*Regular Paper*

# A Personal Navigation System with Functions to Compose Tour Schedules Based on Multiple Conflicting Criteria

Takayuki Shiraishi,† Munenobu Nagata,† Naoki Shibata,††
Yoshihiro Murata,† Keiichi Yasumoto† and Minoru Ito†

In order to allow tourists to travel to multiple destinations efficiently, we need a personal navigation system which computes and shows the best route to the next destination and facilitates composition of a schedule for visiting those destinations taking account of various requirements such as relative importance among destinations, time restrictions, travel expenses, and so on. There is sometimes a tradeoff between these requirements. In this paper, we extend our existing personal navigation system called P-Tour in the following two ways: (1) allowing users to optimize their tour schedules under multiple conflicting criteria such as total expenses and satisfaction degrees; and (2) navigating users to the next destination in a more efficient way. We have implemented the above extensions and integrated them into P-Tour. Through some experiments, we show the effectiveness of the proposed extensions.

## 1. Introduction

Recent progress of portable computing devices (such as PDAs and cell phones) and wireless communication infrastructure (such as wireless LAN and wide-band CDMA) is remarkable. Among the applications with those portable devices with wireless communication capability, pedestrian navigation systems have had much attention.

Various pedestrian navigation systems using portable devices have been developed so far [1)~5)].

(1) proposes a system called *REAL* which detects the user's current context and navigates the user in the most suitable way based on the context.
(2) proposes interfaces through which users can receive navigation services from various terminals.
(3) proposes an indoor navigation system which aims at provisioning the coherent navigation service to different devices.
(4) proposes a system to support planning a bicycle tour by showing different routes for the round trip to a destination.

In Japan, a pedestrian navigation system called *EZ-Naviwalk* [6)] is available through cell phones, where the graphical map around the current user's location is continuously displayed on the display with the route to the specified destination like car navigation systems.

In Ref. 7), we have proposed a personal navigation system for tourism called *P-Tour*, which allows a user to compose the near best schedule to visit multiple destinations under various restrictions on the total distance, arrival and stay time, relative importance among candidate destinations, and so on. P-Tour uses a GA based route search engine to solve a combinatorial optimization problem in practical time.

Existing navigation systems including P-Tour search the route which maximizes one objective function. Consequently, if one wants to deal with multiple conflicting objectives such as minimizing total travel expenses and maximizing satisfaction degrees (i.e., the sum of satisfaction degree assigned to each destination), these conflicting functions have to be unified to one with a certain proportion in advance. The proportion, however, could be different from user to user. A user may want to choose one depending on the situation.

On the other hand, existing navigation systems with only one destination re-computes a new route when a user goes off the route. In a tour, however, the schedule has to be modified if users cannot reach a destination on time. So, an early detection mechanism is needed to prompt users to reach the destinations on time.

In this paper, we extend P-Tour in the following two ways: (1) allowing users to optimize their tour schedules under multiple conflicting criteria such as total expenses and satisfaction degrees; and (2) navigating users to the next destination in a more efficient way. For

† Graduate School of Information Science, Nara Institute of Science and Technology
†† Department of Information Processing and Management, Shiga University

the above (1), we formulated the schedule planning as a multi-objective optimization problem, and implemented an algorithm to derive near pareto-optimal solutions using a GA. This allows a user to choose the best one from multiple solutions with various balances among multiple criteria. For (2), we designed and implemented a new navigation facility which tracks the user's location, checking whether the user is in the appropriate area to be able to reach the next destination on time, and informs the user of how he/she should do to follow the schedule.

Through several experiments, we have confirmed that the proposed schedule planning method is useful for users to intuitively select the best schedule from multiple candidates with multiple criteria and that our new search engine can calculate the near optimal solution in practical time.

## 2. Proposed Extensions to P-Tour

In this section, we explain the outline of our personal navigation system P-Tour [7], and the proposed extensions.

### 2.1 Outline of P-Tour

In general, when we plan a travel schedule, we usually consider the following requirements: (1) number of destinations: we want to efficiently visit as many destinations as possible within allowable time; (2) timeliness: we want to visit some of destinations timely according to starting time of an entertainment show, business hours of the facility, and so on; and (3) preference: we want to select the best subset of all candidate destinations under various restrictions if we do not have enough time to visit all of them.

