Multi-day Trip Planning System with Collaborative Recommendation*

Priska Aprilia  
Department of Computer Science and Information Engineering, Inha University  
(aprilia, 13apriska@gmail.com)

Kyeong-Jin Oh  
Department of Computer Science and Information Engineering, Inha University  
(okjkillo@eslab.inha.ac.kr)

Myung-Duk Hong  
Department of Computer Science and Information Engineering, Inha University  
(hmdgo@eslab.inha.ac.kr)

Myeong-Hyeon Ga  
Department of Computer Science and Information Engineering, Inha University  
(gagaman7777@nate.com)

Geun-Sik Jo  
Department of Computer and Information Engineering, Inha University  
(gjo@inha.ac.kr)

Planning a multi-day trip is a complex, yet time-consuming task. It usually starts with selecting a list of points of interest (POIs) worth visiting and then arranging them into an itinerary, taking into consideration various constraints and preferences. When choosing POIs to visit, one might ask friends to suggest them, search for information on the Web, or seek advice from travel agents; however, those options have their limitations. First, the knowledge of friends is limited to the places they have visited. Second, the tourism information on the internet may be vast, but at the same time, might cause one to invest a lot of time reading and filtering the information. Lastly, travel agents might be biased towards providers of certain travel products when suggesting itineraries. In recent years, many researchers have tried to deal with the huge amount of tourism information available on the internet. They explored the wisdom of the crowd through overwhelming images shared by people on social media sites. Furthermore, trip planning problems are usually formulated as ‘Tourist Trip Design Problems’, and are solved using various search algorithms with heuristics. Various recommendation systems with various techniques have been set up to cope with the overwhelming tourism information available on the internet. Prediction models of recommendation systems are typically built using a large dataset. However, sometimes such a dataset is not always available. For other models, especially those that require input from people, human computation has emerged as a powerful and inexpensive approach. This study proposes CYTRIP (Crowdsource Your TRIP), a multi-day trip itinerary planning system that draws on the collective intelligence of contributors in recommending POIs. In order to enable the crowd to collaboratively recommend POIs to users, CYTRIP provides a shared workspace. In the shared workspace, the crowd can recommend as many POIs to as many requesters as they can, and they can also vote on the POIs recommended by other people when they find them interesting. In CYTRIP, anyone can make a contribution by recommending POIs to requesters based on requesters’ specified preferences. CYTRIP takes input on the recommended POIs to build a multi-day trip itinerary taking into account the user’s preferences, the various time constraints, and the locations. The input then becomes a multi-day trip planning problem that is formulated in Planning Domain Definition Language 3

* This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government (MSIP) (No. 2015-R1A2A2A03006190).
(PDDL3). A sequence of actions formulated in a domain file is used to achieve the goals in the planning problem, which are the recommended POIs to be visited. The multi-day trip planning problem is a highly constrained problem. Sometimes, it is not feasible to visit all the recommended POIs with the limited resources available, such as the time the user can spend. In order to cope with an unachievable goal that can result in no solution for the other goals, CYTRIP selects a set of feasible POIs prior to the planning process. The planning problem is created for the selected POIs and fed into the planner. The solution returned by the planner is then parsed into a multi-day trip itinerary and displayed to the user on a map. The proposed system is implemented as a web-based application built using PHP on a CodeIgniter Web Framework. In order to evaluate the proposed system, an online experiment was conducted. From the online experiment, results show that with the help of the contributors, CYTRIP can plan and generate a multi-day trip itinerary that is tailored to the users’ preferences and bound by their constraints, such as location or time constraints. The contributors also find that CYTRIP is a useful tool for collecting POIs from the crowd and planning a multi-day trip.

**Key Words**: Trip Planning, Collective Intelligence, PDDL, Collaborative Recommendation, Recommendation System

1. Introduction

Planning a multi-day trip is a complex yet time-consuming task. It usually starts with selecting a list of Points of Interest (POIs) worth visiting and then arranging them into an itinerary, taking into consideration various constraints and preferences. Nowadays, most people who plan to get around a new city that they are not familiar with would ask friends to recommend POIs of that city, would search the internet for POIs that suit their preferences, or seek advice from travel agents. However, those options have their limitations. First, recommendations from friends are usually limited to the POIs that they have visited. Second, despite the variety of available options in tourist destinations or attractions publicized on the internet, sometimes the information is too overwhelming for tourists to cope with. Lastly, the suggestions from travel agents might be biased towards the providers of certain travel products.

In recent years, Recommendation Systems (RSs) (Borràs et al., 2014) have been set up to reduce the overwhelming tourism information on the internet and suggest tourist destinations. The RS prediction models are typically built using a large dataset consisting of the characteristics of users, users’ historical data, a list of items (POIs in our domain) described by certain features, and ratings given to various destinations by various users. These datasets allow RSs to correlate the characteristics and preferences of users and the features of the destinations and predict how a user would rate a certain place based on a prebuilt prediction model. The RSs can recommend places that are most likely to be highly rated. However, sometimes that kind of dataset is not available to build the prediction model. For other models, especially those that require input from people,
human computation has emerged as a powerful and inexpensive approach to solving computational problems that are beyond the scope of existing artificial intelligence algorithms.

With the advancement and rapid growth of the Web, human computation systems can now leverage the ability of a number of people to perform complex computation. Wikipedia\(^1\) is one of the most successful human computation systems, allowing multiple users to edit Web content. Another notable success for human computation is the Fold.it\(^2\) project, in which participants are asked to fold virtual proteins in the most efficient way possible. In 2007, a project named Galaxy Zoo\(^3\) was launched. It is an online crowdsourced science project in the astronomy field, which invites people to assist in the morphological classification of a large number of images of galaxies. Other researchers (Zhang et al., 2012; Manikonda et al., 2014) used Amazon Mechanical Turks\(^4\) (AMK), a crowdsourcing internet marketplace, to conduct an experiment on their ideas. They recruited workers from AMK to contribute to planning missions by working on Human Intelligence Tasks (HITs) in their research.

This study proposes CYTRIP (Crowdsourced Your TRIP), a multi-day trip planning system that draws on the collective intelligence of a crowd in recommending POIs. CYTRIP enables a crowd of individuals (friends, and friends of friends) to make use of their travel experiences to collaboratively recommend POIs that may suit the requesters’ preferences. To facilitate the collaborative recommendation of the crowd, CYTRIP provides a shared workspace. The shared workspace enables the crowd to make their recommendations in a collaborative way as well as vote on POIs recommended by other people.

In CYTRIP, instead of asking users to specify a general category (for example, Natural Sites), CYTRIP goes beyond that into subcategories. The assumption is clear; for example, a person might not like visiting a mountain, but visiting a beach is a clear preference. Therefore, in CYTRIP, users can specify what they like, in general, by using subcategories, and can specify what kinds of places they would like to visit for their upcoming trip. With the presence of a crowd to recommend POIs, the search space to find a feasible solution to the planning problem can be reduced. Based on the POIs recommended by the crowd, CYTRIP creates a multi-day planning problem formulated in Planning Domain Definition Language 3 (PDDL3). A predefined sequence of actions in a domain file is used to achieve the goals in the planning problem. CYTRIP employs an automated planner to solve the multi-day trip planning problem. The solution returned by the planner is then parsed and displayed to users as a multi-day trip itinerary.

