A Recommender System for Trip Planners

Rathachai Chawuthai(✉), Prodpran Omarak, and Vitchaya Thaiyingsombat

Department of Computer Engineering, Faculty of Engineering,
King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand
{rathachai.ch,58010804,58011148}@kmitl.ac.th

Abstract. Making a trip plan is a key activity for having a satisfying trip for tourists. However, as we survey, there are many users do not need to spend a lot of time to write the plan, and it becomes a pain point for users. The users prefer to have a simple way to create a whole trip plan from a few users' constraints, and the users just customize some items for their satisfaction. Thus, this work introduces a recommender system that mainly employs the genetic algorithm for generating a trip plan. The approach accepts a few roughly input requirements from users, and then it creates a whole trip schedule and allows users to modify. To have a quality trip plan, any places and times in the plan have to correspond to places' categories, open weekdays, times to spend, favorite daytimes and months, and possible routes. A web application for trip planner is developed to demonstrate the suitability and feasibility of the proposed recommender system. After that, the user feedback and usage statistic present the high degree of user satisfaction and opportunity to improve tourism of any city.

Keywords: Genetic algorithm · Recommender system · Travel itinerary · Trip planner

1 Introduction

It is known that travelling is one of the most popular activities for enhancing living standards [1]. In general, before travelling, users search information about the destination via any travel websites such as Google Travel [2], Expedia [3], Kayak [4], TripAdvisor [5], etc. for finding some popular travel attractions to make an own travel itinerary plan. As discussions with users, searching for travel information and making a satisfying plan took a long time especially in locations that they do not know before. As we provide a questionnaire to 473 people whose age above 20 years old, we found that 30.6% of samples do not like to do a trip plan, 26.3% of samples spend 2 days, 25.2% of samples spend 3 days for making a 3-day trip plan, 36.1% of samples use to have no idea when arriving the target place, and 83.4% of samples prefer to have an application to make a trip plan. Based on our discussion and questionnaire, we would like to introduce a persona of Mr. Dan in order to give a simple explanation about a pain point of users to demonstrate our research problem.

Persona: Dan is a senior salesman working in a company in Europe. He has to go to Chonburi province in Thailand for visiting one customer on Friday and the other

© Springer Nature Singapore Pte Ltd. 2020
P. Sitek et al. (Eds.): ACIIDS 2020, CCIS 1178, pp. 493–504, 2020.
https://doi.org/10.1007/978-981-15-3380-8_43

customer on Monday, so he has free days on Saturday and Sunday. He personally plans to attend the Loi Krathong festival at Pattaya floating market in the evening of that Saturday, and he prefer to go to any temples and beaches anytime. Since he is very busy due to his job, he does not have time to find information for making his trip plan on that weekend. He needs a trip plan that contains a few places from his conditions, and the trip plan must be filled by other possible places in a suitable time and routes.

The scenario of this persona is commonly found in real life when users do not have much time to do a trip plan. For this reason, this paper aims to introduce a recommender system that recommends a trip plan from a few constraints from users. The generated plan is also targeted to satisfy suitable places, categories, times, and routes.

In this paper, the data modelling was designed, the recommendation model was proposed by employing the power of the generic algorithm [6], and a trip planner application was implemented in order to demonstrate the suitability and feasibility of our approach. The evaluation against real users showed the high degree of user satisfaction and opportunity to improve city tourisms.

To this end, the overall of our research is explained in Sect. 1, relate work and computational methods are reviewed in Sect. 2, our recommender system and the application are proposed in the Sect. 3, the results are described and discussed in Sect. 4, and conclusion and future work are drawn in Sect. 5.

2 Literature Review

In this section, related work and the artificial intelligent that benefits to our work are studied.

