Personalized Tour Recommendation using Location-based Social Media

Kwan Hui Lim

ORCID: 0000-0002-4569-0901

Submitted in total fulfilment of the requirements of the degree of

Doctor of Philosophy

School of Computing and Information Systems
THE UNIVERSITY OF MELBOURNE, AUSTRALIA

June 2017

Abstract

Tourism is a popular leisure activity and an important industry, where the main task involves visiting unfamiliar Places-of-Interest (POI) in foreign cities. Recommending POIs and tour planning are challenging and time-consuming tasks for tourists due to: (i) the need to identify and recommend captivating POIs in an unfamiliar city; (ii) having to schedule POI visits as a connected itinerary that satisfies trip constraints such as starting/ending near a specific location (e.g., the tourist's hotel) and completing the itinerary within a limited touring duration; and (iii) having to satisfy the diverse interest preferences of each unique tourist. While tourism-related information can be obtained from the Internet, travel guides and tour agencies, many of these resources simply recommend individual POIs or popular itineraries, but otherwise do not appeal to the interest preferences of users or adhere to their trip constraints.

In contrast to existing works on next-POI prediction and top-k POI recommendation that recommend a single POI or a ranked list of POIs, the task of tour recommendation involves the need to identify a set of interesting POIs and schedule them as an itinerary with various time and space constraints. While there are works on path planning that recommend an itinerary, this itinerary is typically optimized based on a global utility such as POI popularity, and thus offer no personalization for a tourist based on his/her interest preferences. This thesis addresses the challenges associated with the automation and personalization of tour recommendation using data mining techniques to model user interest and POI-related information, and using optimization problems and techniques to formulate and solve more realistic tour recommendation problems. Our main contributions include:

1. Proposing and implementing a framework that utilizes Flickr geo-tagged photos and Wikipedia to automatically determine user trajectories, interest preferences and POI-related information such as POI popularity and visiting times.

2. Proposing the PERSTOUR algorithm for recommending personalized tour itineraries based on POI popularity, users' interest preferences and trip constraints, where POI visit durations are customized based on user interests.

3. Formulating the QUEUETOURREC problem for recommending queue-aware and personalized itineraries that schedule visits to popular and interesting POIs at times with minimal queuing times, and proposing a novel implementation of Monte Carlo Tree Search to solve this problem.

4. Developing the TOURRECINT algorithm for tour recommendation based on a variant of the Orienteering problem with a mandatory POI category, which is defined as the POI category that a tourist has most frequently visited.

5. Formulating and solving the novel GROUPTOURREC problem, which involves recommending tour itineraries to groups of tourists with diverse interests and assigning tour guides with the right expertise to lead each tour group.

6. Illustrating the application of our proposed approach in practice, by presenting a web-based system implementation of our PERSTOUR algorithm, with the front-end component developed using HTML, PHP, jQuery and the Google Maps API, and the back-end based on Python, Java and PHP.

Declaration

This is to certify that

1. The thesis comprises only my original work towards the PhD,
2. Due acknowledgement has been made in the text to all other material used,
3. The thesis is less than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Kwan Hui Lim, June 2017

Acknowledgements

First of all, I would like to thank my supervisors Shanika Karunasekera, Christopher Leckie and Jeffrey Chan for their supervision and guidance during the course of my PhD candidature, and for being supportive of my early PhD completion. In particular, I am grateful for the useful advice they have given and also for providing the freedom to explore various research directions and collaborations. I also thank Benjamin Rubinstein for being part of my advisory committee and his helpful insights into research.

Outside of my PhD, I am happy to have worked on many interesting projects with various collaborators, including Kate E. Lee, Lida Rashidi, Elham Naghizade, Dave Kendal, Stephan Winter, Maria Vasardani, Ee-Peng Lim, Palakorn Achananuparp, Binyan Jiang, Amitava Datta, Xiaoting (Michael) Wang, Lianhua Chi, Nebula Alam, Christopher J. Butler, Youhan Xia, Yi Mei, Kendall Taylor, Tanya Srinidhi, Yehui (Ivan) Chen, Cheong Loong (Alex) Tan, Fu Quan Gao, Teh Ken Wee, Jagdip Singh, Detelina Marinova, Joshua O. Egbuka, Yiwen Chen, Eeshan Srivastava, Elisabeth Smith, and others. I also thank Shanika Karunasekera, Christopher Leckie and Sarah Erfani for the numerous teaching opportunities for Artificial Intelligence and Object Oriented Software Development; Kate E. Lee and MNSI for the research assistantship opportunity; Christopher J. Butler and IBM Research Australia for the research internship opportunity; Jagdip Singh and Detelina Marinova for hosting my research visit at Case Western Reserve University.

I would also like to acknowledge the financial support of the Australian Government, University of Melbourne, School of Computing and Information Systems, NICTA, Data61, and Google Inc. in the form of a Melbourne International Research Scholarship, Melbourne International Fee Remission Scholarship, NICTA Top-Up Scholarship, and Google PhD Fellowship in Machine Learning. I also thank ACM, SIGMOD, IJCAI-AIJ, Google Australia, ICAPS, SIGWEB, UMAP and SIGIR for the numerous grants of conference travel funding.

Last but not least, I am most grateful to my family and relatives for their endless support and love throughout my life. In particular, I would like to express my deepest gratitude to my parents for their upbringing all these years, and my grandma, Auntie Shirley and sister for always being there for me. My greatest accomplishment during this PhD is having Lik Yin say “yes” to being my better half, thank you for all of your unwavering love and support throughout these years, and for being such an important part of my life. Without all of you, this thesis and my achievements would not have been possible.

Preface

This thesis has been written at the School of Computing and Information Systems, The University of Melbourne, Australia. The following is the list of publications that have arisen from this thesis. The abstract of each chapter indicates the corresponding publications arising from that chapter.

1. **Kwan Hui Lim**, Jeffrey Chan, Shanika Karunasekera and Christopher Leckie. Personalized Itinerary Recommendation with Queuing Time Awareness. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'17)*. Accepted to appear (10pp). Aug 2017

2. **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Personalized Trip Recommendation for Tourists based on User Interests, Points of Interest Visit Durations and Visit Recency. *Knowledge and Information Systems*. Accepted to appear (32pp). Springer. Mar 2017

3. **Kwan Hui Lim**, Xiaoting Wang, Jeffrey Chan, Shanika Karunasekera, Christopher Leckie, Yehui Chen, Cheong Loong Tan, Fu Quan Gao and Teh Ken Wee. PersTour: A Personalized Tour Recommendation and Planning System. *Extended Proceedings of the 27th ACM Conference on Hypertext and Social Media (HT'16), Demonstration Track*. Jul 2016.

4. **Kwan Hui Lim**. Personalized Recommendation of Travel Itineraries based on Tourist Interests and Preferences. *Extended Proceedings of the 24th Conference on User Modeling, Adaptation and Personalization (UMAP'16), Doctoral Consortium*. Jul 2016.

5. **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Towards Next Generation Touring: Personalized Group Tours. *Proceedings of the 26th International Conference on Automated Planning and Scheduling (ICAPS'16)*. pp 412-420. Jun 2016.

6. **Kwan Hui Lim**. Recommending and Planning Trip Itineraries for Individual Travellers and Groups of Tourists. *Proceedings of the 26th International Conference on Automated Planning and Scheduling (ICAPS'16), Doctoral Consortium*. pp 115-120. Jun 2016.

7. **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Personalized Tour Recommendation based on User Interests and Points of Interest Visit Durations. *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*. pp 1778-1784. Jul 2015.

8. **Kwan Hui Lim**. Recommending Tours and Places-of-Interest based on User Interests from Geo-tagged Photos. *Proceedings of the 2015 SIGMOD PhD Symposium (SIGMOD'15)*. pp 33-38. May 2015.

9. **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Detecting Location-centric Communities using Social-Spatial Links with Temporal Constraints. *Proceedings of the 37th European Conference on Information Retrieval (ECIR'15)*. pp 489-494. Mar 2015.

10. **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Improving Location Prediction using a Social Historical Model with Strict Recency Context. *Proceedings of the 5th Workshop on Context-awareness in Retrieval and Recommendation (CaRR'15), in-conjunction with the 37th European Conference on Information Retrieval (ECIR'15)*. Mar 2015.

During the course of my PhD candidature, I also had the opportunity to be involved in various research projects that resulted in the following publications. These publications are cited in the thesis where relevant, but otherwise do not constitute a major part of the thesis.

11. Lianhua Chi, **Kwan Hui Lim**, Nebula Alam and Christopher J. Butler. Geolocation Prediction in Twitter Using Location Indicative Words and Textual Features. *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT'16), in-conjunction with the 26th International Conference on Computational Linguistics (COLING'16)*. pp 227-234. Dec 2016

12. Xiaoting Wang, Christopher Leckie, Jeffery Chan, **Kwan Hui Lim** and Tharshan Vaithianathan. Improving Personalized Trip Recommendation to Avoid Crowds Using Pedestrian Sensor Data. *Proceedings of the 25th ACM International Conference on Information and Knowledge Management (CIKM'16)*. pp 25-34. Oct 2016.

13. **Kwan Hui Lim**, Ee-Peng Lim, Binyan Jiang and Palakorn Achananuparp. Using Online Controlled Experiments to Examine Authority Effects on User Behavior in Email Campaigns. *Proceedings of the 27th ACM Conference on Hypertext and Social Media (HT'16)*. pp 255-260. Jul 2016.

14. **Kwan Hui Lim** and Amitava Datta. An Interaction-based Approach to Detecting Highly Interactive Twitter Communities using Tweeting Links. *Web Intelligence*. Volume 14, Number 1, pp 1-15. IOS Press. Feb 2016.

15. **Kwan Hui Lim**, Ee-Peng Lim, Binyan Jiang and Palakorn Achananuparp. Online Experiments of Authority Effects on User Behavior in Email Campaigns. *Proceedings of the 2015 International Conference on Computational Social Science (IC2S2'15)*. 2pp. Jun 2015.

To my family

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Motivation

Tourism is a popular leisure activity involving more than 1.18 billion international overnight tourists [138] who have stayed for multiple days at their holiday destination. International tourism has also enjoyed sustained growth with international tourist arrivals increasing at an average rate of more than 4% per annum, as shown in Figure 1.1. Furthermore, these figures exclude the large number of domestic overnight tourists that travel between states or cities in their respective home countries. For example, there are more than 21.7 million domestic overnight tourists per annum, solely for the state of Victoria in Australia [143]. In addition, there are also many more international and domestic tourists who embark on day trips to other countries and within their countries, respectively. Tourism is also an important industry that generated a revenue of US$7.2 trillion (or 9.8% of the world's Gross Domestic Product), and accounted for 284 million jobs (or 9% of all jobs in the world) [147].

An important challenge for sustaining this growth is outsourcing the task of tour planning. Tour planning is a challenging and time-consuming task as tourists need to select captivating Places-of-Interest (POIs) to visit in unfamiliar cities and then plan these POI visits as a connected itinerary. Planning this itinerary is especially complicated as tourists are often restricted by trip constraints such as starting/ending near a specific location (e.g., the tourist's hotel) and completing the itinerary within the available and

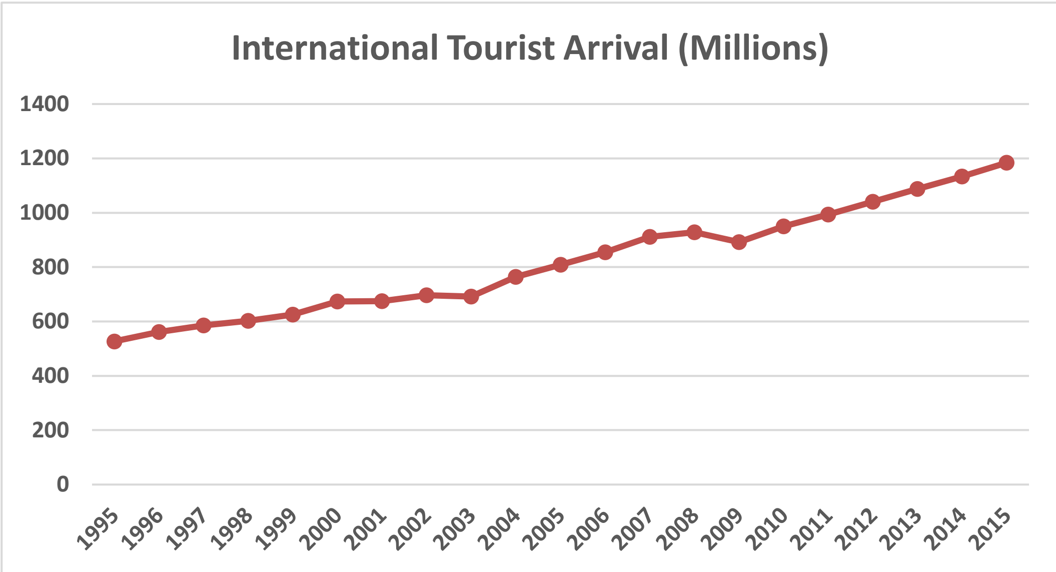


Figure 1.1: Number of international tourist arrivals from 1995 to 2015, based on [138].

often limited touring duration. Furthermore, tourists are often not familiar with the numerous POIs in a city and typically prefer personalized POI recommendations that are based on their unique interest preferences, which adds to the complexity of tour planning. While a tourist is able to engage the services of tour agencies, these tour agencies typically offer standard package tours, which may not cater to the interest preferences of all tourists. In addition, tour agencies offer these tours to groups of tourists and face additional challenges such as recommending POIs that satisfy the tour group as a whole, and assigning tour guides with the right expertise to lead each tour group.

The fields of next-location prediction, top-k location recommendation and travel package/region recommendation are also closely related to our tour itinerary recommendation problem. However, these fields aim to recommend either a single POI, a ranked list of POIs or a set of relevant POIs, but otherwise do not schedule POI visits in the form an itinerary that adheres to various trip constraints. While path planning is related to tour itinerary recommendation, the former aims to find the shortest path or the path with the highest global reward (e.g., POI popularity), without considering the interest preferences of tourists or trip constraints such as a limited touring time. With the prevalence and popularity of location-based social media, there is now an opportunity to address the

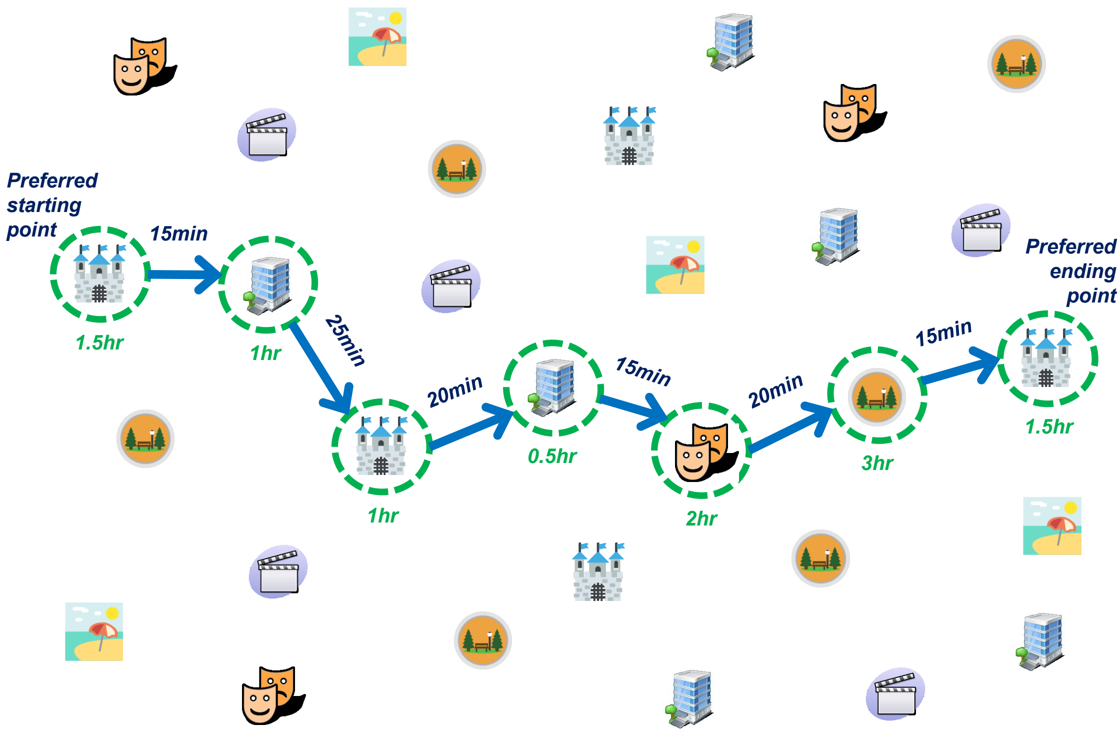


Figure 1.2: Example of the tour recommendation problem.

challenges of tour itinerary recommendation using novel ways to combine data mining techniques and optimization problems.

Figure 1.2 shows a motivating example of the tour recommendation problem, where there are multiple POIs of different categories, and the tourist has to find an itinerary that optimizes the time taken and number of POIs visited, while satisfying various trip constraints. In contrast to works on next-location prediction, top-k location recommendation and travel package/region recommendation, there is the added complexity of recommending a set of POIs based on user interests, and scheduling visits to each POI as an itinerary that can be completed within a certain touring time, considering POI visit durations and travelling times between POIs. Unlike the general path planning problem, this tour recommendation problem aims to model the interest preferences of tourists and recommend POIs based on these interest preferences, resulting in a personalized tour itinerary for each unique tourist.

The main purpose of our research is to automate the trip planning task and improve the quality of tour recommendations for tourists, using both data mining and optimization techniques. First, we introduce and formulate tour recommendation problems that incorporate real-life considerations such as user interests, POI popularity, groups of tourists, and queuing times at attractions, based on variants of optimization-based problems such as the Orienteering problem [136, 142]. Second, we investigate how data mining approaches can be used to automatically determine POI-related statistics and tourist interest preferences based on their past visits, which we infer using the large volume of location-based media that is publicly available. Finally, we propose new tour recommendation algorithms for recommending and planning visits to interesting POIs as a personalized itinerary. Collectively, these studies enable us to better understand the way tourists behave in terms of their travelling group and POI visiting patterns, and allow us to provide a more accurate and personalized tour recommendation service, thus improving their overall touring experience.

1.2 Research Questions and Objectives

Our main aim is to improve the tourist experience by personalizing tour recommendations and automating the task of trip itinerary planning. To achieve this aim, we devise data mining techniques for modelling user interests, and propose novel optimization techniques for planning personalized tour itineraries. Specifically, our key research questions (RQ) and objectives are as follows:

- RQ1: How can we model POI-related information and user interest preferences in a non-intrusive way without explicit input from tourists?
  - Objective: We propose data mining techniques to automatically determine and model tourist interest preferences and other POI-related statistics (e.g., popularity and queuing times) in a fine-grained manner, based on information about their past visits that we derive from location-based media such as geotagged photos.
- RQ2: How can we better personalize tour itinerary recommendations by consider-

ing a more fine-grained measure of user interest and other trip constraints?

  - Objective: We develop personalized tour recommendation algorithms for individual tourists, considering their interest preferences and trip constraints such as starting/ending near specific locations, a limited tour duration, must-visit POI categories, to recommend personalized POIs and visit durations as a connected itinerary.

- RQ3: How can we recommend and schedule itineraries that are personalized to a tourist's interest preferences, as well as conflicting objectives such as POI popularity and POI queuing times?

  - Objective: We develop tour planning algorithms for individual attractions (e.g., theme parks and museums) that appeal to the visitor as a whole, while addressing conflicting objectives such as maximizing POI popularity and minimizing queuing times.

- RQ4: How can we recommend personalized group tours that best satisfy all members of a tour group?

  - Objective: We develop algorithms for recommending customized tour itineraries to groups of tourists that are formed based on their diverse interest preferences, while recommending POIs that satisfy the interest preferences of multiple tourists within the group and assigning tour guides to lead each tour group based on their expertise.

1.3 Contributions of this Thesis

In this thesis, we make the following contributions:

- We perform a comprehensive survey of related work in the area of tour recommendation and introduce a taxonomy to classify the various types of tour recommendation works. (Chapter 2)

- We develop a data mining framework that utilizes Flickr geo-tagged photos [133, 150,151] and Wikipedia to automatically determine the interest preferences of users and derive various POI-related information such as POI popularity, visiting times and queuing times. Based on this framework, we are also able to construct the real-life trajectories of users, including their time of visit and visit durations. (Chapters 3)

- We propose the PERSTOUR algorithm for recommending personalized tour itineraries, with POIs being recommended based on POI popularity, users' interest preferences and trip constraints, where POI visit durations are customized based on user interests. In contrast to earlier works [22, 24, 141, 148], this work is the first to introduce *time-based user interest* with a recency updating mechanism, whereas earlier works either use frequency-based user interest or require users to explicitly state their interests. (Chapter 4)

- We formulate the QUEUETOURREC problem for recommending queue-aware and personalized itineraries that aim to schedule visits to popular and interesting POIs but at a time when the queuing times at these POIs are minimized. To solve the QUEUETOURREC problem, we propose a novel implementation of Monte Carlo Tree Search [27, 45], which out-performs various state-of-the-art baselines in terms of various queuing-time related metrics, itinerary popularity, user interest alignment, recall, precision and F1-score. (Chapter 5)

- We introduce the problem of tour itinerary recommendation with mandatory or "must-visit" POI categories, and propose the TOURRECINT algorithm, based on Integer Linear Programming, where the mandatory POI category is defined as the POI category that a tourist has most frequently visited based on his/her other travel sequences. (Chapter 6)

- We introduce and solve the novel GROUPTOURREC problem, which involves recommending tours to groups of tourists with diverse interests and assigning tour guides with the right expertise to lead each tour group, along with recommending a suitable tour itinerary. Compared to earlier works that study individual parts of this problem [4,40,57,72], our work is the first to investigate group tour recommendations as a holistic problem involving tourist grouping, POI recommendation and tour guide assignment. (Chapter 7)

- We develop the PERSTOURRP system, which is an online-based system for desktops, tablets and mobile phones, that tourists can use for recommending and planning personalized tour itineraries based on their indicated interest preferences and various temporal and spatial constraints. (Chapter 8)

1.4 Structure and Organization of Thesis

The rest of this thesis is structured as follows:

- Chapter 2 presents a comprehensive literature review of tour recommendation research and introduces a taxonomy of this research area.

- Chapter 3 introduces the key notation used in our work, provides background on the generic tour recommendation problem, which is modelled from a variant of the Orienteering problem. We also describe our data mining framework for extracting POI-related information, user interests and trajectories from geo-tagged photos.

- Chapter 4 proposes the PERSTOUR algorithm for recommending personalized tours using POI popularity and user interest preferences, with a customized POI visit duration based on user interest levels. We utilize the timestamps of geo-tagged photos to automatically determine user interest levels and perform a weighted updating of user interests based on the recency of their POI visits. This chapter is derived from the following publications:

  – **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Personalized Trip Recommendation for Tourists based on User Interests, Points of Interest Visit Durations and Visit Recency. *Knowledge and Information Systems*. Accepted to appear (32pp). Springer. Mar 2017

  - **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Personalized Tour Recommendation based on User Interests and Points of Interest Visit Durations. *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*. pp 1778-1784. Jul 2015.

- Chapter 5 proposes the PERSQ algorithm for recommending queue-aware itineraries that take into consideration attraction popularity, user interests and queuing times. Geo-tagged photos are used to automatically determine the queuing times according to the times of POI visits. This chapter is derived from the following publication:

  - **Kwan Hui Lim**, Jeffrey Chan, Shanika Karunasekera and Christopher Leckie. Personalized Itinerary Recommendation with Queuing Time Awareness. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'17)*. Accepted to appear (10pp). Aug 2017.

- Chapter 6 introduces the problem of recommending tour itineraries with a mandatory POI visit category, where this mandatory POI category is based on the POI category that the user has most frequently visited in his/her other travel sequences. This chapter is derived from the following publication:

  - **Kwan Hui Lim**. Recommending Tours and Places-of-Interest based on User Interests from Geo-tagged Photos. *Proceedings of the 2015 SIGMOD PhD Symposium (SIGMOD'15)*. pp 33-38. May 2015.

- In contrast to previous chapters that focused on itinerary planning for individuals, Chapter 7 introduces the novel GROUPTOURREC problem of recommending tours to groups of tourists with diverse interests and assigning tour guides with the right expertise to lead these tours, which we solve by decomposing GROUPTOURREC into a series of more manageable sub-problems, comprising tourist grouping, POI recommendation and tour guide assignment. This chapter is derived from the following publication:

  - **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Towards Next Generation Touring: Personalized Group Tours. *Proceedings of the 26th International Conference on Automated Planning and Scheduling (ICAPS'16)*. pp 412-420. Jun 2016.

- Chapter 8 illustrates how our approach can be implemented in practice, by presenting a web-based system implementation of our PERSTOUR algorithm, with the front-end component developed using HTML, PHP, jQuery and the Google Maps API, and the back-end based on Python, Java and PHP. This chapter is derived from the following publication:

  - **Kwan Hui Lim**, Xiaoting Wang, Jeffrey Chan, Shanika Karunasekera, Christopher Leckie, Yehui Chen, Cheong Loong Tan, Fu Quan Gao and Teh Ken Wee. PersTour: A Personalized Tour Recommendation and Planning System. *Extended Proceedings of the 27th ACM Conference on Hypertext and Social Media (HT'16), Demonstration Track*. Jul 2016.

- Chapter 9 concludes this thesis by summarizing our key contributions and highlighting various directions for future work, substantiated by some preliminary experiments and results. The appendices referenced in this chapter are derived from the following publications:

  - **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Detecting Location-centric Communities using Social-Spatial Links with Temporal Constraints. *Proceedings of the 37th European Conference on Information Retrieval (ECIR'15)*. pp 489-494. Mar 2015.
  - **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Improving Location Prediction using a Social Historical Model with Strict Recency Context. *Proceedings of the 5th Workshop on Context-awareness in Retrieval and Recommendation (CaRR'15), in-conjunction with the 37th European Conference on Information Retrieval (ECIR'15)*. Mar 2015.

Chapter 2

Literature Review

In this chapter, we conduct a comprehensive literature review of studies on tour itinerary recommendation and present a general taxonomy for touring-related research. In addition to reviewing the problems and algorithms proposed in these works, we also discuss the various types of datasets and evaluation methodologies that are typically employed in tour itinerary recommendation research.

2.1 Introduction

Economically, tourism is an important industry, generating more than 284 million jobs and accounting for more than US$7.2 trillion in revenue [147]. Despite its importance and popularity, planning tour itinerary in a foreign city is both challenging and time-consuming due to the need to identify captivating Places-of-Interest (POIs) and plan visits to these POIs as a connected itinerary. Adding to these challenges are the need to personalize the recommended itinerary according to the interest preferences of tourists and to schedule the itinerary based on relevant temporal and spatial constraints, such as having limited time to complete the tour and needing to start and end near certain locations (e.g., the tourist's hotel).

Although tourism-related information can be obtained from the Internet and travel guides, these resources simply recommend popular POIs or generic itineraries but otherwise do not address the unique interest preferences of individual tourists or adhere to their various temporal and spatial constraints. Moreover, the large amount of information available increases the challenge of identifying the relevant information for the tourist. One popular alternative is to engage the services of tour agencies, but likewise, these tour agencies only recommend standard package tours that may not address the interest preferences or trip constraints of all tourists.

To address these issues, many researchers have studied tour itinerary recommendation problems and proposed various algorithms for solving these problems. These problems have their origins in the Operations Research community where the main focus is to schedule an optimal path, where the measure of optimality is typically based on a global metric such as POI popularity, and thus there is no personalization based on unique user interests. With the prevalence of smart phones and location-based social media (e.g., geo-tagged photos), there has been an increased emphasis on data-driven approaches to tour itinerary recommendation to better model the interest preferences of tourists, and recommend personalized tour itineraries that satisfy these interest preferences as well as other trip constraints. In this chapter, we focus on such data-driven tour recommendation works, particularly on the types of data sources used, the problem variants formulated, the algorithms proposed and the evaluation methodology used.

Closely related to the field of tour itinerary recommendation are the adjacent fields of next-location prediction/recommendation [10, 55, 94, 101], top-k location recommendation [88, 93, 153, 154, 158] and travel package/region recommendation [11, 12, 111]. Although these adjacent fields are related to tour itinerary recommendation, there are distinct differences in terms of the problem studied. Next-location prediction and recommendation aim to identify a next location that a user will visit based on his/her previous trajectory, whereas tour itinerary recommendation aims to recommend multiple POIs or locations in the form of a trajectory. Top-k location recommendation and travel package/region recommendation do fulfill the criterion of recommending multiple POIs as part of a ranked list or travel package, but they do not structure these POIs as a connected itinerary. In contrast, tour itinerary recommendation has the additional challenges of planning an itinerary of connected POIs that appeal to the interest preferences of the users, while adhering to the temporal and spatial constraints in the form of a limited time budget for touring and having to start and end at specific POIs (possibly near the tourist's hotel).

In this chapter, we focus our review on works related to tour itinerary recommendation and the different real-life considerations incorporated into this problem. Figure 2.1

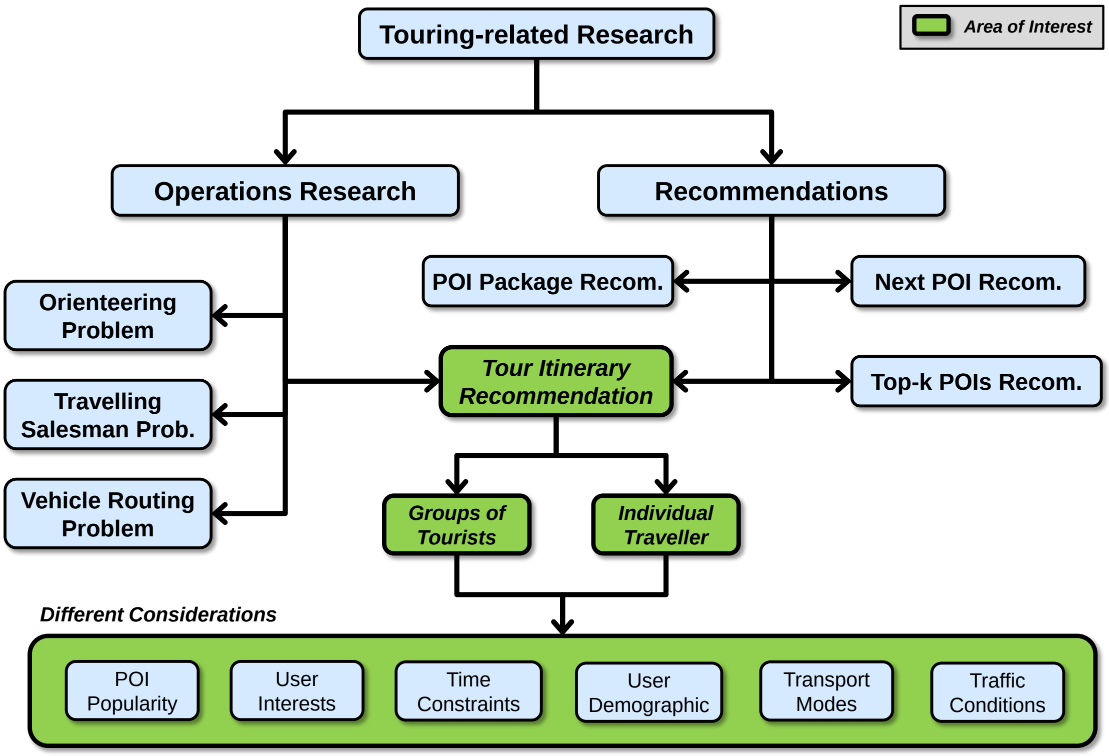


Figure 2.1: Taxonomy of Touring-related Research.

illustrates a taxonomy of the general area of touring-related research, which is further divided into the sub-areas of Operations Research and Recommendations.

2.1.1 Related Surveys and Reviews

There exist a variety of survey and review articles that cover different aspects of the tour recommendation problem. In this section, we aim to discuss these related articles and highlight the difference between this chapter and the earlier articles.

The tour recommendation problem is closely related to the tourist trip design problem covered in the operations research community, and consequently, there have been various survey papers [60, 126] focusing on the aspects of problem formulation, algorithmic design and complexity of this problem. Similarly, many tour recommendation problems are based on variants of the Orienteering problem, and [65, 142] provide in-depth discussions on the Orienteering problem. Researchers such as [18] performed a review of tour recommendation systems, focusing on applications and systems aspects such as the types of interface, the system functionalities, the recommendation techniques and the artificial intelligence methods used. Others studied recommendations in general on location-based social networks [8] and the general types of research utilizing Flickr photos [128], with a small portion of their survey covering tourism-related applications. While these articles offer interesting discussions into different aspects of tour recommendation, this chapter differs from the earlier articles in the following ways: (i) first, we review tour recommendation research as a holistic problem, covering the whole process from data collection, data pre-processing, tour itinerary recommendation, experimentation and evaluation; (ii) second, this chapter provides an updated review of the current state-of-the-art in tour recommendation research (as of 2017).

2.1.2 Structure and Organization

The rest of the chapter is structured as follows: Section 2.2 discusses the different sources and methods for obtaining tourist visit data; Section 2.3 describes various tour recommendation algorithms targetted at the individual traveller; Section 2.4 examines the problem of recommending tours to groups of tourists and examines various algorithms and applications that aim to fulfill this task; Section 2.5 studies the various methodologies that can be used to evaluate the performance of tour recommendation algorithms; and Section 2.6 summarizes and concludes this chapter.

2.2 Methods for Retrieving Tourist Visit Data

In all tour recommendation works, one of the initial steps is to identify an appropriate data source that is representative of tourist real-life trajectories. This dataset is mainly used to evaluate the proposed tour recommendation algorithms and typical sources are based on geo-tagged photos or other social media, Location-based Social Networks, or GPS-trajectory traces. In this section, we discuss these three types of datasets with a main focus on geo-tagged photos, which is also the most prevalent data source used in tour recommendation works.

2.2.1 Mining of Tourist Trajectories using Geo-tagged Photos

The paper by Choudhury et al. [40] was one of the earliest works to study both itinerary recommendation using an optimization-based approach and mining of users' past trajectories based on geo-tagged photos (Figure 2.2). In this section, we focus on the mining of users' past trajectories, while the optimization-based approach to itinerary recommendation is covered later in Section 2.3.1. Using geo-tagged Flickr photos, they construct these past trajectories using the following steps:

1. *Constructing an Ordered Sequence of Relevant Photos*. The entire set of photos are first filtered to remove those that are: (i) not taken in the specific city; (ii) taken in the specific city but not by a tourist; and (iii) stamped with an accurate taken time. This filtering step is performed by only selecting photos of a user that fulfill the following conditions: (i) photos are tagged with a keyword that is relevant to the city's name, e.g., NYC, Manhattan, New York City are considered the same city; (ii) the first and last photos taken are less than N days apart, where $N = 21$, as tourists are more likely to take photos within a shorter time-frame compared to residents; and (iii) if the minutes and seconds of the taken time and upload time differ, as this difference indicates that the photo was not auto-uploaded by Flickr. The remaining photos are then ordered in a temporal sequence.

2. *Mapping Photos to Popular POIs*. The authors then use Lonely Planet guides to obtain a list of popular POIs and determine the latitude/longitude coordinates of these POIs using the Yahoo! Map API. Thereafter, the photos are mapped to a POI if either: (i) the latitude/longitude coordinates of the photo and POI differ by $\leq 100m$; or (ii) the trigram set similarity between the photo tags and POI name is above a threshold of 0.3.

3. *Generating Timed Sequences of POI Visits*. Using the Photo-POI mapping in Step 2, the authors then determine the POI visit duration and POI-to-POI travel duration based on these photos. POI visit duration is based on the first and last photo taken at a specific POI, while POI-to-POI travel duration is based on the time difference between two consecutive but different POI visits. In addition, the sequence of POI

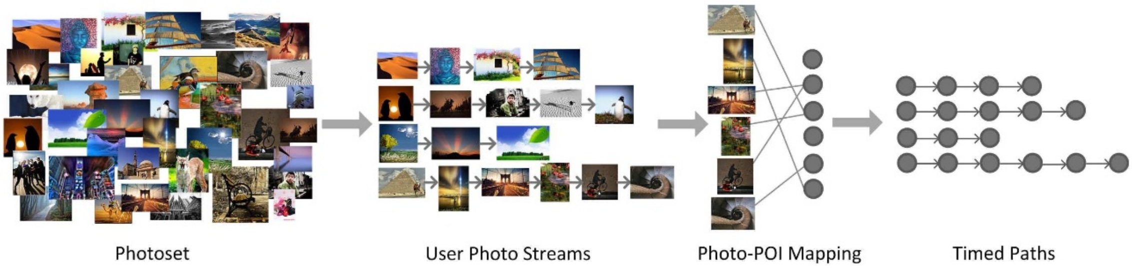


Figure 2.2: Construction of Tourist Trajectories from Geo-tagged Photos. Retrieved from [40].

visits are divided into smaller sub-sequences if two consecutive photos are taken more than 8 hrs apart.

Other authors such as [22,24,86,87,105,107] also adopted variations of this approach in their tour recommendation works. Similarly, this approach can be easily adapted to other forms of social media with a geo-tagged location such as Tweets or Facebook posts. Apart from mining tourist trajectories, many authors further refine this trajectory mining by assigning categories to POIs and using these POI categories to determine the interests of tourists. Many of these approaches are discussed later in Section 2.3.2.

2.2.2 Mining of Tourist Visits on Location-based Social Networks

Similar to the previous section, another source for obtaining tourist trajectories or visits is from Location-based Social Networks (LBSNs) such as FourSquare or JiePang. On such LBSN, users are able to follow one another and form friendship links and more importantly, they are able to explicitly check-in to locations or venues that they have visited. These check-in locations include POIs, restaurants, businesses or general venues and are further divided into categories of Food, Coffee, Nightlife, Fun and Shopping. LBSNs provide another popular source of dataset and have been used by numerous researchers in their tour recommendation and path planning problems [62,85,156,159,160].

2.2.3 GPS-based Trajectory Traces

GPS-based trajectory traces are another popular data source commonly used for tour recommendation and path planning problems [37, 155, 162, 164]. These traces are typically recorded based on GPS-enabled devices such as today's smart phones and dedicated GPS trackers. With the advent of smart phones, such GPS-based trajectory traces are increasingly common but privacy issues prevent such datasets from being publicly shared on a large-scale, unlike datasets based on public geo-tagged photos and LBSNs. Despite the restrictiveness of GPS-based trajectory traces, these datasets provide a very detailed record of a user's movement trajectory based on fine-grained GPS locations, in contrast to geo-tagged photos and LBSNs that only record visits to specific POIs or locations.

2.3 Tour Recommendation for Individual Travellers

In this section, we start our tour recommendation review by discussing various optimization-based approaches that do not include any user personalization. Thereafter, we review a series of data-driven tour recommendation approaches that include personalization based on user interests, consideration for traffic conditions, and incorporation of uncertainty in travelling. Table 2.1 presents a broad overview of the various works on tour recommendation for individual travellers.

2.3.1 Optimization-based Approaches (without Personalization)

Tour recommendation has its roots from the Orienteering problem that is frequently studied in the Operations Research community. We provide an overview of the Orienteering problem before discussing works that cover various variants of the Orienteering problem. One key feature of these works is that they do not incorporate any personalization for individual users and thus all users using the same algorithm will be recommended the same tour itinerary, given the same starting/ending POIs and time budget as inputs.

Table 2.1: Survey of Tour Recommendation for Individual Travellers

Paper Reference	Popularity -based	Interest -based	Determines Interest	Constructs Itinerary	Considers Time	Transport/ Traffic
[62]	Yes	Partly	No	Yes	Yes	No
[40]	Yes	No	No	Yes	Yes	No
[16]	No	Yes	No	Yes	Yes	No
[22, 24]	Yes	Yes	Yes	Yes	Yes	No
[141] Yes	Yes	No	Yes	Yes	No	
[38]	Yes	Yes	Yes	Yes	Yes	No
[149]	Yes	Yes	No	Yes	No	No
[106]	Partly	Yes	Yes	Yes	Yes	No
[34]	Yes	Yes	Yes	Yes	Yes	Yes
[86, 87]	Partly	Yes	Yes	Yes	Yes	Yes
[52]	Yes	No	No	Yes	Yes	Yes
[59]	Partly	Yes	No	Yes	Yes	Yes
[159, 160]	Yes	Yes	Yes	Yes	Yes	Yes
[114]	No	No	No	Yes	No	No
[105]	Yes	Partly	Partly	No	No	No
[54]	No	No	No	Yes	Partly	No
[117]	No	Yes	No	Yes	Yes	No
[31]	No	Yes	Partly	Yes	Yes	Yes
[139]	Yes	Yes	No	Yes	Yes	No
[156]	Yes	Yes	Yes	Yes	Yes	No
[74]	Yes	Yes	Yes	Yes	Yes	No

Orienteering Problem

The Orienteering problem originated from a sport of the same name, which involves participants trying to visit check-points, each with a pre-determined score, in an attempt to maximize their total score within a specific time limit. In recent years, many tour recommendation studies have modelled tour recommendation based on the Orienteering problem and its many variants. Similarly, there have been many web applications [139] developed based on variants of the Orienteering problem. We first describe the original Orienteering problem [65, 136, 142] and how it is applied to the field of tour recommendation.

Many tour recommendation works are focused on individual cities, each of which comprises a set of POIs P. For a tourist visiting a particular city, he/she will have considerations of a certain time or distance budget B, and preferred starting and ending POIs p_1 and p_N, respectively. The budget typically represents the amount of time that a tourist would want to spend on a tour or the distance that he/she is willing to travel. Similarly, the starting and ending POIs reflects the preferences of the tourist to start the tour near a particular point (e.g., the tourist's hotel) and end the tour at another point (e.g., near a restaurant for dinner). Thus, given the set of POIs P, a budget B, starting POI $p_1 \in P$, destination POI $p_N \in P$, our main goal is to recommend a tour itinerary that maximizes a particular score, while adhering to the constraints of the budget, starting and destination POIs. We formally define this as recommending a tour itinerary $I = (p_1, ..., p_N)$ that:

$$Max \sum_{i=2}^{N-1} \sum_{j=2}^{N} x_{i,j} Score(i) \tag{2.1}$$

where:

$$x_{i,j} = \begin{cases} 1, & \text{the itinerary involves travelling from POI } i \text{ to } j. \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

such that:

$$\sum_{j=2}^{N} x_{1,j} = \sum_{i=1}^{N-1} x_{i,N} = 1 \tag{2.3}$$

$$\sum_{i=1}^{N-1} x_{i,k} = \sum_{j=2}^{N} x_{k,j} \leq 1, \qquad \forall\, k = 2, ..., N-1 \tag{2.4}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} Cost(i,j) x_{i,j} \leq B \tag{2.5}$$

Equation 2.1 aims to maximize a certain score that is allocated to POIs in the recommended tour itinerary. This score is typically based on POI popularity, POI alignment to user interests, or some variation of the two. To recommend a tour itinerary that starts and ends at specific POIs, Constraint 2.3 ensures the tour starts at POI 1 and ends at POI N. Following which, Constraint 2.4 ensures that the recommendation is a tour itinerary where: (i) all POIs are connected as an entire path; and (ii) no POIs are visited multiple times. Lastly, Constraint 2.5 ensures that all POIs in the tour itinerary can be visited within the budget B, where the function $Cost(p_x, p_y)$ determines the travelling time or distance cost of visiting POI p_x, followed by POI p_y.

Tour Recommendation by Choudhury et al.

Apart from mining tourist trajectories from geo-tagged photos (Section 2.2.1), Choudhury et al. [40] also proposed the Itinerary Mining Problem, which is based on the Orienteering problem. In this Itinerary Mining Problem, they attempt to find an itinerary that maximizes POI popularity while ensuring that the total time taken for transiting and visiting POIs is within a pre-determined budget. In particular, they model POI Popularity based on the visit count by distinct tourists, transit times between POIs based on the median transit time by all tourists, and POI visit times based on the 75th percentile of visit time by all tourists. To solve their Itinerary Mining Problem, they used a recursive greedy algorithm [32], which tries to estimate the middle node of the itinerary and the associated utility (popularity) gained and cost (time) incurred, then resursively calls itself on both halves of the itinerary.

Tour Recommendation with Specific POI Category Sequence

Gionis et al. [62] approached the tour recommendation problem in a similar fashion as the Orienteering problem, with the constraints of a starting and destination POI, and a specific time/distance budget. The main differences between their proposed problem and the Orienteering Problem are: (i) they consider the category of POIs in their tour recommendation, while the Orienteering problem ignores POI category; and (ii) more importantly, Gionis et al. recommend tours with a specific visit order over all POI categories, e.g., Cafe $\rightarrow$ Parks $\rightarrow$ Restaurants $\rightarrow$ Shopping $\rightarrow$ Museum $\rightarrow$ Beach, while the Orienteering problem does not consider this ordering. Apart from this constraint of a POI category visit order, the authors also consider variations of this ordering constraint, such as:

- *Partial Ordering of POI Categories.* Instead of a total ordering of all POI categories, this relaxed constraint allows for a partial ordering of POI categories. For example, Cafe $\rightarrow$ Parks and Shopping $\rightarrow$ Museum are partial orderings of Cafe $\rightarrow$ Parks $\rightarrow$ Restaurants $\rightarrow$ Shopping $\rightarrow$ Museum $\rightarrow$ Beach.

- *Subset Grouping of POI Categories.* Instead of a specific order of individual POI categories, this relaxed constraint allows the user to state the need to visit any one of a subset of POI categories. For example, a visit order would be Cafe *OR* Restaurants $\rightarrow$ Shopping *OR* Museum *OR* Beach *OR* Parks, instead of the whole range of Cafe $\rightarrow$ Parks $\rightarrow$ Restaurants $\rightarrow$ Shopping $\rightarrow$ Museum $\rightarrow$ Beach.

- *Skipping of POI Categories.* Instead of having to visit all POI categories at least once, this relaxed constraint allows for one or more POI categories to be skipped, i.e., not visited in the recommended tour.

The authors proposed two schemes for evaluating the utility obtained from each tour, namely: (i) an *additive satisfaction function* on the perceived benefit from visiting a particular POI, based on either a general measure (e.g., POI popularity) or personalized measure (e.g., personal satisfaction); and (ii) a *coverage satisfaction function* that determines the number of additional, nearby POIs that can be visited during the tour, i.e., POIs within a certain distance from the tour. To solve this tour recommendation problem and the different variations of the relaxed constraints, the authors used a dynamic programming approach.

Tour Recommendation with POI Category Visit Constraints

Typical tour recommendations attempt to plan tour itineraries that comprise POIs belonging to a category that is aligned to a tourist's interest preferences. However, one consequence of this consideration is that the recommended tour may include too many visits to the same POI categories (e.g., visiting 10 museums for a tour), resulting in a "sensory overload". To overcome this issue, Bolzoni et al. [16] proposed the CLuster Itinerary Planning (CLIP) algorithm that aims to recommend tour itineraries with constraints on the maximum number of times that each POI category can be recommended. This problem is also a variant of the Orienteering problem, which the authors named the Orienteering Problem with Maximum Point Categories (OPMPC). The proposed CLIP algorithm makes extensive use of clustering and pruning techniques to reduce the search time required to generate a tour itinerary. The main steps of the CLIP algorithm are as follows:

1. Based on the set of all POIs, CLIP first uses agglomerative clustering to group POIs into k clusters in a bottom-up manner (i.e., merging individual POIs until k clusters are formed). This clustering is performed based on the location of POIs, thus POIs that are nearer to each other are grouped into the same cluster.

2. For each subsequent tour recommendation request, the algorithm then generates a cluster path that starts from a starting POI S, followed by POI cluster C_1, C_2, ..., C_N, before finishing the tour at the destination POI D. The travel cost is defined as the total cost needed to travel from S to the medoid of C_1, then to all subsequent POI clusters until D, with the assumption that the travel costs within a POI cluster are negligible.

3. Based on each selected POI cluster from Step 2, CLIP proceeds to select a subset of individual POIs from each POI cluster, with the aim of maximizing the obtained utility score. This selection of individual POIs is treated a multi-dimensional knapsack problem with the constraints on the maximum number of visits per user-indicated POI category and total time spent visiting these POIs.

2.3.2 Personalization-based Approaches

After discussing the optimization-based approaches to tour recommendation, we next elaborate on various data-driven approaches to tour recommendation that include various forms of personalization based on user interest preferences.

City Trip Planner

This system was proposed by Vansteenwegen et al. [141] for recommending personalized tours in five cities in Flanders, Belgium (Antwerp, Bruges, Ghent, Leuven and Mechlin) based on user-provided interest preferences. The City Trip Planner works in the following steps:

1. *Solicitation of User Constraints.* Using the form shown in Figure 2.3, the tourist first enters any trip constraints that he/she has, such as the number of days for the trip and the preferred starting and ending location for each day. In addition, the tourist is able to specify a duration for a break anytime during each day, e.g., a lunch break from 1-2pm.

2. *Estimation of User Interests in Specific POI.* The interest of each tourist is determined using the form shown in Figure 2.4. The tourist would indicate their interest in each type of POI (e.g., abbeys, churches, castles, etc) and their level of interest (four levels from "totally not interested" to "absolutely interested"). The user interest preferences are then determined based on a combination of: (i) a *type* score based on matching POIs to the user's interest type and interest level; (ii) a similar *category* score based on the category of the POIs (each POI can belong to multiple categories, e.g., architecture, classical art, science, whereas POIs only belong to one type); and (iii) a *keyword search* score that compares how relevant a POI is based on its textual description, using the Vector Space Model [6] that is described further in [127].

3. *Recommendation of Tour.* The exact tour recommendation problem is then modelled as a Team Orienteering Problem with Time Windows [140], where the time windows correspond to the opening hours of POIs. The authors then solve this problem using a Greedy Randomised Adaptive Search Procedure, which results in a recommended tour for the tourist.

4. *Alterations to Proposed Tour.* The recommended tour generated in the previous step is then shown to the tourist, who can either accept the recommended tour or make

Figure 2.3: Examples of forms for soliciting user trip constraints (left) and user interest preferences (right) in the City Trip Planner. Retrieved from [140].

alterations to it by removing specific POIs. This alteration step then takes place until the tourist is satisfied and accepts the recommended tour.