The rest of this paper is organized as follows. Section 2 discusses the related works, followed by an overview of the CYTRIP system, its methods

\(^1\) https://www.wikipedia.org/
\(^2\) http://fold.it/
\(^3\) http://www.galaxyzoo.org/
\(^4\) https://www.mturk.com
and its algorithms in Section 3. The experimental results are discussed in Section 4. Finally, this paper finishes with conclusions and future works in Section 5.

2. Related Work

Over the years, researchers have been trying to deal with the huge volume of tourism information on the internet. In order to find popular POIs for travel route recommendations, several works (Lee and Sohn, 2006; Lee and Sohn, 2009; YU et al., 2009; Chen et al., 2014; Kurashima et al., 2010; Li, 2013) explored the wisdom of the crowd through overwhelming images shared by people on social media sites like Flickr. Furthermore, travel route search problems are commonly formulated as tourist design problems (Gavalas et al., 2014), such as Orienteering Problem with Time Windows (Chen et al., 2014; Li, 2013; Sylejmani and Dika, 2011; Vansteenwegen et al., 2009) and the Traveling Salesman Problem (Kurata and Hara, 2013). To solve the problems, various search algorithms with heuristics (Gavalas et al., 2014) are employed, such as an iterated local search (Vansteenwegen et al., 2009), a taboo search (Sylejmani and Dika, 2011), and a genetic algorithm (Kurata and Hara, 2013).

On the other hand, Recommendation Systems (RSs) with various techniques (Borràs et al., 2014) have been set up to reduce the huge amount of tourism information on the internet and to offer tourist destinations. The RS prediction models are typically built using a large dataset. However, for other models, especially those that require input from people, human computation has emerged as a powerful and inexpensive approach (Zhang et al., 2012; Manikonda et al., 2014).

The proposed CYTRIP multi-day trip planning system facilitates the contribution of a crowd to the itinerary planning task by providing a single shared workspace, in which the crowd can suggest recommendations in a collaborative way. With the presence of the crowd to recommend POIs, the search space to find a feasible solution to the planning problem can be reduced. In solving the trip planning problem, CYTRIP is similar to other systems (Sebastia et al, 2009), that is, formulating the problem as an Artificial Intelligence (AI) planning problem in PDDL3 (Gerevini and Long, 2005) and using an existing planner to solve it. However, the work by those researchers is only applicable for a single-day trip. In this work, the planning problem is more complex, that is, for multi-day trip planning with various time constraints and various start and end locations. Furthermore, the domain formulation can be applied to both single-day and multi-day trip planning.

PDDL aims to standardize the artificial planning languages, and was inspired by the STRIPS (Fikes, 1971) and ADL (Pednault, 1989) formulations of planning problems. Since its first appearance in 1998 until the time of writing this paper, PDDL had already had five versions: PDDL1.2, PDD2.1, PDDL2.2, PDD3.0, and PDDL3.1. Each version came out with new features that its predecessors did not have. The features used in the formulation of this paper are from PDDL2.1 (Fox and Long,
2003) and PDDL3.0 (Gerevini and Long, 2005). However, this paper uses PDDL3.0 as the reference, since the successor to the previous PDDLs also supports previous features and functionalities. In artificial intelligence, planning is a task that involves choosing a sequence of actions that will transform the states of the world, step by step, to achieve some predefined goals. In PDDL, the planning task is divided into domain and problem definitions. The planning problem is created by pairing up the domain and problem definitions (Fox and Long, 2003). The same domain definition can be paired with many problem definitions, producing different planning problems within the same domain. The characteristics of domain behavior are described by parameterized actions. Meanwhile, the characteristics of a problem instance can be seen in the description of specific objects, initial conditions, goals, constraints and planning metrics.

The multi-day trip planning problem is a highly constrained problem. Sometimes, it is not feasible to visit all the recommended POIs within the available time. This issue results an Over-Subscription Planning (OSP) problem (Smith, 2004). Therefore, in order to cope with unachievable goals (visiting POIs), which can result in no solution for the other goals, CYTRIP employs the A* search to select a set of feasible POIs prior to the planning process. Then, the planning problem is created for the selected POIs and fed into the planner. The solution returned by the planner is parsed and displayed to the user as a multi-day trip itinerary.

3. CYTRIP

In order to embrace collaboration of the crowd in recommending Points of Interests, the recommended POIs are one of the inputs to the multi-day planning problem. In the CYTRIP system, the goal is to automatically generate a multi-day trip itinerary based on collective recommendations from a crowd to satisfy the user’s time constraints, taking into account preferences, time windows of the POIs, and distances among the locations.

The proposed form of collaboration in CYTRIP is a shared workspace that enables a crowd to collaboratively recommend POIs. The shared workspace allows the crowd to see the user’s preferences and the POIs that have been recommended, as well as the recommender, and also allows the crowd to vote on the recommended POIs. Later, CYTRIP formulates the multi-day trip planning problem as an artificial intelligence planning problem using PDDL3. Recommending and planning a multi-day trip to Seoul, the capital city of South Korea, was chosen as a study case.

3.1. Dataset

CYTRIP is currently focusing on generation of an itinerary for a trip to Seoul, the capital city of South Korea. The dataset was retrieved using TourAPI3.0. The list of categories and subcategories was provided by Korean Tourism Organization (KTO) via TourAPI3.0.

5) http://api.visitkorea.or.kr/main.do
3.1. System Architecture and Design

<Figure 1> depicts the system architecture of CYTRIP. CYTRIP is composed of the User Interface (UI), the Controller (CT), the Data Manager (DM), the Problem Generator (PG), the Solution Parser (SP), and the Planner (PL). User, crowd, and system communicate and interact with each other through the UI. The shared workspace is a part of the UI for the crowd to add recommendations and vote on the recommended POIs, as well as to see the itinerary of recommended POIs. The form input from the UI is sent to the CT, and the CT sends the input to the DM. DM handles the database as well as the queries for insert, update, delete and select operations in the database, and also sends the required data to the CT. The PG is in charge of generating the problem file with the input data provided by the CT. After generating the problem file, it is sent to the PL along with the domain file. The PL generates a solution based on the problem using the domain file; that is, the solution is a sequence of actions to achieve the goals specified in the problem. The solution is sent back to the PG. Since all the objects in the problem file use identification, rather than the real name of the points of interest, days, and locations, the solution from the PL is parsed and mapped to the actual data. This process is done by the Solution Parser. The results of the parsing process are then sent back to the CT, which sends the parsed solution (the plan) to the UI where it is displayed to the user as an itinerary.

3.2. Recommendation Request

In order for a user to request POI recommendations from the crowd, the user needs to log into CYTRIP. In the implementation, the login
functionality is integrated with the Facebook Graph Application Programming Interface (API)\(^6\). To request a recommendation, there are two steps. The first is specifying preferences by selecting one or more categories and subcategories. In CYTRIP, there are two types of preference: the general preference (GP), and the current preference (CP). The GP is a list of categories and subcategories of what the user prefers, in general; meanwhile, the CP is a list of categories and subcategories that the user prefers for the current visit.