2.1 Related Work

As we studied the state of the art of research about trip planning [7], some key criteria of the trip planner are emphasized, such as mandatory places (must see), multiple days support, possible routing, and opening times. There are many pieces of research aims to do trip planning in various perspectives. Vansteenwegen et al. estimated user preference through a questionnaire in order to create a city trip plan [8]. Yoon et al. used the statistical analysis on social information including user-generated GPS trajectories to recommend a trip plan [9]. Zeng introduced the trajectory mining for trip recommendation [10]. Lu et al. recommended a trip plan based on trip packages and budgets of tourists, and emphasized on user preferences and temporal aspects [1].

In addition, some applications about trip planning are reviewed. We first studied Google Travel [2]. Due to a very big data held by Google, Google Travel analyzed travel data from many users at the tourist destinations to generate a list of things to do and suggested day plans. For the latter feature, the suggested day plans are itineraries that users commonly traveled. The plans are categorized into interesting groups for users to choose, for example, markets, inside the city, old towns, temples, natures, etc. Second, we study Inspirock [11]. This application provides a simple user scenario to create a trip plan. Users are allowed to create a personal plan based on a day-by-day trip, book some hotels and restaurants nearby the tourist place, and manage the plan by editing them in a

timeline and calendar. Third, we reviewed Roadtrippers [12]. This application is created for travel along a roadside. It also recommends some interesting places nearby users in order to let users have a convenient trip plan.

2.2 Artificial Intelligence

A closer look at our challenge indicates that our research problem is similar to a task scheduling problem and a vehicle routing problem. Both problems are commonly solved by the Genetic Algorithm (GA) [13, 14]. GA is a well-known technique of Artificial Intelligence (AI) [6]. It is a search heuristic that finds an optimal answer using the process of natural selection inspired by the theory of natural evolution of Charles Darwin. There are 4 main steps.

1. **Chromosome Encoding** is a step to transform any features of a solution into a sequence of genes in a chromosome.
2. **Initial Population** is to randomly create the population of candidate chromosomes or candidate solutions.
3. **Fitness Function** is a defined function to value the quality of chromosomes.
4. **Genetic Operator** is an activity to create a new solution from existing ones. There are selection, crossover, and mutation. First, selection is to select high qualitied chromosomes determined by the fitness function. Second, crossover is to exchange some parts of two chromosomes to be new solutions. Last, mutation is to randomly change some genes to other ones.

3 Method

To demonstrate how the proposed method works, this section describes an approach to the trip planner application. Contents includes definitions, key user scenario, system design, data modeling, and a proposed technique.

3.1 Definitions

First, this part expresses some definitions used by our recommender system and application. The terms are formally explained as follows:

poi. The point of interest (poi) is expressed by the relation *(name, loc, avg_visit_hrs, fav_daytimes, fav_months, open_weekdays)*, where *name* is the name of that *poi*; *loc* is a location including the latitude and the longitude of the *poi*; *avg_vist_hrs* is an average hours that most tourists visit that *poi*; *fav_daytimes* is a set of favorite periods in a day for tourist including morning, noon, afternoon, evening, night; *fav_months* is a set of favorite months of the *poi*; and *open_weekdays* is a set of week days that tourists can visit that place including sun to sat (Sunday to Saturday). For example, going to a temple in the morning is more attractive than another time, going to a bazaar at nighttime is better, etc.

cat. Due to requirements from users, users can choose a category of tourist attractions such as nature, building, sport, temple, market, etc. The category is a named category of *pois*.

pcat. The *pcat* is the function mapping between a *poi* and a *category*. The function is denoted by $pcat: POIS \times CATS \rightarrow VAL$; where *POIS* is a set of *pois*, *CATS* is a set of named categories, and *VAL* is a set of real number between 0 and 1 for representing the possibility of a *poi* under that category. For example, in case a place *p* has a 70% possibility to belong to a category *c*, it can be expressed by *pcat(p, c) = 0.7*. The relationship between *pois* and *cats* is many-to-many. For example, some beaches can be categorized as natural places 100% and can be markets 50%.

