ARTICLE IN PRESS

Available online at www.sciencedirect.com

ScienceDirect

Applied Mathematics and Computation xxx (2007) xxx–xxx

APPLIED MATHEMATICS AND COMPUTATION

www.elsevier.com/locate/amc

Heuristic approaches for solving transit vehicle scheduling problem with route and fueling time constraints

Haixing Wang *, Jinsheng Shen

Beijing Jiaotong University, Beijing 100044, China

Abstract

Electric bus scheduling problem can be defined as vehicle scheduling problem with route and fueling time constraints (VSPRFTC). Every vehicle's travel miles (route time) after charging is limited, thus the vehicle must be recharged after taking several trips and the minimal charging time (fueling time) must be satisfied. A multiple ant colony algorithm (ACA) was presented to solve VSPRFTC based on ACA used to solve traveling salesman problem (TSP), a new metaheuristic approach inspired by the foraging behavior of real colonies of ants. The VSPRFTC considered in this paper minimizes a multiple, hierarchical objective function: the first objective is to minimize the number of tours (or vehicles) and the second is to minimize the total deadhead time. New improvement of ACA as well as detailed operating steps was provided on the basis of former algorithm. Then in order to settle contradiction between accelerating convergence and avoiding prematurity or stagnation, improvement on route construction rule and Pheromone updating rule was adopted. A group feasible trip sets (blocks) had been produced after the process of applying ACA. In dealing with the fueling time constraint a bipartite graphic model and its optimization algorithm are developed for trip set connecting in a hub and spoke network system to minimize the number of vehicle required. The maximum matching of the bipartite graph is obtained by calculating the maximum inflow with the Ford–Fulkerson algorithm. At last, an example was analyzed to demonstrate the correctness of the application of this algorithm. It proved to be more efficient and robust in solving this problem.

Keywords: Vehicle scheduling problem with route and fueling time constraints (VSPRFTC); Ant colony algorithm (ACA); Vehicle routing problem (VRP); Bipartite graphic model; Ford–Fulkerson algorithm

1. Introduction

As a splendid and excellent demonstration of "Green Olympics, Scientific Olympics", electric bus would be put into use in scope of regional in some area of Beijing, China. Because of the big difference of dynamical and drive characteristics between electric bus and traditional bus, the charging time (fueling time) of electric bus requires four to five hours and the travel miles (route time) after each charging is limited, it would affect the normal operation of public traffic if characteristic of electric bus was not considered. Studies of scheduling problem of electric bus become more and more important.

* Corresponding author.
E-mail address: haixing21@163.com (H. Wang).

0096-3003/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2007.02.141

Real world transit scheduling is very complicated. The large numbers of trips, links, and paths to be considered rapidly increase the number of variables and constraints in any model developed. The transit scheduling problem belongs to the general class of vehicle scheduling problems (VSP). The VSP is a classical optimization problem which is faced in the operational planning of public transportation systems. It consists in assigning a set of scheduled trips to a set of available vehicles starting from one or more depot have to collectively visit a number of demand points and then return to the depot from which they start, in such a way that each trip is associated to one vehicle and a cost function is minimized [1]. These problems in general are very hard to solve and belong to a class of problems referred to as NP-hard. In general, these problems can be solved using heuristics. Electric bus scheduling problem can be defined as vehicle scheduling problem with route and fueling time constraints (VSPRFTC). Every vehicle's travel miles (route time) after charging is limited, thus the vehicle must be recharged after taking several trips and the minimal charging time (fueling time) must be satisfied. However, in VSPRFTC, the following conditions should be satisfied [2]:

- An objective function given in advance is optimized.
- Each trip is run by exactly one vehicle.
- Every vehicle can only take on one trip at the same time.
- Satisfy the timing between trips every vehicle taking, and the start time of the next trip must posterior to the end time of the previous one.
- Each block of trips starts and ends at the same depot.
- Every vehicle must be refueled when it can not take any trips.
- Satisfy the fueling time of every vehicle, and the vehicle should put into use again when the fueling process is finished.
- Each depot has a given maximum number of vehicles (capacity).
- All operational constraints, including any restriction on the total time a vehicle spends away from the depot, are satisfied.

Due to fuel restrictions, route time constraints had been considered in the formulation for solving real world problems. Ref. [3] referred to a vehicle scheduling problem that considers route time constraints as VSPLPR with the following description: "In the vehicle scheduling problem with length of path restrictions (VSPLPR), constraints are placed on the length of time a vehicle may spend away from the depot or the mileage a vehicle may cover without returning to the depot for service". There is also some recognition of these constraints in the literature in the SDVS context. These models only consider the time difference between one pull-out and the corresponding pull-in. This makes the models unsuitable for considering fuel consumption concerns. Ref. [4] presented a formulation and computational procedure for solving the multiple depot vehicle scheduling with route time constraints.

There is no convenient solution to VSPRFTC of electric bus, as far as we know. The paper try to provide an approach to formulating and solving the VSPRFTC of electric bus in the case of multi-lines through applied ant colony algorithm.

2. Mathematical formulation of VSPRFTC

Ref. [5] the fleet of a public transportation company is subdivided into depots. The set of depots is denoted by D. With each depot $d \in D$ we associate a start point $d+$ and an end point $d-$ where its vehicles start and terminate their daily duty. The number of available vehicles, the depots' capacity is denoted by v. Every vehicle's driving range is denoted by r_k. A given timetable defines L timetabled trips, denoted by I, that are used to carry passengers. We associate with each $i \in I$ a first stop ds_i, a last stop de_i, a timetable departure time s_i' which could shift between $[-se, sl]$, a actual departure time $s_i \in [s_i' - se, s_i' + sl]$, an travel time $e_i = s_i + t_i$, an arrival time $e_i = s_i + t_i$ and a travel range is m_i.