P-Tour finds a near-best schedule from user's input which includes a starting location, departure time, a returning location, arrival time and multiple candidate destinations with relative importance and time restrictions on arrival time and staying hours. Also, P-Tour has a functionality to navigate users to reach the next destination like car navigation systems and prompt them to leave on time at each destination.

As shown in **Fig. 1**, P-Tour consists of a client module which runs on a cell phone or a PDA with a GPS unit and communication capability, and a server module which runs on a PC connected to the Internet. The server module retains a map data, a database of recommended destinations, and other tour related

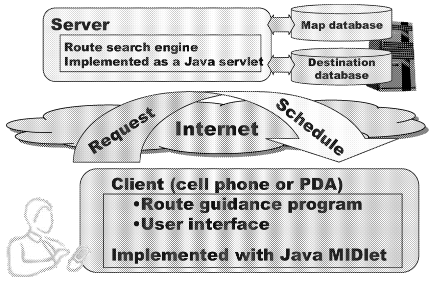



**Fig. 1** System structure of P-Tour.

information such as entrance fees and average staying hours at each destination. Each user with a cell phone or a PDA can receive services (schedule composition and navigation services) from the server module through the network.

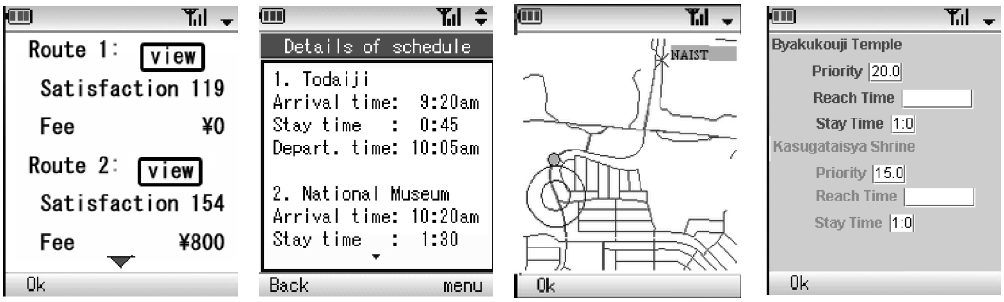
### 2.2 Proposed Extensions

There are many criteria to construct a tour schedule, such as travel expenses, total time/distance to walk, and so on. These criteria often conflict with each other and it is hard to satisfy all of them simultaneously. Thus, we have to find a compromise. However, when formalizing the problem for composing a schedule, we have to specify the proportion among them (i.e., how important each criterion is for a user) beforehand. In this paper, we formalize the problem as a multi-objective optimization problem [8] so that we can obtain multiple solutions with various compromises at the same time. This approach allows users to choose one of the solutions intuitively. For example, it could be hard to specify that "I can pay extra $3 to save 20 minutes." in advance. On the other hand, it would be easy to choose one from two actual routes, say, "a route with 25 minutes walk" and "a route with 5 minutes bus-ride for $3."

The multi-object optimization problem is a problem to find the set of *pareto-optimal* solutions. A pareto-optimal solution is a solution such that any other solutions superior to the solution in one criterion are always inferior at least in another criterion.

Since most problems to find pareto-optimal solutions are known to be NP-hard, it takes huge time to compute the optimal solutions. Thus, we use a genetic algorithm to compute near pareto-optimal solutions. We describe details of our implementation in Section 4.

We also extend a navigation function in P-Tour. The existing navigation function of P-Tour consists of four modes: (i) schedule selection mode; (ii) schedule display mode; (iii) nav-




(a) Schedule selection mode (b) Schedule display mode (c) Navigation mode (d) Interface for user input

**Fig. 2** Display of P-Tour.

igation mode; and (iv) stay mode. In the schedule selection mode, the server module calculates schedules based on user inputs. The server module sends schedules to the user and the user can select one favorite schedule among them (**Fig. 2** (a)). In the schedule display mode, progress of the tour, estimated arrival time of destinations, and their staying hours are displayed as shown in Fig. 2 (b). In the navigation mode, the route to the next destination is displayed like a car navigation system as shown in Fig. 2 (c). In the stay mode, the remaining time for stay, departure time, and information about the current destination are displayed. The navigation mode and the stay mode are automatically switched to each other depending on the user's current location obtained from GPS and clock. Users can switch to the schedule display mode at any time.