Tour Recommendation based on Gender, Age and Race

In [38], Cheng et al. aim to recommend tours based on the current location of a user and his/her demographic details such as gender, age and race, which are automatically detected from Flickr photos. This tour recommendation takes two forms, both of which utilizes the user's demographic details, namely: (i) given the user's current location, recommend the next POI to visit for this user; and (ii) given a starting POI and destination POI, recommend a set of N POIs that are interesting to the user. The three main steps in their work are as follows:

1. *Determining User Demographics.* The user demographics are determined using Flickr photos and comprises the following steps: (i) a facial detection algorithm [144] is performed on each photo to identify the face, in particular facial parts like eyes, philtrum and mouth, and facial features like color and texture; (ii) a set of facial images is then manually annotated and used as training data for a Support Vector Machine classifier; and (iii) Adaboost is then used to select the best set of features for determining a particular demographic attribute, e.g., gender, age or race.

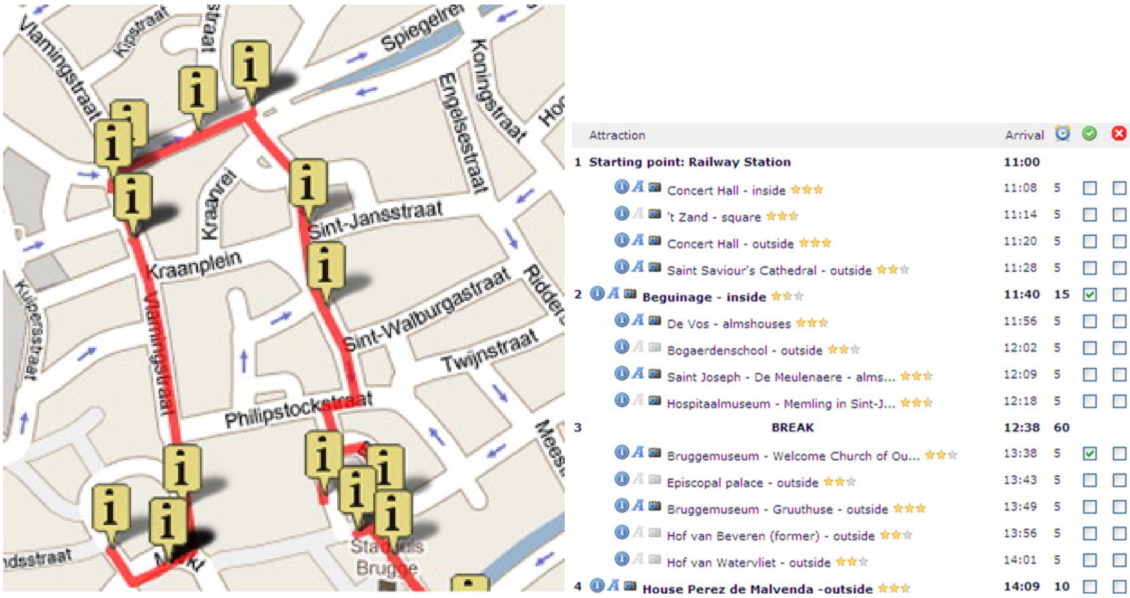


Figure 2.4: Examples of the proposed tour visualized on a map (left) and in the form of a list with options to alter recommended POIs (right) in the City Trip Planner. Retrieved from [140].

2. *Recommending Next POI.* Using the user demographics determined from Step 1, the recommender utilizes a Bayesian learning model that also considers the user's current location and their learned tourist travel model based on travel sequences by other users with similar demographic attributes. However, the learned tourist travel model will not recommend a POI-to-POI travel sequence that has not been taken/observed before, and the authors resolve this issue by adding a smoothing factor to account for unobserved POI-to-POI travel sequences by specific user demographic groups.

3. *Recommending Tour Itinerary.* The tour itinerary recommendation problem is modelled as a variant of the shortest path problem, where the objective is to find the shortest path from a starting POI to destination POI that also covers N other POIs with no repeated POI visits. Each POI is assigned a score based on the POI popularity and how well it aligns to the user-specific demographic profile. While there is no explicit time or distance budget (like in typical Orienteering problems), the authors implemented a penalty function for POI-to-POI path distances thus favouring paths that are shorter.

In a later work [36], the authors extended the work of [38] by also considering the size of the group in which a user is travelling, i.e., individuals, friends, couples or families. They perform this consideration by using facial recognition techniques to detect the number of faces in a photo, thus identifying the number of travellers in a group.

TripBuilder Algorithm

Brilhante et al. [22, 24] developed the TRIPBUILDER algorithm for planning personalized tour itineraries for tourists based on the Generalized Maximum Coverage problem [43]. In this problem, they are interested in planning a tour comprising POIs that maximizes tourists' personal interest while adhering to a specific visiting time budget. The TRIPBUILDER algorithm aims to accomplish this objective in two steps:

1. *Selection of Sub-trajectories.* As part of the TripCover problem, the authors use an approximation algorithm to select a set of sub-trajectories among POIs that best satisfies tourist interests and is within the specified time constraint.

2. *Joining of Sub-trajectories.* As part of the Trajectory Scheduling Problem (TrajSP), the sub-trajectories found in Step 1 are then joined together to form a complete tour itinerary using a local search algorithm.

The TRIPBUILDER algorithm has also been developed as a web-based application with the same name [23].

Aurigo System

Aurigo is a tour recommendation system that aims to recommend personalized itineraries via an End-to-End mode and a Step-by-Step mode [149]. Similar to the Orienteering problem, the End-to-End mode aims to recommend a tour itinerary with specific starting and ending points, while trying to maximize for POI popularity and user interests. In Aurigo, POI popularity is determined based on review counts and ratings given on Yelp, while user interest preferences are explicitly provided by the tourist in the form of 1-5 star ratings given to each POI category, namely Monuments, Museums, Movie filming locations

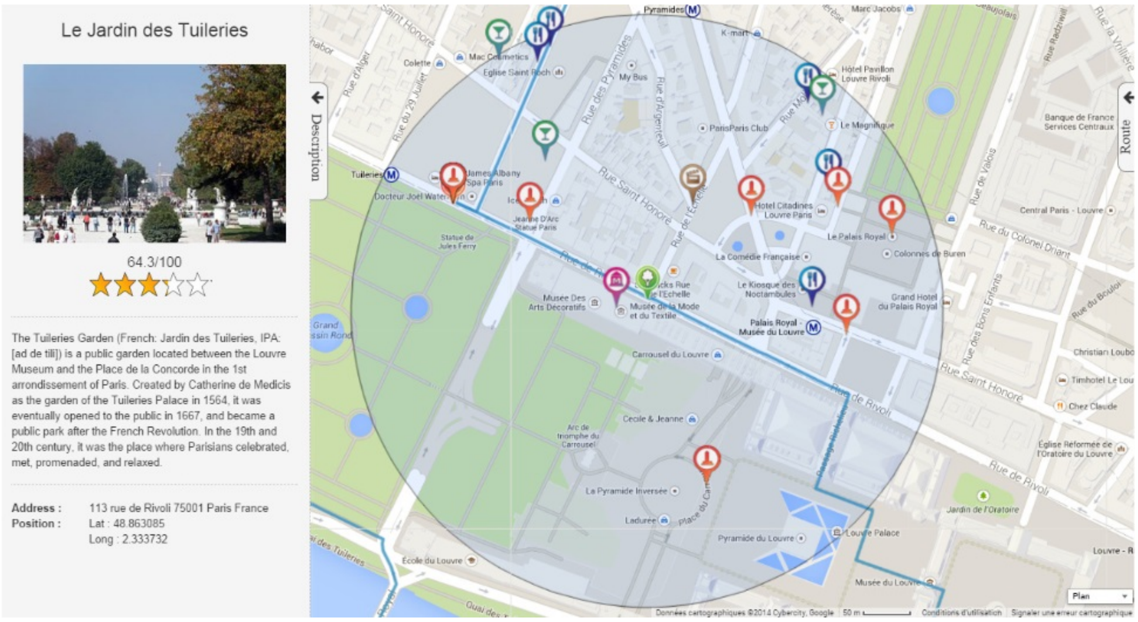


Figure 2.5: Example of an itinerary generated by Aurigo, with the Pop Radius feature (blue circle) that allows tourists to add/delete POIs that are within a certain radius of a selected POI. Retrieved from [149].

and Parks. For the Step-by-Step mode, the tourist first chooses a starting point then iteratively chooses the next POI to visit until he/she is satisfied with the self-constructed itinerary. In both modes, the tourist is able to make changes to the itinerary via the use of a Pop Radius feature (Figure 2.5, which shows all POIs within a specific radius to a selected (clicked) POI. This Pop Radius feature was designed as a lightweight interaction technique for the tourist to fine-tune the recommended itinerary in the End-to-End mode and to build his/her own itinerary in the Step-by-Step mode.

Photo2trip System

Lu et al. [104] proposed a trip planning system that utilizes 20 million geo-tagged photos and 0.2 million travelogues for the main purposes of identifying popular POIs, POI-to-POI path discovery and tour recommendation. More specifically, their trip planning system achieves these functions by:

1. *Identifying popular attractions.* Instead of using a pre-determined list of POIs, Lu et al. used the MeanShift Clustering Algorithm to automatically cluster geo-tagged

photos into clusters based on their location and picked the top 10% largest clusters. Each cluster is then named according to the nearest POI that is listed in the travelogues.

2. *POI-to-POI path discovery.* The authors proposed an Internal Path Discovery algorithm to combine multiple fragments of photo-to-photo paths from different users into a single POI-to-POI path based on the density of the photo fragments and their actual distance. This algorithm aims to overcome the limitation that a single user may not post all photos that cover his/her entire trajectory.

3. *Tour recommendation.* Using the list of POIs (Step 1) and POI-to-POI paths (Step 2), this system then uses a Dynamic programming approach to find an optimal tour that can be completed within a specific time budget. The optimality of a tour is based on the popularity of the POI, and preferences for the POI category and visiting time.

Context-aware Tour Recommendation

Most tour recommendation works based on geo-tagged photos determine POI visits by mapping geo-tagged photos to a list of known POI locations. In contrast, Majid et al. [106] infers the location of POIs and their semantic meaning using clustering approaches on geo-tagged photos. In addition, their approach also infers the popular travel sequences between POIs and considers the context of the tour recommendation, i.e., time, day and weather. In summary, [106] performs their context-aware tour recommendation in the following steps:

1. *Inferring POI Locations.* The authors first use the P-DBSCAN algorithm [79] to cluster geo-tagged photos into clusters of photos, which correspond to POI locations in a city. Thereafter, they assign a semantic meaning to each POI location based on a combination of user tags assigned to the photos and external data from web services such as Google Places.

2. *Mining frequent travel sequences.* Using the list of POI locations, this approach then maps a user's geo-tagged photos to the POIs, thus forming a list of POI visits that

can be combined to form a longer travel sequence. To mine frequent travel patterns, the authors proposed the re-fixSpan algorithm, which is partially based on the Prefix-projected Sequential Pattern Mining algorithm [66] that is used for frequent item-set mining.

3. *Determining Weather Conditions.* As each photo is tagged with a time stamp of when it was taken, the authors are able to associate each photo (POI visit) with the weather conditions when the photo was taken. This weather condition was taken from Wunderground and includes aspects such as the temperature, wind chill, humidity, pressure, wind speed, etc.

4. *POI and Tour Recommendation.* To determine a user interest score, the authors utilized user-based collaborative filtering to rank POIs according to how well a POI satisfies their interest preferences based on their similarities with other users. For a context-aware recommendation, the authors use a joint probability of the likelihood of including a POI in a recommended list, where this joint probability includes the earlier-mentioned interest score and the context in the form of time and weather.

Tour Recommendation with Time-variant Interests

Instead of the Orienteering problem, Yu et al. [156] proposed a tour recommendation problem with a starting POI and limited duration for touring, but with no consideration of a specific destination POI. One key difference between the works of Yu et al. and other interest-based tour recommendation is how Yu et al. proposed the idea of time-variant interest preferences, e.g., visit tourist attractions in the morning and have lunch at a restaurant at noon. This work uses the following steps:

1. *Modelling User Interest Preferences.* To model a user's time-variant interest preferences, the authors first divide a user's interest preferences into six time periods throughout the day, with the exception of midnight to 8am when the user is sleeping. Thereafter, they calculate the interest level of a user in a specific category based on this/her visit frequency to POIs of that category.

2. *Modelling POI Scores.* Yu et al. also assigns a score to each POI based on a combination of POI popularity (based on the number of visits to that POI in a specific month) and the POI rating (as assigned by JiePang users to that POI).

3. *POI Recommendations.* Before constructing the tour itinerary, the next step would be to recommend a list of POIs that are relevant to the user. This step involves the discovery of POIs that are near to the specified starting location and aligned to the user's interest preferences, followed by a ranking of these POIs based on user-based collaborative filtering [161].

4. *Construction of Tour Itinerary.* To construct a tour itinerary, the authors first select a list of top-N POIs [164] for each time period, then construct a tree with the starting POI as the root node and different levels of this tree contain recommended POIs at different time periods. The recommended tour itinerary is then based on a traversal of this tree, where the transition from one node to another is based on a transition probability derived from user interests, POI popularity, remaining touring time and POI-to-POI distances.

Tour Recommendation based on Time and Seasons

Jiang et al. [74] proposed a system to automatically model user interest preferences based on their posted geo-tagged photos and travelogue websites, and recommended tour itineraries are based on popular itineraries that are modified based on user interests. Compared to other tour recommendation system, [74] considers interest preferences, POI admission costs, POI opening hours and the visiting seasons, which they were able to automatically obtain from geo-tagged photos and travelogue websites. Their tour recommendation systems comprises the following steps:

1. *Extracting POI Statistics from Travelogues and Photos.* The authors first employ Natural Language Processing techniques on the tags and textual description of travelogue articles to determine the various topics (categories) associated with POIs, along with additional information such as the admission/visiting cost and opening hours. Using the geo-tagged photos (and associated timestamps), the authors

are also able to determine the visiting distribution at the POIs during the different seasons.

2. *Determining User Interest, Cost, Time and Season Preferences.* Using a user's history of posted geo-tagged photos, the authors are also able to determine the user's interest preferences based on tags associated with their photos, and the Cost, Time and Season Preferences based on the taken timestamps associated with their photos. In addition, a user's posted geo-tagged photos can be used to construct the travel history of this user, i.e., sequence of POI visits.

3. *Tour Itinerary Recommendation.* The recommendation of a tour itinerary to a targetted user works in the following steps: (i) based on the travel history of all users, a list of popular routes are selected based on how well they match the interest, cost, time and season preferences of the targetted user; (ii) using a variant of collaborative filtering [91, 163] based on similar users in terms of interest, cost, time and season preferences instead of co-visits at POIs, the authors then identify a set of top-k POIs to be added to popular routes; (iii) using the popular route identified in Step (i), the k least popular POIs are removed and replaced by the top-k POIs identified in Step (ii), where these top-k POIs are inserted into the route such that they result in the least increase in total distance travelled.

2.3.3 Consideration for Traffic Conditions, Transport Types and Travelling Uncertainty

The consideration of user interest preferences (in the previous section) is an attempt to make tour recommendations more personalized and there are various works that incorporate other real-life considerations. Other considerations include incorporating multiple modes of transport, considering traffic conditions and including uncertainty in travelling times, which we discuss next.

Recommending Tours with Considerations for Traffic Conditions

The TripPlanner [34] framework is a traffic-aware route planning system that aims to recommend a personalized route comprising of a set of must-visit POIs while considering the traffic conditions at different times of the day and days of the week. TripPlanner utilizes a large number of Foursquare check-ins and taxi GPS traces to construct a model of the traffic conditions at different POIs at different days and times. The TripPlanner framework operates based on three main steps:

1. *Generating a Dynamic POI Network Model.* TripPlanner first generates a network model representing the list of POIs (nodes) and paths between these POIs (edges). In this model, POIs (nodes) are static with a fixed popularity, category, location, opening hours and visit duration, while paths (edges) are dynamic with a travelling time between POIs based on a time-dependent traffic condition. The POI-related statistics are derived from the FourSquare dataset and the path-related statistics are derived from the taxi GPS dataset.

2. *Searching for Possible Routes.* The process of route search first requires the user to provide the desired starting/destination POIs, a set of must-visit POIs, and the available time budget to complete this route. If there are no valid routes that satisfy these constraints, TripPlanner iteratively suggest locations to be removed from the must-see list, until a valid route can be found. The total time taken for a route comprises the travelling time between POIs (time-variant based on traffic conditions) and the visiting time at POIs (based on the average visit durations).

3. *Augmenting Routes with Preferred POIs.* In the event that the route found from Step 2 has sufficient time budget left, this route augmentation step then includes additional POIs into this route in order to maximize the satisfaction to the user. Such a scenario can occur when the user only indicated a small number of POIs (or no POIs) in the must-see POI list. These new POIs are selected from a preferred POI list where the POIs are aligned to the interest preferences of the user.

There have also been applications such as the eCOMPASS tourist tour planner [59] that considers time-varying travelling times between POIs based on the traffic conditions as well as transport modes at the time of POI departure.

In the field of Operations Research, there have also been works on route planning that consider multiple transport modes and uncertain travelling times [19, 20, 50]. While these works present interesting results, they differ from our tour recommendation problem as they are mainly concerned with finding the shortest path between a starting and ending location. Similarly, researchers such as [95,96] incorporate traffic flow predictions into their route planning problem, enabling them to recommend routes that avoid traffic congestion and hazards in advance.

Recommending Tours based on Interests and Different Transport Types

Kurashima et al. [86, 87] proposed a method for tour recommendation that considers the current location of the user, his/her interest preferences, available time for touring and available means of transport. The authors use a combined topic and Markov model to recommend POIs that are based on a user's interest and current location. They used Probabilistic Latent Semantic Analysis (PLSA) [69] as the topic model, which considers the interests of a user and models the probability of this user visiting a POI p given a travel history h, that is:

$$P(p|h) = \sum_{z \in Z} P(z|h)P(p|z) \tag{2.6}$$

where Z is the set of topics for the POIs, $P(z|h)$ is the probability of a user being interested in topic z, and $P(p|z)$ is the probability that POI p is selected from topic z.

For the Markov model, the authors employ a first-order Markov model, where the probability of visiting a POI p_t depends on a previous POI visit p_{t-1}. This is formally defined as:

$$P(p_t|p_{t-1}) = \frac{N(p_{t-1}, p_t)}{N(p_{t-1})} \tag{2.7}$$

where $N(p_{t-1}, p_t)$ is the frequency that POI p_t is visited after a prior visit to POI p_{t-1}, and $N(p_{t-1})$ is the total visit count at POI p_{t-1}.

The authors then combine the topic and Markov model as one single model, using the following formula:

$$P(p_t|p_{t-1}, h) = \frac{P(p_t|p_{t-1})}{C(p_{t-1}|h)} \frac{P(p_t|h)}{P(p_t)} \tag{2.8}$$

where $C(p_{t-1}|h)$ is a normalization factor based on unigram rescaling [61]. This combined topic and Markov model is then able to recommend the next POI to visit based on the user's current location and his/her interest preferences. To recommend a tour itinerary of connected POIs, the authors use a best-first search algorithm to select tour itineraries with the highest probability and adhere to the available touring time.

Making use of past geo-tagged photos, Kurashima et al. [86, 87] also estimate the travelling time needed from one POI to another, and consider different types of transport modes. The authors accomplish this in the following steps:

1. For each POI pair, the authors first calculate all possible travelling times based on the users' travel history. The result is a set of travelling times for a single POI pair.

2. Using this set of travelling times, the authors then apply K-means clustering and obtain a set of K centroids, which reflects K different modes of transport.

3. Depending on the preferred mode of transport, the expected travelling time between POIs is then applied to their algorithm to ensure that the tour is within the available touring time.

Recommending Tours with Transport-cost Awareness

Unlike other works that consider transportation in the form of traffic conditions or transport modes, [52] included the consideration for the transport cost associated with a recommended tour. The authors utilized geo-tagged photos to determine the popularity of POIs, visit durations, opening hours and appropriate visual scenes to display at different times of the day. They modelled their tour recommendation problem based on a variant of the NP-hard Vehicle Routing Problem with Time Window [21]. In this problem, their main aim is to visit the largest number of popular POIs and ensure that the routes taken are minimal and smooth (i.e., no long detours), while adhering to the available touring time and transportation cost budget.

Recommending Tours based on Interests, POI Opening Hours and Travel Time Uncertainty

In [159, 160], Zhang et al. studied the tour recommendation problem with the goal of recommending itineraries that are personalized based on the interest preferences of users and can be completed within an available touring time, while adhering to the opening hours of various POIs and considering the uncertainty in travelling time. Their general approach in accomplishing this tour recommendation task is as follows:

- *Modelling User Interest Preferences*. To determine the interest preferences of users, the authors adopted a feature-centric collaborative filtering approach [160]. This approach builds upon traditional collaborative filtering but performs the collaborative filtering on features of POIs, instead of the POIs themselves. Thus, the interest of a user in a specific POI is based on his/her rating on all features of this POI.

- *Modelling Travel Time Uncertainty*. Instead of a fixed value for the traveling time $t_{i,j}$ from POI i to POI j, the authors modelled $t_{i,j}$ as a random variable associated with a particular probability distribution. With this uncertain travelling time, the authors also calculate a *completion probability*, which represents the likelihood of completing this itinerary within a time budget B. An objective of this work is to recommend an itinerary with a *completion probability* above a specific threshold.

- *Modelling POI Opening Hours*. The authors consider POI opening hours by implementing a POI availability constraint, and adopt a tour recommendation approach by iteratively adding a POI to an itinerary if it satisfies this constraint, along with the total time budget constraint. Given a POI i with opening hours $[O_i, C_i]$, arrival time a_i and total time m_i spent at POI i, the POI i is appended to the tour itinerary only if the constraint, $a_i + m_i \leq C_i$, is satisfied.

2.3.4 Other Approaches

While user personalization and real-life traffic/transport considerations are important to tourists, there are other travellers that emphasize different aspects of travelling, such as recommending routes that may not be the most popular but are the most scenic or safest. In the following sections, we present some of these works.

Recommending Beautiful, Quiet and Happy Routes

Instead of recommending popular or interesting routes, Quercia et al. studied the recommendation of routes that are deemed to be emotionally pleasing, in particular beautiful, quiet and happy routes [114]. They first make use of crowdsourcing to ask users to pick between two pictures, the one that is more beautiful, quieter or happier, where each picture corresponds to a specific location in the city. With this crowdsourced dataset, they are able to determine scores for each location based on how beautiful, quiet and happy each one is. Following which, their approach for recommending the best route, in terms of either beauty, quietness or happiness, is as follows:

1. Given a starting point s and destination d, use Eppstein's algorithm [49] to recommend the M shortest route from s to d, where M is set to be arbitrarily large (e.g., 1 million) such that it covers all possible routes.

2. Given a $k \leq M$, calculate the average beauty, quietness or happiness rank for each of the top-k routes (i.e., all locations in each route) and record the route with the best rank. Instead of exploring all M routes, the intuition is to iteratively explore a smaller set of k routes.

3. Iteratively repeat Step 2 for the next set of k routes and all the while recording the route with the best rank. This iterative step repeats until the improvement is less than a threshold ϵ, and the route with the best rank is returned as the most beautiful, quietest or happiest route from s to d.

Extending beyond crowdsourcing, Quercia et al. utilized the tags used in Flickr geotagged photos to determine how beautiful, quiet and happy each photo is by comparing the tags used to the Linguistic Inquiry Word Count dictionary [110]. Apart from Quercia et al. who examined beautiful, quiet and happy routes, the ScenicPlanner system [33] utilized Flickr photos and FourSquare check-ins to determine scenic scores for individual road segments as part of a larger scenic route planning problem. Other researchers have also focused on other aspects of non-touristic tour recommendations, such as [53, 54, 76] who used crime statistics for recommending short but safe (low crime) paths.

Random Walks with Restart

Making use of geo-tagged photos, Lucchese et al. [105] proposed an algorithm for recommending POI visits using random walks on a graph-based representation of past tourist trajectories. This algorithm comprises the following steps:

1. Based on tourist trajectories mined from Flickr geo-tagged photos, the authors first construct an itinerary graph $G = (P, E, W)$, where P is the set of all POIs in a city, E is the set of edges representing two POIs that were co-visited in a tourist's trajectory, and W is the weight given to each edge based on the number of distinct tourists who have visited that POI pair and the number of shared POI categories.

2. With the itinerary graph generated in Step 1 and an itinerary transition matrix, which contains the transition probability between any two POIs, the authors then use the Random Walk with Restart algorithm [134] to compute the steady-state probability distribution for the set of POIs previously visited by the tourist.

3. Using the itinerary transition matrix generated in Step 2 and the set of POIs previously visited by the tourist and its steady-state probability distribution, the authors then calculate the scores of the unvisited POIs based on the product of entries in the steady-state probability distribution. Following which, the algorithm then recommends the top-k POIs with the highest scores.

The output of this algorithm is a set of k POIs, which needs to be constructed as an connected itinerary for the purposes of tour recommendation.

2.3.5 Web and Mobile-based Applications

myVisitPlanner^{GR}

myVisitPlanner^{GR} is a web-based application that is targetted at recommending touristic activities to visitors of Northern Greece [117]. There are three main steps to using the *myVisitPlanner*^{GR} system, namely:

- *Input of User Profile and Constraints*. Users first enter their profile (age, gender and spoken languages) and constraints (preferred start time, conflicts with existing scheduled activities).

- *Recommendation of Suggested Activities*. A list of suggested activities is recommended to the user using a hybrid recommender based on: (i) activities similar to that previously rated by the user based on the Hausdorff distance of the ontology of both activities; and (ii) a variant of collaborative filtering where users are clustered based on their interests and the rating of activities by this user cluster is taken into consideration.

- *Scheduling of Proposed Itinerary*. To schedule a proposed itinerary of the recommended activities, *myVisitPlanner*^{GR} uses the scheduling engine of *SelfPlanner* [116, 118], which also attempts to de-conflict the recommended activities with existing activities (entered by the user). Users are then able to accept the scheduled itinerary or reject it to request re-planning of the itinerary.

This system uses an ontological approach to represent activities where there are multiple hierarchical levels. Activity providers have the flexibility to describe their activity at a higher, more general hierarchical level or a lower, more specific level. In addition, the system keeps track of the ratings that users assign to the various activities and uses these ratings as input to their hybrid recommender system.

SAMAP

[31] proposed the *SAMAP* system for recommending and planning a personalized daily itinerary that considers his/her user profile and available touring time. *SAMAP* was

designed for mobile devices such as smart phones and operates as a multi-agent system, comprising the following agents:

- *User modelling and interface agent*. This agents solicits interests and trip preferences (e.g., available time, preferred transport and payment modes) and personal information (e.g., name, gender, occupation, mobility status) from a tourist and builds a user model of this tourist. This user model is then passed to the case-based reasoning agent, for recommending a set of POIs or activities.

- *Case-based reasoning (CBR) agent*. Using the user model from the user modelling and interface agent, this agent identifies a set of POIs of activities that are located in the visited city and aligned with the tourist's interest preferences, based on this tourist's k nearest neighbours. In particular, the CBR agent retrieves the POIs visited by k users that are the most similar to this tourist based on their interest preferences. Using this set of POIs, the CBR agent identifies the top-k most popular POIs, whose visit duration can be completed within the tourist's available touring time.

- *Planning agent*. Using the list of POIs generated by the CBR agent, this agent then plans and schedules a trip itinerary that includes a subset of these POIs that maximizes the tourist's utility score, while accounting for POI opening hours and transport modes between POIs.

2.4 Tour Recommendation for Groups of Tourists

Tour recommendation works have typically targetted the single traveller as seen in Section 2.3 but in real-life, tours are frequently undertaken by groups of travellers such as couples, friends or families. Such group tour recommendations are challenging due to the need to appeal to multiple travellers within the same tour group. In the following section, we examine some early efforts on resolving this group tour recommendation problem. Table 2.2 presents a broad overview of the various works on tour recommendation for groups of travellers.

Table 2.2: Survey of Tour Recommendation for Groups of Tourists

Paper Reference	Popularity -based	Interest -based	Determines Interest	Constructs Itinerary	Considers Time	Transport/ Traffic	Tour Guides
[3]	Yes	Yes	Yes	Yes	Yes	No	No
[121]	No	Yes	Partly	No	No	No	No
[57,58]	Yes	Yes	Partly	Yes	Yes	No	No
[4]	No	Yes	No	Yes	Yes	Yes	No
[72]	No	Yes	No	No	No	No	No
[36]	Yes	Yes	Yes	Yes	Yes	No	No

2.4.1 Tour Recommendation for Groups

At around the same time as our work in Chapter 7, Anagnostopoulos et al. [3] also proposed similar problems for group tour recommendations, focusing on recommending tour itineraries that best satisfy a group of tourists, where this group is pre-determined in advance. These problems are termed the TOURGROUPSUM, TOURGROUPMIN and TOURGROUPFAIR problems, which differ based on the objective function to be optimized. Instead of solving these problems as an Integer Linear Program, Anagnostopoulos et al. utilized greedy heuristics and Ant Colony Optimization to solve their group tour recommendation problems. Another difference is that [3] did not consider the assignment of tour guides to lead each tour group, but they consider multiple forms of optimization objectives that maximize the overall group interest (TOURGROUPSUM), interest of the least satisfied user (TOURGROUPMIN) and fairness among all members of a group (TOURGROUPFAIR).

2.4.2 e-Tourism System

e-Tourism [57,58] is a system that aims to recommend interesting activities (including POI visits) to either individuals or groups of tourists. There are several main steps in using the *e-Tourism* system, namely:

1. *Provide Tourist Profile and Groupings.* When a tourist first uses the system, he/she provides the following information, namely: (i) demographic details such as their

age, gender, country; and (ii) interest preferences based on a rating of 0 to 100 on specific features. For groups of tourists, the individual tourists that comprise a group must be explicitly listed and the profile for each individual tourist must have first been created.

2. *Recommendation of Activities to Individuals.* This task is performed by a component called the Generalist Recommender System Kernel (GRSK), which analyzes the tourist profile provided in Step 1 to recommend a list of activities (or POI visits) to individual tourists, based on either a demographic-based approach [30], content-based approach [30], general likes-based filtering [67] or a hybrid of the earlier three approaches.

3. *Recommendation of Activities to Groups.* For groups of tourists, the GRSK uses aggregation or intersection techniques to determine a group interest preference, based on the preferences of individual tourists in that group. The aggregation technique uses the average values of interest preferences of all group members to determine the group preference, while the intersection technique determines the group preference using interest preferences that are common among all group members.

4. *Tourist Feedback on Recommended Itinerary.* Upon completion of the recommended tour, each tourist is able to feedback on individual items in the recommended itineraries by giving a rating to each item. This feedback/rating is then stored by the system and used to improve future recommendations to this tourist, particularly by the content-based approach in Step 2.

One unique characteristic of *e-Tourism* is their representation of tourist interest preferences, which are based on a hierarchical taxonomy of features. Thus, this representation results in the following: (i) instead of using POI categories, a more general concept of *features* are used, allowing the system to be easily generalized to other application domains; (ii) the taxonomy of features are structured in a hierarchical manner, with the higher levels being categories and the lowest level being activities or POIs to be recommended; and (iii) specific activities or POIs can belong to multiple features, with varying levels of membership.

2.4.3 Intrigue System

INteractive TouRist Information GUidE (or *Intrigue*) is a web and mobile based system that aims to recommend tours to both individuals and groups of tourists [4]. For a group of tourists, the usage scenario for *Intrigue* is that a specific (lead) tourist will use the system to: (i) indicate the number of tourists in that tour group; (ii) manually specify the sub-group that each tourist belongs to, based on the tourist demographics (e.g., age and background) and interest preferences; and (iii) enter details about each sub-group via a registration form (an optional step). The main idea is that a large tour group could be divided into smaller homogeneous sub-groups, e.g., a particular sub-group could be defined by the characteristics of ages ranging from 30 to 40 years old, backgrounds in engineering, and interests in architecture and museums.

The recommendation of a tour takes place in a few steps that involves various user interactions. These steps are:

- *Selection of POIs.* Using a menu such as in Figure 2.6, the tourist is able to view any categories of POIs in a specific city, e.g., POI category of Civil Buildings in the city of Turin.

- *Recommendation of POIs.* A list of POIs that satisfy the previously selected POI category and city is then shown to the tourist, as shown in Figure 2.6. To solve the issue of conflicting interests in a tourist group, *Intrigue* explains the rationale behind recommending certain POIs to each sub-group.

- *Adding of POIs into Tour.* If the tourist is interested in any POIs recommended in the previous step, he/she is then able to add that specific POI as part of the tour.

- *Proposal of Tour Itinerary.* After various repetitions of the previous step (adding of POIs), the tourist is able to indicate any constraints (such as starting/ending POIs and starting/ending time) and *Intrigue* will schedule an itinerary based on the previously selected POIs and user constraints.

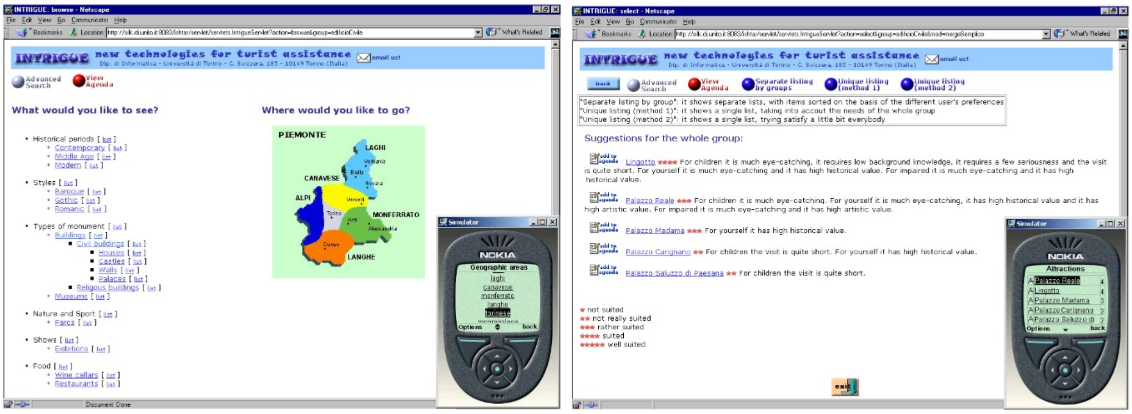


Figure 2.6: Examples of the selection of POIs based on categories and cities (left) and the list of recommended POIs, which the tourist can add to a tour (right), in Intrigue. Retrieved from [4].

2.4.4 Travel Decision Forum

The *Travel Decision Forum* proposed by Jameson et al. [72] was one of the earlier applications that examined the problem of group recommendations in the context of tour planning. This system utilizes group-oriented interfaces and virtual agents (animated characters) to enhance mutual awareness among group members from the stages of indicating interest preferences to removing conflicts from the proposed solution. Some key features of this application are:

- *Solicitation of Interest Preferences.* As shown in Figure 2.7a, this application uses a form to solicit user interest preferences in terms of an assigned rating ("Want it" to "Don't want it") and importance ("Very important" to "Not important") of each category. All users are able to see the interest preferences indicated by other users, before indicating their own preferences.

- *Generation of Proposal.* A proposed solution can be generated using either the average rating, median rating, by random choice or based on a non-manipulable, joint-rating mechanism [44]. The highlighted cells in Figure 2.7a indicates the proposed solution.

- *Discussion of Proposal.* Each user can either accept the proposed solution (i.e., the proposed rating values) or engage in a discussion with other users to come to a

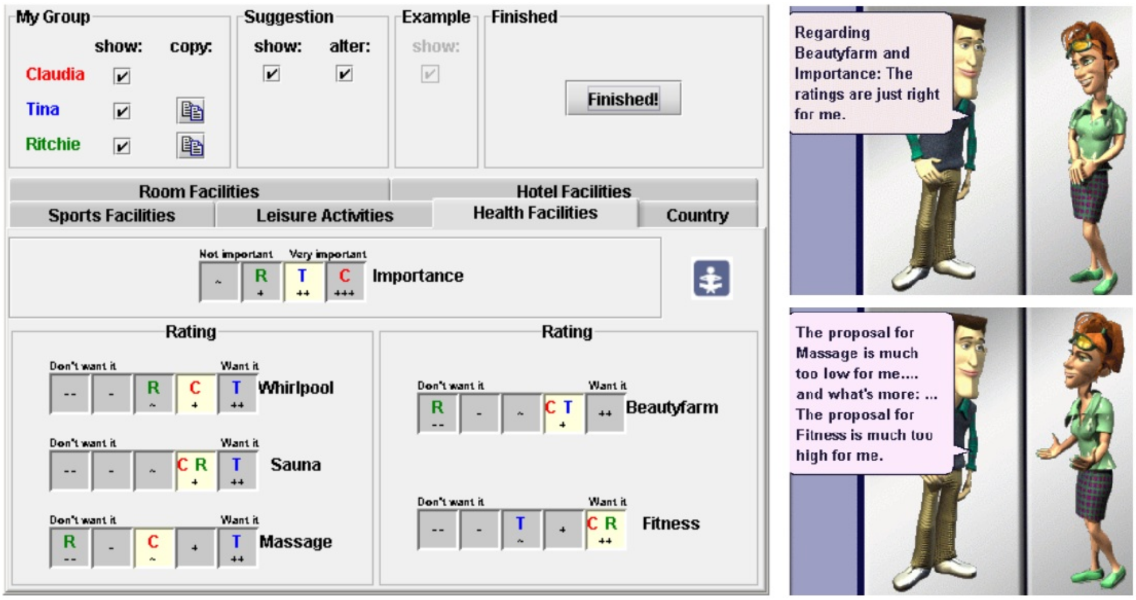


Figure 2.7: Examples of: (a) Preference Elicitation Form (left) for soliciting interest preferences and displaying the proposed solution; and (b) virtual avatars (right) for aiding in group communications. Retrieved from [72].

mutually-agreeable solution. This discussion is represented via the use of virtual agents as shown in Figure 2.7b.

While the *Travel Decision Forum* is used in the context of tourism, its main purpose is for the solicitation of group interest preferences using interaction-based techniques, e.g., group-viewable forms and virtual agents. The main output from this application is the recommendation in terms of group interest preferences that are mutually-agreeable by all members of a group. For the purposes of group tour recommendation, additional work is required to translate these group interests into the recommendation of relevant POIs that are constructed as a connected tour itinerary.

2.4.5 Top-k group recommendation

Closely related to the group tour recommendation problem are the problems of top-k recommendation for groups and group recommendation, where the main objective is to recommend a ranked list or set of items that are of relevance to a group of users. These problems typically focus on retail items such as films, songs or books, with [121] recommending sets of POIs as items but not in the form of an itinerary. More specifically, [121] examined the problem of group formation such that the constructed group comprises members who are more likely to prefer a top-k item recommendation. Others have studied an adjacent problem where the group members are pre-determined, and the objective is to model a collective group preference and/or recommend items to each group. For example, [71] proposed an algorithm for representing group preferences as a set of high level features that are not biased towards specific individuals, using collective deep belief networks and dual-wing restricted Boltzmann machines. Others like [2] tried to derive a group consensus score that maximizes item satisfaction for all members of a group, while minimizing the level of disagreement among members of the group. Yuan et al. [157] proposed a probabilistic model for group recommendations that account for group members with different levels of influence and how user preferences change as individuals compared to in a group.

While these works are targetted at groups of users, their main objective is to recommend either a ranked list of items or a set of items, which are typically retail items and merchandise. Although these works can be adapted to tourism by treating individual POIs as items, they do not recommend these POIs as a connected itinerary and neither do they consider the various spatial and temporal constraints that are associated with tour planning. For a more comprehensive review on group recommendation research, we refer the interested reader to [17].

2.5 Evaluation Strategies

A key process in tour recommendation research is the evaluation of the recommended tour itineraries, namely how well they satisfy the requirements of the individual tourists. However, there are can be many interpretations of these requirements, thus leading to a variety of evaluation strategies used by the various works in this area. In this section, we aim to highlight the various forms of evaluation strategies, and discuss the advantages and disadvantages for each of them.

2.5.1 Real-life Evaluations

We classify an evaluation strategy as a real-life evaluation if the recommended itineraries are compared against the real-life travel history of a tourist. As discussed in Section 2.2, these real-life travel histories can be obtained from various sources, namely: (i) geo-tagged photos; and (ii) location-based check-ins. For both of these sources, the real-life visits of tourists can be narrowed down to individual POIs, which allow a researcher to compare the POIs in a recommended itinerary against the real-life POI visits by tourists. To facilitate such a comparison, various Information Retrieval (IR) based metrics are used, such as:

1. **Precision.** The proportion of recommended POIs that were also visited by the tourist in real-life.

2. **Recall.** The proportion of POIs visited by the tourist in real-life that were also recommended.

3. **F1-score.** The harmonic mean of both precision and recall.

Other variations of Precision, Recall and F1-score include measuring how well the categories of POIs in the recommended tour reflect the POI categories that were visited in real-life. This variation is another measure of the effectiveness of the algorithms in proposing tours (POI categories) that are relevant to the interest preferences of tourists, and has been used in works such as [22, 24]. Others have also proposed variations of F1-score that account for POI visit orders in real-life [35].

2.5.2 Heuristic-based Evaluations

In cases where the real-life travel histories of tourists are not available, or as a supplemental analysis to the IR-based metrics introduced in Section 2.5.1, heuristic-based metrics are often used to evaluate the effectiveness of the recommended itineraries. Some examples of heuristic-based metrics are:

1. **Total POIs Recommended.** The total number of POIs recommended to a tourist as part of an itinerary.

2. **POI Popularity.** The summation of popularity scores of all POIs recommended to a tourist, i.e., total tour popularity.

3. **Tourist Interests.** The summation of interest alignment scores of all POIs recommended to a tourist, i.e., total tour interest.

Apart from using the summation of POI popularity and tourist interests, possible variations include using other statistical values based on the average, median, minimum, maximum or other percentile/quartile values. Particularly for group tour recommendation, the minimum and maximum tour interest values would respectively reflect the least and most satisfied tourist in a group.

2.5.3 Crowd-based Evaluations and User Studies

In contrast to the quantitative measures used in the real-life and heuristic-based evaluations (covered in the previous sections), an alternative evaluation methodology is to utilize qualitative measures, which are typically smaller in scale but more detailed in scope. Examples of such qualitative measures used in evaluating tour recommendations are:

- **User Studies.** Evaluation using user studies involves recruiting a small number of experiment participants who will use the proposed tour recommendation system and other baseline systems. At the end of these experiments, the participants will then answer a series of survey questions that are based on subjective criteria such as the usability of the system. For example, [149] utilized user studies involving 10 participants who evaluated their proposed Aurigo system against the baseline Google Maps.

- **Crowd-based Evaluations.** With the advent of services like Amazon Mechanical Turk (AMT), there has been an increased usage of crowdsourcing-based evaluation for tour recommendations. These evaluations focus more on the recommendation results, i.e., the recommended tour itineraries, unlike user studies that evaluate the entire experience of using the tour recommendation system, i.e., from entering user inputs and retrieving the results. For example, [40, 41] used AMT workers to evaluate their recommended tour itineraries against baseline itineraries by tour agencies.

2.5.4 Online Controlled Experiments

Online companies such as Google, Microsoft and Yahoo! have been using online controlled experiments to evaluate the effects of website user interface and design changes on live users in a real-life setting [84, 89, 129]. Such online controlled experiments are conducted in the form of A/B tests where a proportion of live users are exposed to a specific design or algorithmic variant while another proportion of live users are exposed to an alternative variant. In the area of tour recommendations, web-based tour companies (such as expedia.com and booking.com) can utilize such online controlled experiments on a smaller subset of users before introducing new features. The evaluation of these features could include:

- **Design-based Variants.** The evaluation of changes to a website user interface, which could include the way recommended tours are displayed or the design of a form to solicit user information.
- **Algorithm-based Variants.** The evaluation of changes to the underlying tour recommendation algorithms, e.g., comparing a Naive Bayes recommender against a popularity-based recommender such as in [78].

2.6 Summary

We have provided a comprehensive review of the literature in the area of tour itinerary recommendation and highlighted the key differences between tour itinerary recommendation and the related areas of Operations Research, next-location prediction, top-k location recommendation and travel package/region recommendation. We developed a taxonomy to describe general touring-related research, with a detailed breakdown of tour itinerary recommendations based on various real-life considerations such as POI popularity, user interests, time constraints, user demographics, transport modes and traffic conditions. In addition to reviewing a large selection of tour itinerary recommendation problems and solutions, we also discuss the various types of datasets (geotagged social media, location-based social networks, and GPS trajectory traces) and evaluation methodologies (real-life and heuristic-based metrics, user studies and online experiments) that can be employed in tour itinerary recommendation research.

Although tour itinerary recommendation has been well-studied in recent years, there still remain various research gaps, such as: (i) user interests are typically based on a simple measure of visit frequency, which may not accurately reflect user interests; (ii) most recommendation algorithms focus on the relevance of POIs, but otherwise do not personalize the visit durations based on user interests; (iii) current works are unable to automatically determine and incorporate queuing times at POIs, to recommend a realistic tour itinerary with minimal queuing times; and (iv) tour recommendation studies typically focus on individual travellers, whereas people frequently travel in groups in real-life and may engage the services of tour guides. In this thesis, we aim to address these research gaps by formulating realistic tour itinerary recommendation problems, and utilizing both data mining and optimization techniques to solve these problems. Given this background, in the next chapter we present our formulation of the general tour itinerary recommendation problem.

Chapter 3
Background and Preliminaries

This chapter provides an introduction to the generic tour itinerary recommendation problem and its Integer Programming formulation, which we adapt from a variant of the Orienteering problem. We also introduce some basic notation, preliminary definitions and the data mining framework used for the different variants of the tour recommendation problem, which will be elaborated in later chapters.

3.1 Introduction

Our proposed tour itinerary recommendation problem is closely related to the path planning problems typically studied in Operations Research, such as the Travelling Salesman problem or Orienteering problem. In the tour itinerary recommendation problem, the main objective is to recommend a personalized itinerary for a specific user based on his/her interest preferences as well as other trip constraints. However, unlike the tour itinerary recommendation problem, the Travelling Salesman problem and Orienteering problem aim to generate an optimal solution using a global utility or score that is common among all users, and thus provides no personalization based on user interests

In our work, we model the tour itinerary recommendation problem based on a variant of the Orienteering problem, with the incorporation of user-specific interest preferences in the POI score along with a global popularity measure for each POI[1]. This tour itinerary recommendation problem also considers other trip constraints such as a specific starting and ending location, not revisiting a POI multiple times, and a time budget for completing the itinerary. These trip constraints are aligned with the real-life considerations of tourists, such as preferring to start and end near their hotel, having limited time avail-

[1] In this thesis, we use "POI" and "attraction" interchangeably.

able for touring purposes, and wanting to explore new and relevant POIs.

In this chapter, we provide an overview of the generic tour itinerary recommendation problem and present an Integer Programming formulation of this problem based on a variant of the Orienteering problem. We also present some basic notation and preliminary definitions used in the tour itinerary recommendation problem, as well as a data mining framework for deriving user trajectories, user interests and POI-related information from geo-tagged photos.

3.2 Background on the Orienteering problem

The Orienteering problem [136] has its origins from a navigational competition of the same name, and has been well-studied by the Operations Research community. In this community, the focus of studying the Orienteering problem is on developing algorithms that are able to generate approximate solutions close to the optimal solution, where optimality is typically based on a global measure that is the same for all users.

In an Orienteering competition there are multiple navigational check-points distributed throughout an area, where each check-point is associated with a certain score. The main objective of this competition is for participants to maximize their total score, which is accumulated from visiting the various check-points. Participants are only given a limited amount of time to maximize their scores, and the winner is the participant who has accumulated the highest score. Due to this time limitation, participants have to strategize and select a smaller subset of check-points to visit and decide on the sequence in which to visit these check-points. For a more in-depth review of the Orienteering problem, we refer readers to [142] and [65].

In recent years, various works have used the Orienteering problem to model different variations of the tour recommendation problem [24,35,40,145], due to the similarities between the two problems. For example, the competition participants and tourists alike, are often restricted by a fixed amount of time they have to complete the competition or their tour itinerary, and each check-point or POI is associated with a specific score or utility value. In this chapter, we present an overview of the generic tour itinerary recommenda-

Table 3.1: Description of Key Notation

Notation	Description
P	The set of all Points-of-Interest (POI) or attractions
p_x	A specific POI or attraction, where $p_x \in P$
$x_{i,j}$	A binary value indicating whether a path from POI i to j was chosen
Cat_{p_x}	The category of POI p_x
Dur_{p_x}	The visit duration at POI p_x
$Score(p_x)$	The score or utility of POI p_x
$Pop(p_x)$	The popularity of POI p_x
$Int_u(c)$	The interest of user u in POI p_x
$Trav_{p_x,p_y}$	Travelling time from p_x to p_y
$Cost_{p_x,p_y}$	Cost of travelling from p_x to p_y
$t^a_{p_x}$	The arrival time at POI p_x
$t^d_{p_x}$	The departure time from POI p_x
V_x	A user-POI visit, $V_x = (p_x, t^a_{p_x}, t^d_{p_x})$
S_u	The travel history of a user u, $S_u = ((p_1, t^a_{p_1}, t^d_{p_1}), ..., (p_n, t^a_{p_n}, t^d_{p_n}))$
$Queue^t(p_x)$	The queuing time at POI p_x at time t

tion problem based on a variant of the Orienteering problem, before describing different variants of the tour recommendation problems in subsequent chapters.

3.3 Problem Definition

3.3.1 Basic Notation and Definitions

First, we describe the key notation and definitions that are used in the generic tour itinerary recommendation problem. The list of key notation is given in Table 3.1.

Our tour recommendation problem takes place in the context of a specific city where there is a set of POIs P. Each POI $p_x \in P$ is also associated with a specific category Cat_{p_x}, a recommended visit duration Dur_{p_x}, and a perceived score or utility $Score(p_x)$. In the field of operations research, this POI score $Score(p_x)$ is typically based on a randomly generated value or the POI popularity $Pop(p_x)$, which is the same global value for all users. To provide greater personalization for users, we utilize a user-specific POI score that incorporates both a user-POI interest alignment score $Int_u(c)$ as well as POI popularity $Pop(p_x)$. There is also a perceived cost for travelling between POIs, denoted by $Cost_{p_x,p_y}$, as the cost of travelling from POI p_x to p_y. This cost is typically derived from the travelling time between the two POIs, $Trav_{p_x,p_y}$, depending on the user's mode of transport. Next, we describe the common definitions used in the rest of this thesis.

