A method to plan group tours with joining and forking

Munenobu NAGATA[1] and Yoshihiro MURATA[1] and Naoki SHIBATA[2] and Keiichi YASUMOTO[1] and Minoru ITO[1]

[1] Nara Institute of Science Technology, Japan.
[2] Shiga University, Japan.

Abstract. Group sightseeing has some advantages in terms of required budget and so on. Some travel agents provide package tours of group sightseeing, but participants have to follow a predetermined schedule in tour, and thus there may be no plan which perfectly satisfies the tourist's expectation. In this paper, we formalize a problem to find group sightseeing schedules for each user from given users' preferences and time restrictions corresponding to each destination. We also propose a Genetic Algorithm-based algorithm to solve the problem. We implemented and evaluated the method, and confirmed that our algorithm finds efficient routes for group sightseeing.

1 Introduction

Personal navigation system guides its user through a mobile terminal such as mobile phone or PDA. Until now, there has been a lot of researches on personal navigation system. For example, indoor route guidance system[2] and a system which provides sightseeing information through mobile terminal[3] are proposed. These personal navigation systems' primary objective is to guide the user towards single destination or to provide information, and they lack functionality to guide the user through multiple destinations within limited time period, which is common in sightseeing. We have already proposed a personal navigation system P-Tour which guides user through multiple destinations. P-Tour finds a route schedule to tour multiple sightseeing destinations considering user's preferences[9]. The previous P-Tour provides functionality to guide single user only, but it is quite common that a group of members go sightseeing together in order to save traveling cost by riding on the same vehicle. In this paper, we propose an extension for P-Tour which finds a route schedule on group sightseeing. On group sightseeing, the members would prefer (1) visiting each destination with other members, (2) forking from other members and visiting special destinations, (3) visiting each destination taking into account of all members' preferences, and (4) visiting each destination efficiently within limited time. In order to achieve these objectives, we formalized the problem to find routes for each user from user's preferences and restrictions, designed and implemented a GA-based algorithm to solve the problem. Then, we evaluated the method through experiments, and confirmed that our algorithm finds efficient routes for group sightseeing.

2 Related works

Genetic algorithm is a combinatorial optimization algorithm inspired from evolution in nature, and it uses crossover, mutation, evaluation and selection operations. There are many applications of genetic algorithm including traditional combinatorial optimization problems such as Knapsack problem[8], Traveling Salesperson Problem [10], Set Coverage Problem[1], and so on. Besides them, genetic algorithm can be used for solving engineering problems such as Job Shop Scheduling Problems[7], and multiple processor scheduling problems[4].

We have also used Genetic Algorithm in our already proposed personal navigation system "P-Tour"[9]. P-Tour can plan schedule around multiple destinations with many restrictions.

There are also problems to gain benefits of whole system (or all users) such as group sightseeing schedule planning problem, explained below.

- Theme park problem[5]: A problem to reduce congestion and improve customer's satisfaction in theme park by adjusting presentation of information and booking of each facilities.
- Delivery scheduling problem[6]: A problem to find the optimal delivery path when there are many parcels with labels to deliver, and many vehicles with limited maximum load. The objective is to minimize the total distance of the path and improve customer's satisfaction.

Since these existing studies do not handle forking and joining of passenger groups, these methods are different from our study.

3 Route scheduling for group sightseeing

In the proposed method, we find a sightseeing schedule in which members of group sightseeing fork and join on the way, shown in Fig.1.

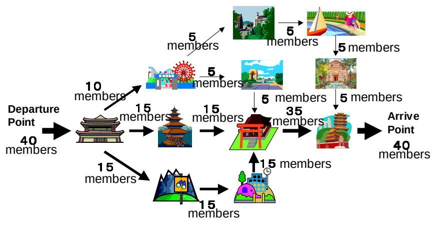

Fig. 1. Concept of group sightseeing schedule

Finding an efficient group sightseeing schedule involves issues which does not exist on scheduling for single person tour.

