c \cdot \left( N_c \right)^2 + w_l \cdot l^2 \] .......................... (11)

where

\[ N: \text{total number of passengers} \]
\[ t_n: \text{waiting time of } n\text{-th passenger} \]
\[ t_{\text{max}}: \text{maximum waiting time among } N \text{ passengers} \]
\[ N_c: \text{total number of passengers experiencing one cage service} \]
\[ l: \text{number of loops of GNP per an hour evaluation} \]
\[ w_1, w_c, w_l: \text{weighting coefficient.} \]

Number of loop of GNP transition is considered in the fitness because it deteriorates the performances of GNP.
All terms in this function are expected to minimize due to its definitions described above. Thus, an individual
with smaller fitness value means that it has a better structure and fitter parameters.

### 4. Simulations and Results

#### 4.1. Simulation Conditions

In this paper, we have studied the effectiveness of the proposed GNP with ACO in a typical office building, which
has 16 floors and 6 double-deck elevators running at the speed of 2.5 m/s. Table 2 shows the specifications of
the system simulator. Simulations are executed under 3 kinds of random sequences considering the probabilistic feature of DDES.

As shown in Table 3, simulations are implemented in
4.2. Results and Discussions

Simulations were implemented using different values of evaporation parameter ρ (ρ = 0, 0.05, 0.2 and 0.5). The fitness curves of the best GNP individual in each traffic are shown in Fig. 4 ((a) Regular Traffic, (b) Up-peak Traffic, (c) Down-peak Traffic). The fitness curve of the best individual is the average over 3 kinds of random sequences.

Compared to simple GNP, the proposed hybrid algorithm converges to a certain value at an early generation. And the different values of the evaporation parameter ρ makes the different convergence speeds which are shown in Fig. 4. In the up-peak traffic, the passenger’s flow is simpler than other traffics. Thus, there are not much differences between GNP with ACO and simple GNP in early generations of the up-peak traffic. The best fitness curve was obtained when ρ = 0.2. It converges around 60th generation. In other case, when ρ = 0.5, the result is a bit similar but not good as ρ = 0.2. The reason is that if the pheromone evaporates too fast, the offspring produced can only use the current pheromone information, not the past pheromone information. When ρ = 0.05, the fitness curve did not improve obviously than simple GNP. Its convergence is around 100th generation. Since the evaporation is slow, each branch would accumulate much pheromone, which means each branch has the identical probability to connect. So the convergence speed is slower when ρ is reduced. Average waiting time (AWT) is the average time until the service cage arrives at the floor after a passenger presses a hall call button. It is one of the important evaluation criteria in EGSCS. Table 5 shows the performance of AWT during the evolution. Also, the percentage of passengers waiting more than 60 s (LWP) and total number of passengers experiencing one cage service (NC) are shown in Tables 6 and 7. To sum up, it can be seen that the evaporation parameter is an important factor whose value influences the effectiveness of EGSCS, i.e., ρ = 0.20 is the best in EGSCS.

Table 3. Flow Ratios of Various Traffic Flow.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>1</td>
<td>19</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4. Evolutonal Conditions of GNP with ACO.

<table>
<thead>
<tr>
<th>Items</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generation</td>
<td>300</td>
</tr>
<tr>
<td>Population Size</td>
<td>200</td>
</tr>
<tr>
<td>— Crossover</td>
<td>80</td>
</tr>
<tr>
<td>— Mutation</td>
<td>122</td>
</tr>
<tr>
<td>— Elite</td>
<td>8</td>
</tr>
<tr>
<td>Node Size</td>
<td>91+Initial Boot Node</td>
</tr>
<tr>
<td>Crossover Rate $P_c$</td>
<td>0.1</td>
</tr>
<tr>
<td>Mutation Rate $P_m$</td>
<td>0.01</td>
</tr>
<tr>
<td>Evaluation Time $[h]$</td>
<td>2</td>
</tr>
<tr>
<td>$w_1, w_2, w_3$</td>
<td>0.007, 0.001, 0.6</td>
</tr>
</tbody>
</table>

Table 5. Performances of AWT during the evolution (unit: sec).

(a) Regular Traffic

<table>
<thead>
<tr>
<th>Number of generations</th>
<th>GNP with ACO</th>
<th>GNP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>36.78</td>
<td>36.68</td>
</tr>
<tr>
<td>30</td>
<td>32.48</td>
<td>32.47</td>
</tr>
<tr>
<td>100</td>
<td>32.41</td>
<td>32.30</td>
</tr>
<tr>
<td>300</td>
<td>32.45</td>
<td>32.23</td>
</tr>
</tbody>
</table>

(b) Up-peak Traffic

<table>
<thead>
<tr>
<th>Number of generations</th>
<th>GNP with ACO</th>
<th>GNP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>32.15</td>
<td>32.73</td>
</tr>
<tr>
<td>30</td>
<td>27.92</td>
<td>27.77</td>
</tr>
<tr>
<td>100</td>
<td>27.84</td>
<td>27.78</td>
</tr>
<tr>
<td>300</td>
<td>27.83</td>
<td>27.75</td>
</tr>
</tbody>
</table>