Definition 1: Travel History. Given a user u who has visited n POIs, we define his/her travel history as an ordered sequence, $S_u = ((p_1, t^a_{p_1}, t^d_{p_1}), ..., (p_n, t^a_{p_n}, t^d_{p_n}))$, with each triplet $(p_x, t^a_{p_x}, t^d_{p_x})$ comprising the visited POI p_x, and the arrival time $t^a_{p_x}$ and departure time $t^d_{p_x}$ at POI p_x. Thus, the visit duration at POI p_x can be determined by the difference between $t^a_{p_x}$ and $t^d_{p_x}$. Similarly, for a travel sequence S_u, $t^a_{p_1}$ and $t^d_{p_n}$ also indicate the start and end time of the itinerary respectively. For brevity, we simplify $S_u = ((p_1, t^a_{p_1}, t^d_{p_1}), ..., (p_n, t^a_{p_n}, t^d_{p_n}))$ as $S_u = (p_1, ..., p_n)$.

Definition 2: Travel Sequence. Based on the travel history S_u of a user u, we can further divide this travel history into multiple travel sequences, i.e., sub-sequences of S_u. We divide a travel history S_u into separate travel sequences if $t^d_{p_x} - t^a_{p_x+1} > \tau$. That is, we separate a travel history into distinct travel sequences if the consecutive POI visits occur more than τ time units apart. Similar to other works [40], we choose $\tau = 8$ *hours* in our experiments. These travel sequences also serve as the ground truth of real-life user trajectories, which are subsequently used for evaluating our various proposed algorithms and baselines. For a user u with n travel sequences, we use $S^1_u, S^2_u, ..., S^n_u$ to denote the different travel sequences in temporal order, such that S^1_u took place before S^2_u.

Definition 3: Point-of-Interest Popularity. Many earlier itinerary recommendation works [24, 40] represent the popularity of a POI based on the number of visits that this POI has received. In our work, we employ a similar measure of popularity for each POI based on the number of times this POI has been visited, which is defined as:

$$Pop(p) = \sum_{u \in U} \sum_{p_x \in S_u} \delta(p_x = p), \quad \forall\, p \in P \tag{3.1}$$

where $\delta(p_x{=}p) = 1$ if POI p_x is the same as POI p, and $\delta(p_x{=}p) = 0$ otherwise.

Definition 4: User-POI Interest Relevance. While POI popularity is the same for all users, the interest relevance of a POI differs from user to user, i.e., each user will have their own unique interest preferences. Given that C represents the set of all POI categories and H the set of photos taken by user u, the interest level of this user in POI category c can be defined as:

$$Int_u(c) = \frac{1}{|H|} \sum_{h \in H} \delta(h = c), \quad \forall\, c \in C \tag{3.2}$$

where $\delta(h = c) = 1$ if photo h is of a POI that belongs to category c, and $\delta(h = c) = 0$ otherwise. Equation 3.2 is a popular definition of user interest where the interest level of a user u in POI category c is derived from his/her number of photos of POI category c, in relation to the total number of photos he/she has taken.

3.3.2 Problem Formulation

We now define the generic tour recommendation problem based on the Integer Programming formulation of the Orienteering Problem [65, 136, 142]. Given a starting POI $p_1 \in P$ and destination POI $p_N \in P$, our goal is to recommend a tour itinerary $I = (p_1, ..., p_N)$ that can be completed within a specific budget B, while maximizing the overall score of all POIs in recommended tour itinerary I. Formally, our main aim is to optimize the following objective function:

$$Max \sum_{i=2}^{N-1} \sum_{j=2}^{N} x_{i,j} Score(i) \tag{3.3}$$

such that:

$$\sum_{j=2}^{N} x_{1,j} = \sum_{i=1}^{N-1} x_{i,N} = 1 \tag{3.4}$$

$$\sum_{i=1}^{N-1} x_{i,k} = \sum_{j=2}^{N} x_{k,j} \leq 1, \quad \forall\, k = 2, ..., N-1 \tag{3.5}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} Cost(i,j) x_{i,j} \leq B \tag{3.6}$$

$$2 \leq p_i \leq N, \quad \forall\, i = 2, ..., N \tag{3.7}$$

$$p_i - p_j + 1 \leq (N-1)(1 - x_{i,j}), \quad \forall\, i, j = 2, ..., N \tag{3.8}$$

where:

$$x_{i,j} = \begin{cases} 1, & \text{if we visit POI } i\text{, followed by POI } j \\ 0, & \text{otherwise} \end{cases}$$

Equation 3.3 shows the objective function that maximizes the total score or utility of all POIs in the recommended tour, where $Score(p)$ is associated with a specific POI p and is the same value for all users. In the context of tour recommendation, we typically define $Score(p)$ based on the popularity of POI p, $Pop(p)$, and a user-POI interest alignment score, $Int_u(c)$. Constraint 3.4 ensures that the recommended tour itinerary I starts and ends at POI P_1 and POI P_N, respectively. Constraint 3.5 ensures path connectivity, i.e., the tour itinerary is connected between POIs, and that no POIs are re-visited. Constraint 3.6 ensures that the total cost incurred from travelling between consecutive POIs is within a pre-determined budget B. In the Orienteering problem, this budget and cost is typically based on the distance between POIs or the time needed to travel between POIs, whereas in our tour recommendation problem, we also include the recommended duration to spend at each POI. In addition, we also include constraints 3.7 and 3.8 for eliminating sub-tours[2] in the recommended tour itinerary I, based on the work of [108], where p_x is the position of POI x in itinerary I.

Figure 3.1: Places-of-Interests from Wikipedia

3.3.3 Tour Recommendation Framework

Our framework for tour itinerary recommendation utilizes geo-tagged photos to derive POI-related statistics and various tour recommendation algorithms for planning personalized itineraries. This framework requires a list of POIs or attractions (with latitude/longitude coordinates and POI categories) and a set of geo-tagged photos (with latitude/longitude coordinates and time taken), which can be easily obtained from Wikipedia and Flickr, respectively, or similar sources. This framework comprises the following steps:

1. **Collection of Geo-tagged Photos**. For each city or locality, we use either the Flickr API [150] to retrieve all geo-tagged photos that were taken within that city or obtain these geo-tagged photos from a public dataset such as the Yahoo! Flickr Creative Commons 100M (YFCC100M) dataset [133,151]. The photos are tagged with 16 levels of geo-location accuracy and we only consider photos with the highest accuracy level of 16. Each of these photos is also tagged with the user ID of the photo owner, timestamp, and latitude/longitude geo-coordinates.

[2]Sub-tours are closed-loop itineraries that are disjoint from the main itinerary. Although such sub-tours satisfy the constraints of no POI re-visits and path connectivity within the sub-tour, these sub-tours result in solutions with multiple disjoint itineraries and hence should be removed.

2. **Get POI/Attraction List**. Extract the list of POIs, latitude/longitude coordinates, and interest categories from Wikipedia or another similar source. Figure 3.1 shows an example of a Wikipedia listing of POIs/attractions in Melbourne and the corresponding category and coordinates of each POI/attraction, which we use for this step.

3. **Mapping of Photos to POIs/Attractions**. Based on the geo-coordinates of the photos and POIs/attractions, we map each photo to a POI/attraction if they differ by <100m according to the Haversine formula [125]. If a photo is within 100m of multiple POIs/attractions, we only map this photo to the nearest POI/attraction, i.e., no photo is mapped to multiple POIs/attractions. At the end of this step, we have a set of POI/attraction visits of all users in a specific city or locality.

4. **Construct Travel History/Sequences.** Based on the POI/attraction visits from the previous step, we can construct the travel history of each user by sorting their POI/attraction visits in ascending chronological order (Definition 1). Using each user's travel history, we then proceed to group consecutive POI visits as an individual travel sequence, if the consecutive POI visits differ by <8 hours (Definition 2). Thus, we are also able to determine the POI visit duration based on the time difference of the first and last photo taken at each POI.

5. **Calculating POI/Attraction Popularity and User Interests**. Using the travel sequences from the previous step, we are able to calculate the POI/attraction popularity and user interests based on Definitions 3 and 4, respectively.

6. **Recommending Personalized Itineraries**. At the conclusion of the earlier steps, we use the computed travel sequences, attraction popularity and user interests to generate itinerary recommendations based on the various tour recommendation algorithms, which will be discussed in the later chapters.

Table 3.2: Differences between Orienteering Problem and Tour Recommendation

Difference	Orienteering Problem	Tour Recommendation
Personalization	No	Yes
Score/Utility	POI popularity or random value	POI popularity and user interests
Cost	Fixed travelling time or distance between POIs	Fixed or variable travelling time, with visit durations at POIs
Dataset	Typically synthetic datasets with known optimal solutions	Typically real-life datasets based on GPS traces or location-based social networks
Evaluation	Focus on optimality of solutions	Focus on adherence to user interest preferences

3.4 Summary

While the Orienteering problem has been well-studied in the Operations Research community, there are various distinct differences with our tour recommendation problem. These differences are highlighted in Table 3.2 and are categorized as follows: (i) whether there is personalization for an individual user's preferences; (ii) the metric that is used as the score or utility of POIs; (iii) the measure of cost for travelling between POIs; (iv) the typical dataset used in experimentation; and (v) the focus of the evaluation methodology.

For the rest of the thesis, we propose various novel formulations of the tour recommendation problem based on variants of the Orienteering problem, and develop various algorithms and evaluation frameworks for these tour recommendation problems. In the next chapter, we examine the personalized tour recommendation problem and proposed the PERSTOUR algorithm for recommending personalized tours based on POI popularity and time-based user interest, along with a weighted updating of user interests based on the recency of their POI visits.

Chapter 4

Personalized Tour Recommendation

In this chapter, we propose an algorithm called PERSTOUR *for recommending personalized tours using POI popularity and user interest preferences, which are automatically derived from real-life travel sequences based on geo-tagged photos. Our tour recommendation problem is modelled using a formulation of the Orienteering problem, and considers user trip constraints such as time limits and the need to start and end at specific POIs. In our work, we also reflect levels of user interest based on visit durations, and demonstrate how POI visit duration can be personalized using this time-based user interest. Furthermore, we demonstrate how* PERSTOUR *can be further enhanced by: (i) a weighted updating of user interests based on the recency of their POI visits; and (ii) an automatic weighting between POI popularity and user interests based on the tourist's activity level. While there have been earlier works that recommend personalized tours, these works only utilize a simple frequency-based user interest without recency updates and do not recommend a personalized POI visit duration. Using a Flickr dataset of ten cities, our experiments show the effectiveness of* PERSTOUR *against various collaborative filtering and greedy-based baselines, in terms of tour popularity, interest, recall, precision and* F_1*-score. In particular, our results show the merits of using time-based user interest and personalized POI visit durations, compared to the current practice of using frequency-based user interest and average visit durations.*

This chapter is derived from the following publications:

- **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Personalized Trip Recommendation for Tourists based on User Interests, Points of Interest Visit Durations and Visit Recency. *Knowledge and Information Systems*. Accepted to appear (32pp). Springer. Mar 2017

- **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Personalized Tour Recommendation based on User Interests and Points of Interest Visit Durations. *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*. pp 1778-1784. Jul 2015.

4.1 Introduction

Tour recommendation and itinerary planning are challenging tasks due to the different interest preferences and trip constraints (e.g., time limits, start and end points) of each unique tourist[1]. While there is an abundance of information from the Internet and travel guides, many of these resources simply recommend individual Points of Interest (POI) that are deemed to be popular, but otherwise do not appeal to the interest preferences of users or adhere to their trip constraints. Furthermore, the massive volume of information makes it a challenge for tourists to narrow down to a potential set of POIs to visit in an unfamiliar city. Even after the tourist finds a suitable set of POIs to visit, it will take considerable time and effort for the tourist to plan the appropriate duration of visit at each POI and the order in which to visit the POIs.

To address these issues, we propose the PERSTOUR algorithm for recommending personalized tours where the suggested POIs are optimized to the users' interest preferences and POI popularity. We formulate our tour recommendation problem based on the Orienteering problem [136], which considers a user's trip constraints such as time limitations and the need for the tour to start and end at specific POIs (e.g., POIs near the tourist's hotel). Using geo-tagged photos as a proxy for tourist visits, we are able to extract real-life user travel histories, which can then be used to automatically determine a user's interest level in various POI categories (e.g., parks, beaches, shopping) as well as the popularity of individual POIs. As tourists have different preference levels between POI popularity and POI relevance to their interests, our PERSTOUR algorithm also allows tourists to indicate their preferred level of trade-off between POI popularity and his/her interest preferences. In cases where the tourist prefers to automate the indication of this trade-off between POI popularity and interest preference, PersTour is also able to determine the appropriate trade-off based on the activity level of the tourist relative to the POI visits of the general population.

In this chapter, our main contributions are as follows:

1. We propose the PERSTOUR algorithm for recommending personalized tour/trip

[1] We use the terms "tourist" and "user" interchangeably, and similarly for the terms "tour" and "trip".

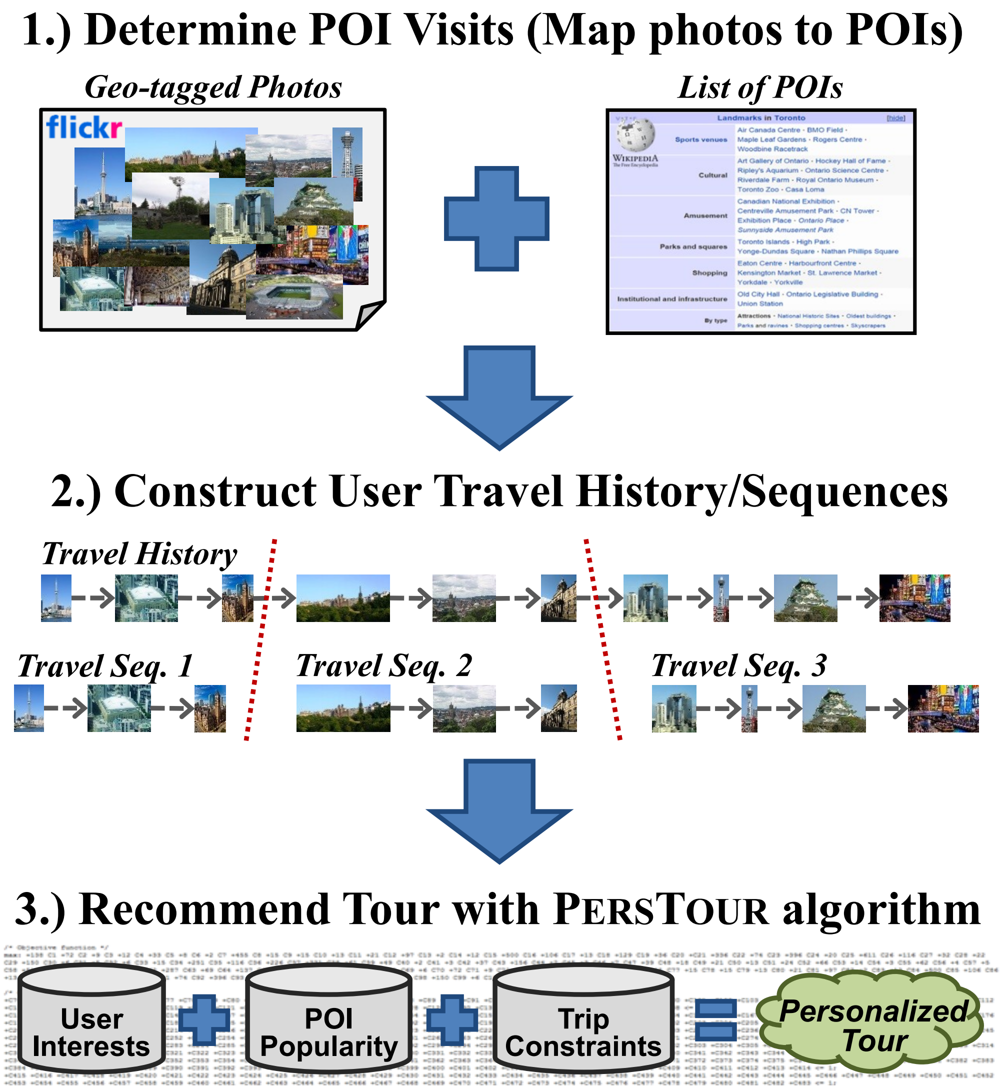


Figure 4.1: Tour Recommendation Framework

itineraries with POIs and visit duration based on POI popularity, users' interest preferences and trip constraints. Our tour recommendation problem is modelled in the context of the Orienteering problem (Section 4.3).

2. We introduce the concept of *time-based user interest* for tour recommendation, where a user's level of interest in a POI category is based on his/her time spent at such POIs, relative to the average user. We also compare our time-based user interest to the current practice of using frequency-based user interest, and show how time-

based user interest results in recommended tours that more accurately reflect real-life travel sequences (Section 4.3.1).

3. We also further enhance *time-based user interest* by implementing an update rule such that user interests are refined based on the recency of their past POI visits. This updating works by giving more emphasis to recent POI visits than those in the more distant past (Section 4.3.1).

4. We demonstrate the *personalization of POI visit duration* using time-based user interest, for the purpose of tour/trip itinerary recommendation. Our results show that personalized visit durations more accurately reflect the real-life POI visit durations of users, compared to the current practice of using average visit duration (Section 4.3.1).

5. While the PERSTOUR algorithm gives tourists the flexibility to indicate their preferred weightage between POI popularity and his/her interests, we also propose two schemes to automatically determine an appropriate weightage based on the tourist's activity level, relative to the general tourist population (Section 4.3.2).

6. We implement a framework (Fig. 4.1) for extracting real-life user travel histories based on their geo-tagged photos, which are then used for training our PERSTOUR algorithm and serve as ground truth for our subsequent evaluation (Section 4.4).

7. We evaluate different variants of PERSTOUR against various baselines using a Flickr dataset spanning ten cities. Our results show that PERSTOUR out-performs these baselines based on tour popularity, user interest, recall, precision and F_1-score (Sections 4.5 and 4.6).

The rest of the chapter is structured as follows: Section 4.2 discusses some related work in tour recommendation; Section 4.3 introduces some preliminaries and defines our research problem; Section 4.4 describes our overall framework for tour recommendation; Section 4.5 outlines our experimental methodology; Section 4.6 discusses our main results and key findings; and Section 4.7 summarizes and concludes this chapter.

4.2 Related Work

In this section, we provide an brief overview of general tour itinerary recommendations and the related areas of top-k POI recommendation and next-location prediction.

4.2.1 General Tour Recommendation

Tour recommendation has been a well-studied field, with researchers studying different aspects of the problem such as tour recommendation for individuals and groups, and proposing different problem formulations based on various real-life constraints. We refer readers to Chapter 2 for a more detailed discussion on the general field of tour recommendation.

4.2.2 Top-k POI recommendation and next-location prediction

The recommendation of top-k POIs and next-location predictions are also closely related to our problem of tour itinerary recommendation. For example, LearNext [10] used Gradient Boosted Regression Trees and Ranking SVMs to predict the (single) next POI that a tourist will visit, while [152] performed a similar next-location prediction task using Markov models, along with seasonal and temporal information. Others like [124] used a category-regularized matrix factorization approach for recommending individual POIs, and [82] proposed a system prototype for recommending individual POIs that are niche and specialized in nature. For top-k POI recommendations, many works utilized variants of matrix factorization or collaborative filtering approaches to recommend a ranked list of k POIs, using information such as friendship links [153], types of activities/users [88] and temporal patterns in POI visits [158].

4.2.3 Other tourism-related work

There are also many interesting tourism-related studies that utilize geo-tagged photos for purposes ranging from identifying popular POIs to analyzing tourist behavior. For example, Ji et al. [73] implemented a graph modeling framework to identify popular

POIs based on photos posted in blogs, while [112] used geo-tagged photos to understand tourist behavior based on their POI visit patterns and time spent. More generally, geo-tagged photos have been used for other purposes such as predicting friendship relationships based on spatio-temporal links [46], identifying local clusters of interesting events and places [79], and estimating the location where a photo is taken [90]. For a more comprehensive discussion of research that utilizes geo-tagged photos, we direct readers to [128], who presented a comprehensive review of current applications and identified various interesting future directions.

4.2.4 Discussion of differences with previous work

While these previous works are the state-of-the-art in tourism-related research, our proposed work differs from these earlier works in various aspects.

1. First, we automatically derive a relative measure of *time-based user interest* using a user's visit durations at POIs of a specific category, relative to the average visit durations of other users, whereas earlier tour recommendation works either use frequency-based user interest (based on POI visit frequency) or require users to explicitly indicate their interest preferences for tour itinerary recommendation.

2. Second, we plan and recommend tour itineraries with *personalized POI visit durations* that cater to individual users based on their time-based user interests, whereas previous works recommend tour itineraries using the same non-personalized POI visit duration for all users (either the average duration or a fixed duration, e.g., 1 hour at all POIs) or do not consider POI visit duration at all.

3. Third, although the works on top-k POI recommendation and next-location prediction are related to our tour itinerary recommendation problem, our proposed problem involves the additional considerations of user interest preferences, POI popularity, time constraints, starting/ending locations and more importantly, recommending a connected tour itinerary that satisfies these considerations, instead of individual POIs.

4. While the other tourism-related works illustrate many interesting applications of geo-tagged photos, these works use such photos to study tourist behavior and identify popular POIs, which are distinctly different from the task of recommending a personalized tour itinerary.

4.3 Background and Problem Definition

In this section, we first examine some preliminary definitions, before introducing a formulation of our tour recommendation problem.

4.3.1 Preliminaries

If there are m POIs for a particular city, let $P = \{p_1, ..., p_m\}$ be the set of POIs in that city. Each POI p is also labelled with a category Cat_p (e.g., church, park, beach) and latitude/longitude coordinates. We denote a function $Pop(p)$ that indicates the popularity of a POI p, based on the number of times POI p has been visited. Similarly, the function $T^{Travel}(p_x, p_y)$ measures the time needed to travel from POI p_x to p_y, based on the distance between POIs p_x and p_y and the indicated travelling speed. For simplicity, we use a travelling speed of 4km/hour, i.e., a leisure walking speed.[2]

Definition 1: Travel History and Sequence. As earlier described in Chapter 3, the travel history of an user is an ordered sequence of his/her POI visits denoted by $S_u = ((p_1, t^a_{p_1}, t^d_{p_1}), ..., (p_n, t^a_{p_n}, t^d_{p_n}))$, while travel sequences are sub-sequences of this travel history where consecutive POI visits occur more than 8hrs apart. For more details, refer to Section 3.3.1.

Definition 3: Average POI Visit Duration. Given a set of travel histories for all users U, we determine the average visit duration for a POI p as follows:

$$\bar{V}(p) = \frac{1}{n} \sum_{u \in U} \sum_{p_x \in S_u} (t^d_{p_x} - t^a_{p_x}) \delta(p_x = p), \quad \forall\, p \in P \tag{4.1}$$

[2] $T^{Travel}(p_x, p_y)$ can be easily generalized to different transport modes (e.g., taxi, bus, train) and to also consider the traffic condition between POIs (e.g., longer travel times between two POIs in a congested city, compared to two equal-distanced POIs elsewhere).

where n is the number of visits to POI p by all users and $\delta(p_x{=}p) = \{^{1,\ p_x=p}_{0,\ otherwise}$. $\bar{V}(p)$ is commonly used in tour recommendation as the POI visit duration for all users [22,24,34], while many earlier works do not factor in POI visit durations at all. In our work, we show how recommended POI visit durations can be personalized to individual users based on their interest (Definition 5), and use $\bar{V}(p)$ as a comparison baseline, i.e., the non-personalized POI visit duration.

Definition 4: Time-based User Interest. As described earlier, the category of a POI p is denoted Cat_p. Given that C represents the set of all POI categories, we determine the interest of a user u in POI category c as follows:

$$Int_u^{Time}(c) = \sum_{p_x \in S_u} \frac{(t_{p_x}^d - t_{p_x}^a)}{\bar{V}(p_x)} \delta(Cat_{p_x}{=}c),\ \forall\, c \in C \tag{4.2}$$

where $\delta(Cat_{p_x}{=}c) = \{^{1,\ Cat_{p_x}=c}_{0,\ otherwise}$. In short, Equation 4.2 determines the interest of a user u in a particular POI category c, based on his/her time spent at each POI of category c, relative to the average visit duration (of all users) at the same POI. The rationale is that a user is likely to spend more time at a POI that he/she is interested in. Thus, by calculating how much more (or less) time a user is spending at POIs of a certain category compared to the average user, we can determine the interest level of this user in POIs of this category.

Definition 5: Personalized POI Visit Duration. Based on our definition of time-based user interest (Equation 4.2), we are able to personalize the recommended visit duration at each POI based on each user's interest level. We determine the personalized visit duration at a POI p for a user u as follows:

$$T_u^{Visit}(p) = Int_u^{Time}(Cat_p) \times \bar{V}(p) \tag{4.3}$$

That is, we are recommending a personalized POI visit duration based on user u's relative interest level in category Cat_p multiplied by the average time spent at POI p. Thus, if a user is more (less) interested in category Cat_p, he/she will spend more (less) time at POI p than the average user.

Definition 6: Frequency-based User Interest. We also define a simplified version of user interest, denoted $Int_u^{Freq}(c)$, which is based on the number of times a user visits POIs of a certain category c (i.e., the more times a user visits POIs of a specific category, the more interested this user is in that category). As using $Int_u^{Freq}(c)$ is the current practice in tour recommendation research [22, 24], we include it for a more complete study and as a comparison baseline to our proposed $Int_u^{Time}(c)$.

Definition 7: Time-based User Interest with Weighted Updates. We improve upon the original *Time-based User Interest* (Definition 4) by giving more emphasis to recent POI visits and less emphasis to POI visits in the distant past. Algorithm 1 details our proposed algorithm. In Line 9 of Algorithm 1, we continuously update user u's interest by minimizing the error between his/her recommended and actual POI visit duration, while $\frac{i}{n}$ ensures that more emphasis is given to more recent POI visits. Lines 6 to 8 calculate the error between the recommended and actual POI visit duration, while Lines 4 and 5 ensure that we perform this update for all POIs in all travel sequences of user u.

Algorithm 1: Time-based User Interest with Weighted Updates

input : $\{S_u^1, S_u^2, ..., S_u^n\}$: The past travel sequences of a user u.
output: $Int_u^{Upd}(c)$: The updated interest levels for user u.

1 **begin**
2   **for** *POI category* $c \in C$ **do**
3         $Int_u^{Upd}(c) \leftarrow Int_u^{Time}(c)$;
4   **for** $i \leftarrow 1$ **to** n **do**
5   **for** *POI* $p \in S_u^i$ **do**
6             $recomTime \leftarrow Int_u^{Upd}(Cat_p) \times \bar{V}(p)$;
7             $actualTime \leftarrow t_p^d - t_p^a$;
8             $error \leftarrow \frac{recomTime - actualTime}{\bar{V}(p)}$;
9             $Int_u^{Upd}(c) \leftarrow Int_u^{Upd}(c) - \alpha \frac{i}{n} error$;

The intuition behind Algorithm 1 is that more recent POI visits are more relevant to a user, and thus should contribute more to the modelling of this user's interest. Similarly, other researchers have also observed people's preference for more recent activities/information, and utilized this recency preference for location-based domain expert identification [92] and personalized music recommendation [123].

Definition 8: Personalized POI Visit Duration with Weighted Updates. Similar to Definition 5, we can then recommend a personalized POI visit duration to POI p for a user u based on his/her *Time-based User Interest with Weighted Updates*, as follows:

$$T_u^{VisitUpd}(p) = Int_u^{Upd}(Cat_p) \times \bar{V}(p) \quad (4.4)$$

Similar to Definition 5, we are personalizing the POI visit duration for user u based on his/her updated interest level in category Cat_p multiplied by the average time that users spend at POI p.

4.3.2 Problem Definition

We now define our tour recommendation problem in the context of the Orienteering problem and its integer problem formulation [136, 142]. Given the set of POIs P, a budget B, starting POI p_1 and destination POI p_N, our goal is to recommend an itinerary $I = (p_1, ..., p_N)$ that maximizes a certain score S while adhering to the budget B.[3] In this case, the score S is represented by the popularity and user interest of the recommended POIs using the functions $Pop(p)$ and $Int(Cat_p)$, respectively. The budget B is based on time spent, and calculated using the function $Cost(p_x, p_y) = T^{Travel}(p_x, p_y) + T_u^{Visit}(p_y)$, i.e., using both travelling time and personalized visit duration at the POI. One main difference between our work and earlier work is that we personalize the visit duration at each recommended POI based on user interest (Definition 5), instead of using the average visit duration for all users or not considering visit duration at all. Formally, we want to find an itinerary $I = (p_1, ..., p_N)$ that:

$$Max \sum_{i=2}^{N-1} \sum_{j=2}^{N} x_{i,j} \Big(\eta Int(Cat_i) + (1-\eta) Pop(i) \Big) \quad (4.5)$$

[3] Although we examine POIs in this work, our tour recommendation problem definition can be easily modified such that a recommended tour itinerary starts and ends at a specific hotel where the tourist is staying.

where $x_{i,j} = 1$ if both POI i and j are visited in sequence (i.e., we travel directly from POI i to j), and $x_{i,j} = 0$ otherwise. We attempt to solve for Equation 4.5, such that:

$$\sum_{j=2}^{N} x_{1,j} = \sum_{i=1}^{N-1} x_{i,N} = 1 \tag{4.6}$$

$$\sum_{i=1}^{N-1} x_{i,k} = \sum_{j=2}^{N} x_{k,j} \leq 1, \quad \forall\, k = 2, ..., N-1 \tag{4.7}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} Cost(i,j) x_{i,j} \leq B \tag{4.8}$$

$$2 \leq p_i \leq N, \quad \forall\, i = 2, ..., N \tag{4.9}$$

$$p_i - p_j + 1 \leq (N-1)(1 - x_{i,j}), \quad \forall\, i, j = 2, ..., N \tag{4.10}$$

Equation 4.5 is a multi-objective function that maximizes the popularity and interest of all visited POIs in the itinerary, where η is the weighting given to the popularity and interest components. Equation 4.5 is also subject to constraints 4.6 to 4.10. Constraint 4.6 ensures that the itinerary starts at POI 1 and ends at POI N, while constraint 4.7 ensures that the itinerary is connected and no POIs are visited more than once. Constraint 4.8 ensures that the time taken for the itinerary is within the budget B, based on the function $Cost(p_x, p_y)$ that considers both travelling time and personalized POI visit duration. Given that p_x is the position of POI x in itinerary I, constraints 4.9 and 4.10 ensure that there are no sub-tours in the proposed solution, adapted from the sub-tour elimination used in the Travelling Salesman Problem [108].

Based on this problem definition, we can then proceed to solve our tour recommendation problem as an integer programming problem. For solving this integer programming problem, we used the lpsolve linear programming package [13]. We denote our proposed algorithm for personalized tour recommendation as PERSTOUR, and shall describe our overall framework and the different PERSTOUR variants in the following section.

Adaptive Weighting

As introduced in Equation 4.5, the η parameter offers tourists the flexibility to indicate their preferences for POI popularity and interest alignment. In this section, we propose two methods that automatically determine an appropriate value for the η parameter based on the POI visits by the general user population.

Given all users U and their set of travel histories $S_{u \in U}$, we define the number of POI visit count for a user u as: $C_u = |S_u|$. Similarly, C_{max} denotes the maximum POI visit count out of all users $u \in U$. We determine the η value (i.e., adaptive weighting) for a user u using the following two methods.

- **Adaptive Weights based on Scaling (PT-AS)**. This method determines the η value for a user u as follows: $\eta = \frac{C_u}{C_{max}}$. In short, we are scaling the POI visit count of a user u by the maximum POI visit count of all users.

- **Adaptive Weights based on Cumulative Distribution (PT-AC)**. This method determines the η value for a user u as follows: $\eta = P(C \leq C_u)$. That is, we are building a probability distribution function based on all users' POI visit counts, and then calculating the probability that a random variable C (i.e., the POI visit count) is less than or equal to the POI visit count of user u.

4.4 Tour Recommendation Framework

Using the framework previously described in Section 3.3.3, we are able to obtain POI popularity, user interest levels and travel sequences, which are used as input to the PERSTOUR algorithm. As described in Section 4.3.2, there can be different variants of PERSTOUR, based on the value of η and the type of interest function chosen. The value of η indicates the weight given to either POI popularity or user interest, while the interest function can be either frequency-based interest (Int_u^{Freq}), time-based interest (Int_u^{Time}) or time-based interest with weighted updates (Int_u^{Upd}). We experiment with the following variants:

- **PERSTOUR using η=0 (PT-0)**. PERSTOUR with full emphasis on optimizing POI popularity, ignoring user interest (i.e., no need to choose between Int_u^{Time} or Int_u^{Freq}).

- **PERSTOUR using Int_u^{Freq} and η=0.5 (PT-.5F)**. PERSTOUR with balanced emphasis on optimizing both POI popularity and *frequency-based* user interest.

- **PERSTOUR using Int_u^{Time} and η=0.5 (PT-.5T)**. PERSTOUR with balanced emphasis on optimizing both POI popularity and *time-based* user interest.

- **PERSTOUR using Int_u^{Upd} and η=0.5 (PT-.5U)**. PERSTOUR with balanced emphasis on optimizing both POI popularity and *time-based* user interest with *weighted updates*.

- **PERSTOUR using Int_u^{Freq} and η=1 (PT-1F)**. PERSTOUR with full emphasis on optimizing *frequency-based* user interest, ignoring POI popularity.

- **PERSTOUR using Int_u^{Time} and η=1 (PT-1T)**. PERSTOUR with full emphasis on optimizing *time-based* user interest, ignoring POI popularity.

- **PERSTOUR using Int_u^{Upd} and η=1 (PT-1U)**. PERSTOUR with full emphasis on optimizing *time-based* user interest with *weighted updates*, ignoring POI popularity.

- **PERSTOUR using Int_u^{Upd} and adaptive weighting η by scaling (PT-AS)**. PERSTOUR with emphasis on optimizing both POI popularity and *time-based* user interest with *weighted updates*, where emphasis is based on adaptive weighting by scaling of POI visit counts.

- **PERSTOUR using Int_u^{Upd} and adaptive weighting η by cumulative distribution (PT-AC)**. PERSTOUR with emphasis on optimizing both POI popularity and *time-based* user interest with *weighted updates*, where emphasis is based on adaptive weighting by cumulative distribution of POI visit counts.

These variants allow us to best evaluate the effects of different η values, and compare between frequency-based interest and time-based interest (with and without weighted updates). As PT-0 does not consider user interest, there is no need to choose between time-based or frequency-based user interest. The PT-0, PT-.5F, PT-.5T, PT-.5U, and PT-1F, PT-1T, PT-1U algorithms allows us to investigate the effect of different emphasis on POI popularity and the different types of user interests, i.e., by adjusting the η parameter. These algorithms offer tourists the flexibility to explicitly specify their preference between the two components of POI popularity and user interests. If the tourist prefers to determine this preference automatically, the PT-AS and PT-AC algorithms provide alternatives where this emphasis (i.e., the η parameter) between the two components of POI popularity and user interests can be automatically learned.

4.5 Experimental Methodology

In this section, we elaborate on the experimental dataset, baseline algorithms, and evaluation metrics that are used for our experimental evaluation.

4.5.1 Dataset

For our experiments, we use the Yahoo! Flickr Creative Commons 100M (YFCC100M) dataset [133, 151], which consists of 100M Flickr photos and videos. This dataset also comprises the meta information regarding the photos, such as the date/time taken, geo-location coordinates and accuracy of these geo-location coordinates. The geo-location accuracy range from world level (least accurate) to street level (most accurate).

Using the YFCC100M dataset, we extracted geo-tagged photos that were taken in ten different cities, namely: Toronto, Osaka, Glasgow, Budapest, Perth, Vienna, Delhi, Edinburgh, Tokyo and London. To ensure the best accuracy and generalizability of our results, we only chose photos with the highest geo-location accuracy and experimented on ten touristic cities around the world. A more detailed description of our dataset is shown in Table 4.1.[4]

[4]This dataset is also publicly available at https://sites.google.com/site/limkwanhui/datacode#ijcai15.

Table 4.1: Dataset description

City	No. of Photos	No. of Users	# POI Visits	# Travel Sequences
Toronto	157,505	1,395	39,419	6,057
Osaka	392,420	450	7,747	1,115
Glasgow	29,019	601	11,434	2,227
Budapest	36,000	935	18,513	2,361
Perth	18,462	159	3,643	716
Vienna	85,149	1,155	34,515	3,193
Delhi	13,919	279	3,993	489
Edinburgh	82,060	1,454	33,944	5,028
Tokyo	55,364	979	15,622	3,798
London	164,812	2,963	38,746	8,373

4.5.2 Baseline Algorithms

We compare our PERSTOUR algorithms against five different baseline algorithms, which can be divided into two broad categories. The first category is based on the popular Collaborative Filtering (CF) recommender systems [119, 154, 158], which utilizes a user's (tourist's) rating on the items (POIs) to recommend a set of item for another user based on their user similarities. Based on two definitions of user ratings, we implemented two variations of CF-based baseline algorithms, namely:

- **Collaborative Filtering based on Photos Uploaded (CF-PHO)**. The user/tourist's rating on each item/POI is based on the number of uploaded photos of that particular POI he/she has uploaded, i.e., a higher number of uploaded photos corresponds to a higher rating for that POI.

- **Collaborative Filtering based on POIs Visited (CF-BIN)**. The user/tourist's rating on each item/POI is based on whether they have visited that particular POI, i.e., a binary rating of 1 (visited) or 0 (not visited).

As CF-based algorithms recommend the top-K individual POIs instead of an itinerary of connected POIs, we implemented additional processing steps to ensure a consistent output result for our tour recommendation problem. Based on a starting POI p_1 (like our PERSTOUR algorithm), the CF-PHO and CF-BIN algorithms will iteratively add in the highest ranked POI from the top-K results, until either: (i) the budget B is exhausted; or (ii) the destination POI p_N is reached.

The second category of baseline algorithms are variations of greedy-based approaches that have also been used in other tour recommendation research [22, 24, 145]. Similar to our PERSTOUR approach, these baseline algorithms commence from a starting POI p_1 and iteratively choose a next POI to visit until either: (i) the budget B is exhausted; or (ii) the destination POI p_N is reached. The sequence of selected POIs thus forms the recommended itinerary, and the three greedy-based baselines are:

- **Greedy Nearest (GNEAR)**. Chooses the next POI to visit by randomly selecting from the three *nearest*, unvisited POIs.
- **Greedy Most Popular (GPOP)**. Chooses the next POI to visit by randomly selecting from the three *most popular*, unvisited POIs.
- **Random Selection (RAND)**. Chooses the next POI to visit by *randomly selecting* from all unvisited POIs.

GNEAR and GPOP are meant to reflect tourists' behavior by visiting nearby and popular POIs respectively, while RAND shows the performance of a random-based approach.

4.5.3 Evaluation

We evaluate PERSTOUR and the baselines using leave-one-out cross-validation [83], i.e., when evaluating a specific travel sequence of a user, we use this user's other travel sequences for training our algorithms. Specifically, we consider all real-life travel sequences with ≥3 POI visits and evaluate the algorithms using the starting POIs and destination POIs of these travel sequences. Thereafter, we evaluate the performance of each algorithm based on the recommended tour itinerary I using the following metrics:

1. **Tour Recall:** $T_R(I)$**.** The proportion of POIs in a user's real-life travel sequence that were also recommended in itinerary I. Let P_r be the set of POIs recommended in itinerary I and P_v be the set of POIs visited in the real-life travel sequence, tour recall is defined as: $T_R(I) = \frac{|P_r \cap P_v|}{|P_v|}$.

2. **Tour Precision:** $T_P(I)$**.** The proportion of POIs recommended in itinerary I that were also in a user's real-life travel sequence. Let P_r be the set of POIs recommended in itinerary I and P_v be the set of POIs visited in the real-life travel sequence, tour precision is defined as: $T_P(I) = \frac{|P_r \cap P_v|}{|P_r|}$.

3. **Tour F$_1$-score:** $T_{F_1}(I)$**.** The harmonic mean of both the recall and precision of a recommended tour itinerary I, defined as: $T_{F_1}(I) = \frac{2 \times T_P(I) \times T_R(I)}{T_P(I) + T_R(I)}$.

4. **Root-Mean-Square Error (RMSE) of POI Visit Duration:** $T_{RMSE}(I)$**.** The level of error between our recommended POI visit durations in itinerary I compared to the real-life POI visit durations taken by the users. Let $I^s \in I$ be the recommended POIs that were visited in real-life[5], and D_r and D_v be the recommended and real-life POI visit durations respectively, RMSE is defined as: $T_{RMSE}(I) = \sqrt{\frac{\sum_{p \in I^s} (D_r - D_v)^2}{|I^s|}}$.

5. **Tour Popularity:** $T_{Pop}(I)$**.** The overall popularity of all POIs in the recommended itinerary I, defined as: $T_{Pop}(I) = \sum\limits_{p \in I} Pop(p)$.

6. **Tour Interest:** $T^u_{Int}(I)$**.** The overall interest of all POIs in the recommended itinerary I to a user u, defined as: $T^u_{Int}(I) = \sum\limits_{p \in I} Int_u(Cat_p)$.

7. **Popularity and Interest Rank:** T^a_{Rk}**.** The average rank of an algorithm a based on its T_{Pop} and T_{Int} scores ranked against other algorithms (1=best, 12=worst).

We selected these metrics to better evaluate the following: (i) time-based versus frequency-based user interest, using Metrics 1-3; (ii) personalized versus non-personalized POI visit durations, using Metric 4; and (iii) PERSTOUR versus baselines, using Metrics 5-7. As personalized POI visit durations only apply to PERSTOUR and not the baselines, we only report T_{RMSE} scores for the PT-0, PT-.5F, PT-.5T, PT-.5U, PT-1F, PT-1T and PT-1U

[5]We can only compare POI visit durations for POIs in itinerary I that were "correctly" recommended (i.e., visited in real-life).

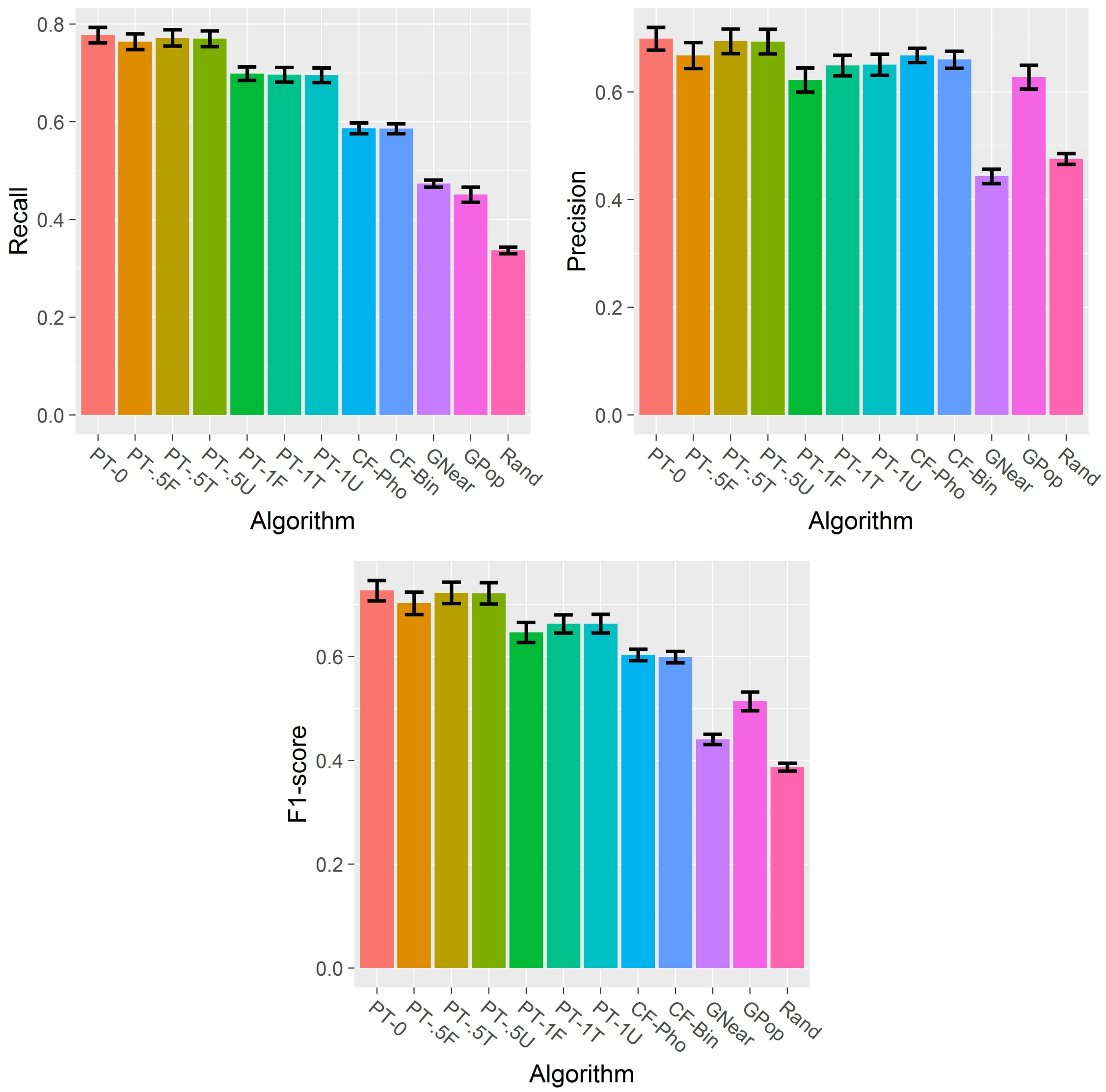


Figure 4.2: Overview of results (average scores) across all ten cities, in terms of Recall (T_R), Precision (T_P) and F_1-score (T_{F_1}).

algorithms. Our baseline for comparing T_{RMSE} are variants of PERSTOUR that use non-personalized POI visit durations, i.e., average POI visit durations.

4.6 Results and Discussion

In this section, we discuss our experimental results and highlight our main findings.

4.6.1 Comparison between Time-based and Frequency-based User Interest

Figure 4.2 presents an overview of results in terms of the average Recall (T_R), Precision (T_P) and F$_1$-score (T_{F_1}) across all ten cities, for the different variations of our PERSTOUR algorithm and the baselines. The results show that all variants of PERSTOUR out-perform the five baselines in terms of T_R and T_{F_1} scores. In terms of T_P scores, the PERSTOUR variants of PT-0, PT-.5T, PT-.5U and PT-.5F out-perform all baselines, while the CF-PHO and CF-BIN out-performs the PERSTOUR variants of PT-1T, PT-1U and PT-1F. We now examine the performance of our PERSTOUR algorithm and the baselines in greater detail.

Moving on to the results for individual cities, we study the performance difference between using time-based user interest and frequency-based user interest, as shown in Table 4.2, 4.3 and 4.4.[6] Comparing the T_{F_1} scores between PT-.5T, PT-.5U and PT-.5F, and between PT-1T, PT-1U and PT-1F, the results show that PERSTOUR using time-based user interest (PT-.5T, PT-.5U, PT-1T and PT-1U) out-performs its counterpart that uses frequency-based user interest (PT-.5F and PT-1F), in most cases. This observation highlights the effectiveness of time-based user interest in recommending tours that more accurately reflect real-life tours of users, compared to using frequency-based user interest. While PT-1T and PT-1U under-perform PT-1F in terms of T_R for some cities, we focus more on the T_{F_1} scores as it provides a balanced representation of both T_R and T_P. Moreover, for all cities, PT-.5T, PT-.5U, PT-1T and PT-1U (time-based user interest) result in higher T_P scores, compared to its PT-.5F and PT-1F counterparts (frequency-based user interest). Another observation is that all PERSTOUR variants also out-perform the five baselines for all cities, in terms of T_{F_1} scores.

The reason for the more accurate recommendations of time-based user interest compared to frequency-based user interest is due to its use of POI visit durations instead of POI visit frequency. Fig. 4.3 illustrates a toy example that highlights the difference between time-based user interest and frequency-based user interest. Consider user A who only visited two parks but spent three or more hours at each of them and user B who

[6] Some metrics are rounded off to the same value, but are different values before rounding. The bold-faced values indicate the best performing metrics.

Table 4.2: Comparison between Time-based User Interest (PT-.5T and PT-1T) and Frequency-based User Interest (PT-.5F and PT-1F), in terms of Recall (T_R), Precision (T_P) and F_1-score (T_{F_1}), for Toronto, Osaka, Glasgow and Edinburgh.