- **Differences of preferences between members**: Each member may have different preferences for each destination. The system has to decide from: (1) making members visit same destinations by ignoring part of members' preferences, (2) making compromised route, or (3) making part of members fork from other members.

– **Differences between starting and ending points of the route of each member**: The system has to handle each member's starting point of the route. Besides it, part of members may start sightseeing later than other members. Different time and locations of ending points should also be considered.

4 Definition of the problem

In this section, we first define the problem to find an efficient schedule for group sightseeing. On group sightseeing, the members usually prefer visiting each destination with other members. But, the members have different conditions regarding to starting location, returning location, and preference of each destination. The objective of the problem is to maximize users' satisfaction which is evaluated by summing up each member's satisfaction value which increases when visiting each destination with other members, and when preferred destinations are included in the route.

4.1 Input

The input of the problem is shown below.

– Map data, given as a directional graph $G = (V, E)$. Each edge is assigned a length. Minimum distance between two given vertices v_1 and v_2 is referred as $dist(v_1, v_2)$.
– Destination data $D = \{d_1, ...\}$, each element is given as tuple of following information.
  1. Name of the destination(e.g. Horyu-ji Temple).
  2. Corresponding vertex $v_h \in V$.
  3. rt_{ij}: The latest arrival time for member u_i at destination d_j. (for example, if rt_{ij} = 12:00, member u_i has to reach destination d_j before 12:00).
  4. dur_{ij}: Restriction of staying time for member u_i at destination d_j.
  5. pre_{ij}: Preference value for member u_i at destination d_j.
– Participant data: each member has following five parameters.
  - $pd_{is} \in D$: Starting point of member u_i.
  - $pd_{ig} \in D$: Returning point of member u_i.
  - pt_{is}, pt_{ig}: Time restriction for member u_i at start / returning points.
  - $speed_i$: Speed of user u_i.

4.2 Notations

The group sightseeing schedule is composed of schedules of each member. The schedule for member u_i is denoted as $s_i = (D_i, Stay_i)$, where D_i is a list of destinations visited by member u_i, defined as $D_i = \langle d'_{i1}, d'_{i2}, ..., d'_{ij}, ..., d'_{i|D_i|}\rangle$, and d'_{ij} is member u_i's j-th destination. $Stay_i$ is the list of stay time for member u_i at each destination, defined as $Stay_i = \langle stay_{i1}, stay_{i2}, ..., stay_{i|D_i|}\rangle$, and $stay_{ij}$

is the stay time for which member u_i stays at destination d_j. The arrival time of member u_i at destination d'_{ij} is denoted as t_{ij}, and calculated as follows : $t_{i(j+1)} = t_{ij} + stay_{ij} + \frac{dist(d'_i j, d'_{i(j+1)})}{speed_i}$. Usually, $stay_{ij}$ is equal to dur_{ij}, but if members have to join at the destination, and some members arrives earlier, other members have to wait at the destination.

4.3 Evaluation function

Evaluation function f is defined as follows:

$$\begin{aligned} f(S) = \sum_{i=1}^{|U|} \{ \alpha \sum_{j=1}^{|D_i|} pre_{ij} \cdot timeok(s_i, d_{ij}) \cdot group(S, i, j) \\ - \sum_{j=1}^{|D_i|-1} \big(\beta \cdot dist(d_{ij}, d_{i(j+1))}) \\ + \gamma \cdot commonpath(S, i, j, j+1)) - \delta \cdot timegoal(S_i, p_{ig}) \} \end{aligned} \qquad (1)$$

α, β, γ and δ are constant values.

Function $timeok(s_i, d'_{ij})$ returns 1 iff destination $d'_i j$ is included in route s_i and both restrictions rt_{ij} and dur_{ij} are satisfied.

Function $group(S, i, j)$ returns the number of members who visit destination d_{ij} together on the route S. By this term, evaluation value increases when many members visit a destination together.