We assume that the server module of P-Tour retains the distance and the average speed at which users can move between any two destinations in the database. However, due to some reasons such as frequent stop at traffic lights, crowdedness along the route, or taking the wrong way, users might not be able to reach the next destination on time if they are not aware of the schedule.

So, in this paper, we extend a function which tracks the user's location and prompts him/her to walk faster/slower or warns him/her to return to the right route, depending on how far he/she is off the appropriate location. We explain details in Section 3.

## 3. Extended Functions in P-Tour

In this section, we explain new functions extended in our navigation system P-Tour.

### 3.1 Finding a Best Schedule Based on Multiple Conflicting Criteria

Through the interface in Fig. 2 (d), a user inputs a starting location, the departure time, a returning location, the arrival time, multiple candidate destinations and multiple criteria to be considered.

Each user can either choose destinations from those registered in the database or input them as coordinates on the map. The database also includes shortest routes and users' moving speeds between any pairs of registered destinations. When a user inputs a destination not registered in the database, the routes and moving speeds are calculated on the fly.

Since our route search engine computes near pareto-optimal solutions which may contain too many candidate solutions, it is not preferable to display all of them. So, we have developed an interface which picks up several representative solutions and indicates them as candidates.

### 3.2 Displaying Representative Solutions

Our system picks up several representative solutions from many near pareto-optimal solutions, and show them to the user as candidate schedules. Here, we use k-means method[11] to cluster the candidate solutions, and solutions nearest to the centroid of each cluster are picked up (**Fig. 3**). If there is no schedule which suits the user's preference, the user can request the system to display other solutions from other candidate solutions, as shown in **Fig. 4**. To do so, the user selects the solution nearest to his/her ideal schedule (e.g., cluster (b) in Fig. 3). Then the selected cluster is further divided into $k$ sub-clusters (cluster (b1), (b2) and (b3) in Fig. 4), and solutions nearest to the centroid of each cluster are picked up and displayed as well. By repeating the above pro-

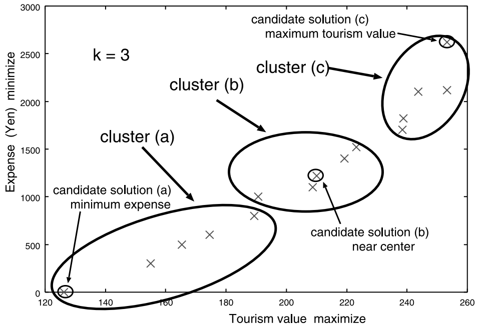



**Fig. 3** Selecting representative solutions.

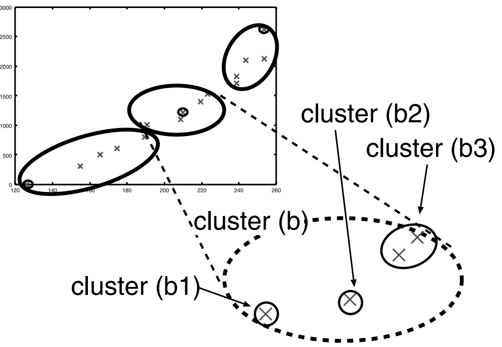



**Fig. 4** Reselecting representative solutions.

cess, the user can explore all the candidates to select the best one.

### 3.3 Prompting Users to Follow the Schedule

This function makes a user aware of if he/she is following the schedule appropriately.

#### 3.3.1 Detecting Undesirable Situations

We have developed a function for detecting undesirable situations that may result in the modification of the current schedule. The basic idea is to define the area in which a user should be at each time so that we can detect an undesirable situation by monitoring if the users go off the area.

In P-Tour, all roads/streets are treated as edges in a graph, and each route between two destinations is approximated by a series of connected edges. We call each connection point of the edges as a *node*. After a schedule is derived, we let the server module calculate the estimated time when the user passes through each node on the route. These data are transfered to the client module.

The algorithm to detect undesirable situations of a user is shown below.

(1) Calculate the expected location of the user $\mathbf{x}_{\mathrm{scheduled}}(t)$ at time $t$, assuming that user moves at a constant speed registered in the database.

(2) Measure the user's actual location $\mathbf{x}_{\mathrm{current}}(t)$ at time $t$, using a GPS unit of the client system.

(3) Find the location $\mathbf{x}_{\mathrm{current\_on\_road}}(t)$ on the route which is the nearest from $\mathbf{x}_{\mathrm{current}}(t)$.