Toronto				**Osaka**			
Algo.	*Recall*	*Precision*	*F_1-score*	*Algo.*	*Recall*	*Precision*	*F_1-score*
PT-.5F	.760±.009	.679±.013	.708±.012	PT-.5F	.757±.025	.645±.037	.687±.032
PT-.5T	***.779±.010***	***.706±.013***	***.732±.012***	PT-.5T	***.759±.026***	***.662±.037***	***.699±.033***
PT-.5U	.773±.009	.698±.013	.726±.011	PT-.5U	***.759±.026***	***.662±.037***	***.699±.033***
PT-1F	.737±.010	.682±.013	.698±.012	PT-1F	.679±.023	.582±.032	.616±.027
PT-1T	***.744±.011***	***.710±.013***	***.718±.012***	PT-1T	***.683±.025***	***.622±.032***	***.641±.029***
PT-1U	.743±.011	.710±.013	.718±.012	PT-1U	***.683±.025***	***.622±.032***	***.641±.029***
CF-PHO	.593±.007	.650±.009	.605±.006	CF-PHO	.618±.018	.707±.034	.635±.018
CF-BIN	.589±.007	.682±.008	.619±.006	CF-BIN	.618±.018	.736±.031	.652±.017
GNEAR	.501±.010	.512±.015	.487±.011	GNEAR	.478±.026	.433±.038	.441±.030
GPOP	.440±.009	.623±.015	.504±.011	GPOP	.439±.034	.649±.038	.517±.035
RAND	.333±.007	.495±.011	.391±.009	RAND	.354±.021	.488±.032	.406±.024

Glasgow				**Edinburgh**			
Algo.	*Recall*	*Precision*	*F_1-score*	*Algo.*	*Recall*	*Precision*	*F_1-score*
PT-.5F	.819±.017	.758±.024	.780±.021	PT-.5F	.740±.006	.607±.010	.654±.009
PT-.5T	.826±.017	.782±.022	.798±.020	PT-.5T	.740±.007	.633±.010	.671±.008
PT-.5U	***.829±.017***	***.783±.022***	***.800±.020***	PT-.5U	***.743±.007***	***.635±.010***	***.674±.009***
PT-1F	***.748±.017***	.728±.022	.726±.019	PT-1F	***.678±.007***	.572±.009	.605±.008
PT-1T	.739±.018	.736±.021	.728±.019	PT-1T	.668±.007	.601±.009	.618±.008
PT-1U	.739±.018	***.739±.021***	***.730±.019***	PT-1U	.671±.007	***.602±.009***	***.621±.008***
CF-PHO	.600±.010	.720±.021	.631±.011	CF-PHO	.561±.006	.648±.007	.581±.005
CF-BIN	.604±.011	.709±.022	.627±.011	CF-BIN	.567±.006	.637±.008	.580±.005
GNEAR	.498±.020	.519±.028	.490±.022	GNEAR	.471±.007	.429±.010	.427±.008
GPOP	.418±.015	.592±.024	.480±.017	GPOP	.486±.008	.640±.010	.539±.008
RAND	.340±.012	.462±.017	.386±.013	RAND	.336±.006	.479±.009	.384±.006

Table 4.3: Comparison between Time-based User Interest (PT-.5T and PT-1T) and Frequency-based User Interest (PT-.5F and PT-1F), in terms of Recall (T_R), Precision (T_P) and F_1-score (T_{F_1}), for Budapest, Perth, Vienna and Delhi.

Budapest			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5F	.679±.008	.550±.011	.596±.010
PT-.5T	***.688±.008***	***.587±.011***	***.624±.009***
PT-.5U	.688±.008	.586±.011	.623±.009
PT-1F	***.633±.008***	.526±.010	.562±.009
PT-1T	.624±.009	***.571±.010***	***.587±.009***
PT-1U	.620±.009	.569±.010	.584±.009
CF-PHO	.542±.007	.598±.008	.550±.006
CF-BIN	.558±.007	.589±.008	.554±.006
GNEAR	.434±.007	.403±.011	.398±.008
GPOP	.408±.007	.538±.011	.450±.008
RAND	.300±.005	.442±.009	.349±.006

Perth			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5F	.798±.030	.697±.045	.735±.039
PT-.5T	***.809±.029***	***.725±.043***	***.757±.037***
PT-.5U	.792±.024	.723±.032	.749±.028
PT-1F	.746±.032	.660±.043	.691±.038
PT-1T	***.746±.030***	.674±.040	.699±.035
PT-1U	.736±.024	***.685±.030***	***.702±.027***
CF-PHO	.612±.022	.681±.024	.634±.019
CF-BIN	.605±.017	.621±.026	.601±.016
GNEAR	.463±.030	.432±.047	.428±.035
GPOP	.427±.029	.561±.038	.477±.031
RAND	.365±.024	.543±.039	.428±.028

Vienna			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5F	.711±.008	.600±.011	.636±.010
PT-.5T	.713±.009	.630±.011	.656±.010
PT-.5U	***.714±.009***	***.632±.011***	***.658±.010***
PT-1F	***.661±.007***	.559±.010	.589±.008
PT-1T	.651±.008	***.596±.010***	***.610±.009***
PT-1U	.651±.008	.593±.010	.607±.009
CF-PHO	.523±.007	.656±.009	.561±.006
CF-BIN	.515±.007	.661±.009	.559±.006
GNEAR	.469±.007	.429±.011	.426±.008
GPOP	.404±.008	.584±.012	.465±.009
RAND	.309±.006	.461±.010	.361±.007

Delhi			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5F	.800±.033	.718±.045	.748±.040
PT-.5T	.807±.036	.746±.045	.769±.041
PT-.5U	***.813±.035***	***.746±.045***	***.770±.041***
PT-1F	.671±.031	.611±.038	.631±.034
PT-1T	.674±.032	.632±.039	.648±.036
PT-1U	***.674±.032***	***.636±.041***	***.648±.036***
CF-PHO	.598±.027	.711±.048	.611±.026
CF-BIN	.593±.028	.700±.049	.603±.027
GNEAR	.506±.028	.422±.038	.449±.031
GPOP	.544±.032	.773±.039	.624±.032
RAND	.327±.020	.433±.026	.368±.021

Table 4.4: Comparison between Time-based User Interest (PT-.5T and PT-1T) and Frequency-based User Interest (PT-.5F and PT-1F), in terms of Recall (T_R), Precision (T_P) and F$_1$-score (T_{F_1}), for Tokyo and London.

Tokyo				London			
Algo.	*Recall*	*Precision*	*F$_1$-score*	*Algo.*	*Recall*	*Precision*	*F$_1$-score*
PT-.5F	.842±.014	.799±.018	.815±.016	PT-.5F	.737±.006	.625±.009	.664±.008
PT-.5T	***.852±.014***	***.813±.017***	***.828±.016***	PT-.5T	***.746±.007***	***.658±.009***	***.690±.008***
PT-.5U	.849±.014	.813±.018	.826±.016	PT-.5U	.744±.007	.657±.009	.688±.008
PT-1F	.755±.014	.720±.017	.732±.016	PT-1F	.679±.006	.581±.008	.612±.007
PT-1T	.763±.015	.736±.017	.745±.016	PT-1T	***.675±.007***	***.614±.008***	***.632±.007***
PT-1U	***.765±.015***	***.739±.018***	***.747±.016***	PT-1U	.674±.007	.612±.008	.631±.007
CF-PHO	.634±.012	.696±.017	.648±.011	CF-PHO	.589±.005	.612±.008	.573±.005
CF-BIN	.624±.009	.677±.017	.632±.009	CF-BIN	.588±.005	.590±.009	.559±.005
GNEAR	.469±.015	.459±.021	.454±.017	GNEAR	.450±.006	.396±.009	.402±.007
GPOP	.524±.014	.706±.021	.592±.016	GPOP	.421±.006	.609±.009	.488±.007
RAND	.355±.011	.495±.017	.407±.012	RAND	.353±.005	.458±.007	.389±.005

visited five parks but spent less than 15 minutes at each of them. Frequency-based interest incorrectly classifies user B as having more interest in the parks category due to his/her five visits, compared to user A's two visits. On the other hand, time-based interest more accurately determines that user A has a higher interest in the parks category due to his/her long visit duration, despite user A only visiting two parks. Furthermore, time-based interest can more accurately capture a user's level of interest based on how much longer this user spends at a POI compared to the average user (e.g., a user is more interested if he/she spends 3 hours at a POI when the average time spent is only 30 minutes). With the availability of user interest levels, we can better personalize POI visit duration for each unique user, which we evaluate next.

Table 4.5: Comparison between Personalized and Non-personalized Visit Durations, in terms of RMSE (T_{RMSE}), for Toronto, Osaka, Glasgow and Edinburgh.

Toronto

Algo.	*Visit Duration*	*RMSE*
PT-0	Personalized	***147.57±10.85***
	Non-personalized	152.44±9.84
PT-.5F	Personalized	***146.33±10.85***
	Non-personalized	152.61±10.09
PT-.5T	Personalized	***143.56±10.89***
	Non-personalized	150.65±10.09
PT-.5U	Personalized	***143.75±10.77***
	Non-personalized	151.67±10.19
PT-1F	Personalized	***137.07±11.40***
	Non-personalized	145.54±10.78
PT-1T	Personalized	***145.20±11.79***
	Non-personalized	148.18±11.29
PT-1U	Personalized	***141.53±11.75***
	Non-personalized	148.64±11.21

Osaka

Algo.	*Visit Duration*	*RMSE*
PT-0	Personalized	***51.35±11.41***
	Non-personalized	54.91±11.91
PT-.5F	Personalized	***56.71±12.43***
	Non-personalized	60.06±13.09
PT-.5T	Personalized	57.09±12.39
	Non-personalized	***55.84±13.18***
PT-.5U	Personalized	57.69±12.39
	Non-personalized	***55.84±13.18***
PT-1F	Personalized	***56.62±13.21***
	Non-personalized	62.24±14.60
PT-1T	Personalized	***53.44±13.05***
	Non-personalized	58.88±14.63
PT-1U	Personalized	***54.12±13.06***
	Non-personalized	58.88±14.63

Glasgow

Algo.	*Visit Duration*	*RMSE*
PT-0	Personalized	***75.98±11.53***
	Non-personalized	85.76±12.07
PT-.5F	Personalized	***88.20±13.03***
	Non-personalized	92.71±12.92
PT-.5T	Personalized	***76.40±11.34***
	Non-personalized	90.33±12.35
PT-.5U	Personalized	***77.14±11.52***
	Non-personalized	90.33±12.35
PT-1F	Personalized	***79.67±12.27***
	Non-personalized	86.24±12.85
PT-1T	Personalized	***73.29±11.94***
	Non-personalized	91.06±13.45
PT-1U	Personalized	***74.08±12.14***
	Non-personalized	90.04±13.44

Edinburgh

Algo.	*Visit Duration*	*RMSE*
PT-0	Personalized	***91.08±4.85***
	Non-personalized	113.15±5.21
PT-.5F	Personalized	***84.56±4.96***
	Non-personalized	99.54±5.14
PT-.5T	Personalized	***89.76±5.85***
	Non-personalized	100.15±5.27
PT-.5U	Personalized	***87.19±5.69***
	Non-personalized	101.29±5.30
PT-1F	Personalized	***69.61±5.04***
	Non-personalized	78.89±5.31
PT-1T	Personalized	***72.11±6.09***
	Non-personalized	74.48±5.29
PT-1U	Personalized	***71.54±5.89***
	Non-personalized	78.01±5.41

Table 4.6: Comparison between Personalized and Non-personalized Visit Durations, in terms of RMSE (T_{RMSE}), for Budapest, Perth, Vienna and Delhi.

Budapest			Perth		
Algo.	*Visit Duration*	*RMSE*	*Algo.*	*Visit Duration*	*RMSE*
PT-0	Personalized	***66.67**±**5.35***	PT-0	Personalized	***51.12**±**15.58***
	Non-personalized	68.32±3.46		Non-personalized	87.03±14.47
PT-.5F	Personalized	***64.79**±**5.56***	PT-.5F	Personalized	***54.15**±**16.62***
	Non-personalized	67.36±3.59		Non-personalized	73.23±13.80
PT-.5T	Personalized	***66.40**±**5.38***	PT-.5T	Personalized	***54.71**±**16.08***
	Non-personalized	68.61±3.78		Non-personalized	73.78±13.61
PT-.5U	Personalized	***67.51**±**5.56***	PT-.5U	Personalized	85.80±19.31
	Non-personalized	68.25±3.75		Non-personalized	***69.88**±**14.57***
PT-1F	Personalized	***64.61**±**5.71***	PT-1F	Personalized	***48.78**±**16.54***
	Non-personalized	67.79±3.92		Non-personalized	75.46±17.24
PT-1T	Personalized	***68.07**±**5.95***	PT-1T	Personalized	***52.84**±**16.51***
	Non-personalized	70.55±4.31		Non-personalized	78.74±16.49
PT-1U	Personalized	***68.84**±**6.28***	PT-1U	Personalized	85.85±21.75
	Non-personalized	70.32±4.30		Non-personalized	***82.07**±**14.86***

Vienna			Delhi		
Algo.	*Visit Duration*	*RMSE*	*Algo.*	*Visit Duration*	*RMSE*
PT-0	Personalized	***70.71**±**3.64***	PT-0	Personalized	***29.57**±**6.59***
	Non-personalized	73.81±3.70		Non-personalized	30.60±6.47
PT-.5F	Personalized	***64.87**±**3.24***	PT-.5F	Personalized	***27.58**±**5.73***
	Non-personalized	68.73±3.61		Non-personalized	31.12±6.61
PT-.5T	Personalized	***69.14**±**4.07***	PT-.5T	Personalized	***26.83**±**5.92***
	Non-personalized	70.22±4.55		Non-personalized	33.92±6.83
PT-.5U	Personalized	***69.68**±**4.63***	PT-.5U	Personalized	***27.25**±**5.73***
	Non-personalized	70.19±3.64		Non-personalized	33.92±6.83
PT-1F	Personalized	***59.92**±**3.88***	PT-1F	Personalized	***29.83**±**6.85***
	Non-personalized	61.37±4.10		Non-personalized	31.85±7.26
PT-1T	Personalized	64.64±4.41	PT-1T	Personalized	***30.02**±**7.06***
	Non-personalized	***62.96**±**4.98***		Non-personalized	35.51±7.76
PT-1U	Personalized	65.26±5.06	PT-1U	Personalized	***30.13**±**7.05***
	Non-personalized	***62.99**±**3.87***		Non-personalized	35.51±7.76

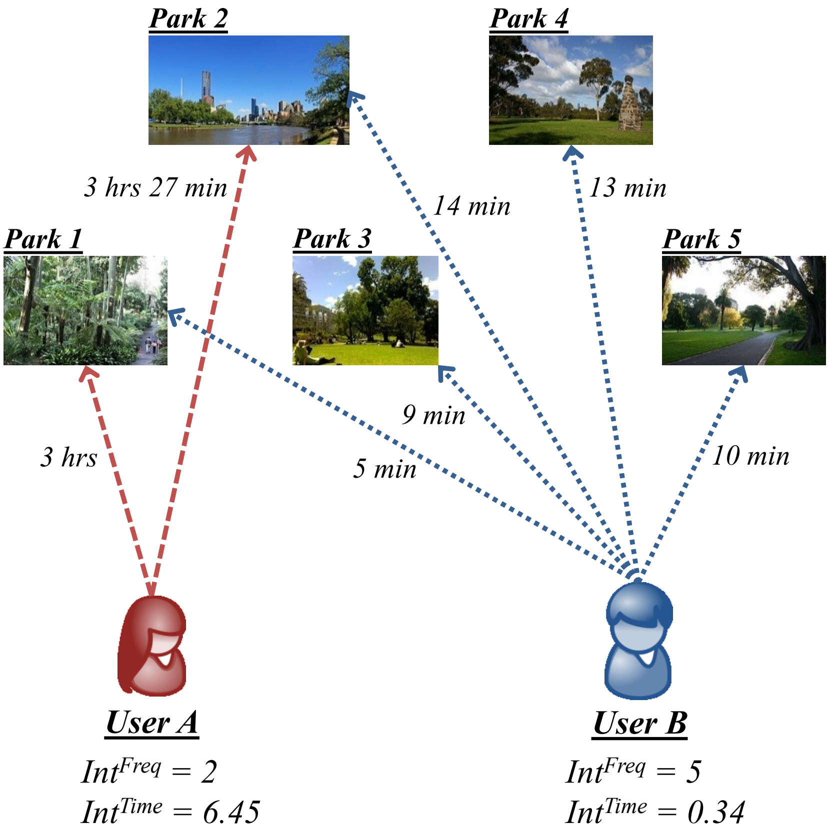


Figure 4.3: Toy example illustrating the difference between Time-based User Interest and Frequency-based User Interest

4.6.2 Comparison between Personalized and Non-personalized Visit Durations

The T_{RMSE} scores in Tables 4.5, 4.6 and 4.7 show that our recommendation of a personalized POI visit duration (Definitions 5 and 8) out-performs the non-personalized version in 62 out of 70 cases, based on a smaller error in the recommended POI visit durations. This result shows that personalizing POI visit duration using time-based user interests more accurately reflects the real-life POI visit duration of users, compared to the current standard of simply using average POI visit duration. Apart from recommending accurate POIs (T_{F_1} scores), recommending the appropriate amount of time to spend at each POI is another important consideration in tour recommendation, which has not been explored in earlier works that also aim to recommend tour itineraries.

Previously, we have observed how time-based interest results in more accurate POI

Table 4.7: Comparison between Personalized and Non-personalized Visit Durations, in terms of RMSE (T_{RMSE}), for Tokyo and London.

Tokyo			London		
Algo.	*Visit Duration*	*RMSE*	*Algo.*	*Visit Duration*	*RMSE*
PT-0	Personalized	***130.14±14.14***	PT-0	Personalized	***24.67±1.80***
	Non-personalized	142.51±10.22		Non-personalized	27.10±1.84
PT-.5F	Personalized	***117.78±10.19***	PT-.5F	Personalized	***25.08±1.86***
	Non-personalized	146.38±10.22		Non-personalized	26.64±1.91
PT-.5T	Personalized	***127.01±13.85***	PT-.5T	Personalized	***25.56±1.88***
	Non-personalized	144.51±10.36		Non-personalized	26.91±1.98
PT-.5U	Personalized	***130.25±14.07***	PT-.5U	Personalized	***25.41±1.90***
	Non-personalized	146.27±10.29		Non-personalized	26.92±1.98
PT-1F	Personalized	***112.26±10.05***	PT-1F	Personalized	***24.19±1.94***
	Non-personalized	144.63±10.52		Non-personalized	25.19±2.00
PT-1T	Personalized	***100.93±9.20***	PT-1T	Personalized	25.78±2.16
	Non-personalized	138.26±10.46		Non-personalized	***22.74±1.84***
PT-1U	Personalized	***106.84±9.54***	PT-1U	Personalized	26.27±2.21
	Non-personalized	139.03±10.42		Non-personalized	***22.83±1.83***

recommendations based on the T_{F_1} scores. Our personalized POI visit duration builds upon this success by customizing the POI visit duration to each unique user based on his/her relative interest level, i.e., spend more time in a POI that interests the user, and less time in a POI that the user is less interested in. Accurate POI visit durations have another important implication in tour recommendation, where spending less time at uninteresting POIs frees up the time budget for more visits to POIs that are more interesting to the user. Similarly, a user might prefer to spend more time visiting a few POIs of great interest, compared to visiting many POIs of less interest to the user.

4.6.3 Comparison of Popularity and Interest

We now present an overview of the results in terms of the average Popularity (T_{Pop}), Interest (T_{Int}) and Rank (T_{Rk}) score for all ten cities, as shown in Figure 4.4. In particular, we

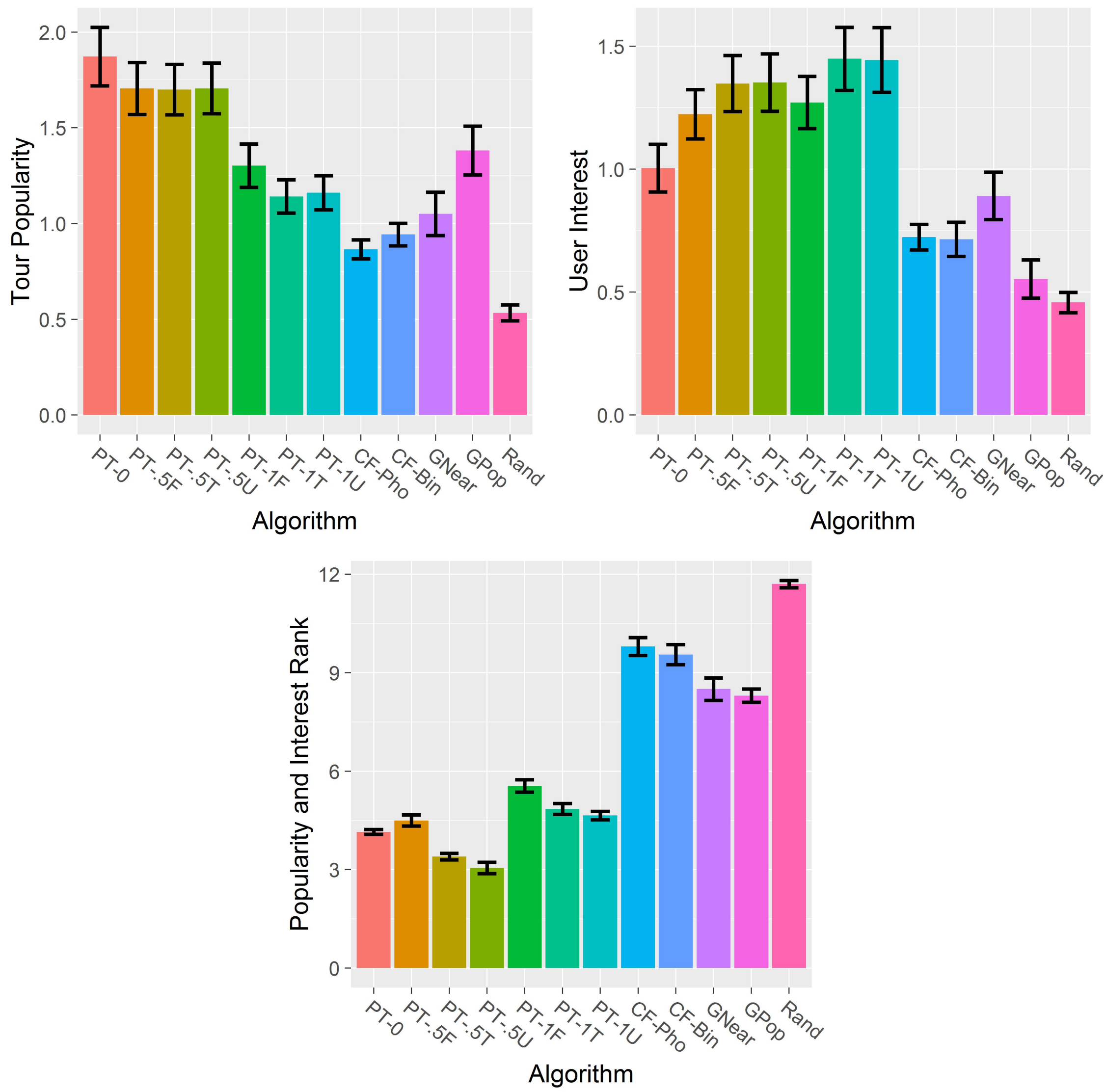


Figure 4.4: Overview of results (average scores) across all ten cities, in terms of Popularity (T_{Pop}), Interest (T_{Int}) and Rank (T_{Rk}). Number within brackets indicate the rank based on Popularity and Interest scores, where 1=best and 12=worst.

are most interested in the T_{Rk} score that is derived from the average rank of an algorithm based on its T_{Pop} and T_{Int} scores, compared to other algorithms. For T_{Rk} scores, a value of 1 indicates the best performance, while 12 indicates the worst performance. Based on this T_{Rk} score, all variants of PERSTOUR out-perform the five baselines, with PT-.5U being the best performer. We observe a similar performance in terms of T_{Int} scores, where all variants of PERSTOUR out-performing the baselines. In terms of T_{Pop} scores, the PERSTOUR variants of PT-0, PT-.5T, PT-.5U and PT-.5F out-perform all baselines, while PT-1T, PT-1U and PT-1F under-performs the GPOP baseline. This performance is understandable as the PT-1T, PT-1U and PT-1F algorithms emphasize fully on user interest preferences, while the GPOP baseline focuses on the most popular POIs thus maximizing the T_{Pop} scores for the latter. Next, we provide a more in-depth discussion of the performance among the various PERSTOUR variants.

Based on the T_{Rk} scores in Table 4.8, 4.9 and 4.10, we observe that PT-.5U (time-based user interest with weighted updates) is overall the best performer, and PT-.5T (time-based user interest) is the second best performer.[7] In addition, we also observe that PT-1U (time-based user interest with weighted updates) out-performs its PT-1F counterpart (frequency-based user interest) for eight out of ten cities, with the same performance for the remaining two cities. These results show the benefits of applying weighted updates to user interests (PT-.5U and PT-1U), compared to simply using time-based user interest without weighted updates (PT-.5T and PT-1T).

Next, we examine how PERSTOUR (with and without weighted updates) performs against the various baselines. Both PT-.5U and PT-.5T out-perform all baselines as well as its PT-.5F counterpart that uses frequency-based user interest. Similarly, PT-1T (time-based user interest) out-performs PT-1F (frequency-based user interest) for six out of ten cities, with the same performance for the remaining four cities. These results show the effectiveness of time-based user interest (both with and without weighted updates) over frequency-based user interest, based on the T_{Rk} scores.

The effects of the η parameter can be observed in the T_{Pop} and T_{Int} scores. A value of $\eta = 0$ (PT-0) results in the best performance in T_{Pop} and worst performance in T_{Int}, while a value of $\eta = 1$ (PT-1F, PT-1T and PT-1U) results in the opposite. While we include the T_{Pop} and T_{Int} scores for completeness, we are more interested in T_{Rk} as it gives a balanced measurement of both T_{Pop} and T_{Int}.

[7] PT-.5T out-performs PT-.5U in only one city, performs the same in five cities, and under-performs in the remaining four cities.

Table 4.8: Comparison of PersTour (PT) against baselines, in terms of Popularity (T_{Pop}), Interest (T_{Int}) and Rank (T_{Rk}), for Toronto, Osaka, Glasgow and Edinburgh. Number within brackets indicate the rank based on Popularity and Interest scores, where 1=best and 12=worst.

Toronto			
Algo.	*Popularity*	*Interest*	*Rk*
PT-0	2.204±.069 (1)	0.904±.048 (7)	4
PT-.5F	2.053±.063 (2)	1.088±.060 (6)	4
PT-.5T	1.960±.064 (4)	1.223±.061 (3)	***3.5***
PT-.5U	1.972±.063 (3)	1.195±.060 (4)	***3.5***
PT-1F	1.583±.048 (5)	1.137±.061 (5)	5
PT-1T	1.419±.044 (9)	1.351±.069 (1)	5
PT-1U	1.420±.043 (8)	1.319±.069 (2)	5
CF-Pho	0.926±.027 (11)	0.807±.042 (8)	9.5
CF-Bin	1.121±.028 (10)	0.572±.033 (10)	10
GNear	1.424±.049 (7)	0.773±.054 (9)	8
GPop	1.566±.050 (6)	0.443±.029 (12)	9
Rand	0.581±.032 (12)	0.467±.037 (11)	11.5

Osaka			
Algo.	*Popularity*	*Interest*	*Rk*
PT-0	1.263±.094 (1)	0.791±.166 (8)	4.5
PT-.5F	1.126±.095 (4)	1.151±.213 (5)	4.5
PT-.5T	1.144±.093 (3)	1.171±.206 (4)	3.5
PT-.5U	1.144±.093 (2)	1.176±.206 (3)	***2.5***
PT-1F	0.809±.075 (8)	1.137±.211 (6)	7
PT-1T	0.737±.067 (9)	1.205±.211 (2)	5.5
PT-1U	0.735±.066 (10)	1.207±.211 (1)	5.5
CF-Pho	0.823±.078 (7)	0.707±.136 (9)	8
CF-Bin	0.953±.076 (5)	0.661±.125 (10)	7.5
GNear	0.500±.059 (11)	0.853±.183 (7)	9
GPop	0.837±.062 (6)	0.223±.066 (12)	9
Rand	0.433±.055 (12)	0.305±.089 (11)	11.5

Glasgow			
Algo.	*Popularity*	*Interest*	*Rk*
PT-0	1.701±.101 (1)	0.459±.069 (8)	4.5
PT-.5F	1.562±.089 (4)	0.563±.091 (5)	4.5
PT-.5T	1.601±.089 (2)	0.625±.084 (4)	***3***
PT-.5U	1.594±.088 (3)	0.626±.084 (3)	***3***
PT-1F	1.128±.069 (6)	0.562±.090 (6)	6
PT-1T	1.001±.052 (7)	0.676±.096 (2)	4.5
PT-1U	0.978±.050 (8)	0.682±.096 (1)	4.5
CF-Pho	0.914±.046 (9)	0.434±.059 (9)	9
CF-Bin	0.874±.045 (10)	0.519±.071 (7)	8.5
GNear	0.874±.064 (11)	0.339±.070 (10)	10.5
GPop	1.399±.075 (5)	0.217±.049 (12)	8.5
Rand	0.483±.048 (12)	0.229±.041 (11)	11.5

Edinburgh			
Algo.	*Popularity*	*Interest*	*Rk*
PT-0	2.269±.046 (1)	1.047±.053 (7)	4
PT-.5F	2.016±.042 (2)	1.383±.068 (6)	4
PT-.5T	2.012±.043 (3)	1.579±.069 (3)	***3***
PT-.5U	2.003±.043 (4)	1.575±.070 (4)	4
PT-1F	1.541±.038 (6)	1.430±.070 (5)	5.5
PT-1T	1.336±.034 (8)	1.722±.076 (1)	4.5
PT-1U	1.355±.034 (7)	1.720±.076 (2)	4.5
CF-Pho	0.941±.023 (11)	0.752±.032 (9)	10
CF-Bin	1.056±.023 (10)	0.740±.032 (10)	10
GNear	1.269±.033 (9)	0.939±.054 (8)	8.5
GPop	1.775±.039 (5)	0.577±.033 (11)	8
Rand	0.656±.025 (12)	0.526±.033 (12)	12

Table 4.9: Comparison of PERSTOUR (PT) against baselines, in terms of Popularity (T_{Pop}), Interest (T_{Int}) and Rank (T_{Rk}), for Budapest, Perth, Vienna and Delhi. Number within brackets indicate the rank based on Popularity and Interest scores, where 1=best and 12=worst.

Budapest			
Algo.	*Popularity*	*Interest*	*Rk*
PT-0	2.921±.075 (1)	1.366±.075 (7)	4
PT-.5F	2.619±.070 (3)	1.596±.081 (6)	4.5
PT-.5T	2.614±.069 (4)	1.859±.087 (4)	4
PT-.5U	2.622±.069 (2)	1.877±.088 (3)	***2.5***
PT-1F	2.090±.064 (6)	1.708±.090 (5)	5.5
PT-1T	1.687±.050 (9)	2.076±.091 (2)	5.5
PT-1U	1.687±.051 (8)	2.109±.096 (1)	4.5
CF-PHO	1.243±.026 (11)	0.919±.045 (10)	10.5
CF-BIN	1.309±.027 (10)	1.114±.051 (9)	9.5
GNEAR	1.746±.057 (7)	1.148±.068 (8)	7.5
GPOP	2.209±.053 (5)	0.900±.050 (11)	8
RAND	0.805±.032 (12)	0.572±.040 (12)	12

Perth			
Algo.	*Popularity*	*Interest*	*Rk*
PT-0	1.854±.154 (1)	1.338±.206 (8)	4.5
PT-.5F	1.732±.146 (4)	1.426±.209 (7)	5.5
PT-.5T	1.744±.152 (3)	1.518±.209 (4)	3.5
PT-.5U	1.773±.127 (2)	1.566±.180 (3)	***2.5***
PT-1F	1.317±.136 (6)	1.490±.218 (5)	5.5
PT-1T	1.170±.131 (8)	1.663±.219 (1)	4.5
PT-1U	1.313±.121 (7)	1.640±.189 (2)	4.5
CF-PHO	0.824±.065 (11)	0.923±.094 (10)	10.5
CF-BIN	0.942±.071 (10)	0.926±.116 (9)	9.5
GNEAR	0.958±.115 (9)	1.430±.186 (6)	7.5
GPOP	1.401±.115 (5)	0.851±.115 (11)	8
RAND	0.529±.077 (12)	0.617±.103 (12)	12

Vienna			
Algo.	*Popularity*	*Interest*	*Rk*
PT-0	1.781±.045 (1)	1.067±.069 (7)	4
PT-.5F	1.550±.043 (4)	1.385±.083 (6)	5
PT-.5T	1.563±.043 (3)	1.559±.085 (4)	3.5
PT-.5U	1.571±.043 (2)	1.595±.086 (3)	***2.5***
PT-1F	1.234±.041 (6)	1.476±.088 (5)	5.5
PT-1T	1.011±.032 (9)	1.676±.087 (2)	5.5
PT-1U	1.030±.033 (8)	1.711±.087 (1)	4.5
CF-PHO	0.676±.018 (11)	0.521±.031 (11)	11
CF-BIN	0.687±.017 (10)	0.382±.023 (12)	11
GNEAR	1.040±.037 (7)	0.957±.070 (8)	7.5
GPOP	1.399±.037 (5)	0.605±.038 (9)	7
RAND	0.470±.021 (12)	0.536±.040 (10)	11

Delhi			
Algo.	*Popularity*	*Interest*	*Rk*
PT-0	1.744±.148 (1)	0.628±.157 (7)	4
PT-.5F	1.620±.142 (2)	0.839±.208 (6)	4
PT-.5T	1.610±.133 (4)	0.954±.252 (3)	***3.5***
PT-.5U	1.620±.135 (3)	0.945±.248 (4)	***3.5***
PT-1F	1.142±.119 (6)	0.923±.238 (5)	5.5
PT-1T	1.129±.089 (8)	1.000±.257 (1)	4.5
PT-1U	1.136±.093 (7)	0.964±.252 (2)	4.5
CF-PHO	0.773±.062 (10)	0.605±.132 (9)	9.5
CF-BIN	0.773±.058 (11)	0.618±.134 (8)	9.5
GNEAR	1.056±.106 (9)	0.524±.120 (10)	9.5
GPOP	1.167±.102 (5)	0.419±.094 (11)	8
RAND	0.431±.059 (12)	0.356±.117 (12)	12

Table 4.10: Comparison of PERSTOUR (PT) against baselines, in terms of Popularity (T_{Pop}), Interest (T_{Int}) and Rank (T_{Rk}), for Tokyo and London. Number within brackets indicate the rank based on Popularity and Interest scores, where 1=best and 12=worst.

Tokyo				London			
Algo.	*Popularity*	*Interest*	*Rk*	*Algo.*	*Popularity*	*Interest*	*Rk*
PT-0	1.396±.051 (1)	1.256±.108 (7)	4	PT-0	1.592±.034 (1)	1.191±.055 (7)	4
PT-.5F	1.335±.049 (4)	1.379±.118 (6)	5	PT-.5F	1.442±.032 (2)	1.426±.062 (6)	4
PT-.5T	1.341±.049 (3)	1.420±.114 (3)	***3***	PT-.5T	1.405±.031 (4)	1.578±.065 (3)	***3.5***
PT-.5U	1.345±.048 (2)	1.409±.115 (4)	***3***	PT-.5U	1.412±.031 (3)	1.567±.065 (4)	***3.5***
PT-1F	1.098±.119 (5)	1.388±.118 (5)	5	PT-1F	1.088±.029 (5)	1.464±.062 (5)	5
PT-1T	1.023±.089 (7)	1.451±.119 (1)	4	PT-1T	0.904±.023 (9)	1.672±.067 (1)	5
PT-1U	1.042±.051 (6)	1.434±.119 (2)	4	PT-1U	0.909±.023 (8)	1.656±.066 (2)	5
CF-PHO	0.737±.033 (10)	0.725±.055 (10)	10	CF-PHO	0.804±.015 (10)	0.845±.031 (10)	10
CF-BIN	0.948±.036 (9)	0.695±.058 (11)	10	CF-BIN	0.766±.013 (11)	0.922±.035 (9)	10
GNEAR	0.694±.044 (11)	0.905±.101 (8)	9.5	GNEAR	0.953±.026 (7)	1.050±.052 (8)	7.5
GPOP	1.006±.043 (8)	0.820±.077 (9)	8.5	GPOP	1.063±.023 (6)	0.481±.024 (12)	9
RAND	0.356±.028 (12)	0.396±.045 (12)	12	RAND	0.597±.019 (12)	0.579±.030 (11)	11.5

4.6.4 Comparison of PersTour with Adaptive Weights

To evaluate the effectiveness of using adaptive weights, we compare PERSTOUR without adaptive weights (PT-.5U and PT-1U) against PERSTOUR with adaptive weights (PT-AS and PT-AC). We focus mainly on the top and bottom 15% of users of each city, based on their number of total POI visits. The reason for choosing these users is that adaptive weights are most beneficial to such outlier users as we can recommend more personalized tours to users with many POI visits and popular tours to users with little POI visits.

Our main evaluation metrics are the T_R, T_P and T_{F_1} scores as they indicate the effectiveness of adaptive weights in recommending tours that correspond to real-life visits. Table 4.11 shows that PT-AS has the overall best performance as indicated by the highest T_R, T_P and T_{F_1} scores for seven, five and six cities, respectively, out of all ten cities. These results show the effectiveness of implementing adaptive weights for different users, i.e., a different level of emphasis between POI popularity and user interest preferences for different users.

Table 4.11: Comparison between PersTour with Weighted Updates and PersTour with Adaptive Weightings, in terms of Recall (T_R), Precision (T_P) and F_1-score (T_{F_1}).

Toronto			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	***.779±.013***	.698±.017	***.728±.015***
PT-1U	.744±.014	***.707±.017***	.716±.015
PT-AS	.767±.012	.685±.017	.715±.015
PT-AC	.766±.013	.700±.017	.723±.015

Osaka			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	.765±.034	.654±.056	.694±.047
PT-1U	.706±.035	.617±.048	.648±.042
PT-AS	***.765±.034***	***.667±.054***	***.702±.046***
PT-AC	.746±.038	.654±.056	.684±.048

Glasgow			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	***.837±.026***	***.781±.036***	***.802±.032***
PT-1U	.732±.027	.718±.032	.715±.029
PT-AS	.831±.025	.767±.036	.789±.032
PT-AC	.831±.026	.775±.035	.796±.031

Edinburgh			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	.722±.010	.583±.013	.634±.012
PT-1U	.682±.010	.566±.012	.606±.011
PT-AS	***.736±.010***	***.595±.014***	***.646±.012***
PT-AC	.723±.010	.592±.014	.640±.012

Budapest			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	.695±.014	.573±.018	.617±.016
PT-1U	.606±.014	.549±.016	.568±.015
PT-AS	***.696±.013***	.574±.018	***.619±.016***
PT-AC	.664±.015	***.579±.018***	.610±.016

Perth			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	.756±.027	.670±.037	.703±.032
PT-1U	.732±.026	.660±.033	.687±.029
PT-AS	***.777±.028***	***.695±.038***	***.726±.033***
PT-AC	.748±.027	.667±.035	.699±.031

Vienna			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	.742±.012	.630±.017	.670±.015
PT-1U	.663±.012	.591±.015	.614±.013
PT-AS	***.744±.012***	.628±.017	.668±.015
PT-AC	.730±.013	***.645±.017***	***.674±.015***

Delhi			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	.750±.056	.639±.073	.677±.065
PT-1U	.665±.056	.600±.072	.624±.066
PT-AS	***.771±.058***	***.656±.078***	***.694±.070***
PT-AC	.722±.057	.637±.075	.670±.068

Tokyo			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	***.812±.021***	.758±.029	.777±.025
PT-1U	.758±.023	.717±.028	.732±.025
PT-AS	.807±.021	.753±.028	.773±.025
PT-AC	.808±.022	***.764±.029***	***.779±.026***

London			
Algo.	*Recall*	*Precision*	*F_1-score*
PT-.5U	.714±.009	***.602±.012***	***.643±.011***
PT-1U	.675±.009	.589±.011	.618±.010
PT-AS	***.718±.009***	.597±.012	.641±.011
PT-AC	.704±.009	.600±.013	.639±.011

4.7 Summary

We modelled our tour recommendation problem based on the Orienteering problem and proposed the PERSTOUR algorithm for recommending personalized tours. Our PERSTOUR algorithm considers both POI popularity and user interest preferences to recommend suitable POIs to visit and the amount of time to spend at each POI. In addition, we implemented a framework where geo-tagged photos can be used to automatically detect real-life travel sequences, and determine POI popularity and user interest, which can then be used to train our PERSTOUR algorithm. Our work improves upon earlier tour recommendation research in three main ways: (i) we introduce *time-based user interest* derived from a user's visit durations at specific POIs relative to other users, instead of using a frequency-based user interest based on POI visit frequency; (ii) we *personalize POI visit duration* based on the relative interest levels of individual users, instead of using the average POI visit duration for all users or not considering POI visit duration at all; and (iii) we introduce two adaptive weighting methods to automatically determine the emphasis on POI popularity and user interest preferences.

Using a Flickr dataset across ten cities, we evaluate the effectiveness of our PERSTOUR algorithm against various collaborative filtering and greedy-based baselines, in terms of tour popularity, interest, precision, recall, F_1-score, and RMSE of visit duration. In particular, our experimental results show that: (i) using time-based user interest results in tours that more accurately reflect the real-life travel sequences of users, compared to using frequency-based user interest, based on precision and F_1-score; (ii) our personalized POI visit duration more accurately reflects the time users spend at POIs in real-life, compared to the current standard of using average visit duration, based on the RMSE of visit duration; (iii) PERSTOUR and its variants out-perform all baselines in most cases, based on tour popularity, interest, precision, recall and F_1-score; and (iv) our adaptive weighting methods further improve the performance of PERSTOUR, based on precision, recall and F_1-score. In the next chapter, we further investigate how we can recommend personalized tours, while optimizing for the conflicting objectives of POI popularity and queuing times, which varies based on the time of POI visit.

Chapter 5

Queue-aware Tour Recommendation

Personalized itinerary recommendation is a complex and time-consuming problem, due to the need to recommend popular attractions that are aligned to the interest preferences of a tourist, and to plan these attraction visits as an itinerary that has to be completed within a specific time limit. Furthermore, many existing itinerary recommendation systems do not automatically determine and consider queuing times at attractions in the recommended itinerary, which varies based on the time of visit to the attraction, e.g., longer queuing times at peak hours. To solve these challenges, we propose the PersQ algorithm for recommending personalized itineraries that take into consideration attraction popularity, user interests and queuing times. We also implement a framework that utilizes geo-tagged photos to derive attraction popularity, user interests and queuing times, which PersQ uses to recommend personalized and queue-aware itineraries. We demonstrate the effectiveness of PersQ in the context of five major theme parks, based on a Flickr dataset spanning nine years. Experimental results show that PersQ outperforms various state-of-the-art baselines, in terms of various queuing-time related metrics, itinerary popularity, user interest alignment, recall, precision and F1-score.

5.1 Introduction

A critical task for any tourist is to plan a trip itinerary that comprises popular and interesting attractions, which can be completed within a specific time limit. This task is especially complex and challenging due to: (i) the need to identify a set of popular attractions that are also aligned with the traveller's interests; (ii) the need to organize these

This chapter is derived from the following publication:

- **Kwan Hui Lim**, Jeffrey Chan, Shanika Karunasekera and Christopher Leckie. Personalized Itinerary Recommendation with Queuing Time Awareness. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'17)*. Accepted to appear (10pp). Aug 2017.

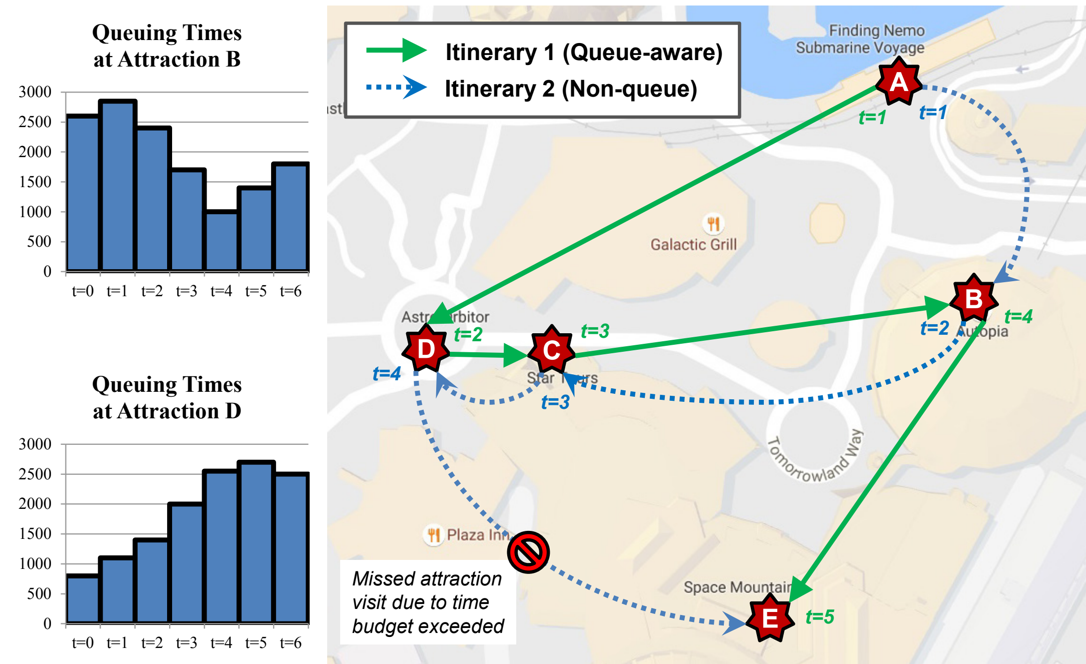


Figure 5.1: Example of itinerary recommendation with queue time consideration

attractions in the form of an itinerary with the constraints of a starting/ending place (e.g., near a traveller's hotel) and limited time for touring; and (iii) the need to plan for travelling, visiting and queuing times at the attractions, where queuing times are dependent on the time of attraction visit. In particular, neglecting to account for queuing times can create a frustrating experience for travellers as they spend an unnecessarily long time queuing instead of enjoying the attractions, and possibly miss attraction visits in their itineraries due to these queuing times exceeding their available touring time.

Figure 5.1 illustrates the importance of queuing time awareness in itinerary recommendation. Itinerary 2 (dotted blue line) does not consider queuing time and recommends an attraction visit sequence of $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$. However, this itinerary is unable to complete the visit to Attraction E, due to excessive queuing times at Attractions B and D that causes the time budget to be exceeded. Moreover, Itinerary 2 schedules visits to Attractions B and D at the peak (longest) of their queuing times. In contrast, Itinerary 1 (solid green line) considers queuing time and recommends that attractions be visited in the sequence of $A \rightarrow D \rightarrow C \rightarrow B \rightarrow E$, as Attraction D has a lower queuing time at $t = 2$, compared to Attraction B at the same time. By the time Itinerary 1 visits

Attraction B (at $t = 4$), its queuing time has shortened drastically, compared to earlier at $t = 2$. In addition, Itinerary 1 schedules visits to Attractions B and D when queues are the shortest, thus making the itinerary more enjoyable with less time wasted on queuing. In real-life, itinerary recommendation is even more complex due to: (i) the need to consider attraction popularity and user interest preferences; and (ii) since queuing times are time-dependent, itinerary recommenders need to also consider the time of visit to attractions and their queuing times.

In this work, we aim to address these issues and propose the PERSQ algorithm for recommending itineraries that are popular, personalized to user interests, and minimize queuing times at attractions. To the best of our knowledge, our work is the first to use geo-tagged photos to automatically determine and incorporate queuing time in personalized itinerary recommendations. Our main contributions are:

- We introduce and formulate the QUEUETOURREC problem for recommending personalized itineraries that aim to maximize attraction popularity and user interest preferences and minimize queuing times, while adhering to a time constraint for completing the itinerary (Section 5.3).

- We propose the PERSQ algorithm, where we have developed a novel implementation of Monte Carlo Tree Search (MCTS) for the domain of personalized itinerary recommendation with queuing time awareness (Section 5.5).

- We implement a framework for automatically extracting attraction popularity, user interests and queuing time from geo-tagged photos (Section 5.6). Using this framework, we have collected a dataset of user visits in five major theme parks across nine years, which is made publicly available at `https://sites.google.com/site/limkwanhui/datacode#sigir17` (Section 5.7.1).

- We introduce various queuing-related evaluation metrics and compare PERSQ against various state-of-the-art baselines. The results show that PERSQ recommends itineraries with the lowest ratio of queuing time to total touring time, and schedules visits to popular attractions at times with the shortest queues. In addition, PERSQ offered the best overall performance in terms of itinerary popularity, user interest, precision, recall and F1-scores (Sections 5.7 and 5.8).

For the rest of this chapter, Section 5.2 discusses related work in itinerary recommendation, while Section 5.4 provides some background on MCTS.

5.2 Related Work

Tour Itinerary Recommendation. Many early works on itinerary recommendation originate from path planning problems such as the Orienteering problem, which typically recommends a generic itinerary for the general population based on a global utility such as POI popularity. Thereafter, more recent works proposed algorithms that aim to recommend itineraries that are personalized to users based on their interest preferences. We previously discussed a large selection of these works and refer readers to Chapter 2 for a more comprehensive review.

Top-k POI Recommendation. An adjacent research area to itinerary recommendation are works on top-k POI recommendation, which has been extensively studied by the Information Retrieval community [88, 93, 153, 154, 158]. Most of these works are based on matrix factorization or collaborative filtering approaches and their main objective is to recommend a ranked list of top-k POIs to a user. To achieve this purpose, they utilize various sources of information such as social links [153], activity/user types [88], and geographical/temporal aspects of POIs [93,154,158]. Although these works propose interesting uses of various POI information, the task of top-k POI recommendation is distinctly different from itinerary recommendation, which encompasses the added complexity of constructing a set of relevant POIs as a time-constrained connected itinerary with the consideration of travelling time and POI visit durations, among others.

Discussion of Related Work. While these earlier works are the state-of-the-art in itinerary and POI recommendation, our work differs from these earlier works in several important aspects, namely: (i) our QueueTourRec problem optimizes for attraction popularity, user interest preferences and queuing times distribution, which is automatically constructed based on geo-tagged photos and then included as part of the travelling

Table 5.1: Description of Key Notation

Notation	**Description**
A	The set of all attractions
a_x	A specific attraction, where $a_x \in A$
Cat_{a_x}	The category of attraction a_x
Dur_{a_x}	The visit duration at attraction a_x
$Trav_{a_x,a_y}$	Travelling time from a_x to a_y
$t^a_{a_x}$	The arrival time at attraction a_x
$t^d_{a_x}$	The departure time from attraction a_x
V_x	An user-attraction visit, $V_x = (a_x, t^a_{a_x}, t^d_{a_x})$
$Pop(a_x)$	The popularity of attraction a_x
$Int_u(c)$	The interest of user u in attraction a_x
$Queue^t(a_x)$	The queuing time at attraction a_x at time t

cost along with transport time and attraction visit duration; (ii) unlike top-k POI recommendations, our QUEUETOURREC problem involves recommending a set of POIs and planning it as a connected itinerary with the associated time constraints of travelling between POIs, visiting POIs and completing the itinerary within a time budget; and (iii) while MCTS is typically used for board game playing [27,45][1], we propose a novel and non-trivial adaptation of MCTS (denoted the PERSQ algorithm) for solving the QUEUETOURREC problem.