There are further types of trips, all running without passengers: A pull-out trip connecting some start point $d+$ with some first stop ds_i, a pull-in trip connecting some last stop de_i with some end point $d-$, and a deadhead trip connecting some last stop de_i with some succeeding first stop ds_{j}. For notational simplicity, we call all these trips unloaded trips. Let t_{ij} denote the duration of the deadhead trip from i to j including some layover time. The travel range of the deadhead trip from i to j is denoted by m_{ij}. The idle time of the deadhead trip from i to j is denoted by $v_{ij} = s_j - e_i$, and $v_{ij} \geqslant t_{ij}$ ensure the timing between trips. Whenever, we call the corresponding deadhead trip compatible A vehicle schedule or duty is a chain of trips such that the first trip is a pull-out trip, and the last trip is a pull-in trip and the trips and unloaded trips occur alternately [6].

$$x_{ijk} = \begin{cases} 1 & \text{vehicle } k \text{ taking deadhead trip from } i \text{ to } j, \\ 0 & \text{else,} \end{cases} \tag{1}$$

$$y_{ki} = \begin{cases} 1 & \text{vehicle } k \text{ taking trip } i, \\ 0 & \text{else,} \end{cases} \tag{2}$$

$$\min Z = \sum_{i=0}^{L} \sum_{j=0}^{L} \sum_{k=1}^{m} v_{ij} x_{ijk}, \tag{3}$$

$$\text{s.t.} \quad \sum_{k=1}^{m} \sum_{j=1}^{L} x_{0jk} \leqslant v, \tag{4}$$

$$\sum_{k=1}^{m} \sum_{i=0}^{L} x_{ijk} = 1, \quad j = 1, \ldots, L, \tag{5}$$

$$\sum_{k=1}^{m} \sum_{j=0}^{L} x_{ijk} = 1, \quad i = 1, \ldots, L, \tag{6}$$

$$\sum_{k=1}^{m} \sum_{i=1}^{L} x_{0ik} - \sum_{k=1}^{m} \sum_{j=1}^{L} x_{j0k} = 0, \tag{7}$$

$$\sum_{i=0}^{L} \sum_{j=0}^{L} x_{ijk}(m_i + m_{ij}) \leqslant r_k, \quad k = 1, \ldots, m. \tag{8}$$

The objective (3) aims at minimizing total idle time; Constraints (4) ensure that the number of vehicle should not exceed the max number of vehicle in depots; Constraints (5) and (6) ensure that every trip should only be take by one vehicle; Constraints (7) ensure that a vehicle is starting from and returning to the same depot; Constraints (8) limit the travel time of every vehicle after being charged.

3. Application of ant colony algorithm on VSPRFTC

This paper presents a multiple ant colony system for VSPRFTC. Algorithm is based on ant colony optimization (ACO), a new metaheuristic approach inspired by the foraging behavior of real colonies of ants.The basic ACO idea is that a large number of simple artificial agents are able to build good solutions to hard combinatorial optimization problems via low-level based communications. Real ants cooperate in their search for food by depositing chemical traces (pheromones) on the floor. Artificial ants cooperate by using a common memory that corresponds to the pheromone deposited by real ants. This artificial pheromone is one of the most important components of ant colony optimization and is used for constructing new solutions. In the ACO metaheuristic, artificial pheromone is accumulated at run-time during the computation. Artificial ants are implemented as parallel processes whose role is to build problem solutions using a constructive procedure driven by a combination of artificial pheromone, problem data and a heuristic function used to evaluate successive constructive steps.

Successful ant colony algorithms have been developed for several combinatorial optimization problems. Such as TSP, VRP (vehicle routing problem), and so on.

Based on the successful of the application of ant colony algorithm to TSP, ant colony algorithm was adopted in solving VSPRFTC. According to the characteristic of this problem, some changes was induced as follows [7].

Please cite this article in press as: H. Wang, J. Shen, Heuristic approaches for solving transit vehicle ..., Appl. Math. Comput. (2007), doi:10.1016/j.amc.2007.02.141

- The difference between the end time of trip i and the start time of trip j is seen as distance.
- A Tabu_k list of forbidden moves also includes the nodes which can not satisfy the timing of trips.
- The limit route time of electric vehicle is considered like the limit supply of vehicle in VRP, and the allowed nodes to which ant could move must satisfy the limit.
- solution sets is divided into several subsets. The subsets contains a set of trips which a vehicle can take on after being charged.
- Arrange the feasible blocks (trip sets) to vehicles according the fueling (charging) time.

3.1. Basic Principles of ant colony algorithm

This section introduces and presents the original ant colony algorithm (ACA) applied to the traveling salesman problem (TSP) Indeed, algorithm has been proposed to solve a VSPRFTC where both the number of vehicles and the deadhead time have to be minimized. This multiple objective minimization is achieved by using 3 artificial ant colonies based on ACA. The TSP is the problem of finding a shortest closed tour which visits all the cities in a given set. ACA is applied to the TSP by associating two measures to each arc of the TSP graph: the closeness η_{ij}, and the pheromone trail τ_{ij}. Closeness, defined as the inverse of the arc length, is a static heuristic value, that never changes for a given problem instance, while the pheromone trail is dynamically changed by ants at run-time. Therefore, the most important component of ACA is the management of pheromone trails which are used, in conjunction with the objective function, for constructing new solutions. Informally, pheromone levels give a measure of how desirable it is to insert a given arc in a solution. Pheromone trails are used for exploration and exploitation. Exploration concerns the probabilistic choice of the components used to construct a solution: a higher probability is given to elements with a strong pheromone trail. Exploitation chooses the component that maximizes a blend of pheromone trail values and heuristic evaluations.

m, the number of ants; τ_{ij}, the intensity in arc (i,j); η_{ij}, the visibility in arc (i,j); $\Delta\tau_{ij}^k$, the quantity of pheromone levels in arc (i,j) of ant k; $P_{ij,}^k$ the probability of moving from present node i to another node j; $\alpha(\alpha \geqslant 0)$, the relative influence of trail; $\beta(\beta \geqslant 0)$, the relative influence of visibility; $\rho(0 \leqslant \rho \leqslant 1)$, the permanence of trail, $1-\rho$ is known as evaporation; N_i^k, the feasible nodes of ant k in node i; Q, the quantity of pheromone the ant leaves [8].