travel_time. The *travel_time* shows the travel time in minutes between 2 pois within a 50-km radius. It is formulated by $travel\_time: POIS \times POIS \rightarrow VAL$, where *VAL* is a set of real number representing minutes between both *pois*.

trip_event. The *trip_event* is a relation (*poi, begin_datetime, end_datetime*) where *poi* is a previously described *poi*, *begin_datetime* is a beginning date time of the event, and *end_datetime* is the end of date time of the event.

trip_plan. The *trip_plan* is a sequence of none-overlapped *trip_event*.

pref_event. The *pref_event* is a preferent event that users give a condition to be an input of the recommender system. It is represented by a relation (*POIS, CATS, begin_datetime, end_datetime*), where *POIS* is a set of *pois*, and *CATS* is a set of categories. Users do not need to give all parameters to *pref_event*, because it is used to be a sketchy event to generate possible *trip_event* for the recommender system. For example, *(any, {nature}, any, any)* means users prefer to go to any natural places at any time in a trip plan.

3.2 Key User Scenario

As the goal of the trip planner, the application allows users to create a complete trip schedule by oneself quickly. Thus, some key requirements are demonstrated by user scenario including the following steps

1. Users enter a start date, duration, and a target city or town.
2. Users add some specific places and specific time periods (input *trip_event*), for example, going to the national museum at 13:00 of the first day.
3. Users give some categories of tourist places with or without specific time periods (input *pref_event*), for instance, going to any market in the morning of the second day.
4. Users get a whole trip schedule (*trip_plan*) generated by the trip planner.
5. Users can ask for regenerating the whole trip schedule; they can fix some trip events and regenerate the other unfix events; and they can ask for regenerating some events directly but not the whole plan.
6. Users can save the final trip plan.

3.3 System Design

To build the trip planner to serve the according scenario, the system is designed and demonstrated through the system architecture diagram as shown in Fig. 1. The system architecture has 5 components that are located in 3 tires: an application tier, a service tier, and a data tier; and all components are described as follows.

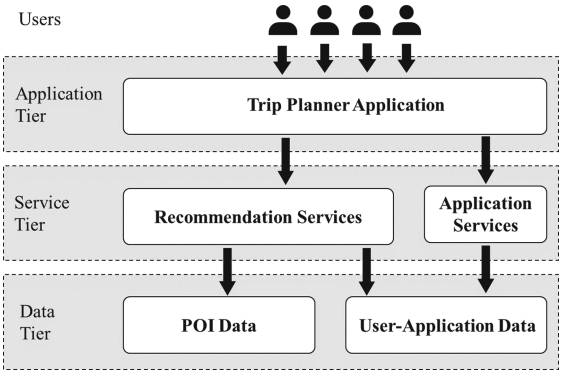

Fig. 1. The system architecture of the trip planner system

Trip Planner Application. A web application that allows users to register, login, logout, specify some requirements, get the whole trip plan, edit the plan, and save. The application has been implemented using HTML and JavaScript that access information from the service tier. Users can input *trip_event* and *pref_event*, and they finally view a *trip_plan* at this application.

Recommendation Services. This service is written using Python for providing core functions of the recommender system. It generates a *trip_plan* from input *trip_events* and *pref_events*. The technical detail of this module is described in hereafter part.

Application Services. This module provides other helpful functions for supporting the trip planner application, for example, register, login, logout, forgot password, edit user profile, etc.

POI Data. The POI Data is a database containing data about trips and places. It covers poi, cat, pcat, travel_time, trip_event and saved trip_plan.

User-Application Data. This module stores other data that used by users and applications, for example, data about user profiles and application configurations.