The level of preference over a subcategory is defined using scores ranging from 10 to 100. The higher the score given to a subcategory, the more the user prefers it; and the higher priority makes it likely to be included in the itinerary. Note that a score of 100 given to a subcategory does not mean a POI that belongs to that subcategory is a must-visit POI, but it does indicate how much the user prefers the subcategory to the other subcategories.

The second step is filling out the recommendation request form. In the recommendation request form, a user needs to specify the travel dates, travel times (starting and ending times) as well as a start location and end location. CYTRIP enables the user to specify time and location constraints, day to day; that is, this produces a multi-location, multi-time-constraint and multi-day planning problem. After the recommendation request form is submitted, CYTRIP reveals the task to other users (the crowd) in the system.

### 3.3. Collaborative Recommendation

The crowd in CYTRIP terminology is anyone who is willing to contribute by recommending

\[\text{(Figure 2) CYTRIP Shared Workspace}\]

\(^6\) https://developers.facebook.com/docs/graph-api
POIs to the user. Once a user requests a recommendation described in Section 3.1, the CT opens and reveals the task to all users. CYTRIP provides the shared workspace to enable the collaborative recommendations from the crowd. The list of POIs recommended by the crowd is used to build a multi-day trip itinerary. <Figure 2> depicts the shared workspace.

Once users open the task, they will be directed to the shared workspace to see the current recommendations. The leftmost side shows the list of preferences (GP and CP) generated by the system based on user input. Meanwhile, the middle is the list of recommended POIs from the crowd, as well as the recommender’s name. To add a new recommendation, click the Add recommendation button, and a recommendation form will open, as depicted in <Figure 3>. In the recommendation form, the contributing crowd needs to select a category to recommend and search for a POI that belongs to that category. Once a POI is selected, CYTRIP employs the Google Map API7) to display the location of the POI and its information on a map. A contributing crowd is required to specify the visit duration of the POI.

In the implementation, CYTRIP stores the identification of each request, the contributing crowd and each POI, and in that fashion, CYTRIP is able to prevent duplication of the same POI for the same request. After submitting the recommendation form, the contributing crowd is given an option to share the recommendations on Facebook, and if agreed, CYTRIP will generate a share link and post it on Facebook on the contributing crowd’s behalf. The crowd is allowed to recommend as many POIs to as many users as possible. Since CYTRIP tends to be a social network in a traveling domain, CYTRIP is equipped with a “Like” feature, that is, a feature where the crowd can vote on the recommended POI. Note that CYTRIP stores voter information for each POI, so each person can vote only once. In that fashion, CYTRIP is again able to prevent multiple votes by a single user.

3.5. Multi-day Trip Planning

After getting the list of crowd-recommended

7) https://developers.google.com/maps/
POIs, a further challenge is how to transform those POIs into a multi-day trip itinerary, given the user’s preferences, travel dates, and time and location constraints. In the current existing works, multi-day trip planning problems are usually formulated as Tourist Trip Design Problems (TTDP), which refers to route-planning problems for tourists interested in visiting multiple points of interest. Furthermore, the TTDP is solved using various algorithms with heuristics. However, in this work, this challenge is answered with formulation as an artificial intelligence planning problem using PDDL3. In AI, planning involves choosing a sequence of actions that will transform the state of the world (the planning domain), step by step, so that goals are achieved. The world is typically viewed as consisting of atomic facts (state variables), and actions that make some information true and some information false through its delete and add effects. In PDDL3 formulation, the planning task consists of eight components: objects, predicates, functions, initial states, goal specification, constraints, planning metrics, and actions or operators. In PDDL3, these components are encoded in two separate problem and domain files, described in the following subsections.

3.5.1. Multi-day Trip Planning Problem Formulation

The problem formulation in this work is dynamic, as it depends on user-specified requirements, such as preferences, travel dates, time and location constraints, and also the number of crowd-recommended POIs. The planning problem consists of five parts.

1. (:objects are composed of three types of objects, which are poi, location and day

2. (:init state is a list of all the ground atoms that are true initially. The ground atoms in the initial state are described by means of functions and predicates. The predicates used in the planning problem are as follows:
   - (located_at p l) defines poi p located at location l. For example (located_at myeong_dong lc_Myeong_dong)
   - (open_on l d) indicates location l is open on day d. For example (open_on myeong_dong day1)
   - (person_at_on l d) indicates a person is initially at location l on day d, e.g. (person_at_on lc_grand_hyatt_seoul day1)
   - (day-now d) defines d as the initial day of the trip. For example, (day-now day1)
   - (day-next d1 d2) indicates day d1 is the day after d2. For example, (day-now day1 day2)
   - (at-now l) indicates l as the initial location. The value of at-now changes upon execution of move and change-day actions (see domain formulation)
   - (at-next l1 l2) defines the location transition between the end location l1 of the current day and the start location l2 of next day.

Meanwhile the functions used in the planning problem are as follows:
   - (visit_duration p) defines visit duration, e.g.
(= (visit_duration myeong-dong) 120)
• (move_duration x y) defines the moving duration from location x to location y. For example, (= (move_duration lc_inha_university lc_insadong) 110)
• (open_hour l), (close_hour l), and (last_admission l) indicate the opening hour, closing hour and last admission hour of location l
• (available_time d), (current_time d), (end_time d), and (total_moving_time d) define the available time, current time, end time, and total moving time of day d.

Note that the numeric value indicated in the functions for open hour, close hour, last admission, visit duration, move duration, current time, available time and end time is a time unit. In the implementation, the values of the functions are transformed into minutes to avoid floating points. For example Gyeongbok Palace is open at 09:30 AM, therefore this information is transformed into (= (open_hour loc_gyeongbok_palace) 570).

The value of the predicates located_at, open_on, day-next, and at-next, and the functions visit_duration, move_duration, open_hour, close_hour, and last_admission will not change upon execution of any action. For example, Insadong is open on Tuesday; that is, no single action in the plan can change this fact, and the same is true for the value of the other aforementioned predicates and functions. However, the value of predicates person_at_on and at-now, and for functions available_time, current_time, end_time, and total_moving_time will change upon execution of actions. For example, for the predicate at-now, let us assume a person is initially at Inha University, and after that, the person moves to Insadong. Obviously, the position of that person has changed from Inha University to Insadong. Similarly, the function current time, upon moving from Inha University to Insadong, will change according to the moving time between the two locations.

3. (:goal defines a list of goals to be achieved. In CYTRIP, the goal is visiting a list of POIs. The goal is defined using predicate (visited p) where p is a poi
4. (:constraints can be hard and soft constraints. Hard constraints must be satisfied in any valid plan. In this work, the following hard constraints are defined:
• Each location can be visited at most once
( forall ( ?d - day ?l - location ) ( at-most-once (person_at_on ?l ?d )))
5. Planning metrics can maximize or minimize a function or a set of functions. For the planning problems, we would like to minimize the objective function total_moving_time.

3.5.2. Multi-day Trip Planning Domain Formulation

The domain file in this work is static and fixed. To achieve the multi-day trip planning goals, three actions are defined, namely move, visit, and change-day. Note that even though the domain formulation is fixed, it is applicable to both single- and multi-day trip planning. Both the move and visit actions are formulated as durative actions, that is, actions that take time.