3.1.1. Route construction

ACA goal is to find a shortest tour. In ACA m ants build tours in parallel, where m is a parameter. Each ant is randomly assigned to a starting node and has to build a solution, that is, a complete tour. A tour is built node by node: each ant iteratively adds new nodes until all nodes have been visited. When ant k is located in node i, it chooses the next node j probabilistically in the set of feasible nodesi according to $p_{ij}^k(t)$.

In the ant colony algorithm original version formula for $p_{ij}^k(t)$ is

$$p_{ij}^k(t) = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha[\eta_{ij}]^\beta}{\sum_{l\in N_i^k}[\tau_{il}(t)]^\alpha[\eta_{il}]^\beta} & j \in N_i^k, \\ 0 & \text{else.} \end{cases} \tag{9}$$

3.1.2. Trail updating

The rules of trail updating can be expressed as three mode, they are ant-density, ant-quantity, ant-cycle. $\Delta\tau_{ij}^k$ of ant-cycle:

$$\Delta\tau_{ij}^k(t, t+n) = \begin{cases} \frac{Q_3}{L^k} & \text{Arc}(i,j) \text{ in the trail of ant } k, \\ 0 & \text{else.} \end{cases} \tag{10}$$

After iteration is complete, that is when all the ants have completed their solutions, the pheromone levels are updated to:

$$\tau_{ij}(t+n) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij}(t, t+n), \tag{11}$$

$$\Delta\tau_{ij}(t, t+n) = \sum_{k=1}^{m} \Delta\tau_{ij}^k(t, t+n). \tag{12}$$

3.2. Improvement on basic principles of ant colony algorithm

The basic principles of ant colony algorithm updating pheromone and confirming the probability of trail select with fixed mode, ignoring the practical state of algorithm searching, and the contradiction between accelerating constringency and avoiding prematurity and stagnancy often comes into being. So Improving on the basic principles of ant colony algorithm is significant.

3.2.1. Route construction

The number of next nodes which ant k in node i could select is represented by $\mathrm{win}^k(i)$. In basic ant colony algorithm, $\mathrm{win}^k(i) = \mathrm{cnt}(\mathrm{allowed}_k)$, $\mathrm{cnt}(\mathrm{allowed}_k)$ denotes the number of nodes in set $\mathrm{allowed}_k$. The value of $\mathrm{win}^k(i)$ should be adjusted dynamically [9].

$$\mathrm{win}^k(i) = \begin{cases} \mathrm{cnt}(\mathrm{allowed}_k) & \mathrm{if}(1 - V(i))^* \mathrm{cnt}(\mathrm{allowed}_k) = \mathrm{cnt}(\mathrm{allowed}_k) \\ \lfloor (1 - V(i))^* \mathrm{cnt}(\mathrm{allowed}_k) \rfloor + 1 & \mathrm{else}. \end{cases} \tag{13}$$

The sum of ants is represented by M, there are r trails from node i, the number of ants pass node i is Y_i, and Y_i is distributing in r trails with number $a_1, a_2, \ldots, a_r$

$$V(i) = \begin{cases} \frac{Y_i}{M}\left(1 - \frac{1}{Y_i}\sqrt{\frac{r\sum_{i=1}^{r}(Y_i/r - a_i)^2}{r-1}}\right) & \text{if } Y_i \neq 0, \\ 0 & \text{else}. \end{cases} \tag{14}$$

Use for reference the idea of Ant-Qalgorithm of Dorigo, the meliorative route construction rule can be expressed as follows:

$$j = \begin{cases} \text{According } P_{ij} \text{ Select } j, \\ P_{ij} = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta [\mu_{ij}]^\gamma [\kappa_{ij}]^\lambda}{\sum_{h \in \mathrm{allowed}_k} [\tau_{ij}]^\alpha [\eta_{ij}]^\beta [\mu_{ij}]^\gamma [\kappa_{ij}]^\lambda} & j \in \mathrm{allowed}_k, \ P \leqslant r_0, \\ \text{According } \max\limits_{j \in \mathrm{allowed}_k} \{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta [\mu_{ij}]^\gamma [\kappa_{ij}]^\lambda\} \text{ Select } j, \ P \geqslant r_1, \\ \text{According } P'_{ij} \text{ Select} j, \\ P'_{ij} = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta [\mu_{ij}]^\gamma [\kappa_{ij}]^\lambda}{\sum_{h \in \mathrm{allowed}'_k} [\tau_{ij}]^\alpha [\eta_{ij}]^\beta [\mu_{ij}]^\gamma [\kappa_{ij}]^\lambda} & j \in allowed'_k, \ r_0 \leqslant P \leqslant r_1. \end{cases} \tag{15}$$

$\mu_{ij} = d_{i0} + d_{0j} - d_{ij}$, called saving.

$\kappa_{ij} = (Q_i + q_i)/Q$, variable induced considering the constraint of the range of electric vehicle.

P is a random number, accords with uniform distribution on $(0,1)$, r_1, r_0 changed dynamic according with the process of the algorithm.

3.2.2. Pheromone updating

Pheromone local updating tactic means ant's one pace moving, during the course of constructing the solution, leads to pheromone's updating at the corresponding arc.

