CYTRIP: 크라우드 소싱을 이용한 POI 추천 기반의 여행 플래닝 시스템

[a]프리스카, [b]오경진, [b]홍명덕, [c]조근식
인하대학교 컴퓨터정보공학과
e-mail : [a]aprilia.19priska@gmail.com, [b]{okjkillo, hmdgo}@eslab.inha.ac.kr, [c]gsjo@inha.ac.kr

CYTRIP: A Multi-day Trip Planning System based on Crowdsourced POIs Recommendation

Priska Aprilia*, Kyeong-Jin Oh*, Myung-Duk Hong*, Geun-Sik Jo**
*Dept. of Computer Science & Information Engineering, Inha University
**School of Computer Science & Information Engineering, Inha University

Abstract

Multi-day trip itinerary planning is complex and time consuming task, from selecting a list of worth visiting POIs to arranging them into an itinerary with various constraints and requirements. In this paper, we present CYTRIP, a multi-day trip itinerary planning system that engages human computation (i.e. crowd recommendation) to collaboratively recommend POIs by providing a shared workspace. CYTRIP takes input the collective intelligence of crowd (i.e. recommended POIs) to build a multi-day trip itinerary taking into account user's preferences, various time constraints and locations. Furthermore, we explain how we engage crowd in our system. The planning problem and domain are formulated as AI planning using PDDL3. The preliminary empirical experiments show that our domain formulation is applicable to both single-day and multi-day trip planning.

1. Introduction

Planning a multi-day trip itinerary before travelling to a city is one of the most important yet a time consuming task. In order to plan a multi-day trip, one needs to find and select a set of worth visiting point of interests (POIs) and arrange the POIs into an itinerary. However, despite the variety of options regarding tourist destinations or attractions available on internet, tourists frequently are not capable to cope with such huge volume of choices. To reduce the options, some people might ask their circle of acquaintances to suggest some interesting places of a city to visit.

In order to find the popular POIs for travel route recommendations, several works [4, 12, 13] have explored the wisdom of the crowd through overwhelming images shared by various people on online media sharing sites e.g. Flickr. Furthermore, travel route search problems are commonly formulated as The Tourist Design Problems [3] such as Orienteering Problem with Time Windows [4, 7, 11, 13] and Traveling Salesman Problem [14]. To solve the problems, various search algorithms with heuristic [3] are employed, namely iterated local search [11], taboo search [7], and genetic algorithm [14].

On the other hand, Recommendation Systems (RSs) with various techniques [6] have been extensively engaged to reduce the huge amount of tourism information on internet and offer tourist destinations. The RSs prediction models are typically built using large data set. However, for which the complete model is not known, especially those that require input from humans, human computation has emerged as a powerful and inexpensive approach [5, 8].

In this paper, we introduce CYTRIP, a multi-day trip planning system that engages human computation (i.e. crowd) to collaboratively recommend POIs through a shared workspace. As stated in [10], there are various motivations (e.g. free will, appeal to knowledge) of people in participating online activities, e.g. recommending POIs to friends and acquaintances. As it is becoming easier to share recommendations and opinions online with people social graphs online, these are increasingly enter personal friend space. Seeing a friend recommend something, either directly or through their actions, may affect ones' desire to try it themselves.

In CYTRIP, instead of asking user to specify the general category, for example, *Natural Sites*, we go beyond into the subcategories. Our assumption is clear, for example, a person might not like to visit a mountain but visiting a beach is in high preference. Thus, in our system, a user can specify what they like in general using subcategories and what kind of places they would like to visit for the current visit. This approach can reduce crowd's work in recommending POIs to the user because *why would they bother to recommend something that a person does not specify?*

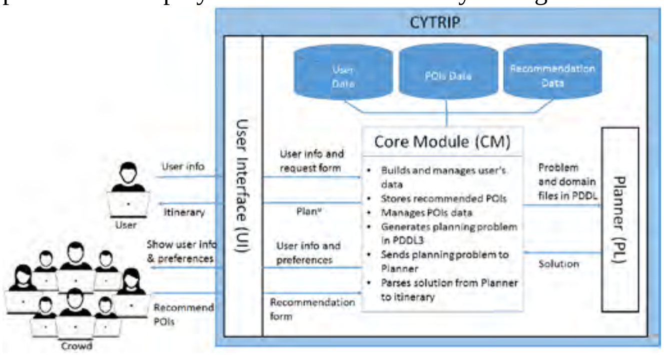
In solving trip planning problem, CYTRIP is similar to [9], that is, formulating the problem as AI planning problem in PDDL and using existing planner to solve it. However, the work in [9] is only applicable for single-day trip, but our planning problem is more complex, that is, a multi-day trip planning with various time constraints and various start and end locations. At the time of writing this paper and to the best of our knowledge, there is no existing published work that formulates multi-day trip planning problem in PDDL3 [1]. Furthermore, our domain formulation can be applied to both single-day and multi-day trip planning.