3.4 Data Modeling

According to the definitions, the database is modeled as shown in Fig. 2. There are 4 main tables for building a trip plan that are POIS, CATS, PCAT, TRAVEL_TIME,

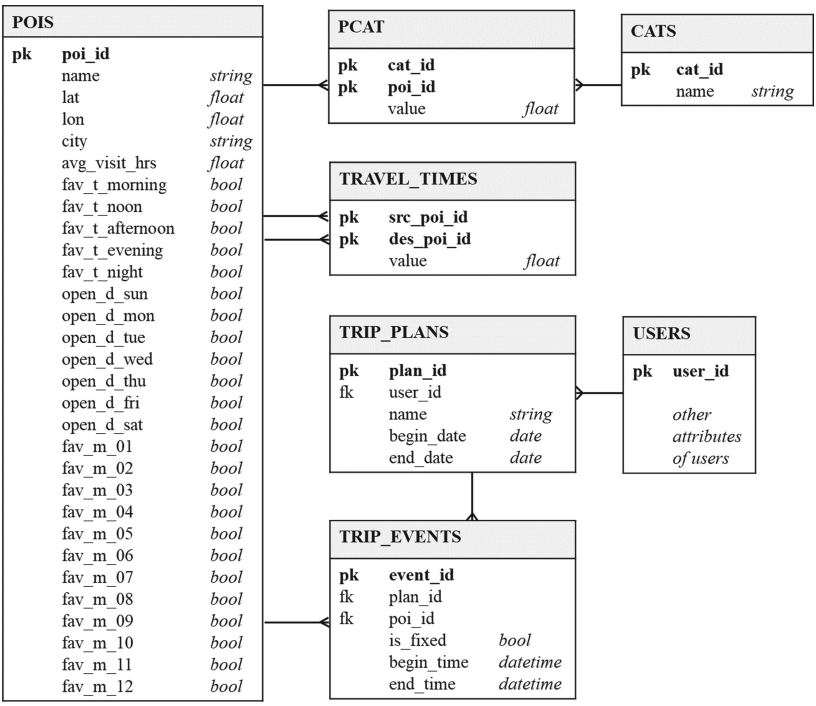

Fig. 2. The data model of the POI database

TRIP_PLANS, and TRIP_EVENTS, and the other tables including USERS to fulfill the application. The table POIS, CATS, PCAT, TRAVEL_TIME, TRIP_PLANS, and TRIP_EVENTS represents the definitions *poi*, *cat*, *pcat*, *travel_time*, *trip_plan*, *trip_event* respectively. In the table POIS, there are many Boolean attributes that correspond to favorite daytimes (*fav_t_* ~), open weekdays (*open_d_* ~), and favorite months (*fav_m_* ~). For the table PCAT, it shows the many-to-many relations between *pois* and categories with a possibility value as described in the definition. In addition, the table TRIP_EVENTS represents the relation between a *poi* and a time period. It also has an attribute "is_fixed" in case the user is satisfied that *trip_event* and informs the recommender system not to change the *poi* and the period after regenerating a new trip plan.

3.5 Trip Planner Technique

To generate a trip plan, the recommender system employs the genetic algorithm to be a key component of our recommendation model. Our recommendation model is demonstrated in a flow as shown in Fig. 3. There are 8 activities in the flow that are described in the following steps.