On the one hand, local updating may enforce all of the ant's cooperation in an iterative cycle, on the other hand, it also can enforce to search the untouched arc, to prevent algorithm maturating early at a certain extent

Attraction: Set the number of ants crossing node i equal to $Y(i)$, the number of crossing directed arc(i,j) equal to a then $F_{ij} = a_j/Y(i)$ means ant's attraction at directed arc(i,j)

While updating the local pheromone, set the quantity of once released pheromone equal to Q, if F_{ij} and Q are too large they will limit the algorithm's global behavior. If they are too small, they will effect the algorithm's convergence speed. So applying the next tactic: variable $Q(t)$ stands for pheromone, when F_{ij} is larger the $Q(t)$ is smaller and

$$Q(t) = Q^*(1 - F_{ij}). \tag{16}$$

Please cite this article in press as: H. Wang, J. Shen, Heuristic approaches for solving transit vehicle ..., Appl. Math. Comput. (2007), doi:10.1016/j.amc.2007.02.141

Algorithm's local updating rules:

$$\tau_{ij}^{\text{new}} = \rho \cdot \tau_{ij}^{\text{old}} + \Delta\tau_{ij} \quad \forall\ i, j, i \neq j, \tag{17}$$

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}, \tag{18}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} Q_1(1 - F_{ij}) & \text{if Ant move from } i \text{ to } j, \\ 0 & \text{else.} \end{cases} \tag{19}$$

After finishing the local solution's optimization, updating every arc's pheromone track globally, applied the following new rules:

$$\tau_{ij}^{\text{new}} = \rho \cdot \tau_{ij}^{\text{old}} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + \sigma \cdot \Delta\tau_{ij}^{*}, \tag{20}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} 1/L^{k} & \text{route of ant } k \text{ including arc}(i,j), \\ 0 & \text{else.} \end{cases} \tag{21}$$

$$\Delta\tau_{ij}^{*} = \begin{cases} 1/L^{*} & \text{if } (i,j) \text{ belongs to the latest optimal route,} \\ 0 & \text{else.} \end{cases} \tag{22}$$

τ_{ij}^{old} and τ_{ij}^{new} stands for the former and updating pheromone thickness at arc(i,j); L^{k} presents for the length of route constructed by the k ant; L^{*} stands for latest optimal route length; σ stands for the quantity of the elitist ant. When updating the tracks, we should consider not only the current iterative's effect on the quantity of track, but also its effect on the latest global optimal solution, this ensures the pheromone thickness included by the optimal solution's arc can be enforced, then this solution can be selected by larger probability in the following iterative, this protect the optimal solution, lest it degenerates.

3.2.3. Multiple ant colonies

A solution model in which each ant builds a single tour (Fig. 1) was presented. The solution is represented as follows: First, the depot with all its connections to/from the trips is duplicated a number of times equal to the number of available vehicles. Distances between copies of the depot are set to zero. This approach makes VSPRFTC closer to the traditional traveling salesman problem [10].

This paper considers a more elaborated VSPRFTC with two objective functions: (i) the minimization of the number of tours (or vehicles) and (ii) the minimization of the total deadhead time, where number of tours minimization takes precedence over deadhead time minimization.

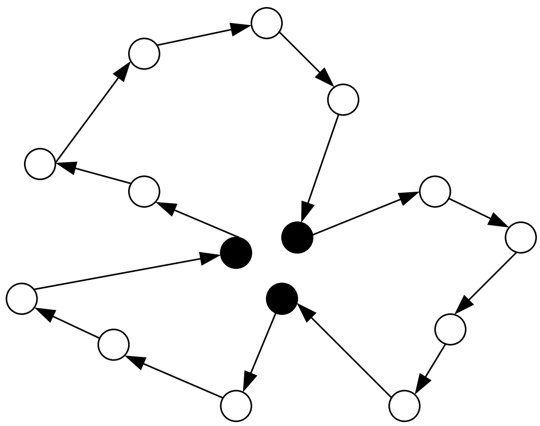

Fig. 1. Feasible solutions for a VSPRFTC (duplicated depots are black points while trips are white points).

ψ^{gb}, feasible solution ;
J_ψ, objective function of ψ ;
Count_Bus (ψ) Count the number of vehicles needed applying ψ ;
Visited_Node (ψ)Count the number of nodes included in ψ ;
1. ascertain the lower value of $V_{\min}$
$V_{Lower} \leftarrow D_m(S) = \max_{t \in [T_1, T_2]} g(t, S)$;
2.get the initialized feasible solution and ascertain the lower value of $V_{\min}$
Load ACA-INI ;
$\psi^{gb} \leftarrow$ applying ACA-INI and get the initialized feasible solution ψ^{ACA_INI} ;
$V_{Higher} \leftarrow$ # Count_Bus (ψ^{ACA_INI}) ;

3. Load ACA-BUS (*V*) to ascertain $V_{\min}$
$V \leftarrow \left\lceil \frac{V_{Higher} + V_{Lower}}{2} \right\rceil$;
Add *V* source nodes as vehicles to the network ;
$\psi^{ACA-BUS} \leftarrow \varnothing$;
Load ACA-BUS (*V*) ;
For Each Ant k
New_Active_Ant (k)
$\forall\, Node\ j \notin \psi^k : IN_j \leftarrow IN_j + 1$;
If $\exists k$:# Visited_Node (ψ^k) > # Visited_Node ($\psi^{ACA-BUS}$) , then
$\psi^{ACA-BUS} \leftarrow \psi^k$;
$\forall\, j : IN_j \leftarrow 0$;
End If
End For
If $\psi^{ACA-BUS}$ is feasible solution, then
$V_{higher} \leftarrow V$;
$\tau_{ij} = (1-\rho)\cdot\tau_{ij} + \rho / J_{\psi^k} \quad \forall (i,j) \in \psi^k$;
If $V_{higher} = V_{lower} + 1$,
Stop ;
$V_{\min} \leftarrow V_{higher}$;
$\psi^{gb} \leftarrow \psi^{ACA-BUS}$;
Else
Update *V* and continue ;
End If
Else
$V_{lower} \leftarrow V$;
Update *V* and continue ;
End If
4. Load ACA-TIME ($V_{\min}$) to get the optimal solution
Add $V_{\min}$ source nodes as vehicles to the network ;
Feasible solution $\psi^{ACA-TIME} \leftarrow \psi^{gb}$;
Load ACA-TIME ($V_{\min}$) ;
For Each Ant k
New_Active_Ant (k)
If $\exists k : \psi^k$ is feasible solution and $J_{\psi^k} < J_{\psi^{gb}}$,
$\psi^{ACA\_TIME} \leftarrow \psi^k$;
$\tau_{ij} = (1-\rho)\cdot\tau_{ij} + \rho / J_{\psi^k} \quad \forall (i,j) \in \psi^k$;
End If
End For
$\psi^{gb} \leftarrow$ applying ACA- TIME ($V_{\min}$) to get ψ^{ACA_TIME} ;