The rest of this paper is organized as follows. Section 2 discusses the architecture and overview of CYTRIP, crowd recommendation and itinerary generation. The preliminary empirical experimental results described in section 3. Finally, we finish with conclusions and further works in section 4.

2. CYTRIP : Architecture and Overview

We propose CYTRIP, a multi-day trip itinerary planning system based on crowdsourced POIs recommendation. The goal of CYTRIP is to automatically generate a multi-day trip itinerary satisfying user's various time constraints taking into account user's preferences, POIs' time windows and the distance among POIs. In our work, we focus on planning the trip in Seoul, the capital city of South Korea.

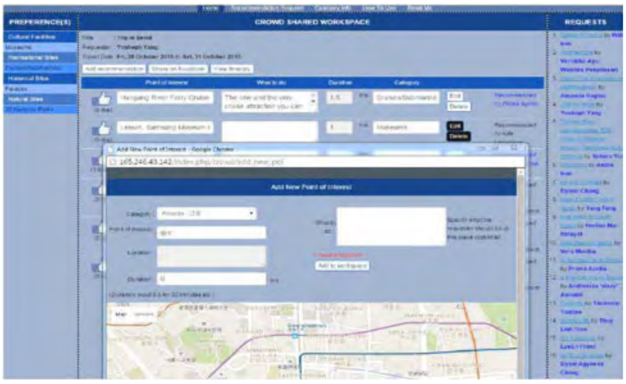
Figure 1 depicts the system architecture of CYTRIP. CYTRIP is composed of *User Interface (UI), Core Module (CM)* and *Planner (PL)*. User, crowd, and system communicate and interact with each other through *UI*. Shared workspace is a part of *UI* for the crowd to add recommendation and vote for the recommended POIs. CM builds and manages User Repository and POIs Repository. It also generates planning problem and sends it to PL. PL solves the problem sent by CM and sends the solution to CM. CM then parses the solution given by AP into a plan. The plan is then displayed to user as an itinerary through *UI*.



(Figure 1) CYTRIP System Architecture

A. Crowdsourced POIs Recommendation

In our term, crowd is anyone who is willing to contribute by recommending POIs. Once a user requests for a recommendation, CM opens and reveals the task. Everyone can see the task, either registered users on the system or outside of the system. The requesting user can also share the task on social media (e.g. Facebook).



(Figure 2) Crowd workspace is a shared workspace for crowd to collaboratively recommend POIs to user.

Figure 2 depicts the shared workspace. The crowd can open the task and will be directed to the shared workspace to see the current recommendation. The leftmost side shows the list of the preferences (categories and subcategories) generated by system based on user's input. In the *Add new poi* workspace (the popup window), crowd chooses one of the subcategories specified by user and system populates all the POIs that belong to that subcategory. The crowd can either select one of them or input a new one. The location and POI selection in our system is integrated with Google Map for the ease of use to users. The visit duration of a POI is determined by crowd as they are assumed to have experience in visiting the POIs they recommend. Once the *Add to workspace* button is clicked, the new recommended POI is saved and shown on the workspace and can be seen by others. If the contributing crowd is connected to social media and agrees to post status on her social media, system will automatically post a status on her social media with something like

"I just recommended Namsan Tower to Priska Aprilia for her visit to Seoul. Please help her by recommending the places you know".

The automatic status update on social media by system is used as a way to attract more people to contribute to the recommendation task.

In our proposed system, another individual of crowd can vote up a recommended POI. The "*like*" button of *Facebook* inspires this "*vote up*" idea. We introduce this "*vote up*" feature due to no duplication allowed in recommending POIs. And it is used to determine how popular the POI among the crowd is. This information is used by system in calculating the POI priority in next section.