Function $commonpath(S, i, j, j + 1)$ returns the number of members who move from d_{ij} to $d_{i(j+1)}$ together on the route S. By this term, evaluation value increases when many members move together.

Function $timegoal(s_i, p_{ig})$ returns an absolute value of difference between arrival time and expected arrival time at destination s_i. Evaluation value decreases when they arrive too early or too late.

the size of touring group between destinations.

4.4 Algorithm

In this section, we describe the GA-based scheduling algorithm for group tours.

Fig. 2 shows the routes of two members represented as a list of genes. One chromosome consists of lists of genes for all members. Each gene in the list represents destination, and the traveling order is represented by the path from left side to right side of the list, in the figure. Genes $P_{11}, \ldots, P_{14}$ represent destinations in the route for member u_1, and genes $P_{21}, \ldots, P_{24}$ represent destinations in the route of member u_2.

P_{ij} is genes called normal gene. R_{11}, R_{12}, R_{21}, R_{22}, R_{23} are called reference genes. Reference genes do not represent destinations directly, but points the destinations visited by another member. Reference genes have three elements: referred member, unique value, hidden gene.

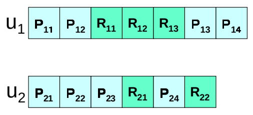

Fig. 2.
Chromosome

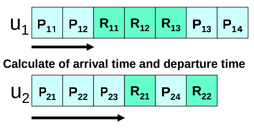

Fig. 3.
Calculation of arrival time and departure time

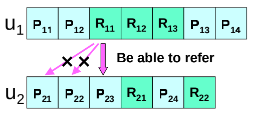

Fig. 4.
Search for gene that can be referred

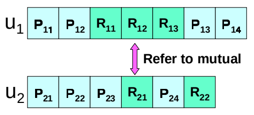

Fig. 5.
Mutual reference

Decoding of reference genes is performed as shown in Fig.3. At first, arrival times and departure times of destinations represented by normal genes are calculated until reference gene appears.

Next, referred gene is searched. Referred gene represents the destination such that the referring member arrives at the destination represented by referred gene before referred member (here, R_{A1} and R_{B1}).

If destination represented by referred gene is included in referring members' route, the hidden gene is activated. Hidden gene represents another destination called hidden destination.

Reference gene may refer to another reference gene, and references between genes can be cyclic(Fig.5). In this case, the unique values of reference genes are compared each other. The hidden gene of the reference gene with the largest unique value is activated.

GA operations are composed of generation of initial population, evaluation, selection, crossover, and mutation. Let M denotes the number of members, N denotes population size, and I denotes the number of generations.

1. Generation of initial population: N chromosomes are randomly generated as initial population. Each chromosome represents candidate solution.
2. Evaluation: The evaluation values are calculated with evaluation function described in section 3.3.
3. Selection: We use elite strategy and tournament selection.
4. Crossover: We use two point crossover. If there are redundant destinations, they are deleted.
5. Mutation: we use random insertion, deletion, exchange, change of reference and conversion.
6. One GA generation is step 2 to 5, and these steps are repeated until a good solution is obtained.

5 Evaluation experiments

To evaluate the proposed method, we conducted experiments to evaluate the following two points.

– Quality of obtained schedules

– Optimality of obtained schedules

In the experiments, we used the map of Nara prefecture (digital map 25000 issued by Geographical Survey Institute of Japan). We executed the proposed algorithm on an ordinary PC with Pentium M 2.0GHz, 512M Memory, Windows XP pro., Java 1.4.2. We assumed that tourists move by car and their speed is 40 km/h.

Also, we set parameter values from preliminary test as follows: $N = 1000$, $I = 200$, $W = 30$, $L = 20000$, $\alpha = 50$, $\beta = 0.015$, $\gamma = 15$ and $\delta = 10$.

5.1 Quality of obtained schedules

We input data of 3 members shown in Table 1, 8 kinds of destination data and evaluated the obtained schedule. Details of the values are as follows: if a user wants to visit one of the destinations, its preference value is set to 5, and if not, it is set to -10. Stay time is 60 minutes for all destinations.