Fig. 2. Optimizing pocess.

New_Active_Ant (k)
$MaxRunningTime$, the running time constraint ;
$RunningTime_k$, current running time of ant k ;
$TripTime_j$, duration of trip j ;
End_i, arrival time of trip i ;
$Begin_j$, departure time of trip j ;
$DeadHeadTime_{ij}$, deadhead time between trip i and j ;
1.initialization
Put k to node i random ;
$\psi^k \leftarrow \{i\}$;
$RunningTime_k \leftarrow 0$;
2. construct the route of ant k
Loop
N_i^k, the feasible node (j) set of ant k in node i, N_i^k can be ascertained considering (3) items :
(1) $j \notin \psi^k$;
(2) $Begin_j - End_i \geq DeadHeadTime_{ij}$;
(3) $MaxRunningTime - RunningTime_k \geq TripTime_j$;
If $N_i^k = \{\}$, then
Select source node j which ant k has not visited ;
$TripTime_j \leftarrow 0$;
Else
$\forall j \in N_i^k$ ascertain j of ant k according P_{ij}^k ;
End If
$\psi^k \leftarrow \psi^k + \{j\}$;
$RunningTime_k \leftarrow RunningTime_k + TripTime_j$;
$\tau_{ij} = (1-\rho)\cdot\tau_{ij} + \rho\tau_0$;
i ← j ;
Until all of the nodes have been visited

Fig. 3. Process of ant cology algorithm.

In the algorithm (Figs. 2 and 3) both objectives are optimized simultaneously by coordinating the activities of 3 ACA based colonies. The goal of the first colony, ACA-INI, is to try to get the initialized feasible solution and ascertain the lower value of vehicles needed; The goal of the second colony, ACA-BUS, is to try to diminish the number of vehicles used, while the third colony, ACA-TIME, optimizes the feasible solutions found by ACA-BUS. Both colonies use independent pheromone trails but collaborate by sharing the variable ψ^{gb}. Initially, ψ^{gb} is a feasible solution found by ACA-INI. Then, ψ^{gb} is improved by the two colonies. When ACA-BUS is activated, it tries to find a feasible solution with one vehicle less than the number of vehicles used in ψ^{gb}. The goal of ACA-TIME is to optimize the total deadhead time of solutions that use as many vehicles as vehicles used in ψ^{gb}. ψ^{gb} is updated each time one of the colonies computes an improved feasible solution. In case the improved solution contains less vehicles than the vehicles used in ψ^{gb}, algorithm kills ACA-TIME and ACA-BUS. Then, the process is iterated and two new colonies are activated, working with the new, reduced number of vehicles.

4. bipartite graphic model for fueling time constraint

Feasible trip sets (blocks) can be presented after the process of applying ant colony algorithm. And the fueling time constraint (trip set connecting problem) should also be considered. A bipartite graphic model and its optimization algorithm are developed for trip set connecting in a hub and spoke network system to minimize the number of vehicle required. First, the trip set connecting problem is converted into the trip set pairing connecting problem, and a bipartite graphic model describing the trip set pairing connecting problem is built.

Thus, an optimal trip set connecting problem is transformed to the maximum matching problem of the bipartite graphic model. Then, an assistant graph with single source and sink is created based on the bipartite graphic model. The maximum matching of the bipartite graph is obtained by calculating the maximum inflow with the Ford–Fulkerson algorithm, and a trip set connecting schedule with minimum vehicle number is accordingly produced. The process can be described as Fig. 4.

$\overline{P}_{\text{free}}$, represents the free vehicles.
Order(i), serial number of every free vehicle, vehicle will be arranged early or late according it.
m_{free}, the number of free vehicles.
$\overline{P}_{\text{free}}$, free vehicle set.
$\overline{F}(i)$, set represents the trip sets taken by vehicle i.