Action 1 depicts the move action to move from one location to another location on a day.
Action 1: Move from x to y on day d

```
(:durative-action move
 :parameters (?x - location ?y - location ?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))
 :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))))
 )
```

The action move takes as a parameter the initial location ?x and the destination ?y and day ?d. The duration for the action is defined in terms of movement duration from location ?x to location ?y. The conditions for this action to be applicable are:

1. a person is at location ?x on day ?d
2. the available time of day ?d is greater than the moving duration from location ?x to location ?y
3. current day is day ?d
4. a person is at location ?x

Conditions (3) and (4) are used to indicate the current location of a person and the current day. Note that, ?x, ?y, and ?d in the action are parameters, and they will be unified with the current existing facts that match the predicates and functions in the conditions. If all of the conditions are true, then the action can be taken. The effects of the move action assert:

1. a person is at destination ?y on day ?d
2. a person is now at ?y
3. a person is no longer at location ?x on day ?d
4. a person is no longer at location ?x
5. the available time, current time, and total moving time are modified according to the moving duration from ?x to ?y

After moving to a location, a visit action needs to be performed in order to achieve the goal of visiting a POI. Therefore, the visit action is defined as shown in Action 2.

Action 2 defines a visit action that indicates visiting poi p located at location l on day d. The

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 (>= (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))))
 )
```

169
Action 3: Change-day d1 to d2

(:action change-day
 :parameters (?l1 ?l2 - location ?d1 ?d2 - day)
 :precondition (and
   (current-day ?d1)
   (day-next ?d1 ?d2)
   (at-next ?l1 ?l2)
   (not (current-day ?d2))
   (at-now ?l1)
   (person_at_on ?l1 ?d1))
 :effect (and
   (not (at-now ?l1))
   (current-day ?d2))
)

conditions for this action to be applicable are as follows:
(1) poi ?p is located at location ?l
(2) a person is already at location ?l
(3) current time of day ?d has to be greater than
or equal to the opening time of location ?l
(4) the current time of day ?d has to be smaller
than the last admission time of location ?l
(5) the activity will be finished before closing
hour of location ?l
(6) available time of day ?d is greater than the
visit duration of poi ?p

The effects of applying the visit action assert
that poi p is visited, and the current available times
are modified according to the visit duration. Finally, to achieve multi-day trip planning, another
action is defined, namely action change-day. This
action is used to transform the current day to the
day after the current day.

In order to explain this action, the following
example is given. Let us assume a user wants to
have a two-day trip with the start location as Inha
University and the end location as Inha University
for first and second days. Since the planning
constraints state that each location can be visited at
most once throughout the plan, CYTRIP needs to
transform these two same locations using the
unique identification for each location instance.
The detailed transformation is described in <Table
1>.

With the location constraints above, CYTRIP
generates the following initial states and
constraints.

In Action 3, the input parameters are location
?l1, location ?l2, day ?d1 and day ?d2. The first
condition of the action states that there must exist
a day ?d1 which results in current-day ?d1 being

Action 4: Initial states and Constraints

(:init
 (person_at_on loc_1 day_1)
 (person_at_on loc_3 day_2)
 (at-now loc_1)
 (at-next loc_2 loc_3)
 (current-day day_1)
 (day-next day_1 day_2)
 (= (move_duration loc_1 loc_2 0)
  (= (move_duration loc_2 loc_3 0)
   (___)
 )
 (:constraints (and (forall (?d - day ?l - location)
   (at-most-once (person_at_on ?l ?d)))
   (at end (person_at_on loc_3 day_1))
   (at end (person_at_on loc_4 day_2))
)

(Table 1) Location instance naming

| Location name      | Location type | Day   | Location id |
|--------------------|---------------|-------|-------------|
| Inha University    | Start         | day 1 | loc_1       |
| Inha University    | End           | day 1 | loc_2       |
| Inha University    | Start         | day 2 | loc_3       |
| Inha University    | End           | day 2 | loc_4       |
true. Following the example, parameter \(?d1\) will be unified with literal \(\text{day}_1\). Under the second condition, \((\text{day-next} \ ?d1 \ ?d2)\) needs to be true. Since we have the fact in the initial state that \((\text{day-next} \ \text{day}_1 \ \text{day}_2)\), \(?d2\) will be unified with \(\text{day}_2\). Now for the third condition, we need to unify \(?l2\) using the predicate \((\text{at-next} \ ?l1 \ ?l2)\), since we have the fact \((\text{at-next} \ \text{loc}_2 \ \text{loc}_3)\). Therefore \(?l1\) and \(?l2\) will be unified with \(\text{loc}_2\) and \(\text{loc}_3\), respectively. Note that, \(\text{loc}_3\) is the start location of \(\text{day}_2\), which is the next-day of \(\text{day}_1\), and \(\text{loc}_2\) is the end location of \(\text{day}_1\). Therefore, for the two last conditions \((\text{at-now} \ ?l1)\) and \((\text{person-at-on} \ ?l1 \ \text{day1})\) to be true, a move action needs to be taken to move to the last location of the current day, which indicates the trip for the current day has ended and a new trip for the next day can be started by executing the action \(\text{change-day}\).

This change-day action will produce a plan maximizing the number of POIs to be visited in a day, given the time constraints. This action is a non-durative action; that is, there is no duration for transforming the day. Note that, if the planning problem is a single day, this change-day action will not be executed.

In CYTRIP, each location has a unique identification (ID), even for the same location. This is because the defined constraint prevents a location to be visited more than one time. However, if locations with different identifications are the same location, in fact, the move duration between them results in zero. In the planning problem formulation described in Section 3.5.1, each object has its own ID. When a solution to the planning problem is found, this identification will be mapped with the instances’ real information stored in the database by the Solution Parser.

3.5.3. Over-Subscription Planning Problem (OSP)

In classic planning, all the conjunctive fluent specified in the goal must be satisfied in the state reached after applying a sequence of actions. In fact, goals in a classic planning problem can be defined as soft goals. That is, a list of goals that we desire to achieve, but that are not necessary to achieve, if it is impossible to satisfy all the goals under the given circumstances (e.g. limited resources). In planning with only soft goals, a plan is still valid even if it achieves a subset of the goals, or even an empty set. This planning problem with soft goals is called the Over-Subscription Planning Problem (OSP) (Smith, 2004).

In PDDL3, the OSP can be expressed using preferences in the goal definition. There are a few existing domain-independent planning systems like SGPlan5 (Hsu et al., 2006) and MIPS-XXL (Edelkamp et al., 2006) that can support planning with preferences. However, in real practical domains, there could be several causes that make it impossible or useless to achieve all the soft goals (Garcia-Olaya et al., 2011). The cause could be that two or more goals are mutually exclusive (i.e. cannot be true at the same time) or there could be limited resources (e.g. not enough time is left). The optimal solution for OSP problems can be computed by finding a plan for each \(2^n\) combinations where \(n\) is the number of goals to be