(c) Down-peak Traffic

<table>
<thead>
<tr>
<th>Number of generations</th>
<th>GNP with ACO</th>
<th>GNP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>32.45</td>
<td>32.62</td>
</tr>
<tr>
<td>30</td>
<td>27.29</td>
<td>27.21</td>
</tr>
<tr>
<td>100</td>
<td>27.21</td>
<td>27.16</td>
</tr>
<tr>
<td>300</td>
<td>27.20</td>
<td>27.17</td>
</tr>
</tbody>
</table>

Table 6. Performances of LWP during the evolution (unit: %).

(a) Regular Traffic

<table>
<thead>
<tr>
<th>Number of generations</th>
<th>GNP with ACO</th>
<th>GNP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>23.48</td>
<td>23.54</td>
</tr>
<tr>
<td>30</td>
<td>12.79</td>
<td>12.70</td>
</tr>
<tr>
<td>100</td>
<td>12.74</td>
<td>12.69</td>
</tr>
<tr>
<td>300</td>
<td>12.76</td>
<td>12.72</td>
</tr>
</tbody>
</table>

(b) Up-peak Traffic

<table>
<thead>
<tr>
<th>Number of generations</th>
<th>GNP with ACO</th>
<th>GNP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>8.86</td>
<td>8.65</td>
</tr>
<tr>
<td>30</td>
<td>8.43</td>
<td>8.51</td>
</tr>
<tr>
<td>100</td>
<td>8.56</td>
<td>8.38</td>
</tr>
</tbody>
</table>

(c) Down-peak Traffic

<table>
<thead>
<tr>
<th>Number of generations</th>
<th>GNP with ACO</th>
<th>GNP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>14.20</td>
<td>14.17</td>
</tr>
<tr>
<td>30</td>
<td>9.22</td>
<td>9.36</td>
</tr>
<tr>
<td>100</td>
<td>9.17</td>
<td>9.16</td>
</tr>
</tbody>
</table>
Fig. 4. Fitness Curves of GNP with ACO and Simple GNP.
5. Conclusions

In this paper, we proposed a hybrid algorithm named GNP with ACO. The essential point of the proposed algorithm is that in the genetic operation phase of the proposed algorithm, the mutation operation carries out a random change to the connections of GNP in common generations, while in special generations the connections of GNP are changed by the pheromone information, which is a kind of positive feedback. The advantages of using the proposed algorithm have been demonstrated through the simulations of the elevator group supervisory control system. GNP with ACO increases the convergence speed and fitness performance by making the “exploitation-exploration” tradeoff.

References:


<table>
<thead>
<tr>
<th>Name</th>
<th>Affiliation</th>
<th>Address</th>
<th>Brief Biographical History</th>
<th>Main Works</th>
<th>Membership in Academic Societies</th>
</tr>
</thead>
</table>
| Shingo Mabu          | Dr., Assistant Professor, Graduate School of Information, Production and Systems, Waseda University | 2-7 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka 808-0135, Japan | 2003 Received M.E. degree from Kyushu University  
2006 Received Ph.D. degree from Waseda University  
2006- Visiting Lecturer at Waseda University  
• The Society of Instrument and Control Engineers (SICE)  
• The Institute of Electrical Engineers of Japan (IEEJ) |
| Jinglu Hu            | Dr., Associate Professor, Graduate School of Information, Production and Systems, Waseda University | 2-7 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka 808-0135, Japan | 1997 Received Ph.D. degree from Kyushu Institute of Technology  
1997- Research Associate at Kyushu University  
• The Institute of Electrical Engineers of Japan (IEEJ) |
| Kotaro Hirasawa      | Dr., Professor, Graduate School of Information, Production and Systems, Waseda University | 2-7 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka 808-0135, Japan | 1966 Joined Hitachi Corp.  
1992- Professor at Kyushu University  
• Association for Computing Machinery (ACM)  
• The Society of Instrument and Control Engineers (SICE)  
• The Institute of Electrical Engineers of Japan (IEEJ) |
| Sandor Markon        | Senior Researcher, the Product Development HQ of Fujitec                     | Big Wing, Hikone, Shiga 522-8588, Japan       | 1973 Received Dipl. Ing. from Budapest Technical University  
1975-1977 Research student at Kyoto University  
1979-1996 Fujitec Co., Ltd.  
1996-1996 Received Ph.D. from Kyoto University | • system analysis, simulation, neural networks, evolutionary computation, software development. | • Institute of Electrical and Electronics Engineers, Inc. (IEEE)  
• International Neural Network Society (INNS)  
• Japanese Neural Network Society (JNNS)  
• Scheduling Society of Japan (SSJ) |