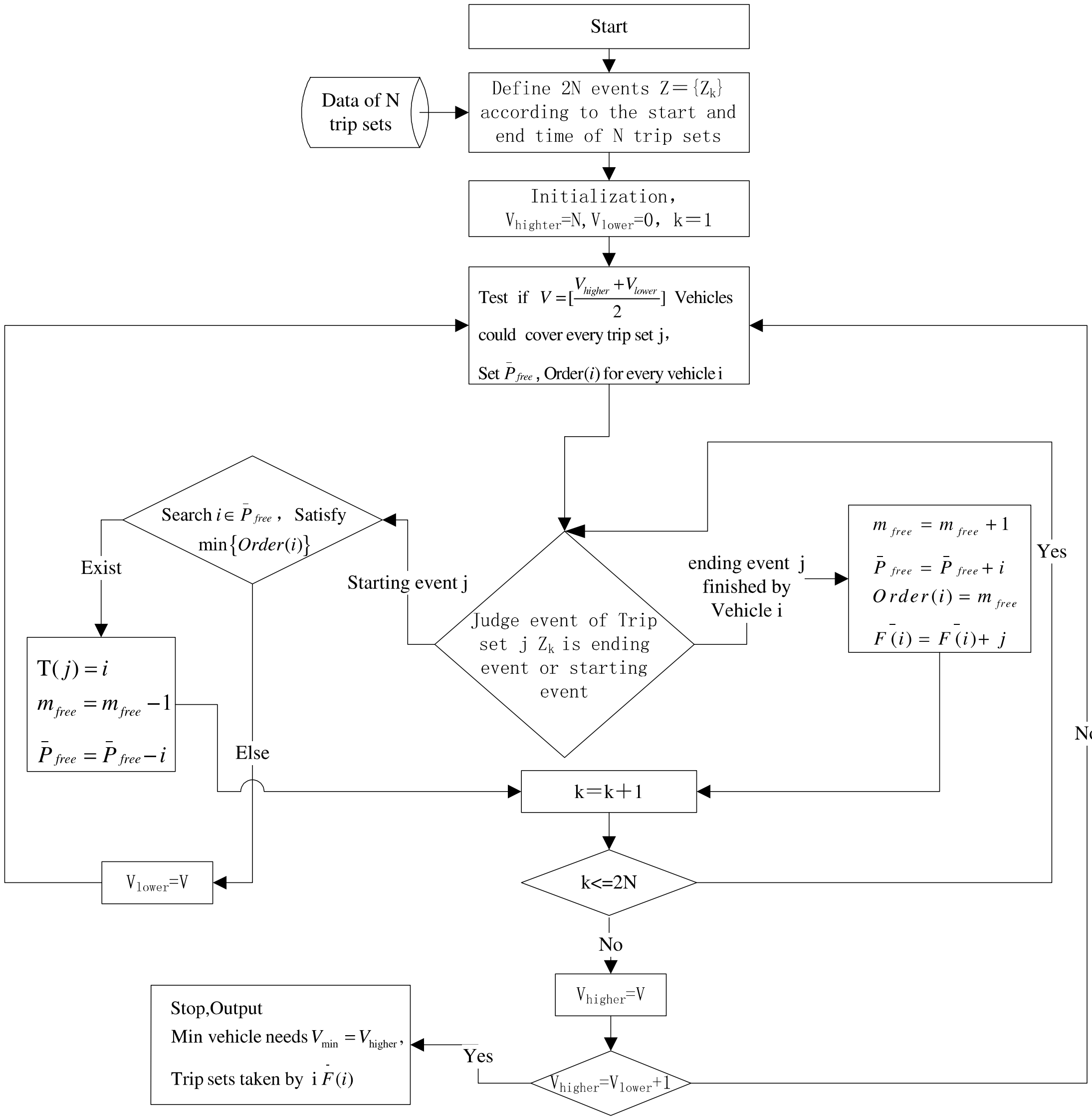

Fig. 4. Process of agorithm.

5. Computational example

The correlative parameter of vehicle is: route time constraint = 420 min, fueling time constraint = 180 min; the applying control parameter is: Ant number $M = 60, nc_{\max} = 40, \tau_{ij} = 10$, The Max pheromones = 100, the Min pheromones = 2, $\alpha = 1, \beta = 1, \gamma = 0.5, \lambda = 0.5, \rho_1 = 0.85, \rho_2 = 0.95, Q_1 = 10, Q_2 = 50, Q_3 = 100, r_0 = r_1 = 1$. Consider the 3 deports 3 routes 261 trips VSPRFTC showing in the Tables 1–5.

Vehicles needed of every deport changing with time can be seen in Fig. 5.Using the algorithms presented, 6 deadhead trips (Table 6) was inserted to the current timetable and three vehicles was saved, 37 feasible trip sets were acquired. Considering that the vehicle can be reused after being fueled, we can see that it is enough for 34 Electric Vehicles (Table 7) to carry out these trips by applying bipartite graphic algorithm. Obviously, this pro-

Table 1
Routes information

Route	Deport	Route time (min)
1	A	48
2	B	48
3	C	59

Table 2
Deadhead time information

Departure deport	Arrival deport	Deadhead time (min)
A	C	20
C	A	20
B	C	15
C	B	15
A	B	25
B	A	25

Table 3
Timetable of route 1

Trip	Departure time	Trip	Departure time	Trip	Departure time	Trip	Departure time	Trip	Departure time
1	5:10	23	7:24	45	9:18	67	14:52	89	17:21
2	5:18	24	7:28	46	9:30	68	15:02	90	17:27
3	5:26	25	7:32	47	9:44	69	15:10	91	17:33
4	5:34	26	7:36	48	9:58	70	15:18	92	17:39
5	5:42	27	7:40	49	10:14	71	15:26	93	17:45
6	5:50	28	7:44	50	10:30	72	15:33	94	17:51
7	5:57	29	7:48	51	10:46	73	15:40	95	17:59
8	6:04	30	7:52	52	11:02	74	15:47	96	18:07
9	6:10	31	7:56	53	11:18	75	15:54	97	18:15
10	6:16	32	8:00	54	11:34	76	16:01	98	18:23
11	6:22	33	8:04	55	11:50	77	16:08	99	18:31
12	6:28	34	8:08	56	12:06	78	16:15	100	18:40
13	6:34	35	8:12	57	12:22	79	16:21	101	18:50
14	6:40	36	8:16	58	12:38	80	16:27	102	19:00
15	6:46	37	8:20	59	12:55	81	16:33	103	19:10
16	6:52	38	8:24	60	13:10	82	16:39	104	19:25
17	6:58	39	8:28	61	13:25	83	16:45	105	19:40
18	7:04	40	8:34	62	13:40	84	16:51	106	19:55
19	7:08	41	8:40	63	13:55	85	16:57	107	20:10
20	7:12	42	8:48	64	14:10	86	17:03	108	20:25
21	7:16	43	8:56	65	14:24	87	17:09	109	20:40
22	7:20	44	9:06	66	14:38	88	17:15	110	20:55