(4) Let $error_{\mathrm{x}}$ be the distance between $\mathbf{x}_{\mathrm{current}}(t)$ and $\mathbf{x}_{\mathrm{current\_on\_road}}(t)$, as shown in **Fig. 5**.

(5) Let $error_{\mathrm{y}}$ be the distance on the route between $\mathbf{x}_{\mathrm{scheduled}}(t)$ and $\mathbf{x}_{\mathrm{current\_on\_road}}(t)$, as shown in Fig. 5. If the user is going behind the schedule, $error_{\mathrm{y}}$ is multiplied by $-1$.

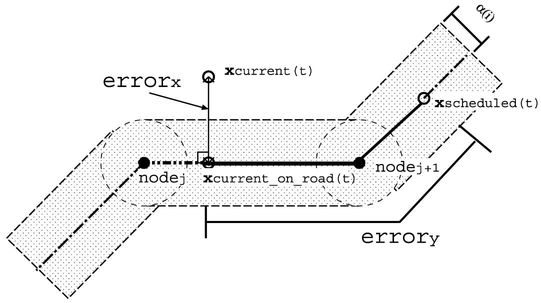



**Fig. 5** Detection of undesirable situations.

The user's situation is defined by the values of $error_{\mathrm{x}}$ and $error_{\mathrm{y}}$ as follows.

| | |
|---|---|
| Wrong route: | $\alpha(i) < error_{\mathrm{x}}$ |
| A little behind schedule: | $\gamma \leq error_{\mathrm{y}} < \beta$ |
| Severely behind schedule: | $error_{\mathrm{y}} < \gamma$ |
| Ahead of schedule: | $\delta < error_{\mathrm{y}}$ |

Here, $\alpha(i)$ is determined by the width of road $i$. $\beta$ and $\gamma$ are constants determined by how tight the current schedule is.

Our navigation system tracks the user's location at regular intervals (for example, every 60 seconds) and reports the user's current situation.

#### 3.3.2 Warnings in Undesirable Situations

When the user's situation changes, the navigation system takes an action. If the system warns a user too frequently, it may make the user unpleasant. On the other hand, if a user is likely to miss a reserved train, the system should prompt the user to hurry. Accordingly, we let the system calculate the negative impact of the schedule change due to the user's delay so that the system warns the user only if the impact is large. The negative impact is calculated by the difference of evaluation values between the current schedule and the new schedule computed with the delay. Details of the warning condition are as follows: (i) When the situation changes to "Wrong route," the system warns that the user has entered the wrong way. (ii) When the situation changes from the normal situation to "A little behind schedule" and the negative impact is large, the system warns the user. (iii) If the situation changes to "Severely behind schedule", it notifies the user that it is difficult to follow the original schedule, and it automatically recalculates a schedule and indicates the new schedule.

In order to evaluate the warning facility, we implemented the algorithm described above on Zaurus SL-C700, and measured the value of $error_y$ when a user walks along a route of 1.5 km. In this experiment, we measured the maximum value of $error_y$ for two cases with and without the warning facility. We used $-15$ m as the value of $\gamma$. With the warning facility, the maximum value of $error_y$ was $-20$ m and without the warning facility, the value was $-60$ m.

## 4. Route Search Engine

In this section, we first define the problem, and then explain the algorithm to solve the problem.

### 4.1 Definition of a Solution Space

A map is given as a graph $G = (V, E)$, where $V = \{v_1, ..., v_n\}$. Let $d_i = (rt_i, dur_i, pre_i, fee_i)$ denote the restrictions for node $v_i$, where $rt_i$, $dur_i$, $pre_i$ and $fee_i$ denote restrictions on the arrival time, staying hours, a preference value (i.e., how important this destination is) of the destination $v_i$ and the entrance fee of a facility located at the destination $v_i$, respectively. Let $D$ denote $\{d_1, ..., d_n\}$. Let $p_a \in V$ and $p_g \in V$ denote a starting location and a returning location of the tour, respectively. Function $dist(v_s, v_g)$ returns the distance between any two destinations $v_s$ and $v_g$. If $v_s$ and $v_g$ are registered in the database, the value of $dist(v_s, v_g)$ should be registered in the database in advance. So, the registered value is used in this case. Otherwise, the value is calculated using $A^*$ algorithm on the map data. A candidate route is represented as a list of destinations $S = (v_{i_1}, ..., v_{i_k}), 1 \le i_j \le n, 1 \le j \le k \le n$ where $v_{i_j}$ denotes $j$-th destination in the route.