Table 1. Data for 30 destinations

requested by all members	none($\emptyset$)	
Yakushi-ji temple(A1)	NAIST(E1)	
Todai-ji temple(A2)	Taima temple(E2)	
Horyu-ji temple(A3)	Kongou shrine(E3)	
	Tamaki shrine(E4)	
	Akisino temple(E5)	
	Suijin tennoryou(E6)	
requested by only u_1	requested by only u_2	requested by only u_3
Kasuga-taisya shrine(B1)	Zinmu tennoryou(C1)	Syoumu tennoryou(D1)
Ishigami shrine(B2)	Hase temple(C2)	Torinoyama kofun(D2)
Tyougaku temple(B3)	Dansan shrine(C3)	Keikou tennoryou(D3)
Houzan temple(B4)	Ishibutai kofun(C4)	Sigi-moutain Nodoka villeage(D4)
requested by u_1, u_2	requested by u_1, u_3	requested by u_2, u_3
Mesuri-moutain kofun(F1)	Kamotoba shrine(G1)	Tennnouzan kofun(H1)
Yashikiyama kofun(F2)	Oogami shrine(G2)	Ruins of Fujiwarakyu (H2)
Oono temple(F3)	Nukataryou(G3)	Suidei kofun(H3)

Next, we evaluate paths obtained by our method. Calculation time for those paths is about 1.3 minutes total. The obtained schedule is shown in Figure 6. Figure 7 is all members' paths overlapped.

Now, we give some explanations regarding to the obtained schedule. Member u_1 departures NAIST(E1) at 8:34, and tours 3 destinations (B3, G3, B1) alone until 11:53. The member sets high preference values for these three destinations. Member u_2 departures Hozanji (B4), joins with u_3, and they tour Tennozan Kofun (H1) together. Member u_2 waits from 9:43 to 11:09 for member u_3 before joining. After that, u_1 joins them at Todaiji(A2), and tours 3 destinations (A2, A1, A3) together. All three members set high preference values for these 3 destinations. Also, member u_2 and u_3 have to wait from 11:38 to 11:45 for member

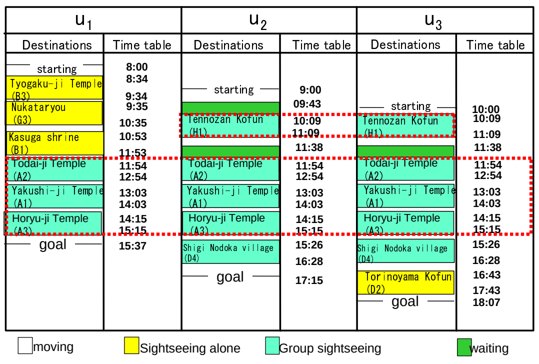

Fig. 6. A time table of obtained schedule

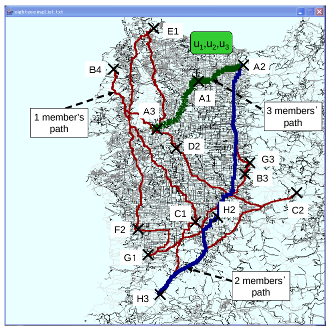

Fig. 7. An obtained path of group

u_1 to join. u_1 forks after touring these 3 destinations, and arrives at his final destination. After that, member u_2 and u_3 tour Shigi Nodoka village(D4) together. Then, member u_2 and u_3 splits. Member u_2 goes to his goal, and member u_3 tours Torinoyama Kofun(D2) before reaching his final destination(Hasedera C2). Member u_3 sets a high preference value for Torinoyama Kofun (D2).

As shown above, our method can obtain schedule with joining and forking, and this schedule includes destinations with high preference values for each member.

5.2 Optimality of obtained schedules

To evaluate optimality of the obtained schedule by the proposed method, we compared the obtained schedules with the optimal schedules obtained by full search. We set the parameters as follows: the number of members is 3, the number of destination is 3 or 4. The results are shown in Table 2 These results are average values of 10 trials.