Table 4
Timetable of route 2

Trip	Departure time	Trip	Departure time	Trip	Departure time	Trip	Departure time	Trip	Departure time
111	5:30	133	8:00	155	12:00	177	16:27	199	18:05
112	5:44	134	8:06	156	12:16	178	16:34	200	18:09
113	5:52	135	8:12	157	12:32	179	16:41	201	18:13
114	6:00	136	8:18	158	12:48	180	16:47	202	18:17
115	6:08	137	8:26	159	13:04	181	16:53	203	18:21
116	6:16	138	8:34	160	13:21	182	16:57	204	18:25
117	6:23	139	8:42	161	13:36	183	17:01	205	18:29
118	6:30	140	8:50	162	13:51	184	17:05	206	18:35
119	6:36	141	8:58	163	14:06	185	17:09	207	18:41
120	6:42	142	9:06	164	14:21	186	17:13	208	18:49
121	6:48	143	9:14	165	14:36	187	17:17	209	18:57
122	6:54	144	9:22	166	14:50	188	17:21	210	19:06
123	7:00	145	9:32	167	15:04	189	17:25	211	19:16
124	7:06	146	9:44	168	15:18	190	17:29	212	19:26
125	7:12	147	9:56	169	15:28	191	17:33	213	19:36
126	7:18	148	10:10	170	15:36	192	17:37	214	19:51
127	7:24	149	10:24	171	15:44	193	17:41	215	20:06
128	7:30	150	10:40	172	15:52	194	17:45	216	20:21
129	7:36	151	10:56	173	15:59	195	17:49	217	20:36
130	7:42	152	11:12	174	16:06	196	17:53	218	20:51
131	7:48	153	11:28	175	16:13	197	17:57	219	21:06
132	7:54	154	11:44	176	16:20	198	18:01	220	21:21

Table 5
Timetable of route 3

Trip	Departure time	Trip	Departure time	Trip	Departure time	Trip	Departure time	Trip	Departure time
221	8:00	230	10:00	239	12:20	248	14:40	257	16:20
222	8:12	231	10:15	240	12:36	249	14:55	258	16:30
223	8:24	232	10:30	241	12:52	250	15:10	259	16:40
224	8:36	233	10:45	242	13:08	251	15:20	260	16:50
225	8:48	234	11:00	243	13:24	252	15:30	261	17:00
226	9:00	235	11:16	244	13:40	253	15:40		
227	9:15	236	11:32	245	13:55	254	15:50		
228	9:30	237	11:48	246	14:10	255	16:00		
229	9:45	238	12:04	247	14:25	256	16:10		

Table 6
Deadhead trips

Dead head trip	Departure deport	Departure time	Arrival deport	Arrival time
DHAC1	A	8:24	C	8:44
DHAC2	A	8:36	C	8:56
DHAB1	A	8:44	B	9:09
DHAB2	A	8:52	B	9:17
DHCB1	C	17:09	B	17:24
DHCB2	C	17:19	B	17:34

ject not only meets the range of public transport electric vehicle time limited but also can meet each dispatch missions request, it is a feasible solution to above vehicle scheduling problem.

Table 7
scheduling result

Vehicle	Trips	Work time	Spread time
1	1 → 8 → 16 → 27 → 41 → 46 → 50 → 54 → 58 → 62 → fueling → 91 → 98	576	841
2	2 → 9 → 19 → 31 → 79 → 90 → 97 → 103 → 107	432	940
3	3 → 10 → 18 → 30 → DHAB1 → 170 → 177 → 187 → 200	409	811
4	111 → 117 → 125 → 135 → 171 → 178 → 190 → 202 → 210	432	864
5	4 → 12 → 22 → 34 → 43 → 47 → 52 → 56 → 59 → 63 → fueling → 93	528	779
6	5 → 13 → 23 → 35 → 74 → 82	288	705
7	112 → 119 → 128 → 136 → 142 → 147 → 151 → 155 → 161 → 165 → fueling	480	760
8	6 → 14 → 24 → 37 → 71 → 78 → 87 → 95 → 101 → 105 → fueling	480	1058
9	113 → 120 → 129 → 137 → 143 → 167 → 172 → 179 → 191 → 203 → fueling	480	977
10	7 → 15 → 26 → DHAC1 → 225 → 251 → 257	341	682
11	114 → 121 → 133 → 140 → 146 → 150 → 154 → 158 → 162 → 166 → fueling → 207 → 213 → 217	624	924
12	115 → 123 → 131 → 173 → 181 → 194 → 206 → 212 → 216 → 220 → fueling	480	1141
13	116 → 124 → 132 → 139 → 145 → 149 → 152 → 156 → 159 → 163 → fueling	528	753
14	11 → 20 → 33 → DHAB2 → 144 → 148 → 153 → 157 → 160 → 164 → fueling	457	707
15	118 → 127 → 174 → 182 → 195	240	727
16	122 → 130 → 138 → 169 → 176 → 185 → 197 → 208	384	763
17	17 → 29 → DHAC2 → 254 → 260	234	651
18	21 → 36 → 44 → 48 → 51 → 55 → 61 → 65 → 70 → 77 → fueling	480	760
19	126 → 134 → 141 → 168 → 175 → 184	288	635
20	25 → 38 → 69 → 76 → 84 → 92 → 99 → 104 → 108	432	821
21	28 → 40 → 67 → 73 → 81 → 89	288	625
22	221 → 226 → 230 → 234 → 238 → 242 → 246 → 250 → fueling	472	669
23	32 → 42 → 66 → 72 → 80 → 88 → 96 → 102 → 106 → 110 → fueling	480	1003
24	222 → 227 → 231 → 235 → 239 → 243 → 247 → 252 → fueling	472	677
25	223 → 228 → 232 → 236 → 240 → 244 → 248 → 253 → fueling	472	675
26	39 → 45 → 49 → 53 → 57 → 60 → 64 → 68 → 75 → 83 → fueling	528	780
27	224 → 229 → 233 → 237 → 241 → 245 → 249 → 255 → fueling	472	683
28	256 → DHCB1 → 189	122	123
29	180 → 192 → 204 → 211 → 215 → 219	59	59
30	183 → 196	96	100
31	86 → 94 → 100	144	145
32	186 → 199 → 209 → 214 → 218	240	266
33	DHCB2 → 193 → 205	111	118
34	188 → 201	96	100