Each user can specify multiple evaluation functions. Below, we show examples of a total satisfaction degree and a total travel expense as the evaluation functions.

**Satisfaction:** It is natural that the satisfaction degree increases when destinations with higher preference values are included in the route. Of course, the satisfaction degree should increase only when the specified time restriction is satisfied for each destination in the route. For example, a satisfaction degree function $f_1$ can be defined as follows.

$$f_1(S) = \alpha \sum_{j \in \{i_1, ..., i_k\}} (pre_j \cdot timeok(rt_j, dur_j)) - \beta \sum_{j=1}^{k-1} dist(v_{i_j}, v_{i_{j+1}}) \qquad (1)$$

where function $timeok(rt_j, dur_j)$ returns 1 only if both restrictions $rt_j$ and $dur_j$ hold. Otherwise, $timeok(rt_j, dur_j)$ returns 0.

$\beta$ is a positive constant to make the evaluation value worse as the total traveling distance becomes long.

**Expense:** Function $f_2$ for the total travel expense (sum of entrance fees, in this case) which the user has to pay is calculated as follows.

$$f_2(S) = \sum_{j \in \{i_1, ..., i_k\}} fee_j \qquad (2)$$

### 4.2 Algorithm

The route search algorithm is as follows.

Let $N$ and $I$ denote the population size (i.e., the number of candidate solutions) and the number of generations (i.e., the number of iterations), respectively.

**Algorithm**

1 **Generation of the initial population:** $N$ individuals are randomly generated. Each individual represents a candidate solution. Let $P$ be a set of individuals in the population.
2 **Evaluation of all individuals:** We initialize variables $n$ and $P'$ with 1 and $P$, respectively.
3 The fitness value is calculated for each individual. Select a candidate solution $p \in P'$ such that any other solutions in $P'$ superior to $p$ in one criterion are always inferior in at least one criterion. Assign the value of $n$ for the pareto rank [8] of $p$.
4 Probe domination between all individuals and the others. Let $pr(q)$ denote a function for returning the pareto rank of an individual $q$. Remove every individual $q$ such that $pr(q) = n$ from $P'$.
5 Set pareto rank to all individual in $P'$. $n := n + 1$. If $P' \neq \emptyset$ goto 2.
6 **Diversity calculation:** Calculate $nn(q)$ rep-

resenting how much an individual $q$ contributes to overall diversity of the population. For a given individual $q$, calculate the minimum distance from $\{r \in P | pr(r) \leq pr(q)\}$. This value is called the distance to the nearest neighbor.

7 **Selection:** Select candidate for parents. Let $S'$ and $i$ be $\emptyset$ and 1, respectively. If $|S' \cup \{r \in P | pr(r) = i\}| < N/2$, let $S'$ and $i$ be $S' \cup \{r \in P | pr(r) = i\}$ and $i+1$, respectively, and go to 7

8 While $|S'| < N/2$, select an individual with the highest value of $nn(r)$ from $(\{r \in P | pr(r) = i\} - S')$ and add it to $S'$.

9 **Crossover:** Two individuals are randomly selected from $S'$, and copied to a buffer. One point is chosen for each individual to separate it in two parts. The parts after the points in those two individuals are exchanged to produce two new individuals. If the same destination appears more than once in each new individual, the redundant ones are removed so that only one remains in the individual.

10 **Mutation:** Solutions in part of newly generated individuals are changed at random as follows. Several destinations are added, removed or exchanged in the individuals, with certain probabilities. These new individuals are added to $S''$. The original two individuals remain in $S'$. If $|S''| < N/2$, goto step 9.

11 Repeat GA generations. Let $P$ be $S' \cup S''$. One GA generation from step 2 to 11 are repeated $I$ times.

12 **Local Search:** Improve solutions. Create a new individual $q$ from $p \in P'$. If $q$ is superior to $p$, add $q$ to $P'$. Calculate $pr(q)(q \in P)$ similarly to step 3.

13 $P' := \{q | pr(q) = 1, q \in P\}$. Return $P'$ as the return value of this algorithm.

## 5. Experiments and Evaluation

In order to evaluate the proposed extensions to P-Tour, we have implemented the algorithm described in Section 4.2 and measured time to find near pareto-optimal solutions.