5.3 Background and Problem Definition

Table 5.1 highlights the key notation used in our work, which we elaborate more in the next section.

[1]More background on MCTS is provided later in Section 5.4.

5.3.1 Preliminaries

For each theme park, there are m attractions, and we represent this set of attractions as $A = \{a_1, ..., a_m\}$. Each attraction a_x is also associated with its latitude/longitude coordinates, category Cat_{a_x} (e.g., roller coaster, water rides, indoor), and the duration Dur_{a_x} needed to visit that attraction (e.g., 10 minutes for a roller coaster). For itinerary recommendation in theme parks, the primary mode of transportation is by walking. Thus, we use $Trav_{a_x,a_y}$ to denote the amount of time taken to walk from attractions a_x to a_y, which is derived from a leisure walking speed of 5km/hour [28] to cover the distance between attractions a_x and a_y.

While we define our problem in the context of a theme park, these definitions and problem formulation are generalizable to any general path planning related context. For example, in the context of a city, there are also various attractions that a tourist can visit, where each attraction is also associated with a specific category (e.g., shopping, beach, park) and visit duration. The travelling speed $Trav_{a_x,a_y}$ can also modified to reflect the transport modes of walking, cycling or driving. Similarly, in the context of a museum, the attractions would be in the form of an artifact or painting, each also associated with a specific category (e.g., classical, modern, gothic) and visit duration.

Definition 1: Attraction Visits. For a user u who has visited n attractions, we represent each visited attraction a_x as a tuple, $V_x = (a_x, t^a_{a_x}, t^d_{a_x})$. In this tuple, a_x denotes the visited attraction, while $t^a_{a_x}$ and $t^d_{a_x}$ denote the time that user u arrived at and departed from attraction a_x, respectively. Given $t^a_{a_x}$ and $t^d_{a_x}$, we can then derive the amount of time that user u spent at attraction a_x, by calculating the difference between the arrival and departure times, i.e., $t^a_{a_x}$ and $t^d_{a_x}$. Note that this time spent includes both the queuing time at attraction a_x and its visiting duration Dur_{a_x}. We later describe (in Definition 5) how these queuing times can be calculated.

Definition 2: Visit Sequence. Given the set of visited attractions (i.e., multiple tuples of $V_x = (a_x, t^a_{a_x}, t^d_{a_x})$) by a user u, we can construct his/her visit sequence by joining consecutive attraction visits to form a chronologically-ordered sequence, $S_u = (V_1, V_2, ..., V_n)$. A visit sequence is split into two separate visit sequences if two consecutive attraction visits take place more than 8 hours apart, i.e., $t^d_{a_x} - t^a_{a_x+1} > 8$ *hours*. Other researchers have adopted a similar approach to divide such visit sequences based on an interval of 8 hours [40]. This list of visit sequences subsequently serves as the ground truth of real-life theme park visits, which is used as part of our experimentation and evaluation of the various algorithms and baselines.

Definition 3: Popularity of Attractions. Similar to many itinerary recommendation works [22, 24, 40], we utilize a simple but effective measure of attraction popularity $Pop(a)$, which is defined as the number of times an attraction is visited. This popularity measure was previously described in Section 3.3.1, where we refer the reader to, for more details.

Definition 4: User-Attraction Interest Relevance. Unlike attraction popularity that is common to all users, the interest relevance of an attraction differs from user to user and each user has their own unique interest preferences. We denote the interest level of a user u in category c as $Int_u(c)$, which is based on the number of photos of attractions that belong to category c, relative to the total number of photos taken by user u. The intuition behind this definition is that a user is more likely to take more photos of an attraction (category) that interests him/her, and less photos otherwise. For more details, refer to Section 3.3.1, where this definition was previously introduced.

Definition 5: Queuing Times of Attractions. In general, each attraction is also associated with a certain queuing time before a user can visit the attraction, e.g., queuing to ride a roller coaster or to buy tickets for a show. Given that V_a is the set of visits to attraction a and t is the visiting time, we represent the queuing times at an attraction a as:

$$Queue^t(a) = \frac{1}{|V_a|} \sum_{u \in U} \sum_{a_x \in S_u} \delta(a_x = a)\big((t^d_{a_x} - t^a_{a_x}) - Dur_{a_x}\big) \tag{5.1}$$

where $\delta(a_x{=}a) = 1$ if attraction a_x is the same as attraction a, and $\delta(a_x{=}a) = 0$ otherwise. In short, we calculate the queuing time at an attraction a based on the total time spent at an attraction (which includes queuing time and visit duration), subtracted by its attraction visit duration (e.g., 10 minutes for a roller coaster). We derive this total time spent based on the departure time $t^d_{a_x}$ and arrival time $t^a_{a_x}$, which we then subtract by the attraction visiting time Dur_{a_x}. For simplicity, we consider the time t of $Queue^t(a)$ in terms of the hour of visit and use the average visit duration of attraction a at a time t to derive its queuing time at time t.[2]

5.3.2 Problem Definition

We first define the problem of recommending a personalized itinerary to a single user, with the main objectives of maximizing the popularity and interest relevance of recommended attractions in this itinerary, while minimizing the queuing times at these attractions. We denote this problem as the QUEUETOURREC problem. Our formulation of QUEUETOURREC is modelled based on a variant of the Orienteering problem [136, 142].

Given the set of attractions A, a time budget B, a starting point a_1 and destination point a_N, our objective is to recommend an itinerary $I = (a_1, ..., a_N)$ that maximizes the total reward $Rwrd(I)$ accumulated from the recommended itinerary I, while ensuring that the itinerary is completed within the time budget B. The reward function $Rwrd(I)$ is calculated based on the popularity, interest relevance and queuing times of the attractions $a_x \in A$ recommended in itinerary I, as denoted by $Pop(a_x)$, $Int(Cat_{a_x})$ and $Queue(a_x)$, respectively. The time budget B is calculated using the function $Cost(a_x, a_y) = Trav(a_x, a_y) + Dur(a_y) + Queue^t(a_y)$, i.e., considering for travelling time, attraction/ride duration and queuing time at $time = t$.

Unlike earlier works that do not automatically determine and consider queuing times, our work differs in the following ways: (i) we derive a queuing time distribution for each attraction (i.e., expected queuing duration at specific times), based on past tourist visits; (ii) we incorporate these queuing times into the itinerary based on the proposed visit time at each attraction; (iii) we recommend an itinerary that optimizes for attraction popularity, user interests and queuing times; and (iv) in contrast to the Orienteering problem with fixed costs, QUEUETOURREC includes a time-dependent cost that affects the chosen attractions and queuing times, thus we cannot utilize traditional algorithms designed for the former.

In essence, our main goal is to plan an itinerary $I = (a_1, ..., a_N)$ that maximizes the

[2] Although we consider t in terms of the hour of visit, t can be easily generalized to smaller or larger units of time (e.g., 30 min or 2-hourly blocks)

following objective function:

$$\max \sum_{a_i \in I} \sum_{a_j \in I, a_i \neq a_j} x^t_{a_i,a_j} \Big(\frac{Int(Cat_{a_i}) + Pop(a_i)}{Queue^t(a_i)} \Big) \tag{5.2}$$

where $x^t_{a_i,a_j} = 1$ if a path from attraction a_i to a_j is taken at time t (i.e., we visit attractions a_i and a_j in sequence), and $x^t_{a_i,a_j} = 0$ otherwise.

Following which, we then attempt to solve for Eqn. 5.2, subject to the following constraints:

$$\sum_{a_i \in I, i \neq 1} x^t_{a_1,a_i} = \sum_{a_j \in I, j \neq N} x^{t+d}_{a_j,a_N} = 1 \tag{5.3}$$

$$\sum_{a_i \in I, k \neq N} x^t_{a_i,a_k} = \sum_{a_j \in I, k \neq 1} x^{t+d}_{a_k,a_j} \leq 1 \tag{5.4}$$

$$\sum_{a_i \in I} \sum_{a_j \in I, a_i \neq a_j} x^t_{a_i,a_j} Cost^t(a_i, a_j) \leq B \tag{5.5}$$

where the cost function used in Equation 5.5 is defined as:

$$Cost(a_i, a_j) = Trav_{a_i,a_j} + Dur_{a_j} + Queue^t(a_j) \tag{5.6}$$

Equation 5.2 is a multi-objective function that aims to recommend attractions in an itinerary I that maximizes the popularity and interest relevance of all visited attractions, while minimizing the queuing times at these attractions. Equation 5.2 can also be enhanced with different weights for $Int(Cat_{a_i})$, $Pop(a_i)$ and $Queue^t(a_i)$ to provide varying levels of emphasis on each component. The optimization of Equation 5.2 is also subjected to Constraints 5.3 to 5.5. Constraint 5.3 ensures that the recommended itinerary begins from a specific attraction a_1, while terminating at attraction a_N.[3] Thereafter, Constraint 5.4 ensures that all paths in the recommended itinerary are connected and no attractions are visited multiple times. Constraint 5.5 ensures that the recommended itinerary can be completed within the time budget B. Specifically, the function $Cost(a_x, a_y)$ (i.e., Equa-

[3]For a more general itinerary recommendation task, such as in a city, a_1 and a_N can be defined as the hotel that the tourist is residing in, and thus any recommended itinerary will start and end at the tourist's hotel.

tion 5.6) is applied to all visited attractions to obtain the total time taken for the itinerary, which includes the consideration for travelling time $Trav_{a_i,a_j}$, attraction/ride duration Dur_{a_j} and queuing time $Queue^t(a_j)$.

The QUEUETOURREC problem formulation is based on a variant of the Orienteering problem, which has been shown to be NP-hard as the Orienteering problem is a specialized instance of the Travelling Salesman Problem [136, 142].[4] Furthermore, this problem formulation incorporates a time-dependent cost function $Cost^t(a_x, a_y)$, which adds to the complexity of this problem. Due to this NP-hard complexity, solving the QUEUETOURREC problem optimally is not feasible. To overcome these problems, we propose the PERSQ algorithm for solving the QUEUETOURREC problem of recommending personalized and queue-aware itineraries. In the following sections, we provide some background on Monte Carlo Tree Search (MCTS) before describing our PERSQ algorithm, which is partially based on the MCTS algorithm.

5.4 Monte Carlo Tree Search

MCTS is a popular search algorithm that has been successfully applied to many board games such as Chess, Go, Othello, among others [27, 45]. We first provide some background information on the MCTS algorithm in its typical application to board games. MCTS approaches the task of board game playing as a tree search problem, where nodes of the tree represent a specific board position and leaf nodes represent a terminal game state, i.e., a win or loss. Thus, moving from a node to a child node corresponds to making a game move that results in a new board position. The main objective is to perform a tree search that results in a set of optimal moves (nodes) leading to a win state. In the MCTS algorithm, the basic idea is that game play initially commences with iterations of random node selection to explore moves, and recording the outcome of choosing those moves. Thereafter, at subsequent game plays, MCTS departs from random moves and progressively builds upon previous successes by converging to moves that result in win

[4]The QUEUETOURREC problem is also NP-hard as it can be generalized to the Orienteering problem, by using attraction popularity $Pop(a_i)$ as a global reward in Equation 5.2 and setting the queuing times $Queue^t(a_i)$ to be uniform. For a more detailed proof on the NP-hardness of the Orienteering problem, please refer to [15, 63].

states.

The MCTS algorithm typically runs for a fixed number of iterations (e.g., run for 100 iterations) or repeats the iterations for a specific running time (e.g., run as many iterations in 10 seconds). At each iteration, MCTS operates with four main steps:

1. ***Selection***. MCTS starts at the root node r and recursively *selects* a child node c to expand based on a certain strategy, until reaching either a leaf/terminal node or an unvisited/unexpanded node. Thus, the root node r corresponds to the start state of the board, while the child node c corresponds to a move that results in a new board position.

2. ***Expansion***. If the selected node c is not a leaf node (i.e., a terminal game condition leading to a win or loss), expand node c and randomly select one of its unvisited child nodes.

3. ***Simulation***. Steps 1 and 2 are then continuously *simulated* (repeated) until reaching the end of the game, i.e., a leaf/terminal node.

4. ***Back-propagation***. At this stage, the game would have ended, resulting in a win or loss to the player. The results of this game is then *back-propagated* to all the traversed nodes (board positions) during this iteration.

Steps 1 to 4 are considered one iteration of MCTS, which are repeated for a fixed number of iterations or for a specific running time. Each node (board position) will be labelled with the number of wins/losses (1/0) and the number of times it has been visited. Step 1 typically employs a selection strategy that exploits nodes with a high win-to-visited ratio, while Step 2 uses a random exploration strategy for previously unvisited nodes. Steps 3 and 4 then simulate the game play to the end and back-propagate the results to all the visited nodes, incrementing their win count (if game was won) and visited count by 1.

5.4.1 Motivations of MCTS for Itinerary Recommendation

Monte Carlo based methods have also been applied to various routing problems, such as variants of the Travelling Salesman Problem [113] and Vehicle Routing Problem [77], with the main purpose of identifying an appropriate next node to visit in the route. Although it is possible to solve such routing-related problems optimally as an Integer Linear Program, the problem complexity increases exponentially with the number of nodes (attractions). In the case of the QUEUETOURREC problem, there is the added complexity of a time-variable in the form of queuing times at attractions. More importantly, real-life applications typically require that solutions be generated in a short time-frame (minutes rather than hours). MCTS is well-poised for solving our QUEUETOURREC problem due to two main reasons [27]: (i) Instead of traversing the entire search tree, MCTS allows us to explore a smaller, more promising region of the solution space, thus leading to a shorter running time; and (ii) MCTS can be adapted to run only for a fixed amount of time, thus making it very suitable for real-life application;

However, the direct application of MCTS to the QUEUETOURREC problem is non-trivial for several reasons: (i) in board games, each move from a node to a child node has a uniform cost of 1, whereas this cost is variable for itinerary recommendation and can be based on distance or travel times; (ii) furthermore, this cost is time-dependent in the QUEUETOURREC problem, due to different queuing times at the same attraction based on the visit time; and (iii) each win/lose state in a board game corresponds to a binary reward of 1 or 0, which leads to a simple back-propagation strategy (Step 4). In contrast, itinerary recommendation results in a complex reward structure that is based on the attractions recommended and their corresponding popularity, interest relevance and queuing times. In the following sections, we describe our proposed PERSQ algorithm, which is an adaptation of MCTS for the QUEUETOURREC problem, and we evaluate the effectiveness of PERSQ against various state-of-the-art methods for personalized tour recommendations.

5.5 Overview of PersQ Algorithm

Algorithm 2 gives an overview of our proposed PERSQ algorithm, which takes as input a desired starting attraction a_1, ending attraction a_N, starting time S and time budget B for completing the itinerary. The output is in the form of a recommended itinerary

Algorithm 2: PersQ - Overview of Algorithm

input : $a_1 \in A$: Starting attraction, $a_N \in A$: Ending attraction, S: Starting time of itinerary, B: Total time budget, $maxLoop$: Number of iterations.

output: $I = (a_1, ..., a_N)$: Recommended itinerary.

1 **begin**
2   Initialize T_{visits} as a tree of visit count;
3   Initialize T_{reward} as a tree of reward collected;
4   Initialize I_{list} as a list of itineraries;
5   **for** $iterations \leftarrow 1$ **to** $maxLoop$ **do**
6   Initialize I_{temp} as a list of attraction visits;
7   Append attraction a_1 to I_{temp};
8         $a_i \leftarrow a_1$;
9         $a_j \leftarrow \emptyset$;
10         $totalCost \leftarrow 0$;
11   **while** $totalCost < B$ **do**
12             $a_j \leftarrow SelectNextNode(a_i, T_{visits}, T_{reward})$;
13   Append attraction a_j to itinerary I_{temp};
14             $totalCost \leftarrow totalCost + Trav_{a_i,a_j} + Dur_{a_j} + Queue^t(a_j)$;
15   **if** $a_j == a_N$ **then**
16   Break loop;
17             $a_i \leftarrow a_j$;
18         $BackpropC(I_{temp}, T_{visits})$;
19   **if** $a_j == a_N$ **then**
20             $R \leftarrow Simulate(I_{temp})$;
21             $BackpropR(I_{temp}, T_{reward}, R)$;
22   Append itinerary I_{temp} to itinerary list I_{list};
23   Return best itinerary I from I_{list};

$I = (a_1, ..., a_N)$, which starts from attraction a_1 at time S and finishes at attraction a_N by time $S + B$.

At the start of Algorithm 2 (Lines 2 and 3), two similar trees are initialized with the root node n_1 as the starting attraction a_1, with its child nodes as the set of attractions $a_i \in A$, up to a depth of $|A|$ (the number of attractions in the theme park). Thus, the traversal of nodes in this tree is equivalent to an itinerary $I = (a_1, ..., a_N)$, with a_1 as the node selected at $depth = 1$, a_2 at $depth = 2$ and so on, until a_N at $depth = N$. The two above-mentioned trees differ in terms of the values associated with each node, T_{visits} contains the number of times a specific node has been selected (Lines 2), while T_{reward}

contains the total reward accumulated by a specific node (Lines 3). Line 4 initializes I_{list}, which will contain a list of explored itineraries $I_{list} = (I_1, I_2, ..., I_{maxLoop})$ at the conclusion of our algorithm.

Algorithm 2 then runs for a fixed number of iterations $maxLoop$ (Line 5) and we set $maxLoop = 1,000$ as this value allows the algorithm to complete in reasonable time ($<$ 3 minutes). Each iteration (Lines 5 to 22) is equivalent to a single run of MCTS. At the start of each iteration, Lines 6 and 7 initialize an itinerary $I_{temp} = (a_1)$ as an ordered list with a_1 as the first attraction to be visited. Following which, Lines 11 to 17 construct an itinerary that can be completed within the time budget B based on travelling time, ride duration and queuing times (Line 14). This itinerary is constructed by iteratively calling the $SelectNextNode()$ method (Line 12) and appending the next recommended attraction to I_{temp} (Line 13). The $SelectNextNode()$ method reflects the Selection and Expansion stages of traditional MCTS and is discussed further in Section 5.5.1. Attractions are continuously appended to I_{temp} until either the time budget B is exceeded (Line 11) or the destination attraction a_N is reached (Line 15). After which, $BackpropC()$ (Line 18) updates/increments the visit count of visited nodes in T_{visits} based on the recommended attractions in itinerary I_{temp}. If itinerary I_{temp} ends at attraction a_N (Line 19), we calculate the reward gained from I_{temp} (Line 20), update the accumulated rewards of visited nodes in T_{reward} accordingly (Line 21), and append I_{temp} to I_{list}. The calculation of reward and back-propagation of visit counts/rewards are described in more detail in Section 5.5.2. At the end of all iterations (Line 23), we examine all explored itineraries $I \in I_{list}$ and return the itinerary with the highest reward.

5.5.1 Selection and Expansion

Algorithm 3 describes the $SelectNextNode()$ method, which is used for selecting the next attraction (node) to visit based on a current attraction a_i, visit count tree T_{visits}, and reward tree T_{reward}. Instead of examining all possible neighbours of attraction a_i, Line 5 focuses the search on attractions that have not been visited. Lines 6 to 10 are a variant of the Upper Confidence Bound [5] applied to Trees, commonly denoted as UCT [80]. The main

Algorithm 3: PersQ - SelectNextNode()

input : $a_i \in A$: Current attraction; $I = (a_1, ...)$: Current itinerary; T_{visits}: Tree of visit counts; T_{reward}: Tree of accumulated reward.
output: $a_n \in A$: Next attraction.

begin
 $visitCount_{a_i} \leftarrow GetVisitCount(a_i, T_{visits})$;
 $a_n \leftarrow \emptyset$;
 $UCT_{max} \leftarrow 0$;
 for $a_j \in A$ *and* $a_j \notin I$ **do**
  $visitCount_{a_j} \leftarrow GetVisitCount(a_j, T_{visits})$;
  $totalReward_{a_j} \leftarrow GetTotalReward(a_j, T_{reward})$;
  $exploit_{a_j} \leftarrow \frac{totalReward_{a_j}}{visitCount_{a_j}} + \frac{Int(Cat_{a_j})+Pop(a_j)}{Trav_{a_i,a_j}+Dur_{a_j}+Queue^t(a_j)}$;
  $explore_{a_j} \leftarrow 2C_p\sqrt{\frac{2\ln visitCount_{a_i}}{visitCount_{a_j}}}$;
  $UCT_{a_j} \leftarrow exploit_{a_j} + explore_{a_j}$;
  if $UCT_{a_j} > UCT_{max}$ **then**
   $a_n \leftarrow a_j$;
 Return a_n;

aim of UCT is to choose a next attraction (node) a_j to visit, that maximizes:

$$UCT^{Orig}_{a_j} = \frac{totalReward_{a_j}}{visitCount_{a_j}} + 2C_p\sqrt{\frac{2\ln visitCount_{a_i}}{visitCount_{a_j}}} \tag{5.7}$$

In Equation 5.7, the first term $\frac{totalReward_{a_j}}{visitCount_{a_j}}$ controls for the *exploitation* of attractions (nodes) that results in itineraries with high rewards, relative to the number of times these nodes were chosen. The second term $2C_p\sqrt{\frac{2\ln visitCount_{a_i}}{visitCount_{a_j}}}$ controls for the *exploration* of nodes (attractions) that have not been previously selected, thus ensuring that different attractions (and hence, itineraries) are considered. The parameter C_p determines the emphasis to give to the *exploration* of nodes and we set $C_p = \frac{1}{\sqrt{2}}$, which Kocsis and Szepesvári [81] has proved to be the best value as it satisfies Hoeffding's inequality.

Our proposed PERSQ algorithm improves upon the original UCT (Equation 5.7) by including an additional heuristic for next node (attraction) selection. Our version of UCT is defined as:

$$\begin{aligned} UCT_{a_j}^{PersQ} = & \frac{Int(Cat_{a_j}) + Pop(a_j)}{Trav_{a_i,a_j} + Dur_{a_j} + Queue^t(a_j)} \\ & + \frac{totalReward_{a_j}}{visitCount_{a_j}} + 2C_p\sqrt{\frac{2\ln visitCount_{a_i}}{visitCount_{a_j}}} \end{aligned} \tag{5.8}$$

Unlike the original MCTS that considers uniform cost (i.e., board games where each move has a cost of 1), the cost in itinerary recommendation is variable based on the selected next node (attraction). Thus, the addition of the heuristic $\frac{Int(Cat_{a_j})+Pop(a_j)}{Trav_{a_i,a_j}+Dur_{a_j}+Queue^t(a_j)}$ favours attractions with a higher interest relevance and popularity but with lower associated travelling and queuing time.

5.5.2 Simulation and Back-propagation

In traditional MCTS for game play, the simulation stage simulates a run from the root node to a terminal node and calculates the reward, which is either a 1 for a win or a 0 for a loss. For the purposes of itinerary recommendation, a simple binary value of 1 and 0 does not accurately reflect the rewards associated with different itineraries. Thus, we chose a reward that reflects the attraction popularity, user interest and queuing times associated with each itinerary, which is defined as:

$$Reward = \sum_{a \in I_{temp}} \left(\frac{Int(Cat_a) + Pop(a)}{Queue^t(a)} \right) \tag{5.9}$$

In Algorithm 2, this reward value is calculated in Line 20 after every iteration of itinerary generation, and subsequently back-propagated to our reward tree T_{reward} in Line 21. Specifically, this reward is back-propagated to all visited nodes only if the itinerary ends at attraction a_N as specified in Constraint 5.3. The reason for this is so that we do not unnecessarily favour itineraries that do not satisfy our destination attraction constraint. On the other hand, we back-propagate the visit count (i.e., increment by one) to our visit tree T_{visits} (Line 18), regardless of whether the itinerary ends at our desired destination attraction a_N. As a result, nodes that are frequently visited but with low/no reward are less likely to be chosen based on our $UCT_{a_j}^{PersQ}$ formulation (Equation 5.8), due to the high denominator of total visit count.

5.6 Itinerary Recommendation Framework

We utilize the same framework previously described in Section 3.3.3 to derive the user visit sequences, user interests and various attraction related information (Definitions 2, 3, 4 and 5) from geo-tagged photos. Thereafter, we use the computed visit sequences, attraction popularity, user interests and queuing times to generate an itinerary recommendation based on our proposed PERSQ algorithm, as described in Section 5.5.

One limitation of this framework is potential noise in the geo-tagged photos, which could affect the calculation of visit sequences and queuing times. An alternative is to use proprietary trajectory data from actual theme park operators, which are typically obtained via tracking devices issued to theme park visitors to monitor their trajectories (via sensors located throughout the theme park). While such datasets have high accuracy, these proprietary datasets are often not publicly available for researchers and furthermore, to construct a similar dataset for another theme park requires the installation of sensors and explicitly tracking visitors in the new theme park. As such, we utilized the above-mentioned framework that is built upon open-source data, i.e., publicly available geo-tagged photos, and can be easily extended for other theme parks or cities.

5.7 Experimental Methodology

5.7.1 Dataset

Our dataset comprises nine years of geo-tagged photos that were taken in five theme parks from Aug 2007 to Aug 2016. The five theme parks are Disneyland, Epcot, California Adventure, Disney Hollywood and Magic Kindgom. This dataset was collected using the Flickr API [150], then mapped to attraction visits and visit sequences, as described in Steps 1 to 3 of Section 5.6. Each attraction a_x is also assigned a category Cat_{a_x} (e.g., roller coaster, water rides, etc) based on its corresponding Wikipedia page. Simi-

Table 5.2: Dataset description

Theme Park	**No. of Photos**	**No. of Users**	**Attraction Visits**	**# Visit Sequences**
Disneyland	181,735	3,704	119,987	11,758
Disney Epcot	90,435	2,725	38,950	5,816
California Adv.	193,069	2,593	57,177	6,907
Hollywood	57,426	1,972	41,983	3,858
Magic Kingdom	133,221	3,342	73,994	8,126

larly, each attraction a_x is also associated with a ride duration Dur_{a_x}, which can also be found on that attraction's Wikipedia page or a website like [135]. While earlier works have constructed datasets from geo-tagged photos, our dataset is the first that includes the queuing time distribution at each attraction based on these geo-tagged photos. The descriptive statistics of this dataset are shown in Table 5.2.

5.7.2 Baseline Algorithms

As there is no existing work that considers all three components of queuing time, attraction popularity and user interest, we select various state-of-the-art baselines that consider both attraction popularity and user interest for recommending personalized itineraries. These baselines are as follows:

- **Iterative Heuristic Approximation (IHA)**. A heuristic algorithm proposed in [159, 160] that commences with an itinerary from the starting attraction a_1 to destination attraction a_N and iteratively adds a new attraction between the current and destination attractions, until the budget is reached. The attraction selected to be added is the one with the maximum heuristic value of $\frac{Int(Cat_{a_j})+Pop(a_j)}{Trav_{a_i,a_j}+Dur_{a_j}}$, i.e., the next attraction with the highest profit relative to its travelling cost.

- **User Based Collaborative Filtering for Itineraries (UBCF-I)**. A variant of the popular User Based Collaborative Filtering (UCBF) [119, 154, 158] that utilizes user interest similarities (based on their ratings on items) to recommend a set of top-k

items for another user. In our adaptation to itinerary recommendations, we define user ratings based on the number of posted photos on a specific attraction, hence more photos represent a higher rating. Thereafter, we take the ranked top-k items (attractions) and iteratively add them to the starting attraction a_1 until the budget is reached, resulting in the recommended itinerary.

- **Personalized Tour Recommender (PERSTOUR)**. An Integer Programming based algorithm proposed in Chapter 4 is used, which recommends personalized itineraries that consider attraction popularity and user interest, with a variable visit duration to attractions based on a user's interest preferences. The PERSTOUR algorithm defines user interest levels based on the length of time a user stays at attractions, relative to the average user.

- **Tour Recommendation With Interest Category (TOURINT)**. The TOURINT algorithm (from Chapter 6) formulates the tour recommendation problem with a mandatory category, which has to be visited at least once in the recommended itinerary. To personalize the recommended itinerary, TOURINT defines this mandatory category as the attraction category that has been most frequently visited in the user's other visit sequences.

- **Trip Builder (TRIPBUILD)**. We adapted the problem formulation in [22, 24] to fit our QUEUETOURREC problem, by adding in constraints of start and destination attractions. TRIPBUILD recommends personalized itineraries that optimize for attraction popularity and user interest, where user interest is based on the number of times a user has visited attractions of a certain category, relative to his/her total attraction visits.

5.7.3 Evaluation

Like many recent itinerary recommendation works [24, 35, 145], our experimental evaluation uses the visit sequences of users as the ground truth of real-life visits by these users. Each visit sequence corresponds to the real-life attraction visits of users in a spe-

Table 5.3: Comparison between PersQ and various baselines, in terms of the mean and standard errors of Maximum Queue-time (MQ_I), Queue:Cost Ratio (QC_I), Queue:Popularity Ratio (QP_I), Popularity (Pop_I), Interest (Int_I), Recall (R_I), Precision (P_I) and F1-score (F_I). Lower values of MQ_I, QC_I and QP_I are preferred, while higher values of Pop_I, Int_I, R_I, P_I and F_I are better. The bold/blue numbers indicate the best result for each metric.

	Algorithm	Maximum Queue-time	Queue:Cost Ratio	Queue:Pop Ratio	Popularity	Interest	Recall	Precision	F1-score
Cali. Adv.	PersQ	**0.059±.002**	**0.257±.007**	**2515±121**	**1.794±.055**	**3.704±.105**	**0.483±.010**	**0.307±.007**	**0.338±.007**
	IHA	0.173±.006	0.346±.009	7669±904	1.394±.043	3.425±.083	0.332±.006	0.296±.006	0.287±.005
	UBCF-I	0.196±.007	0.330±.009	15679±1665	0.636±.025	1.530±.048	0.258±.005	0.282±.006	0.244±.004
	PersTour	0.188±.008	0.330±.011	5583±917	0.902±.033	1.423±.055	0.227±.007	0.223±.007	0.204±.006
	TourInt	0.188±.007	0.339±.011	4583±319	0.910±.033	1.444±.057	0.228±.007	0.225±.007	0.206±.006
	TripBuild	0.197±.008	0.321±.011	5182±360	0.871±.033	1.469±.055	0.225±.006	0.223±.007	0.205±.006
Hollywood	PersQ	**0.184±.005**	**0.298±.011**	**3585±335**	1.28±.053	**2.12±.088**	**0.482±.015**	**0.439±.014**	**0.431±.013**
	IHA	0.283±.009	0.423±.015	4570±223	**1.29±.044**	1.85±.070	0.367±.010	0.417±.012	0.371±.010
	UBCF-I	0.320±.012	0.410±.012	17493±1270	0.62±.029	1.09±.045	0.297±.008	0.370±.010	0.313±.008
	PersTour	0.263±.010	0.402±.013	10287±931	0.82±.033	1.23±.058	0.305±.009	0.367±.012	0.317±.010
	TourInt	0.256±.010	0.395±.013	9543±861	0.81±.033	1.22±.058	0.302±.010	0.364±.012	0.314±.010
	TripBuild	0.272±.011	0.412±.014	12144±1071	0.77±.034	1.24±.058	0.309±.010	0.374±.012	0.322±.009
Epcot	PersQ	**0.113±.004**	**0.284±.008**	**4088±152**	**0.891±.031**	2.407±.086	**0.472±.012**	**0.413±.012**	**0.407±.010**
	IHA	0.279±.006	0.344±.010	6861±285	0.852±.025	**2.511±.069**	0.380±.008	0.368±.008	0.353±.007
	UBCF-I	0.212±.007	0.331±.010	11664±823	0.500±.017	1.232±.044	0.291±.007	0.314±.008	0.283±.006
	PersTour	0.250±.008	0.356±.011	8034±360	0.604±.021	1.490±.060	0.310±.008	0.322±.009	0.297±.007
	TourInt	0.252±.007	0.352±.011	8112±340	0.599±.021	1.459±.058	0.317±.008	0.327±.009	0.303±.007
	TripBuild	0.246±.008	0.342±.011	8208±386	0.584±.021	1.561±.060	0.300±.008	0.312±.009	0.287±.008
Disneyland	PersQ	**0.161±.004**	**0.263±.005**	**5673±263**	**0.731±.017**	2.287±.055	**0.332±.006**	0.295±.006	**0.289±.005**
	IHA	0.198±.004	0.332±.006	13437±419	0.690±.020	**3.531±.068**	0.267±.004	0.270±.004	0.249±.003
	UBCF-I	0.176±.004	0.378±.007	14986±1072	0.495±.012	1.567±.037	0.235±.004	**0.296±.005**	0.240±.004
	PersTour	0.177±.005	0.322±.008	8255±579	0.623±.017	1.523±.045	0.190±.004	0.201±.005	0.180±.004
	TourInt	0.193±.005	0.313±.008	10267±1324	0.612±.017	1.498±.044	0.188±.004	0.200±.005	0.179±.004
	TripBuild	0.193±.005	0.289±.008	10601±1634	0.580±.017	1.454±.043	0.176±.004	0.181±.005	0.165±.004
Magic King.	PersQ	**0.132±.003**	**0.244±.005**	**4275±133**	**0.901±.025**	**3.330±.088**	**0.440±.008**	**0.326±.007**	**0.343±.006**
	IHA	0.240±.004	0.321±.008	6885±329	0.772±.017	3.159±.064	0.305±.005	0.312±.006	0.288±.004
	UBCF-I	0.206±.005	0.323±.007	11328±424	0.474±.012	1.389±.038	0.265±.004	0.304±.006	0.261±.004
	PersTour	0.208±.006	0.303±.010	8822±494	0.486±.016	1.305±.047	0.202±.005	0.201±.006	0.186±.005
	TourInt	0.195±.006	0.316±.010	9410±531	0.490±.017	1.311±.048	0.200±.005	0.200±.006	0.185±.005
	TripBuild	0.191±.006	0.287±.009	9074±456	0.479±.017	1.393±.048	0.193±.005	0.189±.006	0.177±.005

cific theme park, e.g., $Attraction\ a_4 \rightarrow a_{12} \rightarrow a_7 \rightarrow a_{23}$. To ensure a fair comparison with the baselines, which consider user interest, we perform our evaluation only on users with at least two visit sequences. Using these travel sequences, we then apply leave-one-out evaluation [83], where we use one visit sequence for evaluation and the other visit sequences to determine the interest preferences of this user. For each visit sequence, we use the first and last attraction of each visit sequence as input to PERSQ and the baselines, along with the actual time spent in the visit sequence as the time budget. In addition, we also incorporate queuing time into the evaluation and thus attractions may be dropped from recommended itineraries due to the additional queuing time. We repeat the evaluation for all travel sequences in our dataset and compare PERSQ against the various baselines using the following evaluation metrics:

1. **Maximum Queue-time (MQ_I).** The queuing times at attractions recommended in itinerary I, relative to their max. queuing time, defined as: $MQ_I = \frac{1}{|I|} \sum_{a \in I} \frac{Queue^t(a)}{Max_{t \in T}\left(Queue^t(a)\right)}$.
2. **Queue:Cost Ratio (QC_I).** The average ratio of queuing time to itinerary (time) cost of an itinerary I, defined as: $QC_I = \frac{1}{|I|} \sum_{a_i \in I, i \neq 1} \frac{Queue^t(a)}{Trav_{a_{i-1},a_i} + Dur_{a_i} + Queue^t(a_i)}$.
3. **Queue:Popularity Ratio (QP_I).** The average ratio of queuing time to attraction popularity of an itinerary I, defined as: $QP_I = \frac{1}{|I|} \sum_{a \in I} \frac{Queue^t(a)}{Pop(a)}$.
4. **Popularity (Pop_I).** The total popularity based on all attractions in an itinerary I, defined as: $Pop_I = \sum_{a \in I} Pop(a)$.
5. **Interest (Int_I).** The total interest alignment (to a user u) based on all attractions in an itinerary I, defined as: $Int_I^u = \sum_{a \in I} Int_u(Cat_a)$.
6. **Recall: R_I.** The ratio of attraction visits in a user's real-life visit sequence that also exist in the recommended itinerary I. Given that A_r is the set of attractions in the recommended itinerary I and A_v is the set of attraction visits in the real-life travel sequence, we define recall as: $R_I = \frac{|A_r \cap A_v|}{|A_v|}$.
7. **Precision: P_I.** The ratio of attractions in the recommended itinerary I that also exist in a user's real-life visit sequence. Using the same notations for A_r and A_v, we define precision as: $P_I = \frac{|A_r \cap A_v|}{|A_r|}$.

8. **F-score (F_I).** The harmonic mean of the precision P_I and recall R_I of an itinerary I, defined as: $F_I = \frac{2 \times P_I \times R_I}{P_I + R_I}$.

Metric 1 allows us to determine the extent to which an itinerary schedules visits to attractions at their maximum (longest) queuing times (MQ_I=1) or at their minimum (MQ_I=0), where a smaller value of MQ_I is preferred as it means we avoided attractions at their busiest time. Metric 2 measures the ratio of queuing time to the total time spent travelling to, queuing at and visiting/riding attractions, where a smaller QC_I value shows that the user spends less time queuing as part of the itinerary. Metric 3 measures the ratio of queuing time to the attraction popularity, and allows us to better differentiate between popular attractions (which are more likely to have longer queuing times) and unpopular ones with shorter queuing times. Similarly for Metric 3, a smaller value of QP_I is preferred as it indicates short queuing times at popular attractions. Metrics 4 and 5 are standard measures of itinerary popularity (Pop_I) and user-interest alignment (Int_I), respectively, while Metrics 6, 7 and 8 measure how well a recommended itinerary reflects the real-life attraction visits of a user.

5.8 Results and Discussion

Queue-time Metrics. Table 5.3 gives an overview of the experimental results, in terms of the six evaluation metrics introduced in Section 5.7.3. PERSQ outperforms all baselines with the lowest Maximum Queue-time (MQ_I), Queue:Cost Ratio (QC_I), Queue:Popularity Ratio (QP_I). In relative terms, PERSQ out-performed all baselines with a reduction of 8.5% to 65.9% in Maximum Queue-time, 9.0% to 24.6% in Queue:Cost Ratio, and 21.6% to 45.1% in Queue:Popularity Ratio. These results indicate that PERSQ recommends visits to popular attractions at times with the shortest queues, and constructs itineraries that include minimal time spent queuing at attractions.

Recall, Precision, F1-score. In terms of Recall (R_I) and F1-score (F_I), PERSQ also outperforms all baselines for all datasets, with relative improvements of 24.2% to 45.5% for Recall, and 15.3% to 19.1% for F1-score. In terms of Precision (P_I), PERSQ out-performs all baselines in 24 out of 25 cases, with a relative improvement of up to 5.3%. The only exception is for the Disneyland dataset, where PERSQ out-performs all baselines but underperforms UBCF-I by less than 0.34% in terms of Precision. Moreover, we are more interested in the F1-score, as it considers both precision and recall, and PERSQ out-performs all baselines by at least 15.3% in terms of F1-score. These results indicate that itineraries recommended by PERSQ are highly representative of the real-life visits of these users.

Popularity and Interests. In terms of Attraction Popularity (Pop_I) and User Interests (Int_I), PERSQ also offers the best overall performance, while IHA offers the second best performance. PERSQ has the highest Popularity scores for four of five theme parks and the highest Interest scores in three of five theme parks, while IHA leads in Popularity and Interest for one and two theme parks, respectively.

Discussion. The superior performance of PERSQ is due to its three-fold consideration of attraction popularity, user interests and queuing times, which is automatically determined from geo-tagged photos. In contrast, the various baselines only consider attraction popularity and user interests, and hence may recommend visits to attractions that are popular and interesting but with excessive queuing times. This excessive queuing time consumes a large portion of the touring time, as indicated by the high QC_I scores, thus causing attractions later in the itinerary to be missed. Apart from PERSQ, IHA is also able to offer a relatively good performance in terms of attraction popularity and user interests due to its use of the profit (popularity and interest) over cost (travelling time) heuristic.

5.9 Summary

In this chapter, we proposed the QUEUETOURREC problem of recommending personalized itineraries of popular and interesting attractions, while minimizing queuing times. QUEUETOURREC is an NP-hard problem that includes time-dependent queuing times, which we then solve using our proposed PERSQ algorithm that is adapted from MCTS. For determining queuing times, we also implemented a framework that utilizes geotagged photos to determine the distribution of queuing times at each attraction, as well as the attraction popularity and user interest preferences. We evaluated PERSQ on a dataset of five major theme parks and show that PERSQ out-performs the state-of-the-art in terms of maximum queuing times, queuing time to itinerary cost ratio, queuing time to attraction popularity ratio, attraction popularity, user interest alignment, recall, precision and F1-score.

Chapter 6

Tour Recommendation with Mandatory POI Categories

In the previous two chapters, we proposed the PersTour *algorithm and* PersQ *algorithm that aim to recommend personalized tour itineraries with customized visiting durations based on user interests and queue-aware tour itineraries with the consideration of queuing times at attractions. In this chapter, we formulate the problem of tour itinerary recommendation with a mandatory POI visit category and proposed the* TourRecInt *approach for solving this problem. We evaluate our proposed approach on a Flickr dataset comprising three cities and find that our approach is able to recommend tours that are more popular and comprise more places/points-of-interest, compared to various baselines. More importantly, we find that our recommended tours reflect the ground truth of real-life tours taken by users, based on measures of recall, precision and F1-score.*

6.1 Introduction

The prevalence of GPS-enabled camera-phones and photo sharing sites (such as Flickr and Instagram) facilitate users to share geo-tagged photos of interesting places they have visited. The sharing of such photos are increasingly popular in recent years, as illustrated by the 8B existing photos and 3.5M new daily uploads in Flickr [131]. These geo-tagged photos also provide an abundance of location-based information, which can be used to improve the recommendation of tours and POIs to visit. For example, many researchers have utilized these geo-tagged photos to determine the travel history of users and rec-

---

This chapter is derived from the following publication:

- **Kwan Hui Lim**. Recommending Tours and Places-of-Interest based on User Interests from Geo-tagged Photos. *Proceedings of the 2015 SIGMOD PhD Symposium (SIGMOD'15)*. pp 33-38. May 2015.

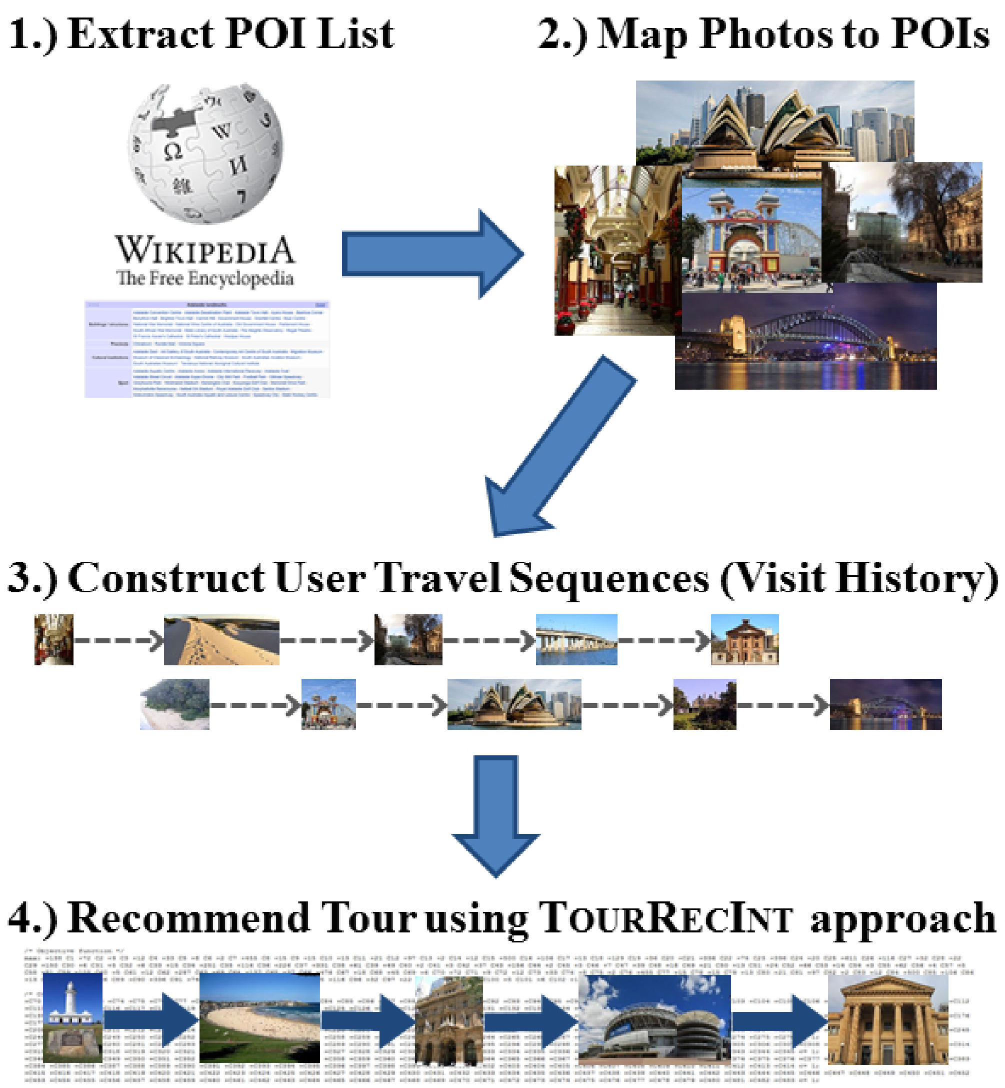


Figure 6.1: Overall Experimental Framework

ommend tours based on these travel histories [38, 40, 86, 87, 114].

In this chapter, we aim to build upon the field of tour recommendation by proposing a novel tour itinerary recommendation problem with a mandatory POI category and the TOURRECINT approach for solving this problem. Specifically, our main contributions are:

- Formulating a novel tour itinerary recommendation problem that includes an additional constraint of a mandatory POI category which must be visited in the recommended tour itinerary. This tour itinerary recommendation problem is modelled in the context of the Orienteering problem [136, 142], with the additional constraint of a mandatory POI category (Section 6.3.1).

- Proposing the TOURRECINT approach for solving our tour itinerary recommendation problem based on an Integer Linear Program, where the mandatory POI category is based on the most visited POI category in a user's past travel sequences (Section 6.3.2).
- Implementing a framework (Figure 6.1) to construct the travel sequences of a user based on his/her geo-tagged photos on Flickr and the list of POIs on Wikipedia (Section 6.3.2).
- Evaluating our TOURRECINT approach against various baselines using a Flickr dataset that comprises geo-tagged photos taken in three cities (Section 6.5).

This chapter is structured as follows. Section 6.2 discusses some related work, while Section 6.3 introduces the problem of tour recommendation with must-visit categories. We describe our experimental methodology in Section 6.4, before discussing the experimental results in Section 6.5. Finally, we conclude and summarize this chapter in Section 6.6.

6.2 Related Work

There is a large body of work that uses geo-tagged photos to recommend tours, and more recent works have also considered the categories of POIs for recommending personalized itineraries, which we have extensively discussed in Chapter 2. Location prediction is another research area that is closely related to tour recommendation, in particular location prediction that is based on user interests. Both [70] and [102] determine user interests based on the time and categories of POIs visited, with [70] employing a topic model and [102] using matrix factorization for location prediction. Based on 68 features such as unique POI categories visited and most visited POI categories, [10] performs location prediction using Ranking Support Vector Machines [75] and Gradient Boosted Regression Trees [165].

Our work differs from these state-of-the-art in tour itinerary recommendation and location prediction in two main ways:

1. Although there are various interesting works on tour itinerary recommendation that consider POI categories, these earlier works are based on problem formulations that do not consider a mandatory POI category, whereas our work recommends tour itineraries with a must-visit category that is based on the most visited POI category in a user's past travel sequences.

2. Despite the similarities between location prediction and tour itinerary recommendation, the major difference is that the former attempts to predict a single POI or location, whereas the latter aims to recommend a set of POIs that is constructed as a connected itinerary. Thus, tour itinerary recommendation involves the additional challenges of recommending multiple POIs and scheduling these POIs as an itinerary that satisfies various temporal and spatial constraints.

6.3 Proposed Approach

We first frame the tour recommendation problem in the context of the Orienteering problem. Thereafter, we elaborate on our proposed approach to recommend tours based on user interests, and describe the main steps of our experimental framework.

6.3.1 Problem Definition

We define our tour recommendation problem based on the Orienteering Problem [136] and use its integer problem formulation from [142], with an additional constraint for a must-visit category based on user interest. Given a set of POIs P, starting POI $p_1 \in P$, and destination POI $p_N \in P$, we want to recommend a tour $T = (p_1, ..., p_N)$ that adheres to a distance budget B, while maximizing the overall profit of POIs in recommended tour

T. Formally, we aim to optimize the following objective function:

$$Max \sum_{i=2}^{N-1} \sum_{j=2}^{N} x_{i,j} Pop(i) \tag{6.1}$$

such that:

$$\sum_{j=2}^{N} x_{1,j} = \sum_{i=1}^{N-1} x_{i,N} = 1 \tag{6.2}$$

$$\sum_{i=1}^{N-1} x_{i,k} = \sum_{j=2}^{N} x_{k,j} \leq 1, \quad \forall\, k = 2, ..., N-1 \tag{6.3}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} Dist(i,j) x_{i,j} \leq B \tag{6.4}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} x_{i,j} \delta(Cat_i = c_m) \geq 1, \quad \forall\, c_m \in C \tag{6.5}$$

where:

$$x_{i,j} = \begin{cases} 1, & \text{if we visit POI } i \text{, followed by POI } j \\ 0, & \text{otherwise} \end{cases}$$

$$\delta(Cat_i{=}c_m) = \begin{cases} 1, & \text{if } Cat_i = c_m \text{ (POI } i \text{ is of category } c_m\text{)} \\ 0, & \text{otherwise} \end{cases}$$

Equation 6.1 shows the objective function that maximizes the total popularity of all POIs in the recommended tour, where $Pop(p)$ measures the popularity of POI p based on the total number of visits to POI p. Constraint 6.2 ensures that the tour starts and ends at POI P_1 and POI P_N, respectively. Constraint 6.3 ensures tour/path connectivity and that no POIs are re-visited. Constraint 6.4 ensures that the total distance travelled between consecutive POIs is within the distance budget B (using the function $Dist(i,j)$ that measures the distance between POI i and POI j).[1]

[1]While we adopt a simple representation of budget using distance, this can be easily generalized to other representations such as travel time by different transport modes and even include the POI visit duration.