6. Conclusion

One aim of creating an ACA for VSPRFTC is to advance our knowledge of the problem itself. Anther is to find a better solution method. New formulations for VSPRFTC of Electric bus were presented. The solving process had been detached into two steps. In step 1, a new ant colony optimization based approach to solve VSPRFTC of electric bus was introduced. In particular, algorithm has been designed to solve VSPRFTC with two objective functions: (i) the minimization of the number of tours (or vehicles) and (ii) the minimization of the total deadhead time, where number of tours minimization takes precedence over deadhead time minimization. Our algorithm introduces a new methodology for optimizing multiple objective functions. The basic idea is to coordinate the activity of different ant colonies, each of them optimizing a different objective. These colonies work by using independent pheromone trails but they collaborate by exchanging information. All trips had been assigned to vehicles after this algorithm finished. In step 2, in dealing with the fueling time constraint a bipartite graphic model and its optimization algorithm are developed for trip set connecting in a hub and spoke network system to minimize the number of vehicle required. The maximum matching of the bipartite graph is obtained by calculating the maximum inflow with the Ford–Fulkerson algorithm. A solution obtained by using this algorithm to solve the real-world VSPRFTC. From the result on the test problem, we can conclude that the model, the heuristic procedures are quite successful in solving VSPRFTC.

Please cite this article in press as: H. Wang, J. Shen, Heuristic approaches for solving transit vehicle ..., Appl. Math. Comput. (2007), doi:10.1016/j.amc.2007.02.141

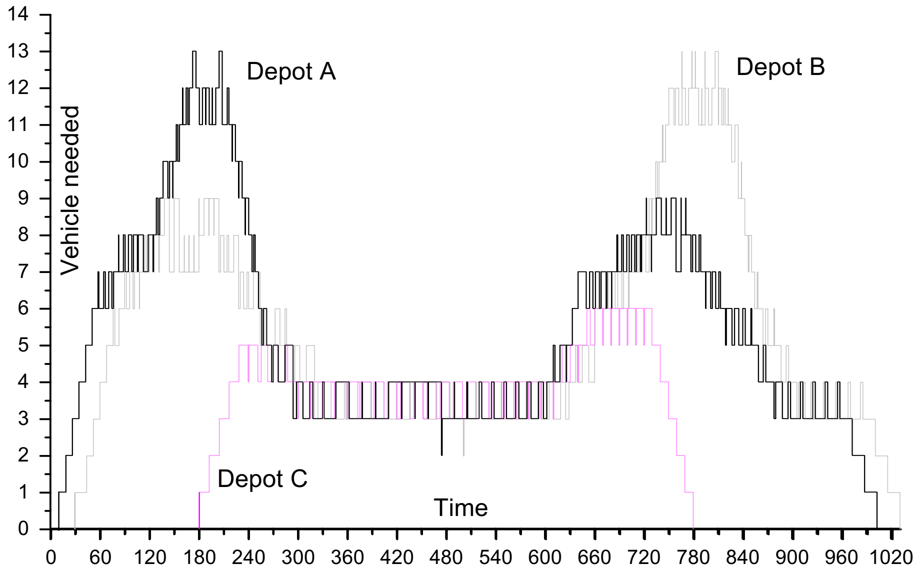

Fig. 5. vehicles needed of every deport changing with time.

References

[1] Ali Haghani, Mohamadreza Banihashemi, Kun-Hung Chiang, A comparative analysis of bus transit vehicle scheduling models, Transportation Research Part B 37 (2003) 301–322.

[2] F. Baita, R. Pesenti, W. Ukovich, D. Favaretto, A comparison of different solution approaches to the vehicle scheduling problem in a practical case, Computers & Operations Research 27 (2000) 1249–1269.

[3] L. Bodin, B. Golden, A. Assad, M. Ball, Routing and scheduling of vehicles and crews the state of the art, Computer and Operations Research 10 (1983) 63–211.

[4] Ali Haghani, Mohamadreza Banihashemi, Heuristic approaches for solving large-scale bus transit vehicle scheduling problem with route time constraints, Transportation Research Part A 36 (2002) 309–333.

[5] Guy Desaulniers, June Lavigne, Francois Soumis, Multi-depot vehicle scheduling problem with time windows and waiting costs, European Journal of Operational Research 111 (1998) 479–494.

[6] Yoshiyuki Karuno, Hiroshi Nagamochi, An approximability result of the multi-vehicle scheduling problem on a path with release and handling times, Theoretical Computer Science 312 (2004) 267–280.

[7] Hoong Chuin Lau, Melvyn Sim, Kwong Meng Teo, Vehicle routing problem with time windows and a limited number of vehicles, European Journal of Operational Research 148 (2003) 559–569.

[8] John E. Bell, Patrick R. McMullen, Ant colony optimization techniques for the vehicle routing problem, Advanced Engineering Informatics 18 (2004) 41–48.

[9] Silvia Mazzeo, Irene Loiseau, An ant colony algorithm for the capacitated vehicle routing, Electronic Notes in Discrete Mathematics 18 (2004) 181–186.

[10] Marc Reimann, Karl Doerner, Richard F. Hartl, D-ants: savings based ants divide and conquer the vehicle routing problem, Computers & Operations Research 31 (2004) 563–591.