In this experiment, we specified Nara Station of Kintetsu railway company as the starting and returning location, 9:00 and 20:00 as the starting and returning time of the tour, and 63 destinations in northern Nara as candidate destinations. This should be a typical scenario to travel around Nara. All of these destinations are registered in the database. It is impossible to visit all of these 63 destinations in one day.

We used two evaluation functions: the satisfaction degree $f_1$ defined by Eq. (1), and the total travel expense (i.e., sum of admission fees at the destinations) $f_2$ defined by Eq. (2). $f_1$ should be maximized, and $f_2$ should be minimized.

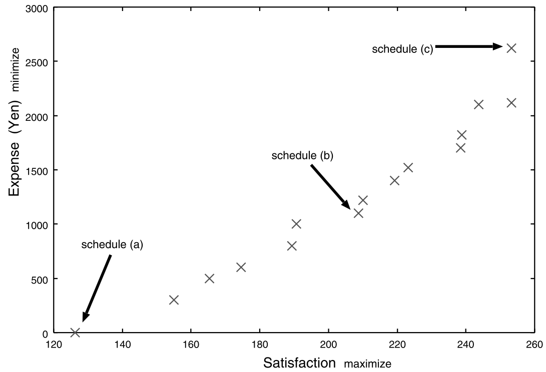


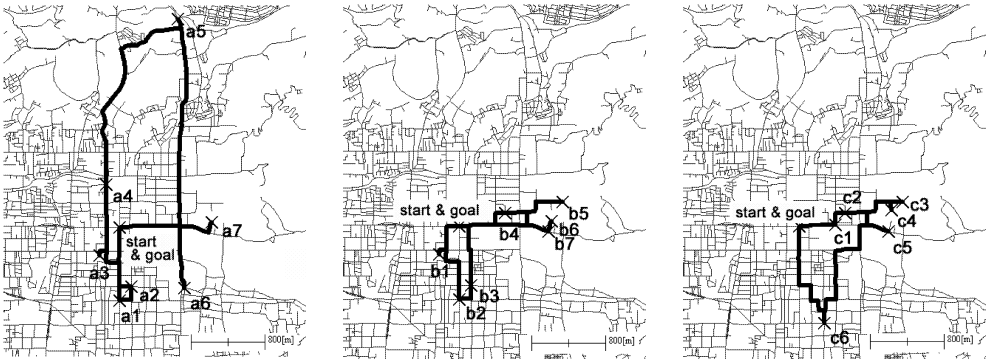

**Fig. 6** Obtained pareto front.

We used a digital map 2,500 published by Geographic Survey Institute of Japan, which includes 29,871 nodes of northern Nara. We used an ordinary PC with Athlon 2500+ and 512 Mbyte Memory on Debian GNU linux. Through the experiments, we used 4 km/h as the user's moving speed. We used parameter values as follows: $\alpha = 1$, $\beta = 0.00125$.

We obtained solutions as shown in **Fig. 6** in 15 seconds. This means that our new search engine has the equivalent performance to our previous one for single-objective problems in Ref. 7)

The schedules (including routes) which were picked up based on the technique in Section 3.1 are shown in **Fig. 7** and **Tables 1**, **2** and **3**. The schedule with the lowest expense and the lowest satisfaction degree for visiting 5 destinations is shown in Fig. 7 (a) and Table 1. The schedule with an average expense and an average satisfaction degree for visiting 7 destinations is shown in Fig. 7 (b) and Table 2. Figure 7 (c) and Table 3 show the schedule with the highest expense and the highest satisfaction degree for visiting 6 destinations. From these results, we see that our system can help users to efficiently compose a good schedule based on multiple criteria, since our system can indicate multiple dissimilar solutions at one calculation.

In order to verify the quality of solutions calculated by our system, we compared solutions derived by our system with the optimal ones calculated by searching the whole solution space. Since 63 destinations are too much to search the whole space, we used the problem sizes between 16 and 19. Taking into consid-




(a) Satisfaction: 126, Expense: 0 Yen
(b) Satisfaction: 210, Expense: 1,220 Yen
(c) Satisfaction: 253, Expense: 2,620 Yen