More recently, authors such as [22, 62] further grouped POIs into different categories (e.g., museums, parks, etc). Similarly, we adopt a set of POI interest categories C and represent each POI $p \in P$ by a unique ID, POI name, category, and latitude/longitude coordinates. Given that Cat_p denotes the category of POI p, Constraint 6.5 ensures that the recommended tour includes at least one visit to a POI belonging to POI category $c_m \in C$. In the next section, we further elaborate on Constraint 6.5 and how we determine the POI category $c_m \in C$. In addition, we also included the constraints for sub-tour elimination as described in [108].

6.3.2 Experimental Approach

Our proposed TOURRECINT approach is based on the Orienteering Problem (outlined in Section 6.3.1), with additional consideration for user interests based on his/her visit history. In addition to the constraints of a starting POI $p_s \in P$, destination POI $p_d \in P$ and distance budget B, we define an additional constraint of a must-visit POI category $c_m \in C$, as highlighted in Constraint 6.5. Some examples of must-visit POI categories include Sports, Parks, Entertainment and Shopping. This constraint ensures that the recommended tour T contains at least one POI of the category $c_m \in C$. Specifically, we define the category $c_m \in C$ as the POI category which the user has most frequently visited in his/her other travel sequences. Finally, we solve this tour recommendation problem as an integer program, using the lpsolve linear programming package [13]. User interest can also be easily generalized to other definitions such as using POI visit times or a relative interest weighting, which we intend to explore as future work.

More details on our evaluation methodology are given next in Section 6.4.

6.4 Experimental Methodology

In this section, we describe the dataset used in our experiments and elaborate on the various evaluation metrics.

6.4.1 Dataset

Our experiments were conducted on the Yahoo! Flickr Creative Commons 100M dataset [151]. This dataset comprises 100M photos and videos that were posted on Flickr, accompanied by their relevant meta information such as the date/time taken, latitude/longitude coordinates and geographic accuracy. For our experiments, we only consider photos with the highest geographic accuracy.

Table 6.1: Description of Dataset

City	No. of POIs	No. of Photos	# POI Visits	# Travel Sequence
Adelaide	65	26,665	4,022	948
Melbourne	96	61,605	14,607	2,780
Sydney	150	53,832	20,876	3,628

Using this dataset, we then extracted photos that were taken in three major Australian cities, namely: Adelaide, Melbourne and Sydney. Table 6.1 shows more details regarding this dataset. As previously described in Section 3.3.3, we first obtained a list of POIs from Wikipedia, then mapped these photos to User-POIs visits. Next, we constructed the user travel sequences, before finally evaluating our proposed approach.

6.4.2 Evaluation Metrics

Our evaluation is performed using the following metrics:

1. **Total POIs in Tour**: The total number of POIs recommended in the tour.

2. **Tour Popularity**: The total popularity of all POIs recommended in the tour, where POI popularity is the number of times a POI is visited.

3. **Tour Recall**: The recall of POIs recommended in the tour, defined as: $\frac{|P_r \cap P_v|}{|P_v|}$, where P_r and P_v are the set of POIs recommended in the tour and visited by the user in real-life, respectively.[2]

[2]We approximate the real-life tours (travel sequences) of users based on the photos they have taken.

4. **Tour Precision**: The precision of POIs recommended in the tour, defined as: $\frac{|P_r \cap P_v|}{|P_r|}$, where P_r and P_v are the set of POIs recommended in the tour and visited by the user in real-life, respectively.

5. **Tour F1-score**: The harmonic mean of both precision and recall, defined as: $\frac{2 \times precision \times recall}{precision + recall}$.

Metrics 1 and 2 reflect the objectives of a typical tourist, which are to visit the largest number of POIs and visit the most popular POIs. Metrics 3 to 5 are standard evaluation metrics used in the Information Retrieval field, which we adopt for evaluating our recommended tours to determine how well they perform against the real-life travel sequences that users embark on.

For determining the must-visit interest category $c_m \in C$ in our proposed TOURRECINT (Section 6.3), we use leave-one-out cross-validation [83] (i.e., when we evaluate a travel sequence of a particular user, we define $c_m \in C$ as the POI category which the user has visited the most in his/her other travel sequences). Similarly, we use the starting/destination POIs and distance covered in these real-life travel sequences as input to TOURRECINT.

6.5 Results and Discussion

In this section, we discuss some results on the distribution of POI visits. Thereafter, we describe the various baseline algorithms used and discuss the results of our proposed TOURRECINT compared to these baselines.

6.5.1 Heavy-tailed Distribution of POI Visits

As shown in Figure 6.2, we observed a heavy-tailed distribution for the POI visits in our dataset. In particular, we find that the top 40 POIs (in terms of visit count) represents 99.4%, 87.9% and 88.5% of all POI visits in Adelaide, Melbourne and Sydney respectively. Given the large representation and popularity of these top 40 POIs, we focus on the top 40 POIs of each city for our tour recommendation experiments.

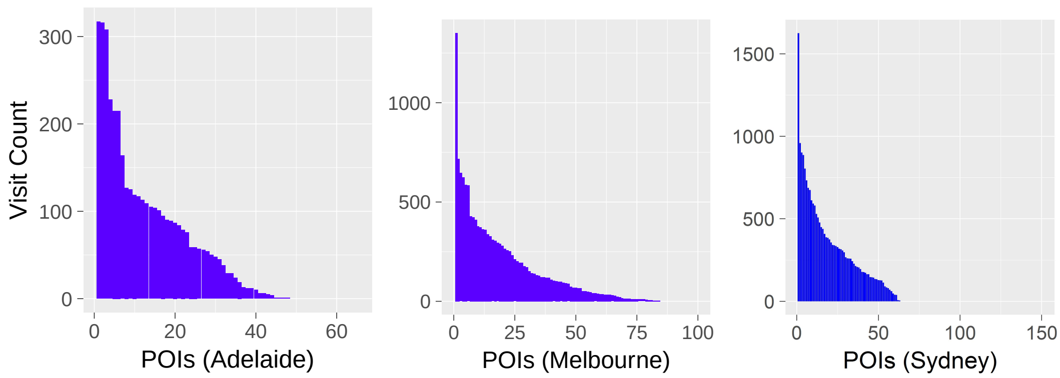


Figure 6.2: Distribution of POI Visit Count

6.5.2 Baseline Algorithms

In our evaluation, we compare our proposed approach with various baselines, as follows:

- **Greedy Nearest (GREEDNEAR).** From a starting POI $p_s \in P$, iteratively select the *nearest, unvisited POI* as the next POI to visit.

- **Greedy Most Popular (GREEDPOP).** From a starting POI $p_s \in P$, iteratively select the *most popular, unvisited POI* as the next POI to visit.

- **Random Selection (RANDOM).** From a starting POI $p_s \in P$, iteratively select a *random, unvisited POI* as the next POI to visit.

All three baselines will iteratively select a next POI to visit, until either: (i) destination POI $p_d \in P$ is reached; or (ii) distance budget B is exceeded.

Similar to the evaluation of TOURRECINT, we evaluate the three baselines using the real-life travel sequences of users. For each travel sequence with ≥3 visited POIs, we use the starting/destination POIs and distance covered in these travel sequence as input to TOURRECINT and the baselines. Thereafter, we measure the performance of each algorithm based on the metrics described in Section 6.4.2, and repeat the evaluation for all travel sequences with ≥3 POIs.

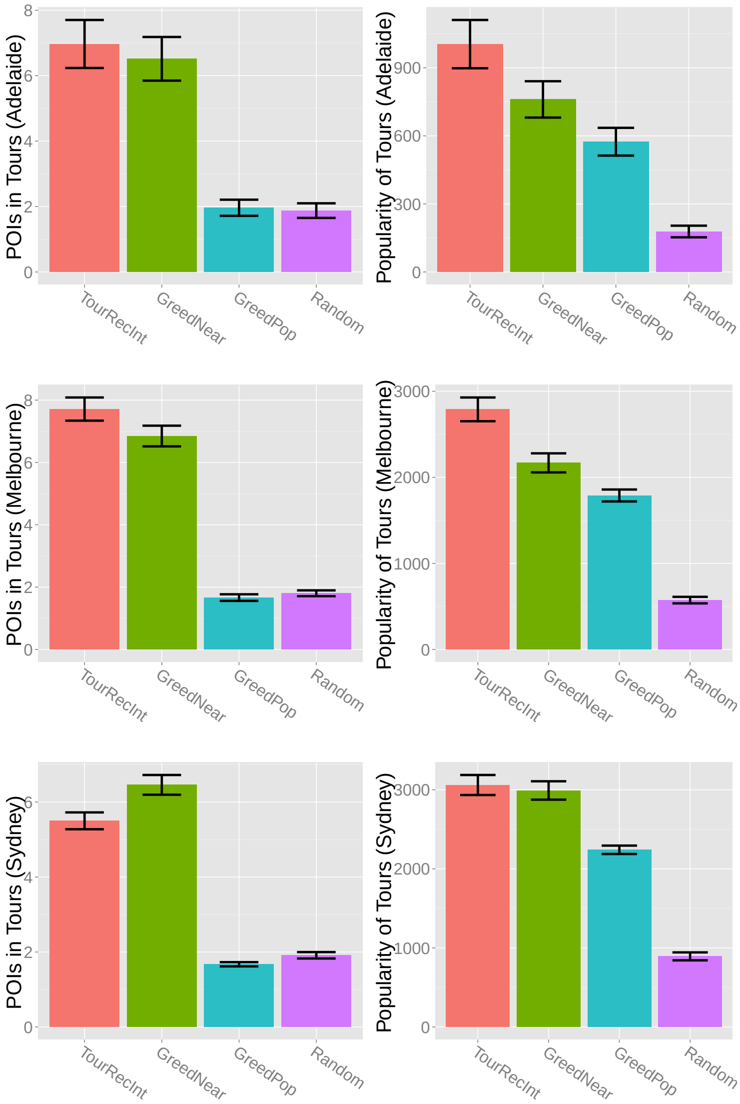


Figure 6.3: Total POI and POI popularity (1st and 2nd column) of tours recommended for the Adelaide, Melbourne and Sydney (1st to 3rd row) datasets. For each graph, the x-axis shows the algorithms evaluated, namely: TourRecInt, GreedNear, GreedPop and Random (left to right).

6.5.3 Tour Recommendation Results

Figures 6.3 and 6.4 show that our proposed TOURRECINT generally out-performs all three baselines (GREEDNEAR, GREEDPOP, and RANDOM) in terms of total POIs, total popularity, precision, recall and F1-score, for all three cities. We now discuss these results in greater detail.

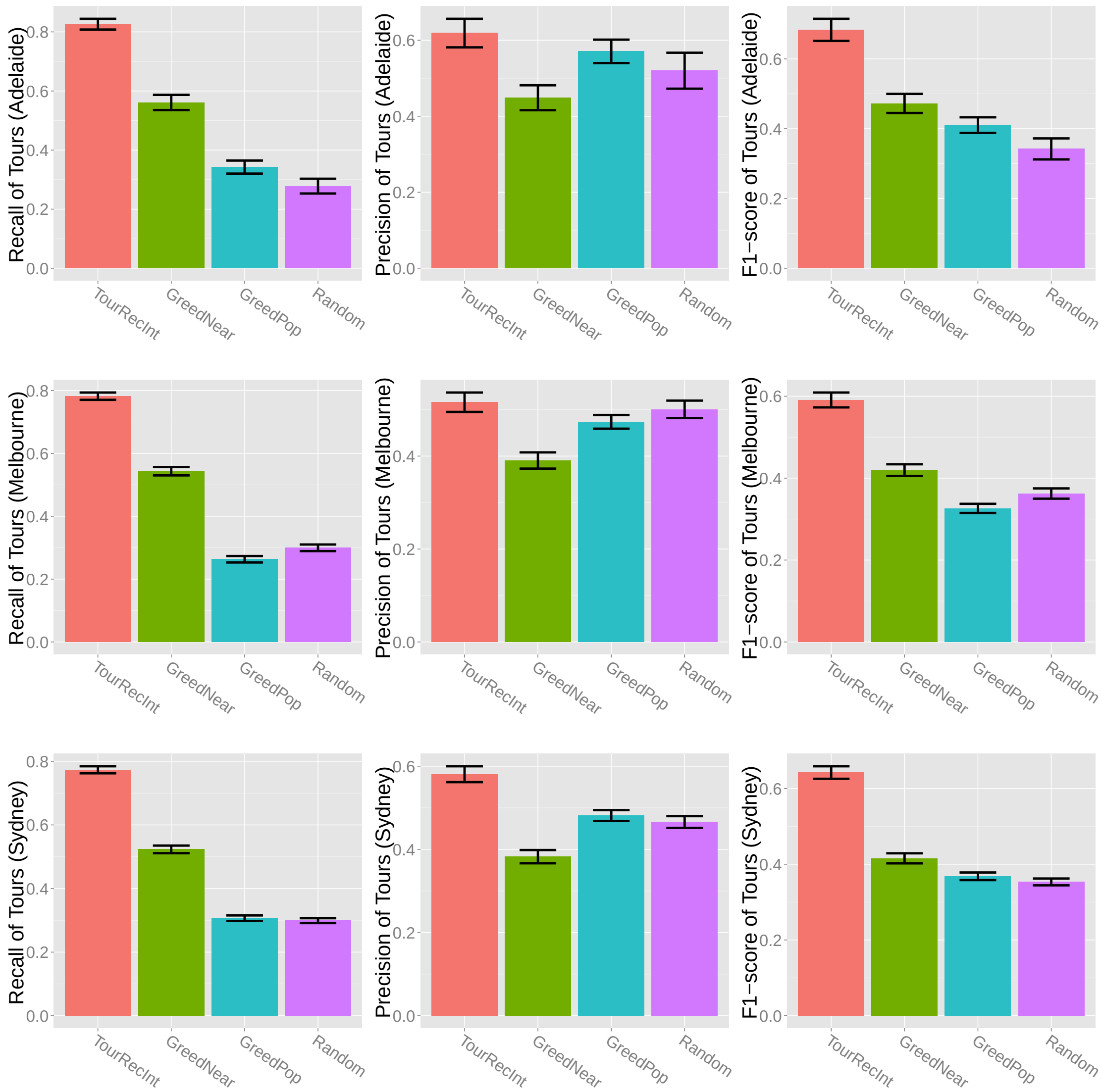


Figure 6.4: Recall, precision and F1-score (1st to 3rd column) of tours recommended for the Adelaide, Melbourne and Sydney (1st to 3rd row) datasets. For each graph, the x-axis shows the algorithms evaluated, namely: TourRecInt, GreedNear, GreedPop and Random (left to right).

Compared to the baselines, TOURRECINT recommends tours that comprise more POIs and are more popular in most cases[3], addressing the typical objectives of tourists to visit as many POIs as possible, with a preference for the more popular ones. GREEDNEAR offers the second best performance as it favours the nearest POI, thus consuming less

[3]Except for Sydney where GREEDNEAR recommends more POIs. However, GREEDNEAR under-performs TOURRECINT in terms of precision, recall and F1-score.

budget (distance) and is able to cover more POIs. Conversely, GREEDPOP is biased towards the most popular POI regardless of distance. However, reaching this POI typically consumes a large proportion of GREEDPOP's budget, thus rendering it unable to visit more POIs. Unsurprisingly, RANDOM provides the worst overall performance based on tour popularity.

In terms of recall, TOURRECINT offers the best performance by a large margin, compared to the three baselines. One contributing factor is the consideration of user interest by TOURRECINT, as users are more likely to visit places that they are interested in [9]. On the other hand, the baseline algorithms do not consider user interest, thus resulting in a poorer performance.

TOURRECINT also performs the best in terms of precision, followed by GREEDPOP and GREEDNEAR, with RANDOM performing the worst. Similarly, the results show that TOURRECINT offers the best performance in terms of F1-score, followed by GREEDNEAR, GREEDPOP and RANDOM. The strong performance of TOURRECINT in recall, precision and F1-score shows that TOURRECINT is able to recommend tours that accurately reflect the ground truth of real-life user travel sequences.

6.6 Summary

We examined tour recommendation in the context of the Orienteering problem modified with the additional constraint of a mandatory POI category, and developed the TOURRECINT approach for recommending tours based on user interest. Our proposed framework involves first using geo-tagged photos and a POI list to construct user visit sequences, then using TOURRECINT for tour recommendation based on these user visit sequences. TOURRECINT is based on a variant of the Orienteering problem, with an additional constraint for a must-visit category based on user interest (i.e., the most visited POI category). Using a Flickr dataset across three cities, we evaluated TOURRECINT against various baselines and observe that TOURRECINT recommends tours that are more popular and comprise more POIs. More importantly, we find that TOURRECINT is able to recommend tours that reflect the ground truth of real-life travel sequences, as indicated by high values of recall, precision and F1-score. Up to now, we have examined various tour recommendation problems that considered aspects such as personalization, queuing times, and mandatory POI categories, which are targetted at the individual traveller. In the next chapter, we study a novel tour itinerary recommendation problem that is targetted at groups of tourists with diverse interest preferences, and tour guides that are tasked with leading these tour groups.

Chapter 7

Group Tour Recommendation

*In the previous chapters, we have explored various problems of tour recommendation for individual tourists and proposed algorithms for solving these problems. However, tourists may want to travel in a group, e.g., extended family, and one possible solution is for tourist to seek the assistance of tour operators. Traditionally tour operators have offered standard tour packages of popular locations, but these packages may not cater to tourist's interests, especially for groups of tourists travelling together with diverse interest preferences. In this chapter, we introduce the novel problem of group tour recommendation (*GROUPTOURREC*), which involves many challenges: forming tour groups whose members have similar interests; recommending Points-of-Interests (POI) that form the tour itinerary and cater for the group's interests; and assigning guides to lead these tours. For each challenge, we propose solutions involving: clustering for tourist groupings; optimizing a variant of the Orienteering problem for POI recommendations; and integer programming for tour guide assignments. Using a Flickr dataset of seven cities, we compare our proposed approaches against various baselines and observe significant improvements in terms of interest similarity, total/maximum/minimum tour interests and total tour guide expertise.*

7.1 Introduction

Both tourists and tour operators play important roles in tourism. A major objective of such tourists is to visit captivating Points-of-Interest (POI) in foreign cities, but they often lack the expertise, familiarity and/or time to plan a suitable tour itinerary. As such, many of these tourists engage the services of tour operators. In turn, tour operators

This chapter is derived from the following publication:

- **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Towards Next Generation Touring: Personalized Group Tours. *Proceedings of the 26th International Conference on Automated Planning and Scheduling (ICAPS'16)*. pp 412-420. Jun 2016.

conduct organized tours to multiple POIs for groups of tourists, and assign tour guides to lead each tour group. However, tour operators typically offer standard tour packages of popular POIs, which may not cater to a tourist's interests. Although tour packages can be customized, it is challenging to construct tours that are interesting to multiple tourists in each tour group and assign tour guides with the right expertise to lead these customized tours. We term this the customized Group Tour Recommendation (GROUPTOURREC) problem (Figure 7.1).

Technically, GROUPTOURREC is a non-trivial problem due to its NP-hard complexity, which is discussed in later sections. Thus, we decompose GROUPTOURREC into a series of more manageable sub-problems, namely:

1. How do we divide the tourists into tour groups, maximizing the interest similarity of all tourists in a group?

2. How do we plan a tour itinerary comprising a subset of POIs that are most interesting to each tour group?

3. How do we assign a tour guide to each tour group, matching the expertise of each guide to the recommended tour?

While there are various works that investigate group recommendations [17] or tour recommendations [60] separately, there has been a lack of work on group tour recommendations as a holistic problem. GROUPTOURREC is an important problem for tourism as most tourists travel in groups and there is increasing demand for customized tours [42].

In this chapter, we make the following contributions:

- We introduce and formulate the novel GROUPTOURREC problem, which involves recommending tours to groups of tourists with diverse interests and assigning tour guides with the right expertise to lead these tours. (Section 7.3)

- To overcome the NP-hard complexity of GROUPTOURREC, we propose a decomposition of GROUPTOURREC into a series of more manageable sub-problems, comprising tourist grouping, POI recommendation and tour guide assignment. (Section 7.4)

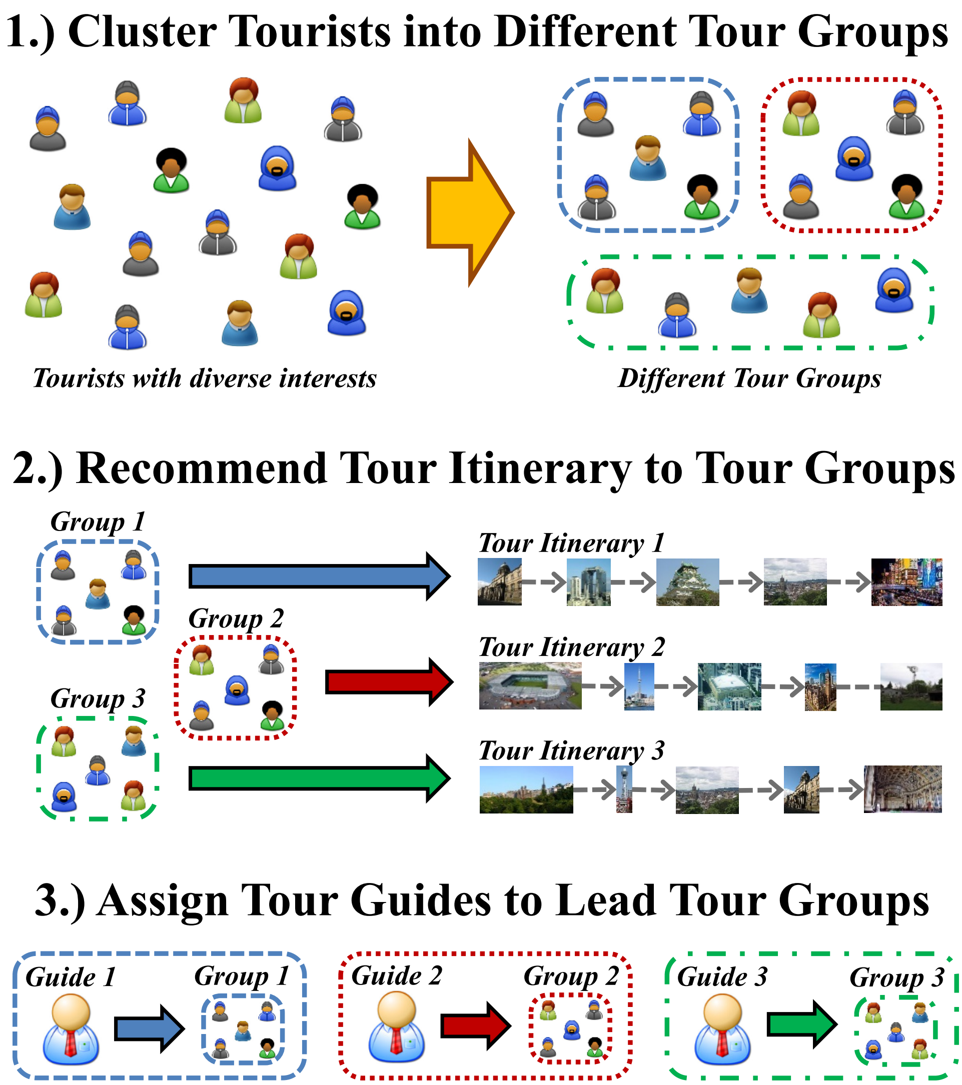


Figure 7.1: Group Tour Recommendation

– For tourist grouping, we model user interest based on their visit duration at specific POI categories, relative to their total visit duration. Thereafter, we use k-means and hierarchical clustering to group these users into different tour groups. (Section 7.4.1)

– For POI recommendation, we optimize a variant of the Orienteering problem, with considerations for user interests, POI popularity, starting/destination POIs and available time budget. We determine user interests and POI popularity using geo-tagged photos. (Section 7.4.2)

– For tour guide assignment, we model tour guide expertise based on the number of times they led a tour to a specific POI, relative to the number of visits by an average user. Thereafter, we use integer programming to assign tour guides to tour groups. (Section 7.4.3)

– We also utilize geo-tagged photos as a form of crowd-sourcing to determine real-life POI visits by users, which is then applied to our model of user interests, POI popularity and tour guide expertise. (Section 7.5.1)

- We evaluate our proposed approaches against various baselines, using a Flickr dataset comprising seven cities. Results show that our proposed approaches outperform these baselines using measures based on tour interest, group interest and tour guide expertise. (Sections 7.5 and 7.6)

For the remaining chapter: Section 7.2 discusses related work; Section 7.3 formally defines the GROUPTOURREC problem; Section 7.4 describes our proposed approach; Section 7.5 outlines our experimental methodology; Section 7.6 highlights our main results; and Section 7.7 concludes this chapter.

7.2 Related Work

Most earlier work has examined either group recommendations or tour recommendations as distinct research problems, instead of group tour recommendation as an integrated problem. Thus, we discuss the key literature in each of these two areas, before highlighting the differences with our work.

Group Recommendations (Retail). Group recommendations on retail items (e.g., movies, books, music) have been well-studied in recent years, such as by [2] that performed group recommendations using a consensus score that maximizes the item relevance to the entire group, while trying to minimize disagreement within the group. Others such as [71] used collective deep belief networks and dual-wing restricted Boltzmann machines to model group preferences as high level features, which are not biased towards specific individuals within the group. In contrast, [121] studied a complementary problem of constructing groups in ways such that recommended items are most relevant to members within these groups. For a more in-depth discussion, [17] provides a comprehensive survey on group recommendation systems.

Group Recommendations (Tourism) and Tour Recommendations. There also exists many interesting applications of group recommendations to the tourism domain. For example, *e-Tourism* [57, 58], *Intrigue* [4] and *Travel Decision Forum* [72] are relevant applications that have also been previously discussed in Chapter 2. In that same chapter, we have also reviewed a large selection of tour recommendation algorithms and systems targetted at individual tourists.

Differences with Related Work. These earlier works are the state-of-the-art in their respective and distinct areas of group recommendations and tour recommendations. However, our work differs from these earlier works in the following ways:

1. Most group recommendations for retail applications attempt to recommend (top-k) items such as movies, books and music, whereas recommending tour itineraries requires different considerations such as starting/ending POIs, time/distance budget, POI popularity and user interest preferences, constructed as a complete itinerary instead of only the top-k items;

2. Although traditional tour recommendation methods are effective for a single tourist, group tour recommendations need to address the additional challenges of grouping tourists, group interest alignment, and assignment of tour guides;

3. However, group recommendations in tourism assume that tour groups are predefined by tourists, whereas we model tourist interests then cluster tourists into groups based on their interests;

4. In addition, group recommendations for tourism do not consider the assignment of tour guides to lead tour groups, whereas we model tour guides' expertise and assign tour guides to tour groups based on their expertise; and

5. Moreover, group recommendations for tourism require users to explicitly enter their demographics details and general interests or select specific POIs, whereas we automatically determine user interests based on their past visits.

To the best of our knowledge, there has been no earlier work that investigates group tour recommendations as a holistic problem, with the exception of [3] that was published after our paper on this problem.

7.3 Background and Problem Definition

In this section, we introduce some preliminaries and formalize the GROUPTOURREC problem.

7.3.1 Preliminaries

For each city, let $T = \{t_1, ..., t_l\}$ be the set of tourists, $U = \{u_1, ..., u_m\}$ be the set of tour guides, $G = \{g_1, ..., g_m\}$ be the set of tour groups, and $P = \{p_1, ..., p_n\}$ be the set of POIs. In other words, there are l tourists, m tour guides and tour groups, and n POIs. Given that $C = \{c_1, ..., c_o\}$ denotes the set of all POI categories, each POI p belongs to a category $Cat_p \in C$. A future extension could include classifying POIs under multiple categories with different weightings for each category.

Definition 1: Tourist and Tour Guide Travel History. We represent the travel history of a tourist t as an ordered sequence $S_t = ((p_1, t^a_{p_1}, t^d_{p_1}), ..., (p_n, t^a_{p_n}, t^d_{p_n}))$, where $t^a_{p_x}$ and $t^d_{p_x}$ respectively denote the arrival and departure time at POI p_x.[1] Based on this travel history S_t, we can determine the duration a tourist t spends at POI p_x by calculating the difference between the arrival time $t^a_{p_x}$ and departure time $t^d_{p_x}$. Similarly, we define the total visiting time a tourist t spends at all POIs as: $V(t) = \sum\limits_{p \in S_t} (t^d_p - t^a_p)$.

Definition 2: POI Popularity and Tourist Interest Preference. As earlier described in Chapter 3, we adopt a simple but effective representation of POI popularity based on the number of times a POI is visited. For more details, refer to Section 3.3.1.

Unlike POI popularity, which is the same for all tourists, interest preferences are unique to each tourist, i.e., different tourists will have different interest preferences for various POI categories. Thus, we calculate the interest level of a tourist t in POI category

[1] $S_t = ((p_1, t^a_{p_1}, t^d_{p_1}), ..., (p_n, t^a_{p_n}, t^d_{p_n}))$ is also written as $S_t = (p_1, ..., p_n)$, for brevity of representation. Similarly, we use S_u to represent the travel history of a tour guide.

c using:

$$Int_t(c) = \sum_{p \in S_t} \frac{(t_p^d - t_p^a)}{V(t)} \delta(Cat_p{=}c), \quad \forall\, c \in C \tag{7.1}$$

where $\delta(Cat_p{=}c) = \{ ^{1,\ Cat_p=c}_{0,\ otherwise}$. That is, the interest level of a tourist t in a POI category c is based on the amount of time he/she spends at POIs of category c, relative to the total time he/she spends visiting all POIs. The basic intuition is that a tourist is more likely to spend more (less) time at a POI category that interests (uninterests) him/her.

Definition 3: Tourist Interest Preferences. Given the interest function $Int_t(c)$ introduced in Equation 7.1, we represent the interest vector of a tourist t as:

$$\vec{v_t} = \langle Int_t(c_1), ..., Int_t(c_o) \rangle, \quad \forall\, \{c_1, ...c_o\} \in C \tag{7.2}$$

Definition 4: Tour Guide Expertise. Next, we model the expertise level of a tour guide u in a specific POI p as:

$$Ept_u(p) = \frac{\sum\limits_{p_x \in S_u} \delta(p_x{=}p)}{\frac{1}{|T|} \sum\limits_{t \in T} \sum\limits_{p_y \in S_t} \delta(p_y{=}p)}, \quad \forall\, p \in P \tag{7.3}$$

where $\delta(p_x{=}p) = \{ ^{1,\ p_x=p}_{0,\ otherwise}$. In short, Equation 7.3 determines the expertise of a tour guide u in POI p based on the number of times he/she has visited this POI, relative to the number of visits by an average user. The basic intuition is that the more times a tour guide has led a tour to POI p, the more experienced this tour guide will be about POI p. We then represent the expertise vector of a tour guide u as:

$$\vec{v_u} = \langle Ept_u(p_1), ..., Ept_u(p_n) \rangle, \quad \forall\, \{p_1, ...p_n\} \in P \tag{7.4}$$

It should be noted that $|\vec{v_u}| = |P|$, i.e., the total number of possible points of expertise of a tour guide corresponds to the total number of POIs in the city. Another possible representation of tour guide expertise is based on his/her expertise in POI categories, instead of specific POIs. However, we chose the representation in Equation 7.4 as this measure is more fine-grained and representative of real-life, i.e., a tour guide is more likely to be an expert in various distinct POIs, rather than having the same level of expertise for all POIs of a certain category.

7.3.2 Problem Definition

We aim to address the GROUPTOURREC problem in terms of its sub-problems of tourist grouping, POI recommendation and tour guide assignment. Given that $T = \{t_1, ..., t_l\}$, $U = \{u_1, ..., u_m\}$, $G = \{g_1, ..., g_m\}$, and $P = \{p_1, ..., p_n\}$ are respectively the set of tourists, tour guides, tour groups, and POIs, our main goal is find $w_{t,g}$, $y_{g,p}$, and $z_{u,g}$ that maximize the following objective function:

$$\begin{aligned} \alpha \sum_{g \in G} \sum_{t \in T} \sum_{p \in P} w_{t,g} y_{g,p} \Big(\eta Int_t(Cat_p) + (1-\eta) Pop(p) \Big) \\ + (1-\alpha) \sum_{g \in G} \sum_{u \in U} \sum_{p \in P} z_{u,g} y_{g,p} Ept(u,p) \end{aligned} \tag{7.5}$$

where:

$$w_{t,g} = \begin{cases} 1, & \text{if tourist } t \text{ is assigned to group } g \\ 0, & \text{otherwise} \end{cases}$$

$$y_{g,p} = \begin{cases} 1, & \text{if group } g \text{ is recommended poi } p \\ 0, & \text{otherwise} \end{cases}$$

$$z_{u,g} = \begin{cases} 1, & \text{if tour guide } u \text{ is assigned to group } g \\ 0, & \text{otherwise} \end{cases}$$

such that:

$$w_{t,g}, y_{g,p}, z_{u,g} \in \{0, 1\} \tag{7.6}$$

$$\sum_{g \in G} w_{t,g} = 1, \qquad \forall\, t \in T \tag{7.7}$$

$$\sum_{g \in G} z_{u,g} = 1, \quad \forall\, u \in U \tag{7.8}$$

$$\sum_{u \in U} z_{g,u} = 1, \quad \forall\, g \in G \tag{7.9}$$

$$\sum_{p \in P} Cost(p_i, p_{i+1}) y_{g,p} \leq B, \quad \forall\, g \in G \tag{7.10}$$

In Equation 7.5, the parameter $\alpha \in [0, 1]$ controls the weighting between the components of: (i) user grouping and POI recommendation; and (ii) tour guide assignment. Thus, Equation 7.5 is maximized when the two components are maximized. We use $\alpha = 0.5$ to give a balanced emphasis on both components. Constraint 7.7 ensures that each tourist is assigned to exactly one tour group. Constraint 7.8 ensures that each tour guide leads exactly one tour group, and Constraint 7.9 ensures that each tour group is led by only one tour guide. For each tour group g, Constraint 7.10 ensures that the total time required to complete the tour itinerary is within a budget B (more details are provided in Section 7.4.2).

While we use Equation 7.5 to formalize the GROUPTOURREC problem, it can be easily generalized to other problems. Consider the application of Equation 7.5 to a project management problem comprising of members, teams, projects and managers. The idea would be to: (i) assign members to teams to either diversify or specialize their skill-set, i.e., $w_{t,g}$; (ii) assign projects that best match the skill-set of specific teams, i.e., $y_{g,p}$; and (iii) assign managers to lead each team based on their experience with the projects, i.e., $z_{u,g}$. Similarly, instead of optimizing for tourist interests, POI popularity and guide expertise in Equation 7.5, we will optimize for the member's skill-set, project requirements and manager's experience.

Equation 7.5 is an instance of a non-linear Integer Programming problem, resembling multiple quadratic assignment problems that are also dependent on each other. Quadratic programming problems are NP-hard [29, 103], thus Equation 7.5 as a whole is also NP-hard. Hence we propose a decomposition of GROUPTOURREC into more manageable sub-problems and use greedy approaches to solve each sub-problem separately.

7.4 Group Tour Recommendation Framework

Due to the NP-hard complexity of GROUPTOURREC, solving Equation 7.5 optimally is not feasible. As stated earlier in Section 7.1, we divide GROUPTOURREC into three more manageable sub-problems, namely:

1. **Tourist Grouping**: How do we divide the l tourists into m tour groups, maximizing the interest similarity of all tourists in a group?

2. **POI Recommendation**: How do we plan a tour itinerary comprising a subset of the n POIs that is most interesting to each tour group?

3. **Tour Guide Assignment**: How do we assign each tour group to a specific tour guide, matching the interest preferences of each group to the expertise of each guide?

In the following sections, we describe our approaches to solving each of these sub-problems.

7.4.1 Tourist to Tour Group Allocation (Tourist2Group)

Given that the l tourists are divided into m tour groups, let $G = \{g_1, ..., g_m\}$ be the set of tour groups, and $g_k = \{t_1, ..., t_q\}$ denote the kth tour group that comprises q tourists. Formally, our goal is to find a grouping G that:

$$Max \sum_{g \in G} \sum_{t_i \in g} \sum_{t_j \in g, t_j \neq t_i} \frac{\vec{v_{t_i}} \cdot \vec{v_{t_j}}}{||\vec{v_{t_i}}|| \; ||\vec{v_{t_j}}||} \tag{7.11}$$

where $\frac{\vec{v_{t_i}} \cdot \vec{v_{t_j}}}{||\vec{v_{t_i}}|| \; ||\vec{v_{t_j}}||}$ is the cosine similarity of two tourists t_i and t_j. This cosine similarity tells us how similar two users are in terms of their interest preferences. Thus in Equation 7.11, we are maximizing the interest preference among all user pairs in each tour group g_k, and for all m tour groups.

Optimal solutions to this clustering problem have been shown to be NP-hard [1]. As such, we employ the following algorithms to obtain approximate solutions to this problem. The algorithms are:

- **k-means clustering (KMEAN).** An iterative algorithm that assigns points (users) to their nearest centroid (group) [68]. This assignment then leads to an update of the centroid, and the assignment and update steps are repeated until the algorithm converges.

- **Hierarchical clustering (HIERA).** An agglomerative, hierarchical clustering algorithm that aims to minimize the variance within groups, using Euclidean distances (based on interests) between users in a group [146].

Given POI categories $C = \{c_1, ..., c_o\}$ and two users i and j, their Euclidean distance is defined as: $\sqrt{(Int_i(c_1) - Int_j(c_1))^2 + ... + (Int_i(c_o) - Int_j(c_o))^2}$.

7.4.2 POIs Recommendations to Tour Group (POI2Group)

One main challenge in recommending and planning tours for a group is the diverse interest preferences among members of the tour group. To address these diverse interest preferences, we construct a collective group interest preference based on the average interest preference of all group members. For a tour group $g = \{t_1, ..., t_q\}$, this collective group interest preference is defined as:

$$\vec{v}_g = \frac{1}{|g|} \sum_{t \in g} \vec{v}_t, \qquad \forall\, g \in G \tag{7.12}$$

Similar to $Int_t(c)$ (Equation 7.1), we define a function $Int_g^{group}(c)$ that determines the interest level of tour group g in POI category c, based on $\vec{v}_g$.

Thereafter, we approach this tour recommendation problem as an instance of the Orienteering problem [136, 142], with a time budget B, starting POI p_1 and destination POI p_N. Our main goal is to recommend a tour itinerary $I = (p_1, ..., p_N)$ that maximizes POI popularity and tourist interest, while staying within the time budget B. As described in Section 7.3.1, POI popularity and tourist interest are defined based on the functions $Pop(p)$ (Equation 3.1) and $Int_t(c)$ (Equation 7.1), respectively. Formally, we want to plan a tour itinerary $I = (p_1, ..., p_N)$ for a tour group g that:

$$Max \sum_{i=1}^{N-1} \sum_{j=2}^{N} x_{i,j} \Big(\eta Int_g^{group}(Cat_i) + (1-\eta) Pop(i) \Big) \tag{7.13}$$

where $x_{i,j} = \{\begin{smallmatrix}1, \text{ travel from POI } i \text{ to } j \\ 0, \text{ otherwise}\end{smallmatrix}$, such that:

$$\sum_{j=2}^{N} x_{1,j} = \sum_{i=1}^{N-1} x_{i,N} = 1 \tag{7.14}$$

$$\sum_{i=1}^{N-1} x_{i,k} = \sum_{j=2}^{N} x_{k,j} \leq 1, \quad \forall\, k = 2, ..., N-1 \tag{7.15}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} Cost(i,j) x_{i,j} \leq B \tag{7.16}$$

$$2 \leq p_i \leq N, \quad \forall\, i = 2, ..., N \tag{7.17}$$

$$p_i - p_j + 1 \leq (N-1)(1 - x_{i,j}), \quad \forall\, i, j = 2, ..., N \tag{7.18}$$

Equation 7.13 aims to maximize the POI popularity $Pop(i)$ and tour group interest $Int_g^{group}(Cat_i)$ of the recommended tour, with $\eta \in [0,1]$ as the weight assigned to POI popularity and group interest. Constraint 7.14 ensures that the recommended tour starts and ends at POI 1 and N, respectively. Constraint 7.15 ensures that no POIs are re-visited and all paths are connected. Constraint 7.16 ensures that the total time needed to visit all POIs in the recommended tour is within the budget B based on the function $Cost(p_x, p_y)$, which considers the travelling time between POIs and visit duration at each POI. To eliminate sub-tours, we adapted the constraints from [108] as listed in Equation 7.17 and 7.18. We then proceed to solve this tour recommendation problem as an integer programming problem, using the lpsolve linear programming package [13].

7.4.3 Tour Guide to Tour Group Assignment (Guide2Group)

Given that $U = \{u_1, ..., u_m\}$ is the set of tour guides and $I_g = (p_1, ..., p_N)$ is the tour itinerary recommended to a tour group $g \in G$, our goal is to find a tour guide u for tour group g that:

$$Max \sum_{g \in G} \sum_{u \in U} \sum_{p \in I_g} z_{u,g} y_{g,p} Ept(u, p) \tag{7.19}$$

In short, we want to best match tour guides to tour groups based on the tour guides' expertise and the POIs recommended to the tour groups. Similar to Section 7.4.2, we solve this as an integer programming problem and denote this approach **OPTIM**.

7.5 Experimental Methodology

In this section, we describe our experimental dataset, the various baseline algorithms and our evaluation methodology.

7.5.1 Dataset

We perform our experiments using the publicly available Yahoo! Flickr Creative Commons 100M (YFCC100M) dataset [132, 151]. This dataset comprises 100M Flickr photos and videos, along with their meta-data such as the date/time taken and latitude/longitude coordinates. The latitude/longitude coordinates are associated with a geo-location accuracy ranging from 1 to 16 (least to most accurate). Such geo-tagged photos serve as a good approximation for real-life tourist visits and are used in similar tour recommendation works [24, 40].

For our experiments, we pre-processed the YFCC100M dataset by extracting geo-tagged photos that were taken in seven different cities, namely: Toronto, Vienna, Osaka, Budapest, Glasgow, Delhi and Edinburgh. We selected these seven touristic cities around the world to ensure the generalizability of our experimental results. In addition, we only considered photos with the highest geo-location accuracy of 16 to ensure the accuracy of our results. These geo-tagged photos are then mapped to a list of POI locations (obtained from Wikipedia), thus providing us with a proxy of users' real-life POI visits, which is then formulated as Definitions 1 to 4 (Section 7.3.1). Based on Definition 1, we use the time taken of a user's first and last photo at a POI to determine his/her POI visit duration. More details on this procedure was earlier described in Sec-

Table 7.1: Summary Statistics of Dataset

City	Number of Photos	Number of Users	Number of POI Visits	Number of Travel Sequences
Toronto	157,505	1,395	39,419	6,057
Vienna	85,149	1,155	34,515	3,193
Osaka	392,420	450	7,747	1,115
Budapest	36,000	935	18,513	2,361
Glasgow	29,019	601	11,434	2,227
Delhi	13,919	279	3,993	489
Edinburgh	82,060	1,454	33,944	5,028

tion 3.3.3. Table 7.1 summarizes the main statistics of our dataset, which is available at https://sites.google.com/site/limkwanhui/datacode#icaps16.

Using this dataset, we first construct the interest preference vector $\vec{v}_t$ for all tourists $t \in T$, as stated in Equation 7.2. Next, we construct the expertise vector $\vec{v}_u$ for all tour guides $u \in U$, as stated in Equation 7.4. In our experiments, we define tour guides as users who visited the most number of POIs and thus have the highest expertise level in the most POIs.[2]

7.5.2 Baseline Algorithms

Tourist2Group Baselines. For the Tourist2Group allocation, we compare the KMEAN and HIERA algorithms against the following baselines:

- **First-Come-First-Allocated (FCFA)**. Tourists are assigned into tour groups based on their time of arrival.
- **Random Allocation (RAND)**. Tourists are randomly assigned into each tour group.

FCFA reflects the modus operandi of tour operators, where they allocate tourists into

[2]Due to a lack of explicitly identified tour guides, we use the most well-visited users as a proxy for tour guides. Our assumption is that tour guides are the most well-visited (compared to the average tourist), due to the tour guide having led many previous tours. In real-life, a tour agency is likely to have a profile of its tour guides or be able to obtain their expertise directly from their tour guides.

tour groups based on the order in which they signed up for the tours. RAND shows the effectiveness of a random-based approach.

POI2Group Baselines. After the Tourist2Group allocation using the KMEAN, HIERA, FCFA and RAND algorithms, we next determine the tour group interest (Equation 7.12) and then proceed to make POI2Group recommendations for each tour group. As described in Section 7.4.2, the POI2Group recommendations can be customized based on the value of η, and we use the following variants:

- **POI2Group with** η=0.5 **(I.5)**. Compute POI2Group recommendations with a balanced weighting on both POI popularity and group interest.

- **POI2Group with** η=1 **(I1)**. Compute POI2Group recommendations with a full weighting on group interest, with no consideration for POI popularity.

We use KMEAN-I.5 to denote the allocation of tourists to tour groups using the KMEAN algorithm, and the recommendation of POIs to these tour groups using η=0.5 (I.5). We use a similar notation for the other Tourist2Group allocations (KMEAN, HIERA, FCFA and RAND) combined with the different POI2Group recommendations (I.5 and I1).[3] Thus for the POI2Group recommendations, we are essentially comparing the {KMEAN, HIERA}-{I.5, I1} algorithms against the {FCFA, RAND}-{I.5, I1} baselines.

In addition, we also compare against a baseline of standard tour packages offered by tour operators, defined as:

- **STDTOUR**. Actual tour itineraries offered by the following tour operators: viator.com (Toronto and Osaka), budapest.com (Budapest), viennacitytours.rezgo.com (Vienna), scottishtours.co.uk (Glasgow), delhitourism.gov.in (Delhi), and edinburgh-tour.com (Edinburgh).

The main purpose of this baseline is to determine how our customized tours perform against standard tour packages, in terms of satisfying the interest preferences of tourists. In order to ensure a fair comparison between our algorithms and the STDTOUR baseline,

[3]Note that we do not use η=0, which results in POI2Group recommendations based only on POI popularity. Such recommendations ignore user interests and are not the focus of this work.

we only evaluate with travel sequences that comprise the same number of POIs as that in STDTOUR (for each city in our dataset).

Guide2Group Baselines. For the Guide2Group assignment, we compare our OPTIM approach (Section 7.4.3) against a **Random Assignment (RANDA)** baseline where each tour group is randomly assigned a tour guide. As our GROUPTOURREC problem comprises the Tourist2Group, POI2Group and Guide2Group components, our overall evaluation involves comparing the {KMEAN, HIERA}-{I.5-OPTIM, I1} algorithms against the {FCFA, RAND}-{I.5, I1}-RANDA baselines.

7.5.3 Evaluation

Our experimental evaluation is based on all travel sequences with $\geq$3 POI visits in our dataset.[4] For each of these travel sequences, we perform the following evaluations:

1. **Tourist2Group Evaluation.** Based on the entire set of users for each city, we randomly select 100 users and group these users into five tour groups using the various Tourist2Group allocation algorithms. We evaluate each Tourist2Group allocation using the following metrics:

   - ***Jaccard Similarity:*** $Jac(g)$**.** The average Jaccard similarity of all pair-wise combinations of tourists in group g. Let $g = \{t_1, ..., t_q\}$ denote a tour group g that comprises q tourists, then Jaccard similarity is defined as: $Jac(g) = \frac{1}{|g|} \sum\limits_{t_i \in g} \sum\limits_{t_j \in g, t_j \neq t_i} \frac{|v_{t_i} \cap v_{t_j}|}{|v_{t_i} \cup v_{t_j}|}$, where v_{t_i} is the binary version of $\vec{v_{t_i}}$ (non-zero values converted to 1s).

   - ***Cosine Similarity:*** $Cos(g)$**.** The average cosine similarity of all pair-wise combinations of tourists in group g. Let $g = \{t_1, ..., t_q\}$ denote a tour group g that comprises q tourists, then cosine similarity is defined as: $Cos(g) = \frac{1}{|g|} \sum\limits_{t_i \in g} \sum\limits_{t_j \in g, t_j \neq t_i} \frac{\vec{v_{t_i}} \cdot \vec{v_{t_j}}}{||\vec{v_{t_i}}|| \, ||\vec{v_{t_j}}||}$.

   - ***Common Top Interest:*** $Com(g)$**.** The largest proportion of tourists in group g with the same interest $c \in C$. Let $g = \{t_1, ..., t_q\}$ denote a tour group g

[4] Rather than pick random POIs to construct artificial itineraries, we use these travel sequences as they reflect real-life itineraries and serve as a more realistic evaluation.

that comprises q tourists, then common top interest is defined as: $Com(g) = \max_{c \in C} \frac{1}{|g|} \sum_{t_i \in g} \delta(Int_{t_i}(c))$, where $\delta(Int_{t_i}(c)) = \{^{1,\ Int_{t_i}(c) \neq 0}_{0,\ otherwise}$.

These similarity measures show how similar users in the allocated group are in terms of their interest, and allow us to determine the effectiveness of our Tourist2Group allocation algorithms against the baselines. As there is no ground truth on the real-life groups, we use these heuristics to measure the effectiveness of the group allocations.

2. **POI2Group Evaluation.** Based on each allocated tour group (from Step 1 above), we run the various POI2Group recommendation algorithms using the starting POIs and destination POIs of these real-life travel sequences. Thus the POI2Group recommendation results in a tour itinerary $I_g = (p_1, ..., p_N)$ of N POIs that is recommended to a group g. We evaluate each POI2Group recommendation using the following metrics:

   - ***Tour Total Interest:*** $Tot_I(t)$**.** The total interest of all POIs in the recommended itinerary I to a tourist t in group g. Tour total interest is defined as: $Tot_I(t) = \sum_{p \in I_g} Int_t(Cat_p)$.
   - ***Tour Maximum Interest:*** $Max_I(t)$**.** The maximum interest out of all POIs in the recommended itinerary I to a tourist t in group g. Tour maximum interest is defined as: $Max_I(t) = \max_{p \in I_g} Int_t(Cat_p)$.
   - ***Tour Minimum Interest:*** $Min_I(t)$**.** The minimum interest out of all POIs in the recommended itinerary I to a tourist t in group g. Tour minimum interest is defined as: $Min_I(t) = \min_{p \in I_g} Int_t(Cat_p)$.