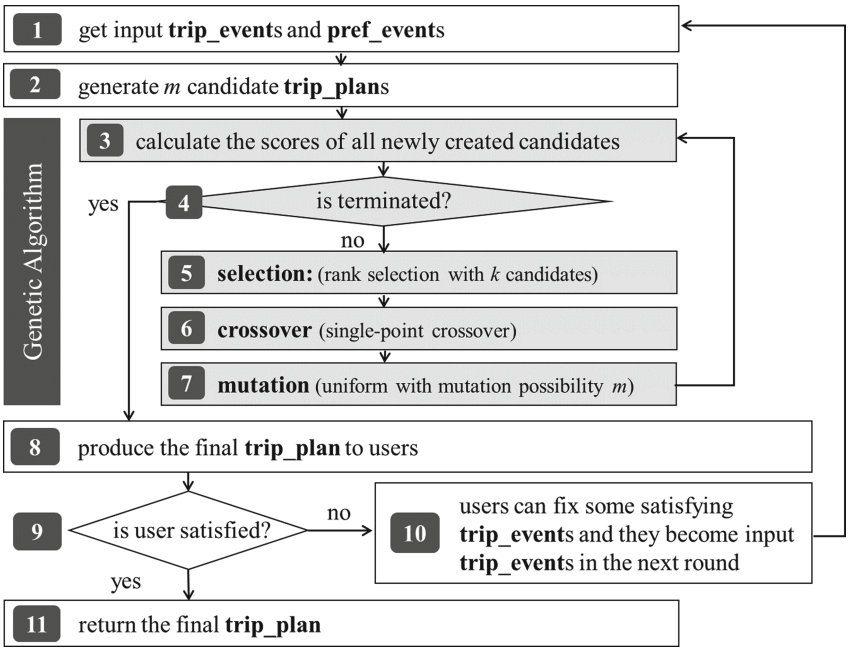

Fig. 3. Trip Planner Recommender System Flow

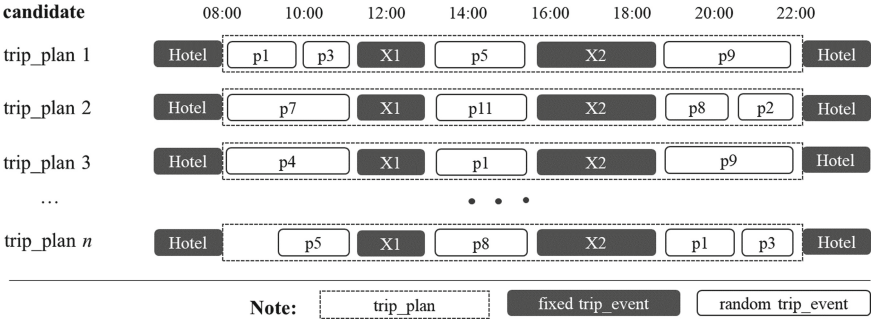

Fig. 4. The idea of generated candidate trip_plans

1. First, the recommender system get data about the user plan including city, start date, and end date. The application also submits *trip_events* and *pref_events* to be key conditions of the plan.
2. Second, the recommender system generates possible *m trip_plans* or candidates as depicted in Fig. 4. It is noted that the *m* is a configured number of candidate *trip_plans*. This work uses *m* being 1,000. The rule is that all *trip_plans* must include all input *trip_events* or fix *trip_events* by assigning the specific *pois* at the specific time periods. After that, the system randomly fills any *pois* at any available spaces. The time period of a random *trip_event* corresponds to the attribute avg_visit_hrs

of that *poi*. There is no *poi* presented more than 1 time in a candidate chromosome. Moreover, any random *pois* must open in the day of the *trip_plans*.

3. Third, the system calculates the score of each candidate trip_plan. The idea is that if many trip_events satisfy cover constraints from users and the condition of *pois*, the score becomes higher. In our experimented system, we define the fitness function to calculate the score $score = 5\ \Sigma\ a_i + b + c + d$.

   a_i is a weight when a random *trip_event* satisfying an input *pref_event*, if the *pref_event* mentions about category; the value of a_i is from the function *pcat*; otherwise, it used as 1. This term is multiplied by 5 because we prefer to place important on an input *trip_event* satisfaction rather than other criteria. However, this number can be configured.
   b is the number of random *trip_events* having a *poi* in a favorite daytime.
   c is the number of random *trip_events* having a *poi* in a favorite month.
   d is the number of random *trip_events* having distances between connected places not more than 30 driving minutes