**Fig. 7** Candidates routes calculated with multiple criteria.

**Table 1** Schedule (a).

| Satisfaction: 126 Expense: 0 Yen | | | |
|---|---|---|---|
| Start | Nara Station | 09:00 | – |
| a1 | Nara Art Museum | 09:11 | 0 Yen |
| a2 | Nara Material Museum | 10:45 | 0 Yen |
| a3 | Hikikawajinja Shrine | 12:27 | 0 Yen |
| a4 | Naramamehikojinja Shrine | 15:16 | 0 Yen |
| a5 | Yuugajinja Shrine | 18:17 | 0 Yen |
| Goal | Nara Station | 19:33 | – |

**Table 2** Schedule (b).

| Satisfaction: 210 Expense: 1,220 Yen | | | |
|---|---|---|---|
| Start | Nara Station | 09:00 | – |
| b1 | Hikikawajinja Shrine | 09:07 | 0 Yen |
| b2 | Nara Art Museum | 10:19 | 0 Yen |
| b3 | Nara Material Museum | 11:53 | 0 Yen |
| b4 | Nara Prefectural Museum | 13:41 | 300 Yen |
| b5 | Toudaiji Temple | 15:52 | 500 Yen |
| b6 | Nara National Museum | 17:05 | 420 Yen |
| b7 | Himurojinja Shrine | 18:36 | 0 Yen |
| Goal | Nara Station | 19:53 | – |

**Table 3** Schedule (c).

| Satisfaction: 253 Expense: 2,620 Yen | | | |
|---|---|---|---|
| Start | Nara Station | 09:00 | – |
| c1 | Kouhukuji Temple | 09:05 | 500 Yen |
| c2 | Nara Prefectural Museum | 10:09 | 300 Yen |
| c3 | Isuien Neiraku Museum | 12:19 | 600 Yen |
| c4 | Toudaiji Temple | 14:22 | 500 Yen |
| c5 | Nara National Museum | 15:35 | 420 Yen |
| c6 | Konjaku Arts Museum | 17:31 | 300 Yen |
| Goal | Nara Station | 19:21 | – |

eration with the practical use of our system, we applied GA operations to evolving solutions for about 15 seconds. On the other hand, the whole search took the time in **Table 4**. **Figure 8** depicts both of approximate solutions by our system and the optimal solutions by the whole search. In this figure, we see that our system can calculate near optimal solutions in wide range. In some points approximate solu-

**Table 4** Time for the whole search.

| # of destinations | calculation time (sec.) |
|---|---|
| 16 | 2,807 |
| 17 | 9,861 |
| 18 | 29,587 |
| 19 | 161,952 |

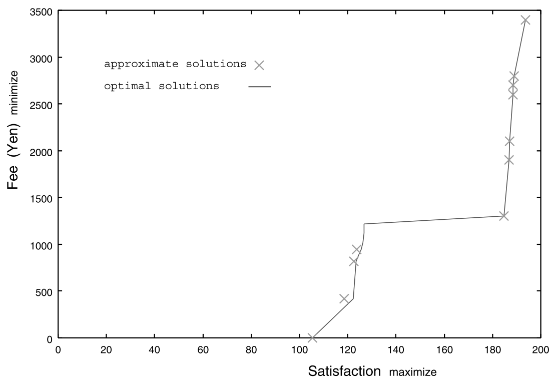



**Fig. 8** Comparison with optimal solutions.

tions do not have the same value as the optimal ones, especially between 500 and 1,000 Yen. This is because we used random values to find the places to apply local search. This problem could be improved in the future.

## 6. Conclusion

In this paper, we extended our personal navigation system P-Tour in two ways. First, we extended our system to treat multi-objective criteria to compose a near best tour schedule intuitively. Secondly, we developed a function for detecting the user's undesirable situation which may result in the schedule re-computation.

We evaluated the proposed method using a digital map of Nara prefecture, Japan, and confirmed that in a typical scenario, it can output various schedules with consideration of multiple criteria in practical time. We believe that our system is helpful for users to compose the best and personalized schedules efficiently.

Currently, we are further extending our system so that it can handle multiple transportations (e.g., car, train, bus, walking, and so on) and schedules over days. Also, we will compare proposed method with other route search methods.