   Total interest allows us to determine how interested all tourists in a tour group are regarding the recommended tour. In addition, we also use maximum and minimum interests to determine the best and worst case of a recommended tour, respectively, i.e., akin to the most interested and least interested tourist in each group.

3. **Guide2Group Evaluation.** Based on the POI2Group recommendations to each tour group (from Step 2 above), we run the various Guide2Group assignment algo-

rithms using the tour guide expertise and recommended tour for each group. We evaluate each Guide2Group assignment using the following metric:

- ***Guide Total Expertise:*** $Tot_E(t)$**.** The total expertise of a tour guide u in all POIs of an itinerary I recommended to group g. Guide total expertise is defined as: $Tot_E(t) = \sum\limits_{p \in I_g} Ept_u(p)$.

4. **Overall Evaluation.** To evaluate GROUPTOURREC as a whole (i.e., Tourist2Group Allocation, POI2Group Recommendation and Guide2Group Assignment as an integrated component), we use an evaluation metric that is based on our main objective score (Equation 7.5).

   - ***Objective Score:*** $ObjScore$**.** Refer to Equation 7.5.

   As we focus more on user interests than POI popularity, we use $\eta = 1$ in this objective score to better measure user interests in the recommended POIs.[5] As stated in Equation 7.5, the α parameter determines the emphasis between the two components of: (i) user grouping and POI recommendation; and (ii) tour guide assignment. In this experiment, we evaluate the performance of the algorithms based on multiple α values from 0 to 1, in intervals of 0.05.

Each of the above-mentioned evaluations (1 to 4) are then repeated multiple times for each city, based on the number of travel sequences in that city (Table 7.1). In the following sections, we report the average score and standard error of each metric, and conduct t-tests to determine if the improvements of our proposed approaches are statistically significant.

7.6 Results and Discussion

In this section, we present and discuss the experimentation results for the overall GROUPTOURREC problem, and the individual components of Tourist2Group Allocation, POI2Group Recommendation and Guide2Group Assignment.

[5]Furthermore, POI popularity is the same for all tourists, while user interests are unique to each tourist. As a result, any evaluation based solely on POI popularity ($\eta = 0$) results in the same score for all tourists.

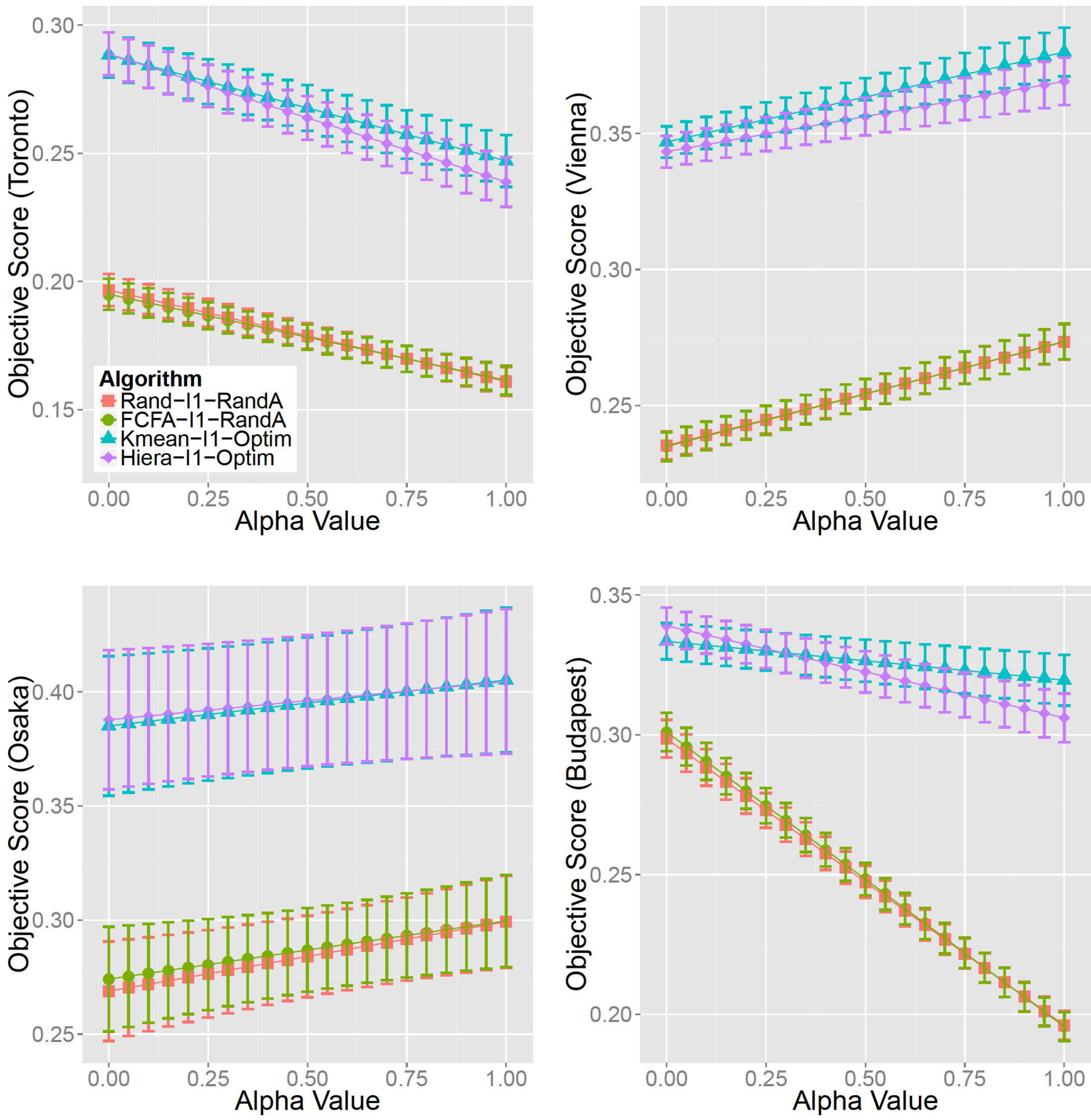


Figure 7.2: Overall comparison of the {KMEAN,HIERA}-I1-OPTIM algorithms against the {FCFA, RAND}-I1-RANDA baselines for Toronto, Vienna, Osaka and Budapest, in terms of our main objective score (described in Section 7.5.3).

7.6.1 Overall Evaluation of GroupTourRec

Figure 7.2, 7.3, 7.4 and 7.5 shows the objective score (described in Section 7.5.3) of our proposed algorithms and baselines at multiple α values. While there are different trends for different cities, our proposed algorithms ({KMEAN, HIERA}-{I.5, I1}-OPTIM) consistently out-perform the baselines ({FCFA, RAND}-{I.5, I1}-RANDA) in all cases and

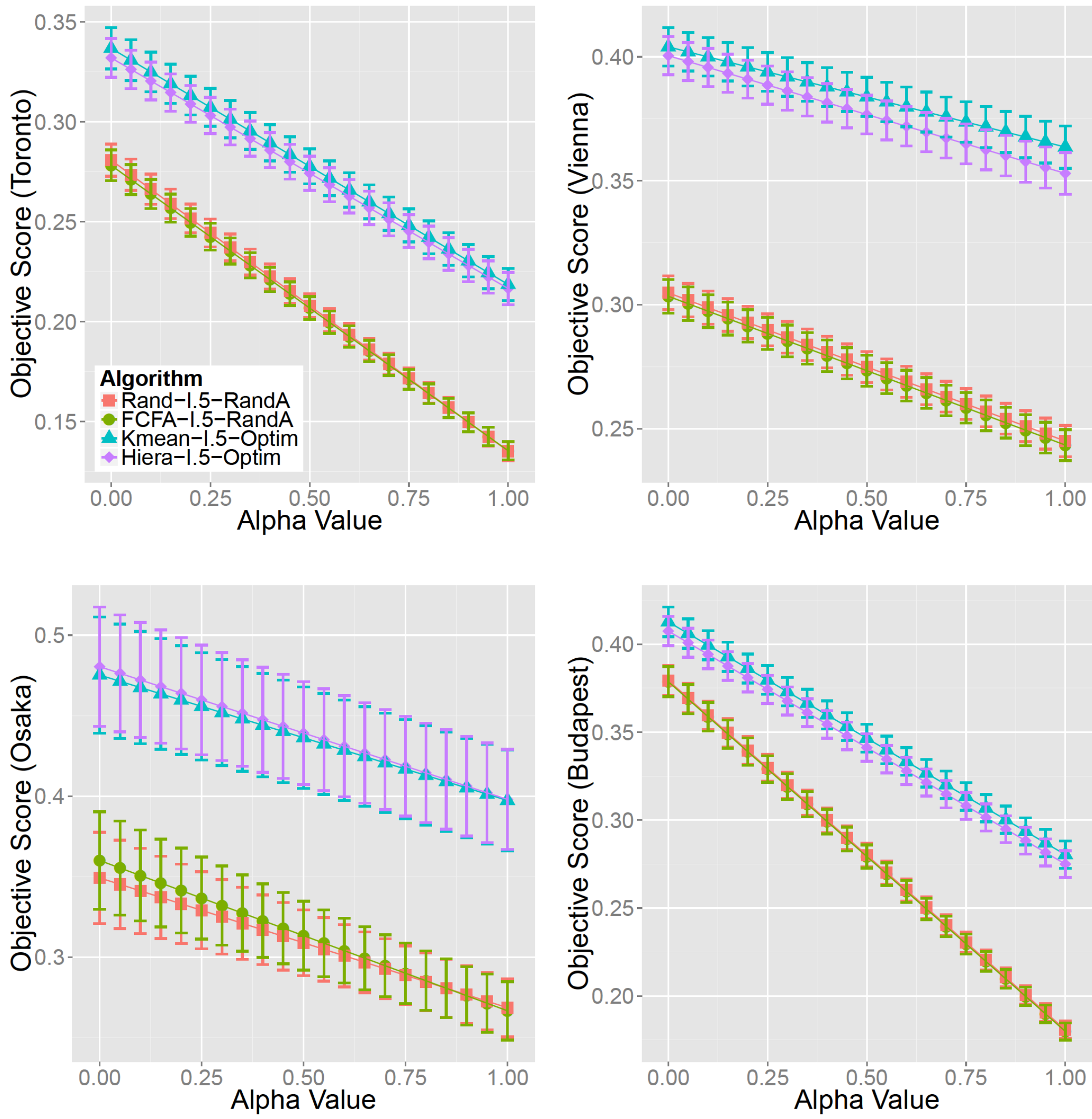


Figure 7.3: Overall comparison of the {KMEAN,HIERA}-I.5-OPTIM algorithms against the {FCFA, RAND}-I.5-RANDA baselines for Toronto, Vienna, Osaka and Budapest, in terms of our main objective score (described in Section 7.5.3).

for all cities, regardless of α values. The consistent out-performance at all α values also shows that our proposed algorithms offer better performance in terms of both components of user grouping and POI recommendation, and tour guide assignment, regardless of the emphasis that a tourist might place on either component. Our proposed algorithms out-perform the various baselines for GROUPTOURREC as a whole, and we further investigate its performance for the sub-problems of Tourist2Group, POI2Group and

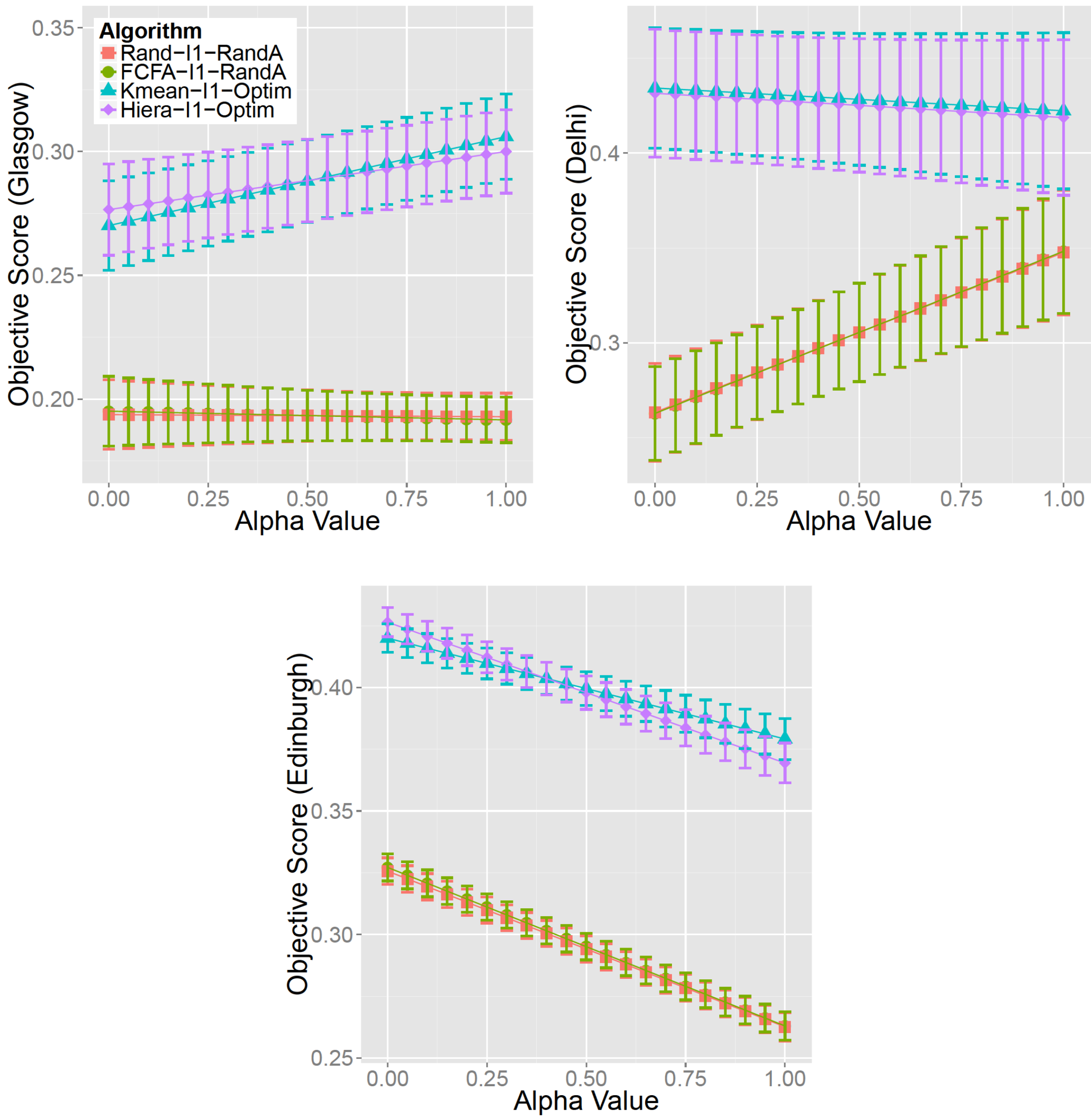


Figure 7.4: Overall comparison of the {KMEAN,HIERA}-I1-OPTIM algorithms against the {FCFA, RAND}-I1-RANDA baselines for Glasgow, Delhi and Edinburgh, in terms of our main objective score (described in Section 7.5.3).

Guide2Group.

7.6.2 Evaluation of Tourist2Group Allocation

We now study the effectiveness of Tourist2Group allocations using our proposed approaches of using the KMEAN and HIERA algorithms, compared to the FCFA and RAND

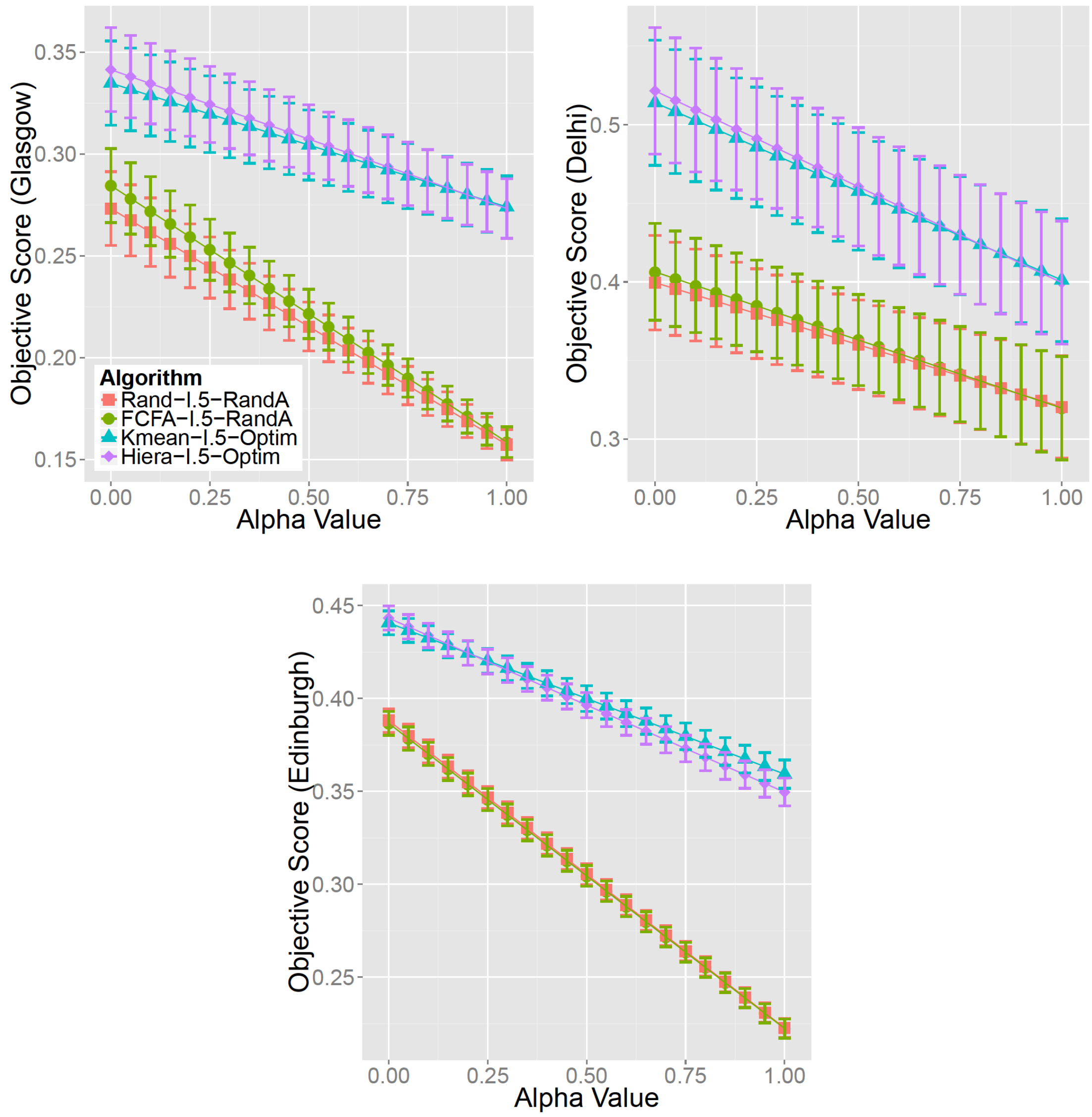


Figure 7.5: Overall comparison of the {KMEAN,HIERA}-I.5-OPTIM algorithms against the {FCFA, RAND}-I.5-RANDA baselines for Glasgow, Delhi and Edinburgh, in terms of our main objective score (described in Section 7.5.3).

baselines, as shown in Table 7.2. The results show that KMEAN and HIERA consistently out-perform the baselines of FCFA and RAND with relative improvements of more than 45.4%, 134.4% and 93.9% in terms of *Jac*, *Cos* and *Com* scores, respectively. Furthermore, the t-test results show that these improvements are statistically significant with p<.0001. These results show how KMEAN and HIERA result in groups comprising members whose interests are more similar to one another, compared to the FCFA and RAND baselines. In

Table 7.2: Tourist2Group comparison of the KMEAN, HIERA algorithms against the FCFA, RAND baselines, in terms of Jaccard Similarity (*Jac*), Cosine Similarity (*Cos*) and Common Top Interest (*Com*). † denotes a significant difference with both FCFA and RAND, based on two-sided t-tests with p<.0001.

Toronto			
Algo.	*Jac*	*Cos*	*Com*
RAND	.383±.002	.263±.001	.355±.002
FCFA	.382±.002	.263±.001	.356±.002
KMEAN	.723±.003†	.836±.004†	**.893±.004**†
HIERA	**.738±.002**†	**.853±.005**†	.882±.005†

Vienna			
Algo.	*Jac*	*Cos*	*Com*
RAND	.495±.002	.338±.001	.422±.002
FCFA	.494±.002	.336±.001	.421±.002
KMEAN	.788±.004†	.816±.004†	.856±.004†
HIERA	**.815±.003**†	**.860±.005**†	**.870±.005**†

Osaka			
Algo.	*Jac*	*Cos*	*Com*
RAND	.414±.004	.373±.003	.459±.005
FCFA	.409±.004	.373±.003	.456±.005
KMEAN	.908±.013†	.921±.008†	.902±.012†
HIERA	**.923±.013**†	**.929±.008**†	**.911±.012**†

Budapest			
Algo.	*Jac*	*Cos*	*Com*
RAND	.561±.003	.319±.001	.357±.001
FCFA	.559±.003	.320±.001	.359±.001
KMEAN	.816±.004†	.837±.003†	**.898±.003**†
HIERA	**.830±.003**†	**.870±.004**†	.887±.004†

Glasgow			
Algo.	*Jac*	*Cos*	*Com*
RAND	.298±.003	.206±.002	.292±.002
FCFA	.296±.003	.205±.002	.296±.002
KMEAN	.674±.006†	.820±.009†	**.879±.008**†
HIERA	**.709±.006**†	**.843±.011**†	.875±.010†

Delhi			
Algo.	*Jac*	*Cos*	*Com*
RAND	.406±.006	.356±.005	.443±.006
FCFA	.411±.006	.360±.006	.441±.007
KMEAN	.791±.014†	.844±.015†	**.860±.017**†
HIERA	**.830±.008**†	**.863±.016**†	.859±.018†

Edinburgh			
Algo.	*Jac*	*Cos*	*Com*
RAND	.475±.002	.321±.001	.397±.001
FCFA	.473±.002	.319±.001	.394±.001
KMEAN	.787±.003†	.871±.002†	**.928±.003**†
HIERA	**.802±.002**†	**.882±.003**†	.906±.003†

Table 7.3: POI2Group comparison of the {Kmean, Hiera}-{I.5, I1} algorithms against the {FCFA, Rand}-{I.5, I1} baselines for Toronto, Vienna, Budapest and Osaka, in terms of Tour Total Interest (Tot_I), Tour Maximum Interest (Max_I) and Tour Minimum Interest (Min_I). † denotes a significant difference with both FCFA and Rand, based on two-sided t-tests with p<.0001.

Toronto			
Algo.	Tot_I	Max_I	Min_I
Rand-I1	0.82±.005	.530±.003	.054±.001
FCFA-I1	0.82±.005	.531±.003	.054±.001
Kmean-I1	**1.26±.006**†	**.737±.002**†	**.092±.001**†
Hiera-I1	1.22±.006†	.728±.002†	.084±.001†
Rand-I.5	0.69±.004	.497±.003	.038±.001
FCFA-I.5	0.69±.004	.496±.003	.038±.001
Kmean-I.5	**1.12±.005**†	**.701±.002**†	**.074±.001**†
Hiera-I.5	1.10±.005†	.698±.002†	.069±.001†

Vienna			
Algo.	Tot_I	Max_I	Min_I
Rand-I1	1.59±.007	.663±.002	.139±.001
FCFA-I1	1.59±.007	.664±.002	.137±.001
Kmean-I1	**2.21±.007**†	**.886±.001**†	**.200±.002**†
Hiera-I1	2.15±.007†	.863±.001†	.186±.002†
Rand-I.5	1.41±.006	.665±.002	.139±.001
FCFA-I.5	1.40±.006	.662±.002	.141±.001
Kmean-I.5	**2.10±.007**†	**.867±.001**†	**.225±.002**†
Hiera-I.5	2.04±.007†	.852±.002†	.210±.002†

Osaka			
Algo.	Tot_I	Max_I	Min_I
Rand-I1	1.52±.023	.658±.007	.172±.005
FCFA-I1	1.52±.022	.676±.007	.163±.005
Kmean-I1	**2.05±.021**†	.871±.005†	**.239±.006**†
Hiera-I1	2.05±.021†	**.872±.005**†	.231±.006†
Rand-I.5	1.36±.019	.691±.007	.137±.005
FCFA-I.5	1.35±.019	.692±.007	.133±.005
Kmean-I.5	**2.01±.021**†	**.872±.005**†	**.249±.006**†
Hiera-I.5	2.01±.021†	.871±.005†	.247±.006†

Budapest			
Algo.	Tot_I	Max_I	Min_I
Rand-I1	1.51±.007	.639±.002	.102±.001
FCFA-I1	1.51±.007	.640±.002	.102±.001
Kmean-I1	**2.46±.008**†	**.875±.001**†	**.167±.001**†
Hiera-I1	2.36±.008†	.845±.001†	.154±.001†
Rand-I.5	1.39±.006	.634±.002	.086±.001
FCFA-I.5	1.38±.006	.635±.002	.086±.001
Kmean-I.5	**2.16±.007**†	**.830±.002**†	**.138±.001**†
Hiera-I.5	2.13±.007†	.812±.002†	.133±.001†

the next section, we further examine the effects of a good Tourist2Group allocation on the subsequent POI2Group recommendations.

7.6.3 Evaluation of POI2Group Recommendation

Next, we evaluate the effectiveness of the various POI2Group recommendations of the {Kmean, Hiera}-{I.5, I1} algorithms, against the {FCFA, Rand}-{I.5, I1} baselines.

Table 7.4: POI2Group comparison of the {KMEAN, HIERA}-{I.5, I1} algorithms against the {FCFA, RAND}-{I.5, I1} baselines for Glasgow, Delhi and Edinburgh, in terms of Tour Total Interest (Tot_I), Tour Maximum Interest (Max_I) and Tour Minimum Interest (Min_I). † denotes a significant difference with both FCFA and RAND, based on two-sided t-tests with p<.0001.

Glasgow				Delhi			
Algo.	Tot_I	Max_I	Min_I	*Algo.*	Tot_I	Max_I	Min_I
RAND-I1	0.71±.008	.502±.005	.048±.002	RAND-I1	1.51±.026	.645±.007	.145±.005
FCFA-I1	0.70±.008	.498±.005	.049±.002	FCFA-I1	1.52±.026	.644±.008	.139±.005
KMEAN-I1	**1.12±.009†**	**.717±.004†**	**.097±.003†**	KMEAN-I1	**1.84±.025†**	**.813±.006†**	.165±.006‡
HIERA-I1	1.10±.009†	.713±.004†	.090±.003†	HIERA-I1	1.82±.025†	.799±.006†	**.169±.006‡**
RAND-I.5	0.58±.007	.447±.005	.036±.002	RAND-I.5	1.40±.024	.648±.007	.109±.005
FCFA-I.5	0.58±.007	.448±.005	.036±.002	FCFA-I.5	1.39±.025	.645±.008	.115±.005
KMEAN-I.5	**1.00±.009†**	**.666±.004†**	.084±.002†	KMEAN-I.5	**1.75±.025†**	**.800±.006†**	.166±.006†
HIERA-I.5	1.00±.009†	.662±.004†	**.089±.003†**	HIERA-I.5	1.74±.024†	.792±.006†	**.169±.006†**

Edinburgh			
Algo.	Tot_I	Max_I	Min_I
RAND-I1	1.49±.006	.668±.002	.082±.001
FCFA-I1	1.49±.006	.668±.002	.084±.001
KMEAN-I1	**2.16±.006†**	**.915±.001†**	**.134±.001†**
HIERA-I1	2.10±.006†	.901±.001†	.123±.001†
RAND-I.5	1.26±.005	.691±.002	.061±.001
FCFA-I.5	1.26±.005	.688±.002	.062±.001
KMEAN-I.5	**2.04±.006†**	**.907±.001†**	**.150±.001†**
HIERA-I.5	1.99±.005†	.892±.001†	.136±.001†

Tables 7.3 and 7.4 show that the {KMEAN, HIERA}-{I.5, I1} algorithms out-perform the {FCFA, RAND}-{I.5, I1} baselines for all cases, with relative improvements of more than 19.7%, 22.2% and 13.8% based on measures of Tot_I, Max_I and Min_I, respectively. Similarly, these improvements are statistically significant as indicated by t-test results with p<.0001. These results show that our proposed algorithms successfully recommend tour itineraries that are more aligned to the interests of tourists in each tour group, compared to those of the baseline methods.

Table 7.5: POI2Group comparison of the {KMEAN, HIERA, FCFA, RAND}-{I.5, I1} algorithms against the STDTOUR baseline for Toronto, Vienna, Osaka and Budapest, in terms of Tour Total Interest (Tot_I), Tour Maximum Interest (Max_I) and Tour Minimum Interest (Min_I). † denotes a significant difference with the STDTOUR baseline, based on two-sided t-tests with p<.0001, ‡ denotes the same with p<.01.

Toronto				**Vienna**			
Algo.	Tot_I	Max_I	Min_I	*Algo.*	Tot_I	Max_I	Min_I
STDTOUR	1.33±.002	.644±.001	.007±.000	STDTOUR	1.90±.003	.800±.001	.022±.000
RAND-I1	1.59±.032†	.722±.011†	.016±.002†	RAND-I1	2.17±.020†	.738±.005†	.097±.003†
FCFA-I1	1.59±.033†	.717±.011†	.023±.003†	FCFA-I1	2.16±.020†	.743±.005†	.093±.003†
KMEAN-I1	**3.10±.030†**	.955±.005†	**.071±.006†**	KMEAN-I1	**3.34±.015†**	**.966±.002†**	**.157±.004†**
HIERA-I1	3.04±.028†	**.973±.004†**	.063±.006†	HIERA-I1	3.14±.017†	.929±.003†	.137±.004†
STDTOUR	1.33±.002	.644±.001	.007±.000	STDTOUR	1.90±.003	.800±.001	.022±.000
RAND-I.5	1.28±.024	.745±.011†	.007±.001	RAND-I.5	2.12±.021†	.782±.006‡	.069±.003†
FCFA-I.5	1.29±.025	.736±.011†	.008±.001	FCFA-I.5	2.14±.021†	.787±.006	.072±.003†
KMEAN-I.5	2.31±.027†	**.978±.004†**	.012±.002‡	KMEAN-I.5	**3.34±.019†**	**.955±.003†**	**.181±.005†**
HIERA-I.5	**2.34±.028†**	.943±.006†	**.014±.002†**	HIERA-I.5	3.26±.020†	.948±.003†	.159±.005†

Osaka				**Budapest**			
Algo.	Tot_I	Max_I	Min_I	*Algo.*	Tot_I	Max_I	Min_I
STDTOUR	1.09±.006	.630±.003	.032±.001	STDTOUR	2.31±.003	.792±.001	.005±.000
RAND-I1	1.25±.025†	.668±.011‡	.155±.008†	RAND-I1	3.06±.046†	.781±.008	.051±.003†
FCFA-I1	1.25±.024†	.671±.011‡	.151±.008†	FCFA-I1	3.04±.044†	.772±.008	.048±.003†
KMEAN-I1	**1.79±.020†**	**.918±.006†**	**.204±.009†**	KMEAN-I1	**5.66±.034†**	**.980±.003†**	**.092±.005†**
HIERA-I1	1.77±.020†	.902±.007†	.198±.009†	HIERA-I1	5.33±.040†	.944±.004†	.078±.005†
STDTOUR	1.09±.006	.630±.003	.032±.001	STDTOUR	2.31±.003	.792±.001	.005±.000
RAND-I.5	1.25±.027†	.656±.012	.143±.009†	RAND-I.5	2.98±.042†	.769±.008‡	.049±.003†
FCFA-I.5	1.25±.029†	.653±.013	.152±.009†	FCFA-I.5	2.95±.043†	.757±.008†	.051±.003†
KMEAN-I.5	1.97±.020†	**.952±.006†**	.232±.011†	KMEAN-I.5	**4.62±.033†**	**.973±.003†**	.068±.004†
HIERA-I.5	**2.00±.020†**	.944±.006†	**.239±.011†**	HIERA-I.5	4.55±.037†	.934±.005†	**.072±.004†**

Tables 7.5 and 7.6 show the comparison of the {KMEAN, HIERA, FCFA, RAND}-{I.5, I1} algorithms against the STDTOUR baseline, in terms of $Tot_I(t)$, $Max_I(t)$ and $Min_I(t)$. The results shows that the {KMEAN, HIERA}-{I.5, I1} algorithms out-perform the STD-

Table 7.6: POI2Group comparison of the {KMEAN, HIERA, FCFA, RAND}-{I.5, I1} algorithms against the STDTOUR baseline for Glasgow, Delhi and Edinburgh, in terms of Tour Total Interest (Tot_I), Tour Maximum Interest (Max_I) and Tour Minimum Interest (Min_I). † denotes a significant difference with the STDTOUR baseline, based on two-sided t-tests with p<.0001, ‡ denotes the same with p<.01.

Glasgow				Delhi			
Algo.	Tot_I	Max_I	Min_I	*Algo.*	Tot_I	Max_I	Min_I
STDTOUR	0.49±.002	.460±.002	.002±.000	STDTOUR	.603±.004	.555±.003	.048±.001
RAND-I1	0.71±.014†	.554±.010†	.013±.002†	RAND-I1	.807±.020†	.570±.013	.238±.011†
FCFA-I1	0.72±.014†	.555±.010†	.018±.002†	FCFA-I1	.813±.020†	.579±.013	.234±.011†
KMEAN-I1	**1.30±.015†**	**.884±.007†**	**.059±.004†**	KMEAN-I1	**.959±.017†**	**.711±.011†**	**.248±.010†**
HIERA-I1	1.27±.015†	.855±.007†	.056±.004†	HIERA-I1	.945±.017†	.700±.011†	.244±.010†
STDTOUR	0.49±.002	.460±.002	.002±.000	STDTOUR	.603±.004	.555±.003	.048±.001
RAND-I.5	0.66±.016†	.520±.011†	.017±.002†	RAND-I.5	.705±.017†	.548±.012	.157±.008†
FCFA-I.5	0.66±.016†	.510±.011†	.018±.002†	FCFA-I.5	.699±.017†	.540±.012	.159±.008†
KMEAN-I.5	**1.30±.019†**	**.818±.009†**	**.063±.005†**	KMEAN-I.5	**.920±.017†**	**.689±.011†**	.231±.009†
HIERA-I.5	1.25±.018†	.801±.009†	.049±.004†	HIERA-I.5	.909±.017†	.677±.011†	**.232±.009†**

Edinburgh			
Algo.	Tot_I	Max_I	Min_I
STDTOUR	0.88±.001	.766±.001	.015±.000
RAND-I1	1.01±.007†	.631±.004†	.071±.002†
FCFA-I1	1.02±.007†	.636±.004†	.072±.002†
KMEAN-I1	**1.62±.006†**	**.953±.002†**	**.124±.002†**
HIERA-I1	1.58±.006†	.938±.002†	.115±.002†
STDTOUR	0.88±.001	.766±.001	.015±.000
RAND-I.5	1.00±.006†	.694±.004†	.049±.001†
FCFA-I.5	1.00±.007†	.688±.004†	.052±.002†
KMEAN-I.5	**1.73±.006†**	**.966±.001†**	**.162±.003†**
HIERA-I.5	1.67±.006†	.950±.002†	.143±.003†

TOUR baseline for all cities, with relative improvements of more than 50.7%, 16.1% and 71.4% based on measures of Tot_I, Max_I and Min_I, respectively. Results from t-tests also

Table 7.7: Guide2Group comparison of the OPTIM algorithm against the RANDA baseline, in terms of Guide Total Expertise (Tot_I). † denotes a significant difference with RANDA, based on two-sided t-tests with p<.0001.

Guide Total Expertise (Tot_E)				
Algo.	*Toronto*	*Vienna*	*Osaka*	*Budapest*
RANDA	149.53±1.17	269.36±1.63	50.29±1.08	105.00±0.65
OPTIM	**174.78±1.33**†	**321.22±1.74**†	**60.40±1.23**†	**119.51±0.68**†

Guide Total Expertise (Tot_E)			
Algo.	*Glasgow*	*Delhi*	*Edinburgh*
RANDA	64.72±1.15	46.48±1.11	251.49±1.08
OPTIM	**76.05±1.27**†	**54.45±1.17**†	**291.41±1.17**†

show that these improvements are statistically significant with p<.0001.[6]

In contrast, the {FCFA, RAND}-{I.5, I1} algorithms (our previous baselines) show mixed results against the STDTOUR baseline in terms of Tot_I and Max_I scores, while showing general improvements in terms of Min_I scores. However, some of the improvements by the {FCFA, RAND}-{I.5, I1} algorithms are not statistically significant. On the whole, these results show that our {KMEAN, HIERA}-{I.5, I1} algorithms are the best performers in all cases and that recommending customized tours better satisfy the interest preferences of tourists, compared to recommending standard tour packages offered by tour operators.

7.6.4 Evaluation of Guide2Group Assignment

We next evaluate the effectiveness of the Guide2Group assignment using the OPTIM algorithm compared to the RANDA baseline. Based on the Tot_E scores, Table 7.7 shows that OPTIM out-performs RANDA with relative improvements of 13.8% to 20.1%. Similarly, the t-test results show that these improvements are statistically significant with p<.0001. The experiments on the Glasgow, Delhi and Edinburgh datasets show similar results. On

[6]Except for Toronto in terms of Min_I scores, which show statistically significant improvements at p<.01.

the whole, the results for the Tourist2Group, POI2Group and Guide2Group evaluations show that our proposed approaches significantly out-perform the various baselines, for all seven cities.

7.7 Summary

We introduced and formulated the novel GROUPTOURREC problem, which involves recommending tours to groups of tourists and assigning tour guides to lead these tours. Our approach to solve this NP-hard GROUPTOURREC problem involves decomposing it into more manageable sub-problems of Tourist2Group allocation, POI2Group recommendation and Guide2Group assignment. For Tourist2Group allocations, we modeled the interests of tourists based on their past POI visits (extracted from geo-tagged photos) and proposed the use of k-means and hierarchical clustering to allocate these tourists into tour groups. For POI2Group recommendations, we recommended tours to these tour groups based on their collective group interest and using a variant of the Orienteering problem, which also considers various trip constraints. For Guide2Group assignments, we proposed a model of tour guide expertise which is then matched to the recommended tour for each tour group as an Integer Programming problem. Lastly, experimental results on a Flickr dataset of seven cities show that our proposed approaches significantly out-perform the various baselines, based on measures of Jaccard/Cosine/top interest similarity, total/maximum/minimum tour interests and total tour guide expertise.

Chapter 8

Tour Recommendation System Demonstration

In the preceding chapters, we have focused on the problem formulation and algorithmic aspects of various tour recommendation problems. In this chapter, we present a system demonstration of a tour recommendation system, partially based on our PERSTOUR *algorithm that was previously described in Chapter 4. We term this system the Personalized Tour Recommendation and Planning (PersTourRP) system. The main motivation behind our PersTourRP system is to alleviate the time and effort that tourists have to expend to identify interesting attractions or POIs and structure these POIs in the form of a time-constrained tour itinerary. Our PersTourRP system is able to plan for a customized tour itinerary where the recommended POIs and visit durations are personalized based on the tourist's interest preferences. In addition, tourists have the option to indicate their trip constraints (e.g., a preferred starting/ending location and a specific tour duration) to further customize their tour itinerary.*

8.1 Introduction

For a tourist visiting an unfamiliar city, there are numerous challenges such as: (i) identifying attractions or POIs that appeal to his/her interest preferences, rather than simply visiting popular POIs; (ii) structuring these POIs as a tour itinerary that considers the tourist's preferences for starting/ending locations and time constraints for touring; and (iii) providing detailed directions on how to get from one POI to another, including rec-

This chapter is derived from the following publication:

- **Kwan Hui Lim**, Xiaoting Wang, Jeffrey Chan, Shanika Karunasekera, Christopher Leckie, Yehui Chen, Cheong Loong Tan, Fu Quan Gao and Teh Ken Wee. PersTour: A Personalized Tour Recommendation and Planning System. *Extended Proceedings of the 27th ACM Conference on Hypertext and Social Media (HT'16), Demonstration Track*. Jul 2016.

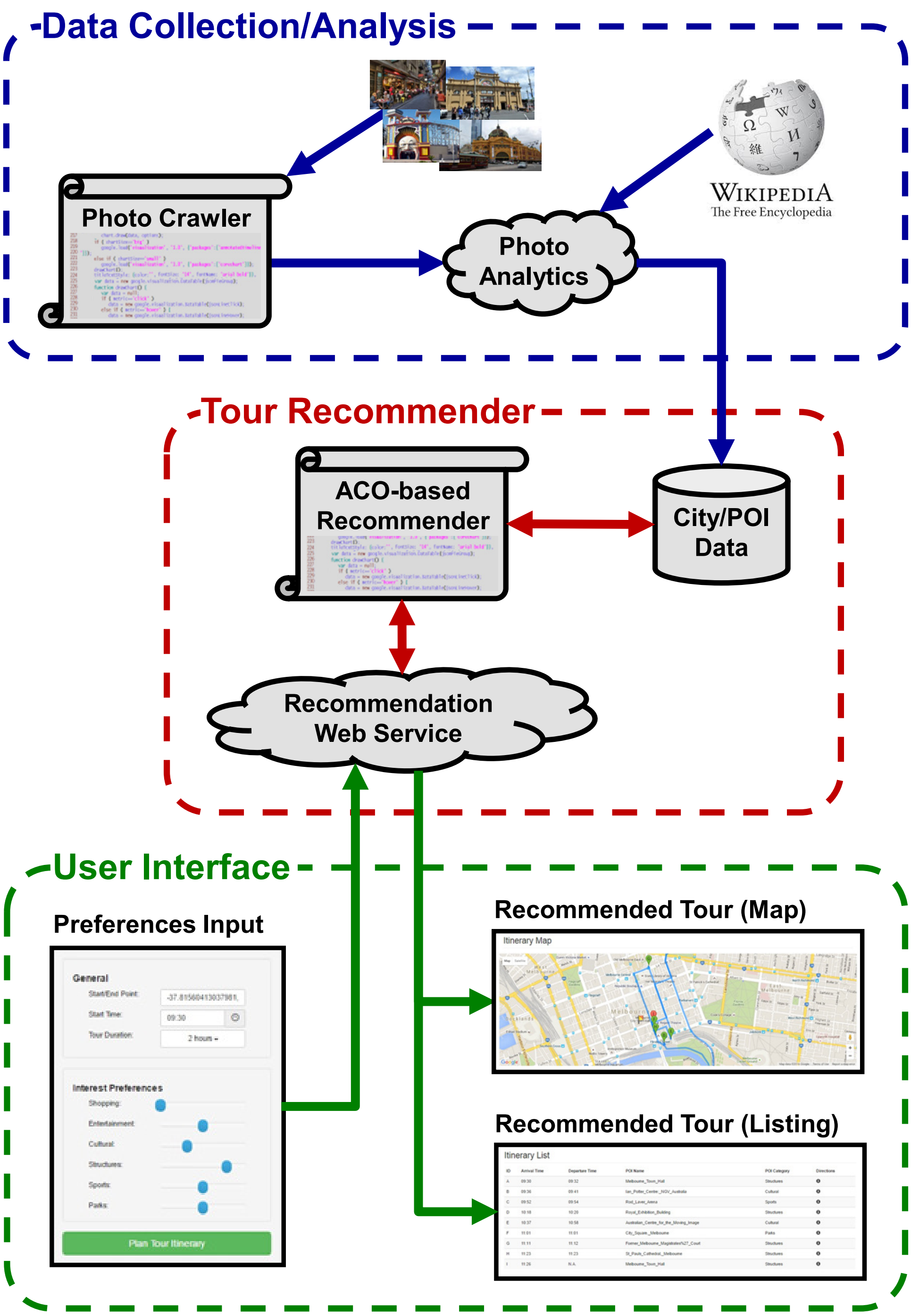


Figure 8.1: PersTourRP System Architecture.

ommendations for POI visit durations based on the tourist interest preferences.

To alleviate these challenges faced by tourists, we propose the Personalized Tour Rec-

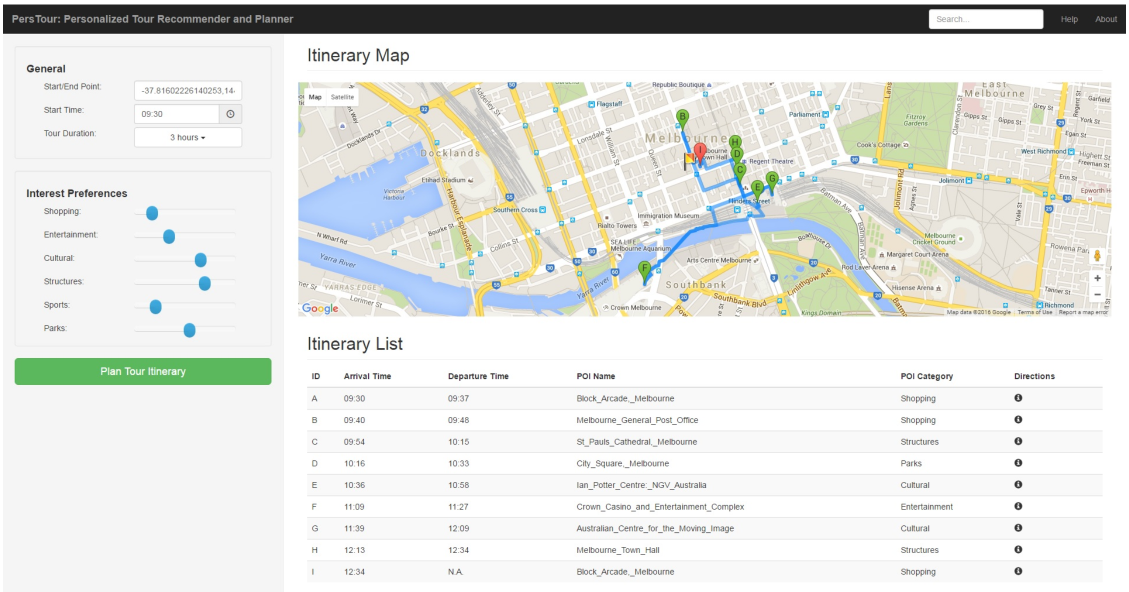


Figure 8.2: User Interface of the PersTourRP System.

ommendation and Planning (PersTourRP) System. While there exist various interesting tour planning applications [23, 31, 117, 122, 137, 139, 149], our PersTourRP system differs from them in one or more of the following ways:

- Tourists are able to select any starting/ending location (instead of a specific POI, which the tourist may be unfamiliar with) and PersTourRP will recommend an itinerary that starts/ends at a POI near that selected location.

- In addition to a personalized itinerary recommendation (comprising POIs of interest to the tourist), PersTourRP also personalizes the recommended visit duration at each POI based on the tourist's interest preferences.

- PersTourRP uses publicly available geo-tagged photos and Wikipedia to determine POI-related statistics and information.

8.1.1 Contributions

Our main contribution is in developing the PersTourRP system (Figure 8.1) that is able to recommend POIs that are interesting to the tourist and plan these POIs in the form of a tour itinerary. The key features of this system are as follows:

- Able to consider tourist trip constraints such as starting and ending at specific locations (e.g., near the tourist's hotel) and having limited time for touring.

- Utilizes geo-tagged photos and Wikipedia to: (i) determine the popularity of POIs; (ii) derive the average time tourists spend at each POIs; and (iii) classify POIs into distinct categories.

- Able to recommend tours based on either POI popularity or tourist interest preferences. In addition, recommended POI visit durations are tailored based on the interest levels of the tourist, i.e., a longer visit duration for POIs that are interesting to the tourist.

- Adapted the Ant Colony Optimization algorithm for the purpose of tour recommendation, with considerations for trip constraints and interest preferences.

- Recommendation results are displayed in an intuitive graphical and textual form (Figure 8.2). The graphical form allows for a quick overview of the tour itinerary on a map, while the textual form provides detailed information about getting from one POI to another.

8.1.2 Structure and Organization

The rest of this chapter is structured as follows. Section 8.2 illustrates the technical architecture of our PersTourRP system in terms of its three main components, while Section 8.3 describes two use case scenarios for a popularity-based and interest-based tour recommendation. Finally, Section 8.4 summarizes this chapter.

8.2 System Architecture

Our PersTourRP system was developed as a web-based application with a responsive interface that allows for viewing on desktops, tablets or mobile phones. The front-end component was developed using HTML, PHP, jQuery and the Google Maps API [64], while the back-end was developed using Python, Java and PHP. Our PersTourRP system comprises three main components, namely:

- **Data Collection and Analysis Component**. This back-end component is mainly responsible for the retrieval of geo-tagged photos and analyzing these photos to infer POI popularity, average POI visit durations and POI categories.
- **Tour Recommendation Component**. This back-end component uses the processed POI data (from the Data Collection and Analysis component) for recommending and planning personalized tour itineraries that are then passed to the User Interface component.
- **User Interface Component**. This front-end component solicits the trip constraints and interest preferences from the tourist, then communicates with the Tour Recommendation component to obtain a personalized tour itinerary, which is then displayed to the tourist.

In the following sections, we will describe each component in greater detail.

8.2.1 Data Collection and Analysis Component

The Data Collection and Analysis component performs two main tasks, which are: (i) the crawling of geo-tagged photos from the Flickr photo sharing website; and (ii) the analysis of these photos to infer the popularity of POIs, average POI visit duration and the interest categories associated with each POI.