4. Next, the top-score *trip_plan* is evaluated. If all *trip_events* satisfy all conditions from the user, pois, and time; it becomes a perfect plan. In case the perfect one found; the algorithm is terminated and produces the final *trip_plan* to the 8^{th} step; otherwise, the genetic operations which are selection, crossover, and mutation are performed in the next steps until t times to terminate. The t is configurable in our application. We use the t as 50.
5. The selection operation using the technique of rank selection is firstly done by selecting top k plans having high scores. The k is a configurable parameter in the application, our work uses it k as 100.
6. Next, the crossover operation using the single-point crossover technique is operated. All selected candidates are paired. In each pair, a fix *trip_event* from ones in the middle is randomly selected to be a crossover point, and then exchange the parts of both plans based on that point. In this case, two child *trip_plans* are newly created.
7. After that, the mutation operation using the uniform mutation method is run. The mutation probability m is configured as 0.1 (or 10%). For doing mutation, a random *trip_event* is replaced by another random *trip_event* in the range of *trip_event*s that satisfy the user conditions to be a new *trip_plan*.
8. In this step, the final *trip_plan* is provided to the user.
9. After that, the user can decide to choose the generated *trip_plan*.
10. If the generated *trip_plan* is not satisfying, the user will ask to regenerate the *trip_plan* again. In case the user is satisfied some *trip_events*, he or she can fix those *trip_events* to be input *trip_events* for the next round. After that, the algorithm moves to the first activity. In addition, the users can ask for randomly changing an output *trip_event* directly.
11. At last, when the user is satisfied the generated *trip_plan* from the sixth activity, the final *trip_plan* is saved.

4 Result and Discussion

The recommender system and the trip planner application are implemented to testify the possibility and practicability of our approach. In this section, the application, evaluation, and the discussion are described.

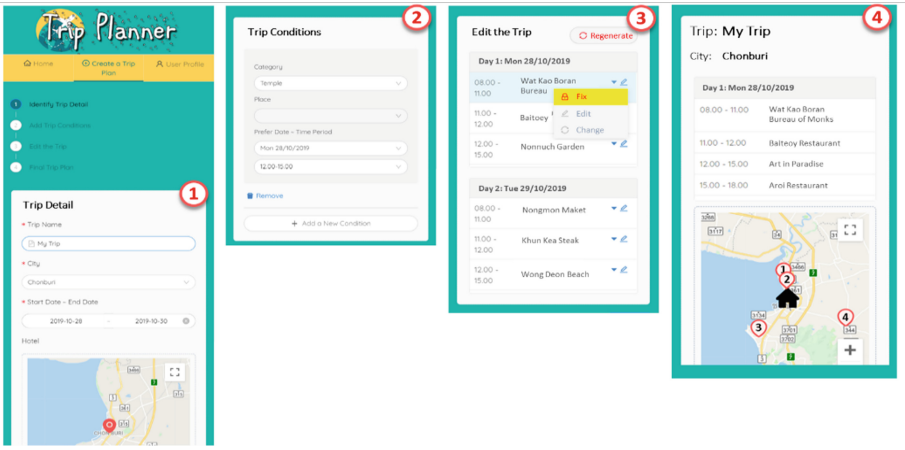

Fig. 5. The captured screens of the trip planner application

4.1 Application

The prototype of the trip planner application is developed based on the design and method in the third section. It is a mobile-friendly web-based application. In the application, there are 3 main menu items: Home, Create a Trip Plan, and User Profile. The key feature is to Create a Trip Plan as shown in Fig. 5. In this feature, there are 4 steps as follows:

1. First, to identify a trip detail as shown in Fig. 5 (1), a user gives information about trip name, city, start date, end date, and pins the location of the hotel.
2. Second, to add trip conditions as depicted in Fig. 5 (2), the user adds input *trip_events* and *pref_events* in this step. The user interface provides input boxes of *pois'* categories, *pois*, date, and time period. In case of adding a *pref_event*, the user does not need to give all inputs, for example, giving only a category and leave the date and time period blank. After submitting this form, the recommender system executes the algorithm and generates a *trip_plan* that is displayed in the next step.
3. Third, the step to edit the trip is demonstrated in Fig. 5 (3). In this step, the user can get the generated *trip_plan* and the plan can be edited. There are 2 ways to edit the plan: (1) to regenerate the whole plan, and (2) to rechange a *trip_event* directly. If some generated *trip_events* are satisfying, the user can mark to fix them in the schedule. In this case, when the user asks to regenerate the whole trip plan, the *trip_plan* is altered while all fixed *trip_events* are unchanged.