achieved (or in CYTRIP, the number of POIs to be visited) and then selecting the goals with the maximum utility. It is obvious that it is infeasible to compute all the possible solutions except for very small n. Hence, some works (García-Olaya et al., 2011; Smith, 2004) do a priori selection of the goals. The goals can be selected to find the best subset of goals to plan for, prior to the planning process. The later step, the planning process, deals with only the selected goals.

In the planning problem for a given user, not all activities are likely to be included in the plan, since the plan schedule will depend on the user’s time constraints and the restrictions of the environment, such as the opening hours of locations, as well as whether a location is open on the day a user would like to make the trip. Since it is a highly constrained problem (that is, one with the aforementioned constraints), the problem is how to select the subset of the recommended POIs, maximizing the user’s preferences while minimizing the overall time spent, in order to visit more places. Therefore, a multi-day trip planning problem with limited resources might lead to an OSP.

3.5.4. Heuristic Search for OSP

Similar to existing research (Smith, 2004; García-Olaya et al., 2011), CYTRIP performs a priori selection of the goals (the list of POIs to be visited) to plan for before feeding the planning problem to an automated planner. In this work, CYTRIP adopts the A* tree search with heuristics to solve the OSP problems, which is similar to other work (Benton et al., 2005).

A* employs best-first search and looks for a path with the lowest cost from a given initial node to one goal node \( G_i \) (out of one or more possible goal nodes). As A* traverses the graph, it builds a tree of partial paths. A priority queue is used to store the leaf nodes of the tree (called the open set of fringes). This priority queue arranges the leaf nodes in ascending order using a cost function that combines an estimated heuristic of the cost to reach the goal, as well as the distance traveled from the initial node. More formally, the cost function is

\[
f(i) = g(i) + h(i) \quad (1)
\]

Here, \( g(i) \) is the known cost of reaching goal node \( G_i \) from initial node \( S \). This value is recorded and tracked by A*. Meanwhile, \( h(i) \) is a heuristic estimate of the cost to reach any goal node from \( S \).

In CYTRIP, node \( S_j \) in A* represents the starting location of current day \( D_j \), each node \( G_i \) represents a POI and its location, and finally, the ending location for \( D_j \) is represented by \( E_j \). Hence, in multi-day trip planning, the goal selection is finding a series of paths \( P_j = S_j, G_i, G_{i+1}, \ldots, G_n, E_j \) where \( i \leq \text{N} \), is the total recommended POIs, and \( j \leq \text{D} \) where \( j \) represents the day with index \( j \) and \( \text{D} \) is the total day. The path \( P_j \) should minimize the cost \( f_j(n) \).

Note that, if the planning problem is a single day, or if the recommended POIs can be all visited within a single day, the objective of A* becomes finding a single path.
The goal of CYTRIP is to find a trip itinerary that has the total satisfaction factor as high as possible, while the travel time remains as low as possible. Hence, CYTRIP needs to define an evaluation function for A* to expand a node. The evaluation function is defined as follows:

\[ EF_i = w_1[S_{\text{norm}}] + w_2[V_{\text{norm}}] + w_3[T_{\text{norm}}] \quad (2) \]
\[ w_1 + w_2 + w_3 = 1 \quad (3) \]

In (2) \( S_{\text{norm}} \), \( V_{\text{norm}} \), and \( T_{\text{norm}} \) represent the normalized value of the total score, total vote and total travel time, respectively. Meanwhile, the parameters \( w_1 \), \( w_2 \) and \( w_3 \) represent the weight coefficients for the particular components of the evaluation function. These parameters can be adjusted. In order to have a proportional effect in the evaluation function when the value of the cost of reaching a node in the search tree changes, the components should be normalized:

\[ S_{\text{norm}} = 100 \times \frac{TS_i}{TS} \quad (4) \]
\[ V_{\text{norm}} = 100 \times \frac{TV_i}{TV} \quad (5) \]

where terms are defined as follows:

- \( TS_i \) - total score of POIs achieved from initial node \( S \) to the current goal node \( G_i \)
- \( TS \) - total score of the recommended POIs
- \( TV_i \) - total vote of POIs achieved from initial node \( S \) to the current goal node \( G_i \)
- \( TV \) - total vote of the recommended POIs

Since the approach used in the evaluation function maximizes the value of its three components, for the travel time, it should be the opposite: the value of the travel time should be minimized. Hence, in order to minimize the travel time, its value should be complemented, which in fact will minimize the travel time. The complementary value of the travel time is defined as follows:

\[ T_{\text{norm}} = 100 \times \left[ 1 - \frac{TVD_i + TMD_i}{TD_i} \right] \quad (6) \]

where terms are defined as follows:

- \( TVD_i \) - total visit duration of POIs from initial node \( S \) to the current goal node \( G_i \)
- \( TMD_i \) - total move duration from initial node \( S \) to the current goal node \( G_i \)
- \( TD_i \) - tour duration of day

The aforementioned evaluation function in equation (2) is similar to work by Sylejmani and Dika (2011). However, the evaluation function in this work is extended to having the vote as the third component. A* evaluates the node to expand to build a partial path using the cost function \( f(i) \), and the evaluation function was previously maximized. Therefore, CYTRIP needs to transform its evaluation function \( EF_i \) in equation (2) to a cost function \( f(i) \). The first two components of \( EF_i \) are used as \( g(i) \), and the third component is used as \( h(i) \). The transformation to both functions \( g(i) \) and \( h(i) \) is done by complementing the value of the involved components represented in equations (7) and (8), respectively:

\[ g(i) = 100 \times \left[ 1 - \frac{w_1[S_{\text{norm}}] + w_2[V_{\text{norm}}]}{100} \right] \quad (7) \]
similarly,
\[ h(i) = 100 \times \left[ 1 - \frac{w_3 \times \left\lceil T_{\text{norm}} \right\rceil}{100} \right] \]  \hspace{1cm} (8)

since the value of \( g(i) \) and \( h(i) \) are normalized and both will form an intact \( f(1) \). Therefore, both components should be multiplied by 0.5:

\[ g'(i) = 0.5 \times g(i) \]  \hspace{1cm} (9)
\[ h'(i) = 0.5 \times h(i) \]  \hspace{1cm} (10)

So the final cost function \( f'(i) \) is defined as

\[ f'(i) = g'(i) + h'(i) \]  \hspace{1cm} (11)

CYTRIP employs the A* search (described in Section 2.3) with the aforementioned propagated cost function \( f'(i) \) to evaluate the partial path. The A* algorithm is run iteratively for at most \( |D| \) iterations, where \( |D| \) is the number of days for the trip. The days are arranged in ascending order based on date. For each iteration, the start node is \( S_j \), which is the starting location of \( D_j \). The aim of each iteration is to find \( P_j = \{S_j, G_i, G_{i+1}, ..., G_n, E_j\} \) that minimizes \( f'_j(i) \).

Let us suppose the A* search has finished running for \( D_j \). In the next iteration, \( D_{j+1} \), the A* algorithm only considers the nodes (POIs) that are not included in the previous iteration, \( D_j \). Those nodes will be generated as the successors of root node \( S_{j+1} \). If there is no unscheduled node (POI), the iteration \( D_{j+1} \) will not proceed.

Note that both \( g'_j(S) \) and \( h'_j(S) \) are initially zero in the start node, and the values of both functions are propagated from the start node down to node \( n \). Each node in the search tree is annotated with its location ID, score, visit duration, vote, its predecessor, and the predecessor’s propagated cost, current time, and available time, and finally it also records final location \( E_j \). The moving cost (duration) from node \( G_i \) and its successor \( G_{i+1} \) is retrieved from a moving time matrix. The moving time matrix is pre-generated prior to the running of A* with the combinations of all locations in two directions.