User can specify general preference (GP) and current preference (CP) by selecting one or more categories of POIs. The categories are based on the categories provided by Korean Tourism Organization (KTO). In our preference selection, we have categories and subcategories. For example, *Natural Sites* category has *Beaches*, *Caves*, *Waterfalls*, *Botanical Gardens*, etc. as the subcategories. The general assumption oPrf having CP is user might have different categories of preference for each visit which might also be different from GP. For example, a user does not specify *Museums* as her GP but for the current visit, she travels with her friend and her friend likes visiting a museum. In this case, the user can specify *Museums* as CP. The level of preference over a subcategory is defined using score ranging from 10 to 100. The higher the score given by a user to a subcategory, the more the user prefers it. Note that, 100 score given to a POI does not mean it is a must visit POI. It indicates how much a user prefers a subcategory to the others.

In order for system to generate multi-day trip itinerary, user is required to enter her timetables and specific start and end location. If the timetables and start and end locations are same for each day, user just needs to input one otherwise user can adjust it. Note that start and end locations are not POIs but they are locations from and to where the user would like to start and end the journey, e.g. hotels. The accommodation recommendation is out of this paper discussion.

B. Itinerary Generation

After getting a list of crowd recommended POIs, the further challenge is how to transform those POIs into a multi-day trip itinerary. In our work, we formulate the multi-day trip planning problem as Artificial Intelligence planning problem using PDDL3 [1] taking as input the recommended POIs and user's requirements which makes our planning problem formulation dynamic. The planning problem consists of five parts:

1. Objects are composed of three type of objects which are *day*, *Location*, and *poi*.
2. Initial state is a list of all the ground atoms that are true initially. The ground atoms in the initial state are described by the means of functions and predicates. The predicates used in our planning problem are as follows
   - (`located_at p l`) defines poi `p` is located at location `l`
   - (`open_on l d`) indicates location `l` is open on day `d`
   - (`person_at_on l d`) indicates a person is at location `l` on day `d`
   - (`current-day d`) defines `d` as the initial day of the trip
   - (`day-next d1 d2`) indicates day `d1` is the day after `d2`. Note that, if the trip is single-day, `day-next` will be not specified.
   - (`at-now l`) indicates `l` as the current location. The value of `at-now` is changed upon the execution of `move` action (see domain formulation)
   - (`at-next l1 l2`) defines the location transition between the end location `l1` of current day and the start location `l2` of next day.

   Meanwhile, the functions used in our planning problems are as follows
   - (`visit_duration p`) and (`priority p`) define visit duration and priority of POI `p`, respectively.
   - (`move_duration x y`) defines the moving duration from location `x` to location `y`.
   - (`opening_hour l`), (`closing_hour l`), and (`last_admission l`) indicate the opening hour, closing hour, and last admission of location `l`, respectively.
   - (`available_time d`), (`current_time d`), (`end_time d`), (`total_priority d`) define the available time, current time, end time, and total priority of day `d`, respectively.
3. Goal defines all the goals to be achieved. In our planning problem, the goals are the execution of visiting a list of POIs. The goal in our planning problem is specified using predicate (`visited p`) where p is a `poi`.
4. Plan constraints can be hard and soft constraints [1]. Hard constraints must be satisfied in any valid plan. In our planning problem, we define following hard constraints
   - Each location can be visited at most once
     (`forall` (`?d - day ?l – location`) (`at-most-once` (`at ?d ?l`)))
   - At the end of day `d`, we want a plan to return to user's specified location `l`
     (`at end` (`person_at_on l d`))
5. Planning metric can be maximizing or minimizing a function or a set of functions. For our planning problem, we would like maximizing the objective function `total_priority` of each day in order to satisfy user preference.

CM calculates the priority value for each recommended POI with the following equation:

$$\Pr_{poi} = \begin{cases} Score_{poi} + N_{vote} * \gamma & \text{if poi in specific preference } SP_U \\ Score_{poi} * \alpha + N_{vote} * \gamma & \text{if poi in specific preference } GP_U \end{cases} \quad (1)$$

For example, POI "*Insadong*" specified in CP is given score 90, and it gains one vote, therefore its priority $Pr_{Insadong}$ = 90 + 1 * 10 = 100 (γ = 10). If a POI is defined in GP, its score is multiplied by α (=0.5) in order to make the POIs in the GP have lower priority than the ones in the CP. Note that, if a category is specified in both GP and CP, system considers the score given in CP.