4. The last step, to display the final *trip_plan* as displayed in Fig. 5 (4), the user can access the full trip plan including trip schedule of each date together with a map having pins of the hotel and the running number of all *poi*.

4.2 Evaluation

The application provides a feature to recommend a trip plan based on user criteria. It also allows users to fix some events and regenerate the whole trip. In general, any recommendation models are directly evaluated against supervised data, such as user-item datasets. However, in this work, the supervised data about trip plans of any users are hardly gathered. Thus, the evaluation in this work has to be mainly focusing on user satisfaction. In this case, we delivered the application to real users and collected feedback from them. There are 2 experiments: (1) one is to get feedback from users directly, and (2) the other one is to check the behavior when they use the application.

In the first experiment, the evaluation was done by using our application and filling a survey form. Volunteers were firstly screened by choosing ones who did not like to waste time writing travel plans. This is the target group for our application as mentioned in the introduction. Therefore, there were 40 potential users who were picked to evaluate our approach. The form asks about user satisfaction from 0 to 5 where the higher score means higher satisfaction. As the feedback was summarized, the average score all users are 4.25 of 5.00. It is resulted in the high ranking of user satisfaction.

In the second experiment, we analyzed the behavior from the usage of the application. According to the application, users can save the final trip plan. In this case, we assume that users ask for regenerating the plan until they satisfy and save the final trip plan. Thus, we evaluate this behavior by counting the number of times the button "Regenerate" or "Change" is clicked until the final plan is saved. The experiment was conducted with the same group of users. In this experiment, we analyzed only 31 of 40 users who gave 4–5 mark in the previous survey. It is found that the average times of regenerating the whole trip per one user is 0.714, and the average times of regenerating some part of the trip plan is 1.142. It can be interpreted that users regenerated the trip plan not more than 3 times until they were satisfied and saved the plan.

4.3 Discussion

Our recommender system and application provide a key feature to produce a trip plan based on a few user criteria. Some criteria are clearly defined (input *trip_event*), and some are flexible (input *pref_event*). This feature is suitable for users who do not have time to make a trip plan as mentioned by the persona in the introduction section. Users sometime need a complete trip schedule, but they do not need a really perfect trip plan, because they prefer to spend the any free time for traveling activities. Thus, this application can support their requirement. As we mentioned about the result of our survey in the introduction section, there are 30.6% of users who do not prefer to do a trip plan. It is because they do not need to spend a lot of time to do the plan, and 83.4% of samples need recommended trip plans that allow them to customize easily.

As the process of our recommendation model, the trip plan is corresponding to the features of attractions such as open weekdays, times to spend, favorite times and months, and feasible routes. As we reviewed, most related work [1, 8–10] placed important to the technique of trip pattern recognition and trajectory mining. They did not much mention about the aspects of *pois*. In addition, As comparing to related applications, Google Travel [2], Inspirelock [11], and Roadtrippers [12] provide information and ready recommend plans, but they do not have a flexible way to customize the trip based on user constraints and features of tourist places. Since the mentioned constrains are the key features of our trip planner application, the consensus view seems to be that it can demonstrate the novelty of our recommendation system.

Since our recommender system produces a trip plan from a few flexible user criteria, our work is directly evaluated from user feedback directly. According to the result of evaluation in the previous part, users are highly satisfied this trip planner application. As they occasionally regenerated the plan, it shows that users are pleased with the result generated from our recommender system.