## References


1) Baus, J., Krüger, A. and Wahlster, W.: A Resource Adaptive Mobile Navigation System, *Proc. 7th Int'l. Conf. on Intelligent User Interface* (*IUI'02*), pp.15–22 (2002).
2) Kruger, A., Butz, A., Muller, C., Stahl, C., Wasinger, R., Steinberg, K.-E. and Dirschl, A.: The connected user interface: Realizing a personal situated navigation service, *Proc. 9th Int'l. Conf. on Intelligent User Interface* (*IUI'04*), pp.161–168 (2004).
3) Butz, A., Baus, J., Krüger, A. and Lohse, M.: A Hybrid Indoor Navigation System, *Proc. 6th Int'l. Conf. on Intelligent User Interface* (*IUI'01*), pp.25–32 (2001).
4) Ehlers, M., Jung, S. and Stroemer, K.: DESIGN AND IMPLEMENTATION OF A GIS BASED BICYCLE ROUTING SYSTEM FOR THE WORLD WIDE WEB (WWW), *Int'l. Symp. on Geospatial Theory, Processing and Applications* (*ISPRS2002*) (2002).
5) Malaka, R. and Zipf, A.: DEEP MAP — Challenging IT research in the framework of a tourist information system, *Proc.ENTER 2000*, pp.15–27 (2000).
6) au by KDDI, "EZ Navi Walk". http://www.au.kddi.com/ezweb/au_dakara/ez_naviwalk/
7) Maruyama, A., Shibata, N., Murata, Y., Yasumoto, K. and Ito, M.: P-Tour: A Personal Navigation System for Tourism, *Proc. 11th World Congress on ITS* (2004).
8) Zitzler, E.: Evolutionary Algorithms for Multiobjective Optimization, *Evolutionary Methods for Design, Optimisation, and Control, CIMNE*, pp.19–26 (2002).
9) Zipf, A. and Malaka, R.: Developing Location Based Services (LBS) for tourism — The service providers view, *Proc.ENTER 2001*, pp.83–92 (2001).
10) Kanoh, H. and Nakamura, N.: Route Guidance with Unspecified Staging Posts using Genetic Algorithm for Car Navigation Systems, *Proc. IEEE Int'l. Conf. on Intelligent Transportation Systems* (*ITSC 2000*), pp.119–124 (2000).
11) Bishop, C.M.: Neural Networks for Pattern Recognition, *Oxford University Press* (1995).


(Received May 31, 2005)
(Accepted September 2, 2005)
(Online version of this article can be found in the IPSJ Digital Courier, Vol.1, pp.528–536.)

**Takayuki Shiraishi** graduated from Faculty Office of Computer Science and System Engineering, Okayama Prefectural University in 2003. He received the M.E. degree in Graduate School of Information Science, Nara Institute of Science and Technology in 2005. He currently works on Denso Create. His research interests include car navigation systems.

**Munenobu Nagata** graduated from Department of Regional Development Faculty of Life and Environmental Science, Shimane University in 2004. He is currently a M.E. student in Graduate School of Information Science, Nara Institute of Science and Technology. His research interests include personal navigation systems, intelligent transport systems and genetic algorithms.

**Naoki Shibata** received Ph.D. in Engineering from Osaka University, Japan, in 2001. Currently, he is an associate professor at Shiga University. His main research focus is on distributed systems, design and implementation of multimedia systems, and genetic algorithms.

**Yoshihiro Murata** received the B.E. degree in Department of Computer and Systems Sciences, College of Engineering, Osaka Prefecture University in 1997. He received the M.E. degree in Graduate School of Information Science, Nara Institute of Science and Technology in 1999. He is currently a Ph.D. student in Graduate School of Information Science, Nara Institute of Science and Technology. His research interests include agent technology and genetic algorithms.

**Keiichi Yasumoto** received the B.E., M.E. and Ph.D. degrees in Information and Computer Sciences from Osaka University, Osaka, Japan, in 1991, 1993 and 1996, respectively. He joined the faculty of Shiga University in 1995. Since 2002 he has been an Associate Professor of Graduate School of Information Science at Nara Institute of Science and Technology. His current research interests include design methodology of distributed cooperative systems. He is a member of IEEE/CS.

**Minoru Ito** received the B.E., M.E. and Ph.D. degrees in information and computer sciences from Osaka University in 1977, 1979, and 1983, respectively. From 1979 to 1986, he was a Research Associate, from 1986 to 1989, an Assistant Professor and from 1989 to 1993, an Associate Professor at Osaka University. Since April 1993, he has been a Professor in Graduate School of Information Science, Nara Institute of Science and Technology, Nara, Japan. His current research interests include design theory of relational database, applications of object-oriented databases, and designing DNA probes. He is a member of the IPSJ, the ACM, and IEEE Computer Society.