Since the planning problem described in this thesis has user time constraints and location time windows, if a node does not meet the time function conditions specified in the visit and does not comply with the move actions in the domain formulation stated in Section 3.5.2, the node will not be generated. The additional condition for a node to be feasible is that the available time after node \( G_i \) is generated has to be sufficient to move from \( G_i \) to final location \( E_j \); otherwise, it will not be generated. If, in the current node \( G_i \) given the current available time, there is no feasible successor, \( G_{i+1} \), then it generates goal \( E_j \) and records it as one of the solution paths. A* will evaluate all the paths to that reach \( E_j \) from \( S_j \) and return the path with the lowest cost, \( f'(i) \).

It is obvious that the graph traversing the value of \( g'_j(i) \) is getting smaller as more POIs are visited. Meanwhile, the value of \( h'_j(S) \) is getting higher as the travel time gets higher. A smaller
\( g'_{j(i)} \) and a higher \( f'_{j(i)} \) mean the path is closer to goal node \( E_j \). The traversing of a path for day \( D_j \) stops when it reaches goal node \( E_j \), which represents the final location of \( D_j \).

3.5.5. Planning for the Selected Goals

The best path found by the A* search algorithm from \( \mathcal{O} \) to \( \mathcal{G} \) produces the list of goals to plan for. CYTRIP then generates the problem file with the selected goals and feeds it to a temporal planner (Hsu et al., 2006) all at once. In the online experiments, the goals given to the planner are no longer oversubscribed in all planning problems of the participating users. However, if the current list of goals cannot be solved by the planner, CYTRIP will remove the last goal in the order, one at a time, and create a new planning problem. This process is done until a solution is found or no more goals can be removed.

4. Experimental Result

An experiment was conducted to evaluate and prove the effectiveness of the proposed method in dealing with complex multi-day trip planning to satisfy the user’s preferences, taking into account the user’s time and location constraints. To solve the multi-day trip planning problem, SGPLAN5 (Hsu et al., 2006) was employed. The reason for selecting this planner was due to the wide range of PDDL3 features that it supports, which are used in CYTRIP to express a dynamic planning problem.

An experiment result comparison against other similar works did not publish the datasets they used for their experiments. Therefore, in order to verify our approach and method, an online experiment with the involvement of participants was conducted.

4.1. Online Experiment

In the online experiment, some people were asked to contribute. The participants were divided into two groups. The first group was requesters, whose main role was to request recommendations by specifying their preferences and travel dates, as well as time and location constraints. The requesters were mainly people who live outside Korea and who are not familiar with Seoul, but a few of them had been to Seoul before. The second group was recommenders, whose task was to recommend places they know to the requesters, given the requesters’ preferences. This group of people comprised Inha University students, and they were familiar with the tourist attractions in Seoul because they travel frequently. In this experiment, the total number of participants was 38. However, we only used the results from 25 people because some data included too few recommendations, and some included errors, so we excluded data from 13 people. The participants were divided into two groups, 12 requesters, and 11 recommenders, while the remaining two participated in both groups.

In order to enable users to participate in the online experiment, CYTRIP was implemented as a web-based application built using PHP on a CodeIgniter Web Framework8). The system was hosted on an Ubuntu server running on a virtual
machine with base memory of 1GB. The SGPLan5 planner used for the experiment was integrated with CYTRIP. This integration enabled online generation of the itineraries and their presentation to users whenever they wanted to see them.

In the online experiments, the group of requesters was asked to specify their preferences and planning details, including dates, times, and locations. Based on the specified preferences, the group of recommenders was asked to contribute by recommending POIs that they know. The planning problems specified by the requesters varied, from a single day to seven-day plans with various time and location constraints, as well as preferences. The itineraries from the planning problem were automatically generated online. First, the system ran the algorithm to select a set of goals to plan for, and then created the planning file for the selected goals. Next, the planning problem was fed to the planner to solve the problem. The first generation of planning problem results is presented in Table 2, and all planning problems were solved. The processing time is presented in the sixth column. In the worst case scenario, it took less than 0.5 seconds to generate an itinerary for a seven-day trip.

As Table 2 suggests, User 2, User 4, User 8, User 12 and User 14 planning problems were oversubscribed (OSP), so some of the POIs were not scheduled in the itineraries due to time limitations. However, these OSP problems could easily be solved by running the proposed A* algorithm to select the goals prior to the planning process by SGPLan5.

It is obvious that CYTRIP is indeed able to generate multi-day trip planning with real-world

Table 2) CYTRIP’s Performance based on Number of days and Suggested POIs

| User  | No. of Days | No. of Used Days | Total Recommended POIs | Total POIs Included in the Plan | Processing Time (s) |
|-------|-------------|------------------|------------------------|--------------------------------|---------------------|
| User 1| 2           | 1                | 4                      | 4                              | 0.13                |
| User 2| 1           | 1                | 3                      | 2                              | 0.14                |
| User 3| 2           | 1                | 6                      | 6                              | 0.14                |
| User 4| 1           | 1                | 6                      | 4                              | 0.14                |
| User 5| 7           | 1                | 8                      | 8                              | 0.39                |
| User 6| 4           | 2                | 9                      | 9                              | 0.21                |
| User 7| 7           | 1                | 7                      | 7                              | 0.44                |
| User 8| 1           | 1                | 6                      | 5                              | 0.14                |
| User 9| 1           | 1                | 3                      | 3                              | 0.21                |
| User 10| 2           | 2                | 4                      | 4                              | 0.13                |
| User 11| 3           | 1                | 4                      | 4                              | 0.14                |
| User 12| 3           | 2                | 7                      | 6                              | 0.15                |
| User 13| 4           | 1                | 2                      | 2                              | 0.14                |
| User 14| 4           | 4                | 16                     | 14                             | 0.6                 |

Average 0.22