To achieve specified goals in the problem, we specify three actions in our domain, namely `move`, *`visit`*, and `change-day`. Note that, our domain definition can be used to solve both single and multi-day trip planning. Figure 3 depicts action *move* formulated in PDDL3 as a durative action.

```
Action 1: Move from x to y on day d
(:durative-action move
    :parameters (?x - location ?y - location ?p - poi ?d - day)
    :duration (= ?duration (move_duration ?x ?y))
    :condition (and (at start (person_at_on ?x ?d))
        (at start (> (available_time ?d) (move_duration ?x ?y)))
        (at start (current-day ?d))
        (at start (at-now ?x))
        (at start (open_on ?y ?d)))
    :effect (and (at end (person_at_on ?y ?d))
        (at start (at-now ?y))
        (at start (not (at-now ?x)))
        (at start (not (person_at_on ?x ?d)))
        (at start (increase (current_time ?d) (move_duration ?x ?y)))
        (at start (decrease (available_time ?d) (move_duration ?x ?y)))
        (at start (increase (total_moving_time ?d) (move_duration ?x ?y)))))
```

(Figure 3) Action *move* formulated in PDDL3.

The parameters input for action `move` are the initial location `?x`, destination location `?y`, and day `?d`. The conditions for this action are (1) A person is at initial place `?x` on day `?d` (2) Available time on day ?d is sufficient to move from `?x` to `?y` (3) the current day is `?d` (4) location `?y` is open on day `?d`. The effects of this action assert that a person is at `?y`, available time, current time and total moving time of day `?d` are modified by moving time.

```
Action 2: Visit poi p located at l on day d
(:durative-action visit
    :parameters (?p - poi ?l - location ?d - day)
    :duration (= ?duration (visit_duration ?p))
    :condition (and (over all (located_at ?p ?l))
        (over all (person_at_on ?l ?d ))
        (at start (not (visited ?p)))
        (at start (>= (current_time ?d) (opening_hour ?l)))
        (at start (> (last_admission ?l) (current_time ?d)))
        (at start (>= (closing_hour ?l) (+ (current_time ?d) (visit_duration ?p))))
        (at start (> (available_time ?d) (visit_duration ?p))))
    :effect (and (at end (visited ?p))
        (at start (increase (current_time ?d) (visit_duration ?p)))
        (at start (decrease (available_time ?d) (visit_duration ?p)))
        (at start (increase (total_visit ?d) 1))))
```

(Figure 4) Action *visit* formulated in PDDL3.

After moving, a `visit` action is needed to be taken. The action `visit` is shown in Figure 4. The action `visit` takes three parameters as input poi `?a` and its corresponding location `?p`, and also day `?d`. The conditions for the *`visit`* action to be applicable are (1) poi `?p` is located at location `?l` (2) a person is already at location `?l` (3) the poi `?p` is not yet visited (4) current time of day `?d` has to be greater or equal to the opening time of location `?l` (5) the current time of day `?d` has to be smaller than the last admission time of location `?l` (6) the activity will be finished before closing hour of location `?l` (7) available time of day `?d` is greater than the visit duration of poi `?p`. The effects of this action assert that (1) the poi `?p` is visited (2) available and current time of day ?d are modified according to visit duration of poi `?p` and (3) the total visit of day `?d` is increased by 1.

For multi-day planning, we need to formulate an additional action, namely `change-day` (in Figure 5). This action is a *non-durative* action. Let us assume, user wants system to build a two-day trip itinerary. The user specifies that the starting location and ending location of day 1 are *A* and *B*, respectively. Meanwhile the starting location and ending location of day 2 are *B* and *C*, respectively. Therefore, in order to start a new day trip for day 2, a person needs to return to *B* first (see constraints in problem definition) which indicates the trip for day 1 has ended (line 3-4). Line 5 indicates that there is another day `?d2` such day `?d2` is the next day of `?d1` and there is location `?l2` such that location `?l2` is the next location of `?l1`. The value of predicates `next-day` and `at-next` are grounded in the initial state meanwhile the value of other predicates in this action are modified throughout the states expansions through the actions. The `change-day` action will not be executed if the planning problem is a single-day.

Action 3: Change-day from day d1 to day d2

```
1 (:action change-day
2       :parameters (?l1 ?l2 - location ?d1 ?d2 - day)
3       :precondition (and (current-day ?d1) (at-now ?l1)
4             (person_at_on ?l1 ?d1)
5             (next-day ?d1 ?d2) (at-next ?l1 ?l2))
6       :effect (and
7             (at-now ?l2) (current-day ?d2))
              (not (at-now ?l1)) (not (current-day ?d1)))
```

(Figure 5) Action *change-day* formulated in PDDL3.