Table 2. Comparison of obtained schedules with the optimal solutions

Number of destinations	Number of combinations (search space)	computation time proposed method	computation time full search	error rate (%)
3	$(21)^3$	about 10(sec.)	about 2(min.)	0%
4	$(142)^3$	about 13(sec.)	about 18(hour)	0%
5	$(12336)^3$	-	-	-

In the cases of both 3 and 4 destinations, our method obtains the optimal schedule. Also, our method was a lot faster than the full search algorithm. Since the calculation time of the full search algorithm grows exponentially when the number of destinations increases, we could not observe the case above 4 destinations.

Also, In order to evaluate effect of introducing reference genes, we compared proposed method with and without reference genes.

In the case without reference genes, our method with reference genes achieved 42.5% to 76.5% better fitness values than one without reference genes.

In the case without reference genes, members can join only if their schedules happen to include the same point at the same time. Since members has different starting points, it can find only few schedules with joining.

6 Conclusion

We proposed a method to find a schedule for group tour with different route for each member. Each member of group sets own preference and restrictions. In this paper, we formalized the scheduling problem on group sightseeing. And to solve this problem, we designed scheduling algorithm based on GA. Moreover, we evaluated the method through experiments by using the map data on the northern part of Nara Prefecture, and confirmed that our algorithm finds efficient routes for group sightseeing.

We are planning to improve the search efficiency by introducing the local search method.

References

1. Al-Sultan, K.S., Hussain, M.F., Nizami, J.S., "A Genetic Algorithm for the Set Covering Problem", *Journal of the Oper.Res. Society*, Vol. 47, pp. 702–709, 1996.
2. Butz, A., Baus, J., Krüger, A., Lohse, M., "A Hybrid Indoor Navigation System", *Proc. of IUI2001: International Conf. on Intelligent User Interfaces 2001*, ACM Press, New York, pp. 25–33, 2001.
3. Cheverst, K., Davies, N., Michell, K., Friday, A., "The Design of an Object Model for a Context-Sensitive Tourist Guide", *Computers Graphics Journal*, Vol. 23, No. 6, pp. 883–891, 1999.
4. Hou E.S.H., Ansari N., Ren, H., "A Genetic Algorithm for Multiprocessor Scheduling", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 5, No. 2, pp. 113–120, 1994.
5. Kawamura, H., Kurumatani, K. and Ohuchi, A.: "Modeling of Theme Park Problem with Multiagent for Mass User Support", *Working Note of the IJCAI-03 Workshop on Multiagent for Mass User Support*, pp. 1–7 (2003)
6. Ohta, M., Shinoda, K., Noda, I. and Kunifuji, S., "Usability of Demand-bus in Town Area", *Domestic Conference of Intelligence Transportation Systems* (in Japanese), Vol. 115, pp. 239–245, 2002.
7. Ono, I., Yamamura, M.,Kobayashi, S., "A Genetic Algorithm for Job-shop Scheduling Problems Using Job-based Order Crossover", *Proc. of 1996 IEEE International Conf. on Evolutionary Computation*, pp. 547–552, 1996.
8. Richard, S., "Solving large knapsack problems with a genetic algorithm", *In IEEE International Conf. on Systems, Man and Cybernetics*, Vol. 1, pp. 632–637, 1995.
9. Shiraishi, T., Nagata, M., Shibata, N., Murata, Y., Yasumoto, K., Ito, M., "A Personal Navigation System with a Schedule Planning Facility Based on Multiobjective Criteria", *Proc. of 2nd Int'l. Conf. on Mobile Computing and Ubiquitous Networking*, pp. 104–109, 2005.
10. Whitley, D., Starkweather, T., Fuquay, D., "Scheduling problems and traveling salesmen: The genetic edge recombination operator", *In Proc. of the Third International Conf. on Genetic Algorithms and their Applications*, pp. 133–144, 1989.