Overall, this study focuses on the problem representation of trip planner using the data model and the fitness function, and the model through the computing process together with the system architecture and the user flow. The data model mentions about the schema and the description of pois, categories, trip plans, and trip events. Some aspects such as multiple-possible categories, recommended visit times, and times spent visiting are introduced in this data model to fulfill requirements from users. Besides, the computing process takes advantage of the genetic algorithm to find an appropriate solution. The formation of a chromosome and a fitness function are also proposed to respect the conditions from the tourism domain and the traveler domain. The suitability and feasibility of this work are demonstrated by the prototype.

In addition, some discussions with smart city experts found that our approach are benefits to tourism development in each city. Since the favorite time periods and months are mentioned in the model, they can be adapted to be some special events and festivals. The model should take advantage of social media in order to improve the precision of open time, close time, suggested visit time, recommended time spent visiting, and festivals as well.

5 Conclusion and Future Work

This paper introduced a recommender system for trip planners. In our research methodology, we firstly surveyed the needs of users, and found that users prefer to have an application to recommend a trip plan from a few constrains and they can modify the generated plan according to their satisfaction. We formulate the research requirements into a fitness function to measure the quality of any generated trip plan. This function gives a high score if a plan and tourist attractions in the plan correspond to specific place, places' categories, open weekdays, times to visit, favorite daytimes, favorite months, and feasible routes. These features are designed in the proposed data model. To execute the recommender system, users just input a few criteria such as some places, places' categories, and some specific time periods; and then the recommender system employed the genetic algorithm to produce the final trip plan having the highest score. To demonstrate

the suitability and feasibility of the model, a trip planner prototype, which is a mobile-friendly web application, has been developed. Lastly, our work is evaluated using the prototype. It has been found that users appreciate the application and they are satisfied most of initially generated plans, because they spend a few times to regenerate the plans until they are satisfied. It is also discussed that our work can be enhanced to improve tourism development of any cities.

In order to increase the capability of our work, the user interface and user experience should be considered. In addition, social media should be studied and utilized to improve the precision of tourist attractions' aspects and recommended trip plans.

References

1. Lu, E.H., Fang, S.H., Tseng, V.S.: Integrating tourist packages and tourist attractions for personalized trip planning based on travel constraints. GeoInformatica **20**(4), 741–763 (2016)
2. Google Travel. https://www.google.com/travel. Accessed 10 Oct 2019
3. Expedia. https://www.expedia.com. Accessed 10 Oct 2019
4. Kayak. https://www.kayak.com. Accessed 10 Oct 2019
5. TripAdvisor Homepage, https://www.tripadvisor.com. Accessed 10 Oct 2019
6. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-662-05094-1
7. Souffriau, W., Vansteenwegen, P.: Tourist trip planning functionalities: state–of–the–art and future. In: Daniel, F., Facca, F.M. (eds.) ICWE 2010. LNCS, vol. 6385, pp. 474–485. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16985-4_46
8. Vansteenwegen, P., Souffriau, W., Berghe, G.V., Oudheusden, D.V.: The city trip planner: an expert system for tourists. Expert Syst. Appl. **38**(6), 6540–6546 (2011)
9. Yoon, H., Zheng, Y., Xie, X., Woo, W.: Social itinerary recommendation from user-generated digital trails. J. Pers. Ubiquit. Comput. **16**(5), 469–484 (2012)
10. Zheng, Y.: Trajectory data mining: an overview. ACM Trans. Intell. Syst. Technol. (TIST) **6**(3), 1–41 (2015)
11. Inspirock. https://www.inspirock.com. Accessed 10 Oct 2019
12. Roadtrippers. https://roadtrippers.com. Accessed 10 Oct 2019
13. Omara, F.A., Arafa, M.M.: Genetic algorithms for task scheduling problem. In: Abraham, A., Hassanien, A.E., Siarry, P., Engelbrecht, A. (eds.) Foundations of Computational Intelligence, vol. 3, pp. 479–507. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01085-9_16
14. Mohammed, M.A., Ghani, M.K.A., Hamed, R.I., et al.: Solving vehicle routing problem by using improved genetic algorithm for optimal solution. J. Comput. Sci. **21**, 255–262 (2017)