data that is complex. Given users’ preferences and the number of visiting days, and given the active crowd to make suggestions, CYTRIP is effective in generating a multi-day trip itinerary that is

(Figure 4) Sample of Generated Itinerary

(Figure 5) Itinerary for Trip in Seoul with Route Shown on Daum Map
tailored to a user’s preference. A sample generated itinerary for one of the requesters is shown in <Figure 4>, and another sample itinerary is shown on the Daum Map depicted in <Figure 5>.

### 4.2. CYTRIP Usability Questionnaires

In order to evaluate the satisfaction of the users participating in the online experiment, two surveys were conducted by asking users to fill out two questionnaires. One questionnaire assessed the satisfaction of requesters, and the other was aimed at recommender satisfaction, as presented in <Table 3> and <Table 4>, respectively.

The results in <Table 3> suggest that the requesters were satisfied overall with CYTRIP, and they found CYTRIP a useful tool for generating an itinerary with the help of other people. With the help of the crowd, CYTRIP was able to help users notice the places they might not have thought of before. Meanwhile, the recommenders’ satisfaction with CYTRIP was average, as seen in <Table 4>. The reason the evaluation score for the entire system is low was because participants were not

|   | Question                                                                 | Average (out of 7) |
|---|--------------------------------------------------------------------------|--------------------|
| Q1 | How satisfied are you with CYTRIP overall?                               | 4.92               |
| Q2 | Did the use of CYTRIP heighten your expectations of sightseeing in Seoul / Korea | 5.29               |
| Q3 | Did you feel clearer than you did before using CYTRIP in what you would like to do at the destination? | 5.14               |
| Q4 | Did the recommendation from users of CYTRIP help you notice tourist spots that you would not consider in your usual planning? | 5.58               |
| Q5 | Was the tour plan you made with CYTRIP useful for your trip?              | 5.58               |
| Q6 | Did the itinerary’s automatic generation save time compared to manual planning? | 4.36               |
| Q7 | Did the recommendations from people save time compared to searching a list of places to visit on the internet? | 5.71               |
| Q8 | How useful is a tool like CYTRIP as a crowdsourced recommendation and automatic planning system? | 5.93               |
| Q9 | Were you able to draw up a travel plan to your taste (based on the preferences you specified)? | 5.57               |

|   | Question                                                                 | Average (Out of 7) |
|---|--------------------------------------------------------------------------|--------------------|
| Q1 | How satisfied are you with CYTRIP overall?                               | 4                  |
| Q2 | Was the CYTRIP recommendation form easy to use?                          | 3.6                |
| Q3 | Was the information (e.g. preferences, travel dates) that the user provided through CYTRIP helpful in guiding you to make a recommendation? | 4.1                |
| Q4 | How useful is a tool like CYTRIP as a crowdsourced recommendation and automatic planning system? | 5                  |
familiar with the system’s UI. Although we received around 5 to 6 points on the system’s usefulness aspect and for the evaluation of the entire recommended schedule, the score for the question “Was the CYTRIP recommendation form easy to use?” is 3.6. Also, the participants gave a lot of feedback about being unfamiliar with the system’s interface. Since recommenders undertook the task of recommending places to people, they apparently need a more user-friendly and nicer interface to discover the places. Even so, they agreed that a tool like CYTRIP is useful, even though the rating was not as high as from the requesters.

5. Conclusion and Future 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 considering various constraints. With the advancement and the fast growth of the Web, tourism information can easily be found online. However, the huge amount of information on the internet may be too overwhelming to cope with. In recent years, Recommendation Systems (RSs) have been set up to reduce the immense options on the Web. Such RSs usually need a large dataset in order to build prediction models. However, such a dataset is not always available. For other models, especially those that require input from people, human computation has emerged as a powerful and inexpensive approach to solving many computational problems that are beyond the scope of existing artificial intelligence algorithms. With the growth of the Web, human computation can now easily leverage a number of people to perform complex computation in solving problems they are good at.

This work proposes CYTRIP, a multi-day trip itinerary planning system that engages people (i.e. a crowd) for recommending POIs. The contribution of this paper is using collective intelligence based on PDDL to make a better content and planning system for multi-day trips. CYTRIP provides a shared workspace from which to draw on the collective intelligence of a crowd by enabling the crowd to collaboratively contribute to the task of POI recommendation. Later, the recommendations from the crowd become an AI planning problem formulated in PDDL3.

Multi-day trip planning is a highly constrained problem, even though the crowd is there to reduce the search space by recommending POIs to the user. The list of recommended POIs can still result in an over-subscription planning problem (OSP). That is, within limited resources, it is not likely that all POIs can be visited within the given number of days and the available time. To solve the OSP, CYTRIP proposes a method to determine which goals can be achieved within the given constraints. That method employs the A* search algorithm to select a set of goals to plan for, with benefits as high as possible.

In order to verify and evaluate the proposed system, an online experiment was conducted. The online experiment involved a group of participants. The online experiment shows that, given the suggestions from the crowd, CYTRIP can
automatically generate a trip itinerary that is tailored to the user’s preferences. Therefore, with the presence and help of the crowd in recommending POIs, and combined with planning technology, trip planning can be much easier. In addition, CYTRIP can also reduce the search space for both the goal selection algorithm and the planning algorithm. Even though the approach to solving a multi-day trip planning problem is domain-specific, the author believes that the same approach and logic can be applied to other planning and scheduling problems, for example, a vehicle routing problem or robotic planning.

Finally, from the results of surveys completed using questionnaires, the author can conclude that the requesters are more interested in the functionality of the system as a crowdsourced trip recommendation and planning system. On the other hand, the recommenders are interested in the user interface. From the point of view of a recommender, this is understandable, as they need to recommend places to people, so a nicer user interface might help them to quickly find the places to recommend and provide better navigation. However, with the present method, CYTRIP still has room for improvement. Even though the experiments were conducted online, future experiments with large-scale users and a large dataset should be conducted. The current itinerary-generation process does not involve both requesters and the crowd, because system unilaterally decides the “best” itinerary for the requester. Future work should involve users in the planning process, so that users can help to deal with the OSP problem by having the system prompt for infeasible POIs to be removed by the user. In addition, the system should let a user decide the priority, whether it is list of popular places or of places that suit the user’s taste, even though they might not be popular.