3. Preliminary Empirical Experimental Result

We used the dataset provided by Korean Government. TourAPI[1] was used to get the POIs of Seoul. SGPLan [2] version 522 was used to solve our planning problem. The planner participated in The Fifth International Planning Competition (IPC-5)[2] in 2006. In our experiments, CM calculates the moving duration using Google Map service. Different experiments were done to evaluate the feasibility of our approach and domain formulation. The first one was the experiments dealing with the single-day trip planning, meanwhile the second one was dealing with multi-day trip planning (i.e. two-day and three-day) with various time constraints and various start and end locations. The experiments used the same domain formulation, the difference laid on the problem formulations.

4. Conclusion and Further Work

Planning a multi-day trip itinerary is often a difficult and complex problem, starting from selecting a list of POIs and arranging them into an itinerary with various constraints. To reduce the complexity and the state space of a planning algorithm to find a solution to a multi-day trip planning problem, human computation can be engaged to contribute to the POIs generation process. In this paper, we present CYTRIP, a multi-day trip itinerary planning system which engages human (i.e. crowd) in recommending POIs. CYTRIP provides a shared workspace to draw collective intelligence of crowd by enabling the crowd to collaboratively contribute to the task of POIs recommendation. Later, the recommendations from crowd becomes AI planning problem. Our preliminary empirical experimental results show that our domain formulation in PDDL3 can be applied to both single and multi-day trip planning by defining more actions and predicates. However, since we are still developing our system user interface, we leave experimenting with the involvement of the crowd as our further work for further validation of our hypotheses.

[1] Available on http://api.visitkorea.or.kr/main.do accessed on April 10th, 2015

[2] http://ipc.informatik.uni-freiburg.de/Planners accessed on April 27th, 2015

Acknowledgement

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No. R0101-15-0054, WiseKB: Big data based self-evolving knowledge base and reasoning platform)

References

[1] A. Gerevini, D. Long, “Plan constraints and preferences in PDDL3,” in Technical Report, Dept. Electronics for Automation, University of Brescia, Italy, 2005.

[2] C.W. Hsu, B. Wah, R. Huang, Y. Chen, “New Features in SGPlan for Handling Preferences and Constraints in PDDL3.0,” in Proc. 5th Int. Planning Competition, 2016.

[3] D. Gavalas, C. Konstantopoulus, K. Mastakas, G. Pantziou, “A survey on algorithmic approaches for solving tourist trip design problems,” in *J Heuristics*, vol. 20, issue 3, pp. 291-328, June 2014.

[4] G. Chen, S. Wu, J. Zhou, A.K.H. Tung, “Automatic Itinerary Planning for Travelling Services,” in *IEEE TKDE*, vol. 26, no. 3, March 2014

[5] H. Zhang, E. Law, R. Miller, K. Gajos, D.C. Parkes, E.Horvitz, “Human computation tasks with global constraints,” in CHI’12, 2012.

[6] J. Borràs, A. Moreno, A. Valls, “Intelligent tourism recommender systems: A survey,” in *ESWA*, vol. 41, issue 16, pp. 7370-7389, 15 November 2014.

[7] K. Sylejmani, A. Dika, “Solving touristic trip planning problem by using taboo search approach,” in *IJCSI*, vol. 8, issue 5, no. 3, September 2011.

[8] L. Manikonda, T. Chakraborti, D. Talamadupula, S. Kambhampati, “AI-MIX: Using Automated Planning to Steer Human Workers Towards Better Crowdsourced Plans,” in IAAI-14, 2014.

[9] L. Sebastia, I. Garcia, E. Onaindia, C. Guzman, “E-tourism: a tourist recommendation and planning application,” in *IJAIT*, vol. 18, issue 05, October 2009.

[10] P. Organisciak, “Why Bother? Examining the Motivations of Users in Large-Scale Crowd-Powered Online Initiatives,” M.S. thesis, Dept. Humanities Computing, Univ. of Alberta, Edmonton, Alberta, Canada, 2010.

[11] P. Vansteenwegen, W. Souffriau, G.V. Berghe, D.V. Oudheusden, “Iterated local search for the team orienteering problem with time windows,” in *COR*, vol. 36, pp. 3281-3290, 2009.

[12] T. Kurashima, T. Iwata, G. Irie, K. Fujimura, “Travel Route Recommendation Using Geotags in Photo Sharing Sites,” in CIKM’10, October 26-30, 2010.

[13] X. Li, “Multi-Day and Multi-Stay Travel Planning using Geo-Tagged Photos,” in ACM SIGSPATIAL GEOCROWD ’13, November 5, 2013.

[14] Y. Kurata, T. Hara, “CT-Planner4: Towards a more user friendly interactive day-tour planner,” in *Information and Communication Technologies in Tourism*, Springer International Publishing, pp. 73-86, 2014.