References

Benton, J., M. B. Do, and S. Kambhampati, “Over-subscription planning with numeric goals,” *International Joint Conferences on Artificial Intelligence Organization*, (2005), 1207–1213.

Borràs, J., A. Moreno, and A. Valls, “Intelligent tourism recommender systems: A survey,” *Expert Systems with Applications*, Vol.41, No.16(2014), 7370–7389.

Chen, G., S. Wu, J. Zhou, and A. K. Tung, “Automatic Itinerary Planning for Travelling Services,” *IEE Transaction on Knowledge and Data Engineering*, Vol.26, No.3(2014), 514–527.

Edelkamp, S., S. Jabbar, and M. Nazih, “Large-Scale Optimal PDDL3 Planning with MIPS-XXL,” *International Planning Competition Booklet*, (2006), 28–30.

Fox, M. and D. Long, “PDDL2.1 : An Extension to pddl for Expressing Temporal,” *Journal of Artificial Intelligence Research*, Vol.20, No.1(2003), 61–124.

Garcia-Olaya, A., T. Rosa, and D. Borrajo, “Using the relaxed plan heuristic to select goals in oversubscription planning problems,” *Advances in Artificial Intelligence*, (2011), 183–192.

Gavalas, D., C. Konstantopoulos, and K. Mastakas, “A survey on algorithmic approaches for
solving tourist trip design problems,” *Journal of Heuristics*, Vol.20, No.3(2014), 291–328.

Gerevini, A. and D. Long, “Plan constraints and preferences in PDDL3,” *The Language of the fifth International Planning Competition-Technical Report*, University of Brescia, 2005.

Kurata, Y. and T. Hara, “CT-planner4: Toward a more user-friendly interactive day-tour planner,” *Information and Communication Technologies in Tourism*, (2013), 73–86.

Hsu, C. W., B. W. Wah, R. Huang, and Y. Chen, “New features in SGPlan for handling preferences and constraints in PDDL3.0,” *Proceedings of the Fifth International Planning Competition*, (2006), 39–41.

Kurashima, T., T. Iwata, G. Irje, and k. Fujimura, “Travel route recommendation using geotags in photo sharing sites,” *Proceedings of the 19th ACM international conference on Information and knowledge management*, (2010), 579–588.

Lee, J. H. and M. M. Sohn, “Traveling Product Bundling on Web Service Composition in Ubiquitous Computing Environment,” *Journal of Intelligence and Information Systems*, Vol.12, No.2(2006), 49–65.

Lee, J. H. and M. M. Sohn, “Framework for Information Integration and Customization Using Ontology and Case based Reasoning,” *Journal of Intelligence and Information Systems*, Vol.15, No.4(2009), 141–158.

Li, X., “Multi-day and multi-stay travel planning using geo-tagged photos” *Proceedings of the Second ACM SIGSPATIAL International Workshop on Crowdsourced and Volunteered Geographic Information*, (2013), 1–8.

Manikonda, L., T. Chakraborti, S. De, K. Talamadupula, and S. Kambhampati, “AI-MIX: using automated planning to steer human workers towards better crowdsourced plans,” *Second AAI Conference on Human Computation and Crowdsourcing*, (2014), 42–43.

Pednault, E. P. D., “ADL: Exploring the middle ground between STRIPS and the situation calculus,” *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, (1989), 324–332.

Fikes, R. E. and N. J. Nilsson, “STRIPS: A new approach to the application of theorem proving to problem solving,” *Artificial Intelligence*, Vol.2, No.3-4(1971), 189–208.

Sebastia, L., I. Garcia, E. Onaindia, and C. Guzman, “e-Tourism: a tourist recommendation and planning application,” *International Journal on Artificial Intelligence Tools*, Vol.18, No.5(2009), 717–738.

Smith, D. E., “Choosing Objectives in Over-Subscription Planning,” *International Conference on Automated Planning and Scheduling*, Vol.4(2004), 393–401.

Sylejmani, K. And A. Dika, “Solving touristic trip planning problem by using taboo search approach,” *International Journal of Computer Science Issues*, Vol.8, No.5(2011), 139–148.

Vansteenwegen, P., W. Souffriau, G. V. Berghe, and D. V. Oudheusdena, “Iterated local search for the team orienteering problem with time windows,” *Computers and Operations Research*, Vol.36, No.12(2009), 3281–3290.

Yu, Y. H., S. J. Cha, and G. S. Jo, “Hybrid Heuristic Applied by the Opportunity Time to Solve the Vehicle Routing and Scheduling Problem with Time Window,” *Journal of Intelligence and Information Systems*, Vol.15, No.3(2009), 137–150.
Zhang, H., E. Law, R. C. Miller, K. Z. Gajos, D.C. Parkes, and E. Horvitz, “Human computation tasks with global constraints,” Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, (2012), 217~226.
국문요약

협업적 추천 기반의 여행 계획 시스템

프리스카*・오경진*・홍명덕*・가명현*・조근식**

여행을 계획하는 일은 매우 복잡하고 많은 시간을 필요로 한다. 여행 계획을 정할 때에는 보통 관심지점(point of interests, POIs)을 선택하고 그에 따른 다양한 제약 조건들을 고려하여 일정을 계획한다. 관심 지점을 선정할 때 친구들에게 의견을 묻거나 인터넷에서 직접 정보를 찾으며 여행사의 도움을 받기도 한다. 하지만 이러한 방법들은 다음과 같은 어려움이 있다. 친구들에게 의견을 묻는 경우에는 친구들이 방문해 보지 못한 장소에 대한 정보를 얻기 어렵고 인터넷에서 정보를 찾는 경우에는 오히려 너무 많은 여행 정보들 때문에 필요한 정보를 탐색하고 정리하는데 많은 시간이 필요하며 여행사의 도움을 받을 때에는 여행 일정이 여행을 제공해주는 업체들 쪽으로 편중될 우려가 있다. 이러한 문제를 해결하기 위해 본 논문에서는 여행 일정 계획 시스템인 CYTRIP을 제안한다. CYTRIP은 웹 기반의 추천 시스템으로서, 여행 정보를 공유할 수 있는 공간을 제공하고, 이를 통해 참여자들의 집단 지성에 따른 관심 지점을 추천 받는다. 그리고 PDDL3를 통해 추천된 지점들의 시간적, 공간적 제약조건 따라 여행 일정이 자동으로 생성되며 이렇게 생성된 일정은 지도 위에 표시되어 사용자에게 제공된다. 여행을 계획할 때에 정해진 기간 동안 모든 추천 관심지점을 방문할 수 없는 경우가 발생한다. 이러한 문제를 피하기 위해 저런 시간에 방문 가능한 관심지점들의 후보 집합을 선택하고 이 후보 집합들에 대한 여행 일정을 생성한다. 제안하는 시스템의 성능평가를 위해 사용자 평가를 실시하였다. 사용자 평가를 위해 한국관광공사에서 제공하는 데이터를 활용하였고 평가 결과 제안하는 시스템이 여러 참여자들의 집단 지성을 통해 여행 일정을 계획하는데 유용하다는 것을 알 수 있었다.

주제어: 여행 계획, 집단 지성, PDDL, 협업적 추천, 추천 시스템.

논문접수일: 2016년 2월 5일 논문수정일: 2016년 3월 14일 게재확정일: 2016년 3월 15일
원고유형: 일반논문 교신저자: 조근식

* 인하대학교 컴퓨터정보공학과
** Corresponding Author: Geun-Sik Jo
Department of Computer and Information Engineering, Inha University
100 Inharo, Nam-gu, Incheon 402-751, Korea
Tel: +82-32-860-7447, Fax: +82-32-875-5863, E-mail : gsjo@inha.ac.kr
저 자 소 개

Priska Aprilia
She received a B.S degree of Computer Science and Mathematics from Binus University, Jakarta, Indonesia, in 2012. Now, she is pursuing a M.S degree in Computer Science and Information Engineering in Inha University, Korea and is expected to graduate in February 2016. Her research interests include Recommendation Systems, Planning and Data Mining.

Kyeong-Jin Oh
Received a B.S. degree Computer and Information Engineering from Inha University, Korea, in 2006, and a M.S. degree in Information Engineering from Inha University, Korea in 2008. He is a Ph.D. Candidate in Information Engineering of Inha University, Korea. His research interests include Data Mining and Semantic Web.

Myung-Duk Hong
Received a B.S. degree Computer Science, Seoul Digital University, Korea, in 2008. And a M.S. degree in Information Engineering from Inha University, Korea, in 2011. He worked for Kuwoo Information Technology as a researcher 2005 to 2008. He is currently a Ph.D. Candidate in Information Engineering of Inha University, Korea. His research interests include Recommender System, Semantic Web, Ant Colony Optimization and Meta-Heuristic.

Myeong-Hyeon Ga
Received a M.S. degree Computer and Information Engineering from Inha University, Korea in 2014. He is currently Ph.D. Candidate in Information Engineering of Inha University, Korea. His research interests include Recommender System and Data
Geun-Sik Jo
Is a Professor in Computer and Information Engineering, Inha University, Korea. He received the B.S. degree in Computer Science from Inha University in 1982. He received the M.S. and the Ph.D. degrees in Computer Science from City University of New York in 1985 and 1991, respectively. He has been the General Chair and/or Technical Program Chair of more than 20 international conferences and workshops on artificial intelligence, knowledge management, and semantic applications. His research interests include knowledge-based scheduling, ontology, semantic Web, intelligent E-Commerce, constraint-directed scheduling, knowledge-based systems, decision support systems, and intelligent agents. He has authored and coauthored five books and more than 300 publications.