Data collection. For the first task, we are interested in all photos taken within a specific city of interest, particularly the associated meta-data such as the latitude/longitude coordinates, photo time taken and photo owner/taker.[1] The usefulness of geo-tagged photos for tour recommendation purposes has also been demonstrated in many recent research works [24, 128]. A future enhancement would involve the use of computer vision techniques to analyze the photos themselves to determine the number of humans

[1] While we use Flickr geo-tagged photos for the purpose of this system demonstration, our PersTourRP system can be easily generalized to other photo sharing sites (e.g., Instagram) or any social media that is tagged with geo-location information (e.g., geo-tagged tweets).

in each photos (i.e., travelling alone, in pairs or larger groups) and demographic details (e.g., age group, gender, etc).

Data analysis. For the second task, we analyze the meta-information of each photo to determine the popularity of each POI based on the number of photos taken at each POI, i.e., a proxy for real-life POI visits as the user has to visit the POI to take a photo.[2] We are also able to determine the amount of time spent visiting each POI based on the time difference between the first and last photo taken at a POI. Lastly, we utilize Wikipedia to derive the category (e.g., Shopping, Entertainment, Cultural, Structures, Sports and Parks) that each POI belongs to, based on the Wikipedia article describing the POIs in each city.

These two tasks (data collection and data analysis) can then be conducted for each city of interest. Upon completion, the results of the analysis are provided to the Tour Recommendation component, which utilizes the computed POI popularity, POI categories, distance between POIs, and average POI visit duration for recommending and planning tour itineraries. We next discuss the details of the Tour Recommendation component.

8.2.2 Tour Recommendation Component

Using the POI-related information provided by the Data Collection and Analysis component, the Tour Recommendation component recommends and plans a tour itinerary according to the interest preferences and trip constraints of the tourist. The interest preferences corresponds to the POI categories in the city, while trip constraints are in terms of the tourist's preferred starting/ending location and available touring time.

The back-end tour recommendation algorithm is based on a modified version of the Ant Colony Optimization algorithm [48]. We first discuss the basic Ant Colony Optimization algorithm before describing our proposed modifications to adapt it for our purpose of personalized tour recommendation and planning. The basic Ant Colony Optimization algorithm utilizes a number of agents (ants) that start from a specific POI with the aim to finding the best path to a desired destination. This algorithm works in the following

[2]We only use publicly available data and do not release any personal information in our subsequent recommendations.

main steps:

1. At the start of the algorithm, all agents initially select the next POI to visit (based on the utility of visiting that POI), until they reach the destination.

2. At the end of Step 1, the best path taken among all agents is selected and remembered for a period of time, before being gradually forgotten.

3. Steps 1 and 2 are then repeated for a fixed number of iterations. The main difference is that the selection of the next POI to visit (i.e., Step 1) will be biased towards paths that have been taken recently.

The intuition behind the Ant Colony Optimization algorithm is that agents are more likely to follow a path that is "better" and has been taken recently. This preference subsequently leads to the positive reinforcement of choosing a single path over time, resulting in that path being selected as the best solution.

Our modifications to the Ant Colony Optimization algorithm are largely based on our earlier work, the PERSTOUR algorithm from Chapter 4, and include the following modifications:

1. The utility of each POI is based on a combined POI popularity score and tourist interest alignment.

2. The cost of travelling from one POI to another is based on a fixed travelling cost and dynamic POI visit duration, which is personalized based on tourist interest levels.[3]

In most cases, this algorithm takes less than 0.5 seconds to recommend and plan a personalized tour.

8.2.3 User Interface Component

The User Interface component serves three main responsibilities, namely: (i) obtaining user inputs in the form of the tourist's trip constraints (starting/ending location

[3]As we currently focus on city tours, we compute travelling costs based on the transport mode of walking but this can be extended to other transport modes such as cycling and cars by changing the appropriate travelling speeds.

and available touring time) and their interest preferences; (ii) communicating with the Tour Recommendation component by providing the tourist's trip constraints and interest preferences, and retrieving the recommended tour itinerary; (iii) displaying the recommended tour itinerary in an easy to understand visual and textual format.

Obtaining user input. For the first task, a tourist can pick a preferred starting and ending location by simply clicking on any point on the map. Similarly, the tourist can enter a desired tour start time and select a preferred tour duration. For a more personalized tour, the tourist is also able to indicate their interest preferences via slider bars that represent their interest level in the six POI categories (Shopping, Entertainment, Cultural, Structures, Sports and Parks). The slider bars allow tourists to state their interest level at varying levels, ranging from "not interested" to "very interested", which is represented by values of 1 and 100, respectively. By default, all interest levels are set to a neutral "neither interested nor uninterested", i.e., a value of 50.

Communication between components. The second task commences when the tourist clicks on the "Plan Tour Itinerary" button. Upon clicking, the User Interface component makes a web service call to the Tour Recommendation component, along with the various trip constraints and interest preferences provided. In turn, the Tour Recommendation component invokes its recommendation algorithm to plan a personalized tour based on the provided parameters. This personalized tour is then returned to the User Interface component in the form of a JSON response, containing the recommended POIs and the time to spend at each POI.

Displaying recommendation results. For the third task, the User Interface component parses the returned JSON response for display in a visual and textual format. Utilizing the Google Maps API, the visual representation is in the form of waypoints (POIs) that are plotted on a map and connected lines that indicate the route to take between POIs. The textual representation provides more information on the recommended tour, indicating the time to arrive at and depart from each POI, along with the name and category of each POI. In addition, the tourist is also able to click on the "information" icon to the right of each POI for more detailed step-by-step directions, i.e., which road to take, how far to travel and which road junctions to turn at, as shown in Figure 8.3.

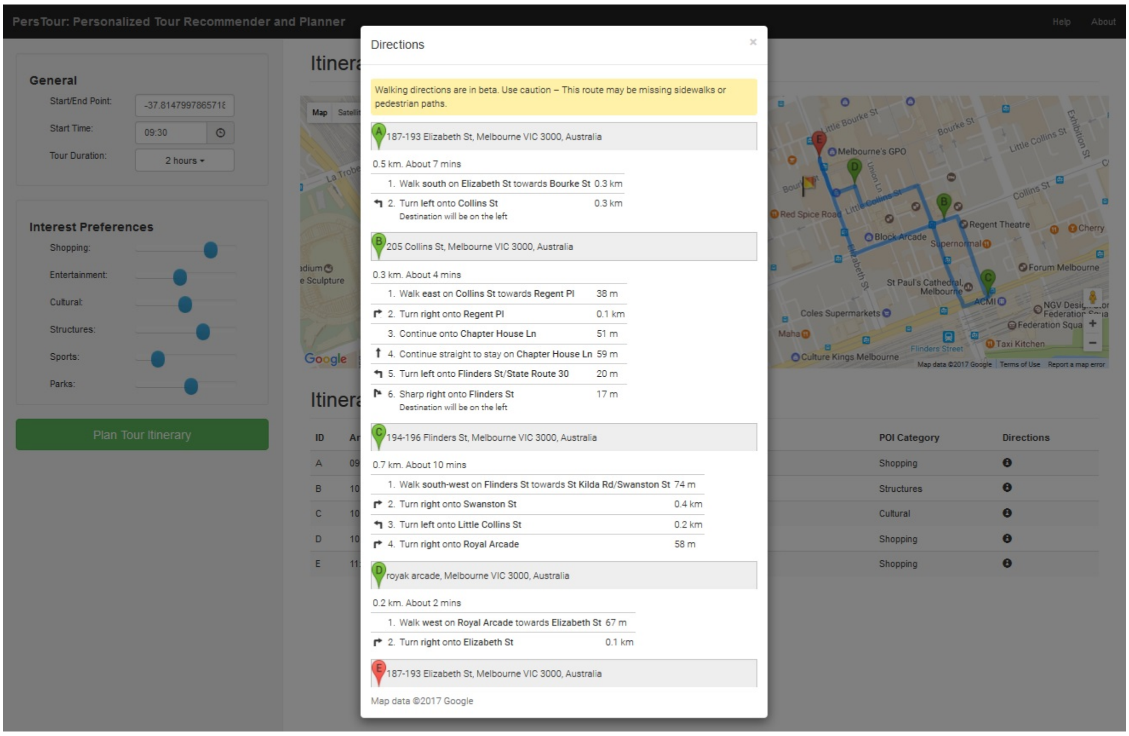


Figure 8.3: Walking directions displayed in the PersTourRP System.

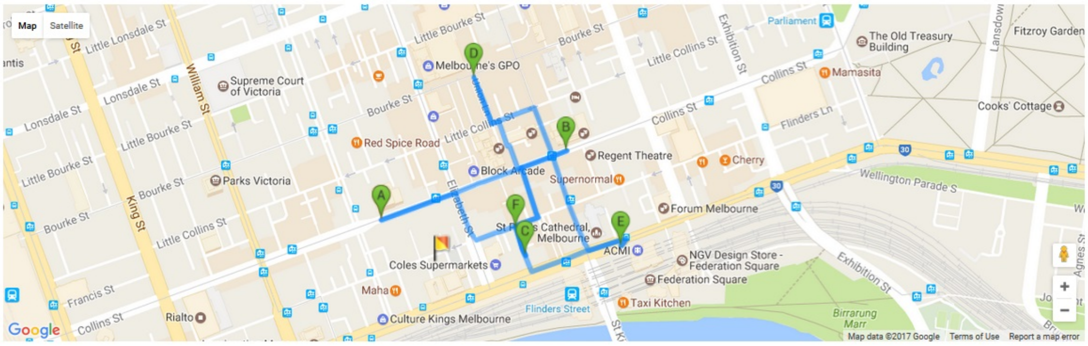
8.3 Use Case Scenarios

As part of our system demonstration, we highlight two scenarios where a tourist might use PersTourRP to obtain a popularity-based and interest-based tour recommendations.

8.3.1 Popularity-based Tours

Consider a tourist Alice who is staying at The Sebel Melbourne Flinders Lane and is planning for a tour that starts near her hotel. Using our PersTourRP system, she can simply click on the location of her hotel (or anywhere on the map) as her desired starting/ending point. Furthermore, Alice selects a starting time of 10am, a tour duration of 3 hours and then clicks on the “Plan Tour Itinerary” button to get a customized tour itinerary recommendation. Based on the selected starting/ending location, tour start time and preferred tour duration, PersTourRP recommends a set of popular POIs to visit within Alice’s preferred tour duration. This recommendation is displayed as a graphical tour itinerary on the map as well as in textual form with detailed information about the POI visit sequence

Itinerary Map



Itinerary List

ID	Arrival Time	Departure Time	POI Name	POI Category	Directions
A	10:00	10:18	Collins_Street,_Melbourne	Shopping	ℹ
B	10:23	10:38	Melbourne_Town_Hall	Structures	ℹ
C	10:41	10:57	Degraves_Street,_Melbourne	Shopping	ℹ
D	11:02	11:19	Bourke_Street,_Melbourne	Shopping	ℹ
E	11:26	11:43	Ian_Potter_Centre:_NGV_Australia	Cultural	ℹ
F	11:47	12:02	Flinders_Lane,_Melbourne	Shopping	ℹ
G	12:05	N.A.	Collins_Street,_Melbourne	Shopping	ℹ

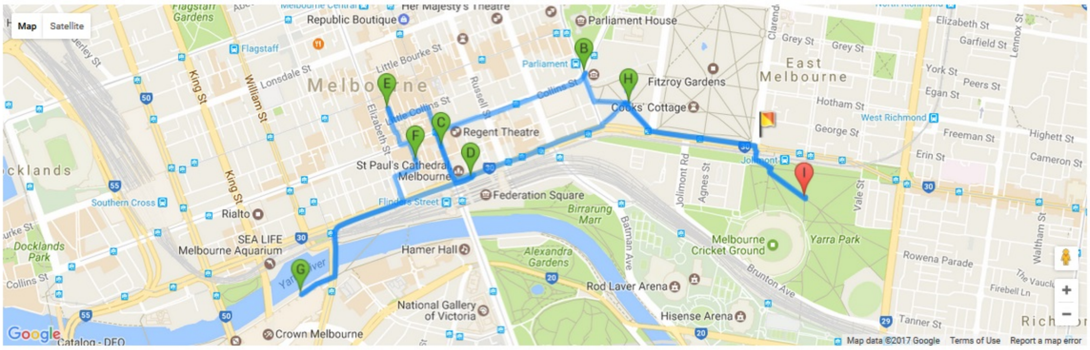
Figure 8.4: Example of the tour itinerary recommended by the PersTourRP System to Alice (no interest preferences).

with the appropriate time to arrive at and depart from each POI. If Alice requires more detailed directions, clicking on the "information" icon beside each POI listing will display a set of detailed instructions for directions. An example of the recommended tour for Alice is shown in Figure 8.4.

8.3.2 Interest-based Tours

Consider another tourist Bob who prefers a more personalized tour based on his specific interest preferences. Similar to what Alice has done, Bob also selects his preferred starting/ending point (near "Pullman Melbourne On The Park"), tour start time (1pm) and preferred tour duration (5 hours). In addition, Bob can indicate his interested preferences via a set of slider bars that correspond to each POI interest categories. For example, Bob is very interested in Sports and Parks, moderately interested in Shopping and Entertainment, and less interested in Structures and Cultural. As such, Bob adjusts the slider bars

Itinerary Map



Itinerary List

ID	Arrival Time	Departure Time	POI Name	POI Category	Directions
A	13:00	13:23	Yarra_Park	Parks	❶
B	13:36	13:48	Old_Treasury_Building,_Melbourne	Structures	❶
C	13:56	14:19	City_Square,_Melbourne	Parks	❶
D	14:22	14:32	Ian_Potter_Centre:_NGV_Australia	Cultural	❶
E	14:39	14:54	Royal_Arcade,_Melbourne	Shopping	❶
F	14:57	15:15	Centre_Place,_Melbourne	Shopping	❶
G	15:25	15:51	Crown_Casino_and_Entertainment_Complex	Entertainment	❶
H	16:11	16:37	Treasury_Gardens	Parks	❶
I	16:59	N.A.	Yarra_Park	Parks	❶

Figure 8.5: Example of the tour itinerary recommended by the PersTourRP System to Bob (high interest in Sports and Parks, average interest in Shopping and Entertainment, and low interest in Structures and Cultural).

for each POI interest category accordingly before clicking on the "Plan Tour Itinerary" button. In this case, PersTourRP takes into account Bob's interest preferences and recommends a personalized tour itinerary comprising POIs that are more likely to include POIs of the Sports and Parks categories and less of the Structures and Cultural categories. Similarly, PersTourRP recommends a longer duration to spend at POIs of the Sports and Parks categories and a shorter duration at POIs of the Structures and Cultural categories, given Bob's interest preferences. An example of the recommended tour for Bob is shown in Figure 8.5.

8.4 Summary

In this chapter, we proposed the PersTourRP system for recommending and planning personalized tour itineraries. This system comprises three main components that perform the following functions, namely: (i) a Data Collection and Analysis component that uses geo-tagged photos and Wikipedia to derive POI-related statistics and information; (ii) a Tour Recommendation component that uses a modified Ant Colony Optimization algorithm to recommend tour itineraries, which adhere to trip constraints and consider interest preferences; and (iii) a User Interface component that uses an intuitive graphical and textual interface to solicit user input and display recommendation results. In addition, the PersTourRP system is able to recommend a suitable starting/ending POI based on a tourist-selected location and personalizes the recommended POI and visit duration based on tourist interest preferences.

Some future work to enhance the PersTourRP system includes the following: (i) incorporate restaurant visits (e.g., breakfast, lunch and dinner) and consider POI visiting costs (e.g., entrance fees) as part of tour recommendation; (ii) cater for tour recommendations to groups of tourists with diverse interest preferences, in the same spirit as that of GROUPTOURREC from Chapter 7; (iii) automatically build a tourist interest profile, possibly by analyzing a tourist's social media posts such as in [7]; and (iv) apart from POI popularity and tourist interest, also consider the beauty, peacefulness and enjoyability of routes taken in a tour [114].

Chapter 9
Conclusion

This chapter concludes the thesis by summarizing our main contributions in the domain of tour itinerary recommendation, in terms of proposing and developing novel research problems, recommendation algorithms and frameworks for experimental evaluation. We also highlight some interesting directions for future work in tour itinerary recommendation and other related research domains.

9.1 Summary of Contributions

In this thesis, we have made various contributions to the field of tour itinerary recommendation, and more generally to the related fields of optimization and data mining. A summary of our contributions are as follows:

- In Chapter 2, we performed a comprehensive survey of literature on tour recommendation research and proposed a taxonomy to more distinctly classify the various sub-categories of research in this area.
- In Chapter 3, we provided an overview of the generic tour recommendation problem and described how this problem can be formulated as a variant of the Orienteering problem. We also introduced various key notations used throughout this thesis, and described our data mining framework that uses geo-tagged photos to derive user interests and trajectories, and POI-related information.
- In Chapter 4, we introduced and formulated a novel personalized tour recommendation problem based on a variant of the Orienteering problem and proposed the PERSTOUR algorithm for solving this problem. Our PERSTOUR algorithm considers both POI popularity and user interest preferences to recommend suitable

POIs to visit and a customized duration of time to spend at each POI. In addition, PERSTOUR is able to automatically determine user interest levels and perform a weighted updating of user interests based on the recency of their POI visits. Our PERSTOUR algorithm improves upon existing tour recommendation algorithms in various ways: (i) we introduced the idea of *time-based user interest*, derived from a user's visit durations at specific POIs relative to other users, in contrast to earlier works that use frequency-based user interest based on POI visit frequency; (ii) we proposed the use of *personalized POI visit duration* based on the relative interest levels of individual users, compared to earlier works that use the average POI visit duration for all users, or do not consider POI visit duration at all; and (iii) we proposed *two adaptive weighting methods* to automatically determine the level of emphasis to place on POI popularity and user interest preferences.

- In Chapter 5, we proposed and formulated the QUEUETOURREC problem of recommending personalized itineraries of popular and interesting attractions, while minimizing queuing times at these attractions. QUEUETOURREC is a complex problem that includes time-dependent queuing times, and we developed the PERSQ algorithm for solving this queue-dependent tour recommendation problem. Our PERSQ algorithm is based on a novel adaptation of Monte Carlo Tree Search for the domain of personalized itinerary recommendation with queuing time awareness. We also implemented a framework that utilizes geo-tagged photos to automatically extract the distribution of queuing times at each attraction, in addition to the popularity of attractions and interest preferences of tourists. In particular, we determine the queuing time at an attraction by subtracting its attraction visit duration (e.g., 10 minutes for a roller coaster) from the total time spent at an attraction (which includes queuing time and visit duration).
- In Chapter 6, we formulated the problem of recommending tour itineraries with a mandatory or must-visit POI visit category that must be included in the recommended tour itinerary. This problem is based on a variant of the Orienteering problem, with considerations of starting/ending POIs and a distance budget for completing the tour itinerary, and an additional constraint for a must-visit POI cat-

egory. To solve this problem, we developed the TOURRECINT algorithm, in the form of an Integer Linear Program, for recommending such tours. In our proposed approach, the mandatory POI category is defined as the POI category that a tourist has most frequently visited based on his/her other travel sequences, and the starting/ending POIs and distance budget are based on the tourist's real-life travel sequences derived from geo-tagged photos. Apart from applying our TOURRECINT algorithm for tour recommendations, TOURRECINT can also be adapted to general route planning problems, such as driving from Melbourne to Sydney with compulsory visits to the POI category of "Petrol Station", or walking from home to a friend's BBQ with mandatory visits to the POI categories of "Supermarket" and "Hardware Store".

- In Chapter 7, we introduced and formulated the novel GROUPTOURREC problem, which involves recommending tours to groups of tourists with different interest profiles, and assigning tour guides with different types of expertise to lead these tours. To solve this GROUPTOURREC problem, we decompose it into a series of more manageable sub-problems, comprising tourist groupings, POI recommendations and tour guide assignments. For each sub-problem, we adopted the following approaches: (i) for tourist grouping, we built a model of interest preferences of tourists based on their past POI visits and used the k-means and hierarchical clustering algorithms to assign these tourists into various tour groups; (ii) for POI recommendations to each tour group, we derived a collective group interest, then recommended interest-based tour itineraries to each tour group based on a variant of the Orienteering problem, which also considers various trip constraints; and (iii) for tour guide assignments, we constructed a model of tour guide expertise based on their experience in leading tours to specific POIs, then matched these types of tour guide expertise to each recommended tour as an Integer Programming problem. As far as we are aware, our work is the first to investigate group tour recommendations as a holistic problem involving the entities of tourists, tour groups and tour guides.

- In Chapter 8, we presented the Personalized Tour Recommendation and Planning system (denoted the PERSTOURRP system), which is an online-based implementation of our PERSTOUR algorithm. The PERSTOURRP system comprises three main components, as follows: (i) the Data Collection and Analysis Component, which collects geo-tagged photos and analyzes these photos to infer POI related statistics; (ii) the Tour Recommendation Component, which is the back-end algorithm for recommending and planning personalized tour itineraries; and (iii) the User Interface Component, which solicits user input from the tourist and displays the recommended tour itinerary. Our PERSTOURRP system was designed to provide a responsive and easy-to-use interface on desktops, tablets or mobile phones for tourists to indicate their interest preferences and various trip constraints (start time, available touring duration, starting/ending locations) and to present the recommended tour itinerary in an intuitive graphical form (map overview) and textual form (detailed travelling directions).

9.2 Future Directions in Tour Itinerary Recommendations

Although we covered numerous aspects of tour itinerary recommendations, there are still various interesting directions for future work. In this section, we describe some of these possible directions for future work, as follows:

- Future tour itinerary recommendation research can consider multiple modes of transport (e.g., walking, bus, train, taxi, car), instead of a single type of transport. The main motivation of this future work would be to offer users the flexibility to switch between different modes of transport, while excluding certain types (e.g., either bus, train or taxi but no walking). Such constraints for transport modes would be useful for making tour recommendations to different demographics of tourists, such as students (prefer buses or trains), working professions (all transport modes), families with babies or the elderly (all transport modes except walking).
- Furthermore, when considering public transport (e.g., bus, train, tram), we can recommend tour itineraries that consider the arrival and departure times of public transport to minimize the waiting time incurred by the tourists for their respec-

tive public transport services to arrive. Another consideration is to incorporate the switching of different modes of transport (e.g., bus to train) and to minimize the waiting time when switching between different transport modes. Furthermore, we can also model uncertainty in the arrival times and travelling time of different transport modes, especially when there are connections between multiple transport modes.

- The modelling of uncertainty can also be extended to the predicted visit durations and possible queuing times at POIs. The main consideration for this future work is to incorporate some uncertainty in the amount of time recommended at various POIs due to delays caused by crowds (e.g., POIs are more crowded during weekends than weekdays, thus causing possible delays). Thus the uncertainty in visiting durations and queuing times can also be varied based on the day of the week and time of the day.

- As travel plans are subject to changes due to various circumstances (e.g., bad weather, human fatigue, POI/road closure, traffic congestion), another possibility for future work is to develop dynamic tour recommendation algorithms that consider these changing contexts during the course of a pre-planned tour. For example, a tourist may have visited the first three POIs in a recommended tour itinerary with eight POIs before the weather turned rainy, in which case such an algorithm should then recommend a new tour itinerary considering the POIs visited, time spent so far and the type of environmental change, i.e., no outdoor POIs due to rainy weather.

- Another possible future enhancement to tour recommendation systems is to incorporate a mechanism for explicit tourist feedback at the completion of a tour itinerary. This feedback can then be used to improve the tour recommendation system in various ways, namely: (i) as an explicit evaluation metric to measure how well a recommended tour itinerary satisfies a tourist; (ii) to build an interest profile for returning tourists, who use the system multiple times, based on their feedback on specific POIs in an itinerary; and (iii) to improve future tour itinerary recommendations by avoiding POIs with negative feedback and including POIs with favourable feedback.

- Traditionally, tour recommendation algorithms aim to propose tours that maximize the personal profit of individual tourists. One limitation of this approach is that while the individual tourist benefits, the entire tourist population could potentially "lose" (e.g., everyone going to the most popular POI, but ends up overcrowding and creating long queues at that POI, leading to a poor tour experience for most people). To address this problem, future work can adopt a game theoretic approach to tour recommendations where we model POI "crowdedness" as a common utility and derive equilibrium strategies to recommend tours that will benefit all tourists as a whole. Potential applications of this work would be in optimizing for queuing times at attractions/rides in theme parks and preventing over-crowding at exhibits within museums. For example, instead of recommending the most popular attraction in a theme park to all visitors and increasing the queuing times, we may want to recommend some less popular attractions that have shorter queuing times to a subset of visitors.

- In addition to the considerations of user interest preferences, POI popularity and queuing times derived from geo-tagged photos, future work can consider user sentiments towards various types of POIs and a more fine-grained categorization of POIs. For example, image recognition techniques could be applied to photos or sentiment analysis techniques on tweets to determine how tourists feel regarding various places. Furthermore, natural language processing techniques such as topic modelling can be used on the user tags on photos or textual content of tweets to determine the different activities associated with different POIs.

- In terms of experimental evaluation, a possible direction is to adopt a more qualitative form of evaluation, compared to the quantitative evaluations that our work focuses on. Such qualitative evaluations will involve human participants that explicitly or implicitly evaluate the recommended tour itineraries, for example by: (i) using Amazon Mechanical Turk for a survey of user opinions on recommended tour itineraries in relation to a baseline recommendation, such as in [40, 114]; and

(ii) using online controlled experiments to better understand user behaviour and their fine-grained actions when deciding between recommended tour itineraries and other baselines, such as in [78, 100].

- For the wide-spread adoption of automated tour recommendation systems, there is also a need to address the cold-start problem of determining the interest preferences of new users who have not previously used the system. Future works can solve this problem by: (i) inferring the interest preferences of users based on other sources of information, such as their demographic details, the interest preferences of their friends, their user accounts and activities on other social networking sites; and (ii) developing an intuitive interface for soliciting user interest preferences without imposing excessive cognitive effort on the user, e.g., asking users to indicate their preferences based on photos instead of requiring them to fill up a long questionnaire.

9.3 Future Directions in Other Related Domains

Apart from the future directions in tour itinerary recommendation, our proposed algorithms and findings in this thesis can also be applied to other related domains, such as for the purposes of detecting location-centric communities and predicting the next visit location. We discuss some future directions in these related domains and substantiate this future work with some preliminary results.

9.3.1 Location-centric Community Detection

We studied the GROUPTOURREC problem (in Chapter 7), which involves recommending tours to groups of tourists, in contrast to the more frequently studied problem of tour recommendation for individual travellers. Extending this problem to LBSNs, one future research direction is to identify groups or communities of users who tend to visit the same set of places, i.e., detecting location-centric groups or communities.

As a preliminary study, we proposed an approach to find such location-centric communities by augmenting social ties (i.e., friendships) with temporal and spatial information. We evaluated this approach on two Foursquare datasets and were able to detect communities of users that display strong similarities in terms of the places they visit and reside in. Previously, we have also found that communities of users with the same interests tend to share strong similarities in their residential city [97, 99]. Similarly, other researchers have found that city-level social networks comprise a large number of user triads that visit common places [26]. For more information, please refer to Appendix A.

9.3.2 Next Visit Location Prediction

Location prediction and POI recommendation are closely related to the tour itinerary recommendation problem we studied in this thesis. With the prevalence and popularity of LBSNs, location prediction and POI recommendation are especially important as knowing the next place a user intends to visit will allow businesses to optimize their marketing strategy by displaying products and services relevant to these predicted locations. Previously in Chapter 4, we introduced a weighted updating of user interests based on the recency of their POI visits, which can also be adapted to the task of next visit/check-in location prediction.

Towards this effort, we performed some preliminary experiments in improving the prediction accuracy of a user's next check-in location on LBSNs by considering the recency of their location visits. Our preliminary results show that LBSN users exhibit a *recency preference* where they are more likely to re-visit recently visited places than those visited in the distant past. We also introduced *place-links*, which are links where the two users share a friendship and a common daily check-in. Our work improved the Social Historical Model (SHM) [55], which uses a language model [130] to predict a user's next check-in location based on this user's previous check-ins and that of his/her friends (i.e., social links). Using two Foursquare LBSN datasets, we showed how the incorporation of both *recency preference* and *place-links* can improve the performance of next check-in prediction over the original SHM and various baseline location prediction algorithms. For more information, please refer to Appendix B.

9.4 Closing Remarks

Tourism is an important industry that accounts for 9% of the world's jobs and close to 10% of world's Gross Domestic Product. The main consumers of this industry are the billions of international and domestic tourists, whose main purpose is to select captivating POIs to visit in various touristic cities and then schedule these POI visits as a connected tour itinerary subject to their constraints of starting/ending near a specific location (e.g., the tourist's hotel), completing the itinerary within a limited touring time and satisfying their unique interest preferences. The prevalence and growth of tourism makes the task of tour itinerary recommendation an important real-life problem, while the various considerations and constraints in scheduling a tour itinerary make it a complex optimization problem. Furthermore, there is an increased demand for personalizing such recommended tour itineraries, which involves the use of data mining techniques to accurately model the unique interest preferences of these tourists.

This thesis has focused on tour itinerary recommendations and we have made numerous contributions to this field by: (i) formulating novel tour recommendation problems with real-life considerations (e.g., user interests, POI popularity, groups of tourists, queuing times), based on variations of classical optimization problems; (ii) developing various algorithms for solving these tour itinerary recommendation problems; (iii) implementing various data mining techniques and frameworks to model the real-life trajectories of tourists and determine their interest preferences based on geo-tagged photos; and (iv) proposing various evaluation frameworks and metrics for measuring the effectiveness of recommended tours based on measures of interest alignment, POI popularity, group similarity, queuing-related metrics and relevance to real-life travel sequences.

Appendix A

Location-centric Community Detection

A.1 Introduction

In Chapter 7, we examined the GROUPTOURREC problem of recommending tours to groups of tourists, in contrast to tour recommendation for individual travellers, which is more frequently studied by researchers. The distinction of recommending different tours for individuals or groups is important as a tourist might want to visit certain POIs if he/she was travelling alone, but may visit a different set of POIs if he/she is travelling with his/her family or friends (a group). Building upon this problem, future work can also aim to identify groups or communities of users who tend to visit the same set of places.

With the rising popularity of LBSNs, it is now possible to add a spatial aspect to these traditional social links for the purpose of detecting such communities. Many researchers have used such social-spatial links to detect location-centric communities on LBSNs [25, 26]. In addition to applications in group-based tour recommendations, the detection of these location-centric communities is especially important for companies embarking on location-based and mobile advertising, which are increasingly crucial to any company's marketing efforts [47]. Location-centric communities allow companies

This appendix is derived from the following publication:

- **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Detecting Location-centric Communities using Social-Spatial Links with Temporal Constraints. *Proceedings of the 37th European Conference on Information Retrieval (ECIR'15)*. pp 489-494. Mar 2015.

Table A.1: Types of Links

Link Type	Description
Social (SOC)	Links based on explicitly declared *friendships* (i.e., topological links)
Social-Spatial-Temporal (SST)	*Social links* where two users share a *common check-in*, on the *same day*
Social-Spatial (SS)	*Social links* where two users share a *common check-in*, regardless of time
Spatial-Temporal (ST)	Links based on two users sharing a *common check-in*, on the *same day*

to better understand their customers and categorize customers based on their locality similarity. Furthermore, location-centric communities offer the opportunity to improve prediction models by utilizing similar features among members of these communities. We posit that the detection of such location-centric communities can be further improved by adding a temporal constraint to such social-spatial links, and demonstrate the effectiveness of this approach using two LBSN datasets. In addition, we study the effects of social, spatial and temporal links on the resulting communities, in terms of various location-based measures.

A.1.1 Methodology

Our proposed approach to detecting location-centric communities involves first building a social network graph $G = (N, E_t)$, where N refers to the set of users and E_t refers to the set of links of type t (as defined in Table A.1). SOC links are essentially topological links that are used in traditional community detection tasks, while SS links were used in [25] to detect location-focused communities with great success. Our work extends [25] by adding a temporal constraint to these links, resulting in our SST links.[1] Furthermore, we also use ST links to determine the effects of adding this temporal constraint solely to the spatial aspects of links (i.e., without considering social information). While there are many definitions of links, these four types of links allow us to best investigate the effects of social, spatial and temporal information on location-centric communities.

Then, we apply a standard community detection algorithm on graph G, resulting

[1] While SST links can also be defined as two friends who share a common check-in within D days, our experiments show that a value of D=1 offers the best results, hence the current definition of SST links. More importantly, using higher values of D days converges SST links towards SS links, which we also investigate in this work.

in a set of communities. Thus, the different types of links (SOC, SST, SS and ST) used to construct the graph G will result in the different types of communities that we evaluate in this paper. We denote the detected communities as Com_{SOC}, Com_{SST}, Com_{SS} and Com_{ST}, corresponding to the types of links used. In this experiment, Com_{SST} are the communities detected by our proposed approach, while Com_{SOC}, Com_{SS} and Com_{ST} serve as baselines.

For the choice of community detection algorithms, we choose the Louvain [14], Infomap [120] and LabelProp [115] algorithms. Louvain is a greedy approach that aims to iteratively optimize modularity and results in a hierarchical community structure, while Infomap is a compression-based approach that uses random walkers to identify the key structures (i.e., communities) in the network. LabelProp first assigns labels to individual nodes and iteratively re-assigns these labels according to the most frequent label of neighbouring nodes, until reaching a consensus where the propagated labels denote the different communities. In principle, any other community detection algorithms can be utilized but we chose these community detection algorithms for their superior performance [51], and to show that our obtained results are independent of any particular community detection algorithm.

A.1.2 Experiments and Results

Datasets. Our experiments were conducted on two Foursquare datasets, which are publicly available at [56] and [55]. Foursquare dataset 1 comprises 2.29M check-ins and 47k friendship links among 11k users, while dataset 2 comprises 2.07M check-ins and 115k friendship links among 18k users. Each check-in is tagged with a timestamp and latitude/longitude coordinates, which is associated with a specific location. In addition, dataset 1 provides the hometown locations that are explicitly provided by the users. We split these datasets into training and validation sets, using 70% and 30% of the check-in data respectively. The training set is used to construct the set of SST, SS and ST links, which will subsequently be used for community detection as described in Section A.1.1.

Evaluation Metrics. Using the validation set, we evaluate the check-in activities and locality similarity of users within each Com_{SOC}, Com_{SST}, Com_{SS} and Com_{ST} community. Specifically, we use the following evaluation metrics:

1. **Average check-ins**: The mean number of check-ins to all locations, performed by all users in a community.

2. **Average unique check-ins**: The mean number of check-ins to unique locations, performed by all users in a community.

3. **Average days between check-ins**: The mean number of days between consecutive check-ins, performed by all users in a community.

4. **Normalized all-visited locations**: The number of times when all users of a community visited a unique location, normalized by the community size.

5. **Ratio of co-visited locations**: Defined as $\frac{1}{|C|}\sum_{i \in C} \frac{|L_i \cap L_C|}{|L_C|}$, where L_i is the set of unique locations visited by user i, and L_C is the set of unique locations visited by all users in a community C.

6. **Ratio of common hometown**: The largest proportion of users within a community that share the same hometown location.

Evaluation metrics 1 to 3 measure the level of user check-in activity, while metrics 4 to 6 measure the user locality (check-in and hometown) similarity within each community. Ideally, we want to detect communities with high levels of check-in activity and locality similarity. As Metrics 1 to 3 are self-explanatory, we elaborate on Metrics 4 to 6. Metric 4 (normalized all-visits) determines how location-centric the entire community is based on how often the entire community visits the same locations. We normalize this metric by the number of users in a community to remove the effect of community sizes (i.e., it is more likely for a community of 50 users to visit the same location than for a community of 500 users). Metric 5 (co-visit ratio) measures the similarity of users in a community (in terms of check-in locations) and a value of 1 indicates that all users visit the exact set of locations, while a value closer to 0 indicates otherwise. Similarly, a value of 1 for Metric 6 (hometown ratio) indicates that all users in a community reside in the same location, while a value of 0 indicates otherwise.

Results. We focus on communities with $>$30 users as larger communities are more useful for a company's location-based and mobile advertising efforts. Furthermore, there has

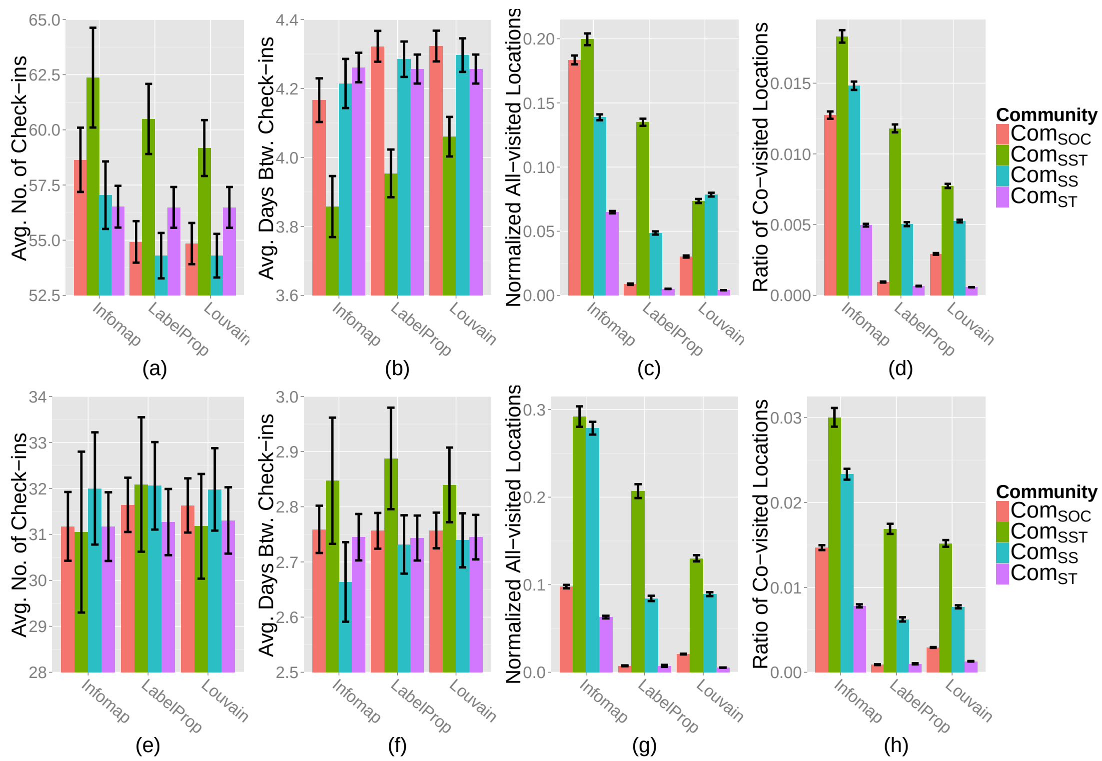


Figure A.1: Average number of check-ins, average days between check-ins, normalized number of all-visited locations and ratio of co-visited locations for Foursquare dataset 1 (top row) and dataset 2 (bottom row). For better readability, the y-axis for Fig. A.1a/b/e/f do not start from zero. Error bars indicate one standard deviation. Best viewed in colour.

been various research that investigated the geographic properties of communities with $\leq$30 users [25, 109]. In particular, [109] found that communities with $>$30 users tend to be more geographically distributed than smaller communities. Instead of repeating these early studies, we investigate the check-in activities and locality similarity of communities with $>$30 users.

Fig. A.1 shows the average number of check-ins, average days between check-ins, normalized number of all-visited locations and ratio of co-visited locations, for the two datasets. The average number of unique check-ins is not shown due to space constraints. The general trend shows that Com_{SST} outperforms all other communities (Com_{SOC}, Com_{SS} and Com_{ST}) in terms of all six evaluation metrics (except for the average number of check-ins and unique check-ins on dataset 2 where all four communities display no clear difference). We shall now discuss the results in greater detail with respect to each evaluation

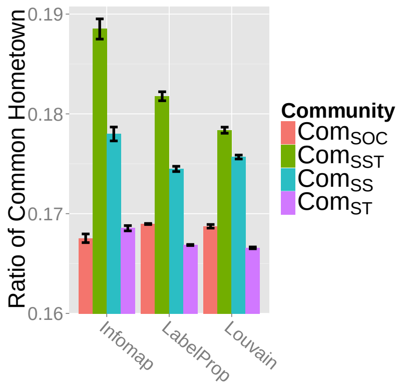


Figure A.2: Common hometown ratio for dataset 1.

metric.

In terms of the average number of check-ins (Fig. A.1a/e), unique check-ins (not shown due to space constraints) and days between check-ins (Fig. A.1b/f), Com_{SST} outperforms Com_{SOC}, Com_{SS} and Com_{ST} on dataset 1, regardless of which community detection algorithm used. However for dataset 2, the performance of Com_{SST} is largely indistinguishable from that of Com_{SOC}, Com_{SS} and Com_{ST}.[2] For both datasets, there is no clear difference among Com_{SOC}, Com_{SS} and Com_{ST} in terms of the average number of check-ins, unique check-ins or days between check-ins. These results show that our proposed SST links can be used to effectively detect communities that are more active in terms of check-in activity (for dataset 1), and such communities serve as a good target audience for a company's location-based and mobile advertising efforts. There is no clear difference among using SOC, SS and ST links (for both datasets). For the detection of location-centric communities, the locality similarity of these communities is a more important consideration, which we investigate next.

We examine locality similarity of the four communities in terms of the normalized number of all-visited locations (Fig. A.1c/g), ratio of co-visited locations (Fig. A.1d/h) and ratio of common hometown (Fig. A.2). We only compare the ratio of common hometown for dataset 1 as this information is not available for dataset 2. For both datasets, Com_{SST} offers the best overall performance in terms of these three locality similarity metrics, while Com_{SS} offers the second best overall performance.[3] On the other hand, Com_{ST}

[2]With an exception in Fig. A.1f where Com_{SST} marginally underperforms Com_{SOC}, Com_{SS} and Com_{ST}.
[3]With exceptions in Fig. A.1c where Com_{SS} (using Louvain) outperforms Com_{SST}, and Com_{SOC} (using

resulted in the worst performance for both datasets. These results show that using our proposed SST links results in communities comprising users who tend to frequently visit similar locations and reside in the same geographic area. Such location-centric communities are useful for the purposes of providing meaningful location-relevant recommendations and to better understand LBSN user behavior.

A.1.3 Discussion

We demonstrate how standard community detection algorithms can be used to detect location-centric communities by augmenting traditional social links with spatial information and a temporal constraint. We evaluate our proposed approach on two Foursquare LBSN datasets and show its effectiveness in detecting location-centric communities that comprise users who were shown to frequently visit the same locations. Furthermore, we demonstrate the general consistency of our proposed approach in detecting such location-centric communities, regardless of the community detection algorithms used. Our evaluations on two Foursquare LBSN datasets show that: (i) augmenting social links with spatial information allows us to detect location-centric communities; (ii) however, using spatial/temporal information (without considering social links) results in communities that are less location-centric than communities based solely on social links, thus spatial/temporal information should not be used independently; and (iii) our proposed approach of augmenting social links with both spatial and temporal information offers the best performance and results in location-centric communities, which display high levels of check-in and locality similarity.

For future work, we can combine our location-centric community detection algorithm with that of [98] to detect communities with common interests and similar spatial attributes, e.g., home locations or frequently visited place, which will be useful for location-based and mobile advertising purposes. Apart from using location-centric communities for location-based and mobile advertising, such communities could also be used to improve location or link prediction models. We can employ different prediction models for each community, which are trained using data (e.g., check-ins) generated by users within

Infomap) outperforms Com_{SS}.

that community. As users in these location-centric communities are more similar, we are essentially training each prediction model using data that is most relevant to a particular group of users.

Appendix B
Next Visit Location Prediction

B.1 Introduction

Location prediction and POI recommendation are research problems that are closely related to the tour itinerary recommendation problem. For example, Chapter 4 introduced a weighted updating of user interests based on the recency of their POI visits, which can also be adapted to the task of next visit/check-in location prediction. LBSN such as Foursquare and Facebook Places have gained immense popularity in recent years, fueled by the prevalence of smart phones with GPS technology, and these LBSNs enable users to check-in to any place they visit and share these check-ins with their friends. Next visit or check-in location prediction is important as knowing the next place a user intends to visit will allow businesses to optimize their marketing strategy by displaying products and services relevant to these predicted locations. Towards this effort, we present some preliminary work and results in improving the prediction accuracy of a user's next check-in location on LBSNs by considering the recency of their location visits. Our work improved the Social Historical Model (SHM) [55], which uses a language model [130] to predict a user's next check-in location based on this user's previous check-ins and that of his/her friends (i.e., social links).

This appendix is derived from the following publication:

- **Kwan Hui Lim**, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Improving Location Prediction using a Social Historical Model with Strict Recency Context. *Proceedings of the 5th Workshop on Context-awareness in Retrieval and Recommendation (CaRR'15), in-conjunction with the 37th European Conference on Information Retrieval (ECIR'15)*. Mar 2015.

B.1.1 Original Social Historical Model

The SHM comprises both the components of a Historical Model (HM) and Social Model (SM). As restated from [55], we first describe the HM and SM components before elaborating on how they constitute the SHM.

The HM predicts a user's next check-in location using their past check-in history. A user's check-in history is modeled as a Hierarchical Pitman-Yor (HPY) process [130], which is a language model that extends the traditional Pitman-Yor process and generates a probability distribution of check-in locations, with a discount parameter to capture the power-law effect. In addition, the HPY process also uses an n-gram model to capture the short-term effect where the latest check-in has more importance than an earlier check-in.[1] The HM is formally defined as:

$$P^i_{HM}(c_{n+1} = l) = P^i_{HPY}(c_{n+1} = l) \tag{B.1}$$

where $P^i_{HPY}(c_{n+1} = l)$ is the probability of user i's next check-in c_{n+1} at location l, calculated using the HPY process with user i's previous check-ins, $c_1, c_2, ..., c_n$.

The SM first computes the similarity between a user i and his/her friend j where the user-friend similarity $sim(i, j)$ is based on their frequency of common check-ins over their total number of check-ins. The computation of this similarity is then repeated for the entire set of user i's friends, denoted F_i. Thereafter, the SM attempts to predict the next check-in location of user i based on his/her friends. Formally, the SM is defined as:

$$P^i_{SM}(c_{n+1} = l) = \textstyle\sum_{j \in F_i} sim(i, j) P^j_{HPY}(c_{n+1} = l) \tag{B.2}$$

where $P^j_{HPY}(c_{n+1} = l)$ is the probability that a user i's next check-in c_{n+1} is at location l, calculated using the HPY process with friend j's previous check-ins (unlike Eqn. B.1 that uses user i's previous check-ins, Eqn. B.2 uses the previous check-ins of his/her friend, user j). In short, the SM predicts user i's next check-in location based on the predicted check-ins of user i's friends and their similarity to user i.

[1] In this appendix we do not discuss the HPY process in detail and refer readers to [130] for more information.

The SHM then uses both the HM and SM to predict next check-in locations and is defined as:

$$P^i_{SHM}(c_{n+1} = l) = \eta P^i_{HM}(c_{n+1} = l) + (1 - \eta) P^i_{SM}(c_{n+1} = l) \tag{B.3}$$

where η is the weighting given to the HM and SM. The original authors experimented with various values of η and found that a value of 0.7 works the best. For more details on the SHM, refer to [55].

B.1.2 Proposed Modifications to SHM

Our modifications to SHM include adopting more stringent definitions of check-ins' recency and social links for the HM and SM respectively.

Specifically for the HM, we adopt a stringent *recency criterion* for check-ins rather than use all of a user's past check-ins regardless of their time. Let T_n be the time of the latest check-in c_n of a user i. We modify the HM (Eqn. B.1) such that we only train the HPY process with user i's previous check-ins, $c_m, c_{m+1}, ..., c_n$, where $T_n - T_m = X\ month$, i.e., we only use check-ins by a user that is within the most recent X months, as training data. In contrast, the original HM use all of a user's past check-ins. As the HPY process emphasizes on both the power-law distribution (frequency) and short-term effect (recency) of check-ins [55], a location that is frequently visited in the past could be incorrectly given a high probability. Our proposed modifications further constrain the HPY process to a much smaller set of recent check-ins, ensuring that the emphasis on check-in frequency is only on recent data. We denote this modified HM (using strict recency) as HM-SR.

Instead of using social links for the SM, we only consider *place-links*, which are defined as social links where the connected users have checked-in to the same venue on the same day. Referring to Eqn. B.2, we consider and calculate $sim(i, j)$ only for users i and j who share a common check-in that is performed on the same day. While the SM utilizes an effective user-friend similarity based on their common check-ins, the temporal aspects of such common check-ins have not been considered (e.g., two friends with common check-ins that are months apart). Place-links introduce this temporal criterion and ensure that we only consider friends who are similar in both the temporal and spatial aspects of check-ins. We denote this modified SM (using place-links) as SM-PL.

Similar to original SHM (Eqn. B.3), our modified SHM (denoted SHM-PLSR) then combines the results from both HM-SR and SM-PL with a η value of 0.7. We next present some experiments and data analysis that show the effectiveness of these proposed modifications.

B.1.3 Experimental Methodology

Datasets. We use two large LBSN datasets from Foursquare that comprise 2.29M and 2.07M check-ins, coupled with 47k and 115k friendship links among the 11k and 18k users. All check-ins are time-stamped and tagged to a specific location while user friendships are bi-directional links. The two datasets differ in terms of their time range, one is from Jan 2011 to Dec 2011 while the other is from Mar 2010 to Jan 2011. Both datasets are publicly available at [56] and [55].[2]

Evaluation. Using the two datasets mentioned previously, we divide each dataset into 10 equal time bins and consider users with >1 check-in at each time bin for our experiment. At each time bin, we hide the last check-in location of each user and attempt to predict it based on the preceding data. For example, for an evaluation using time bin 5, we predict the last check-in location for each user in time bin 5 based on the preceding time bins 1 to 5 (minus the last check-in). Thereafter, we evaluate the various models using the average prediction accuracy, which is based on the total number of correct predictions (for all users over all time bins) out of the total number of predictions made.

B.1.4 Experiments on Historical Model

Temporal Re-visit Trends. As our proposed HM-SR is built on the premise that users tend to visit (check-in to) more new places than old ones (previously visited places), we first investigate the temporal trends of how users re-visit such old places. Using the set of unique places to which a user has performed a check-in in the first time period, we

[2]While there are other public LBSN datasets like Gowalla and Brightkite [39], we use Foursquare as it still in service while the former two are no longer in operation.

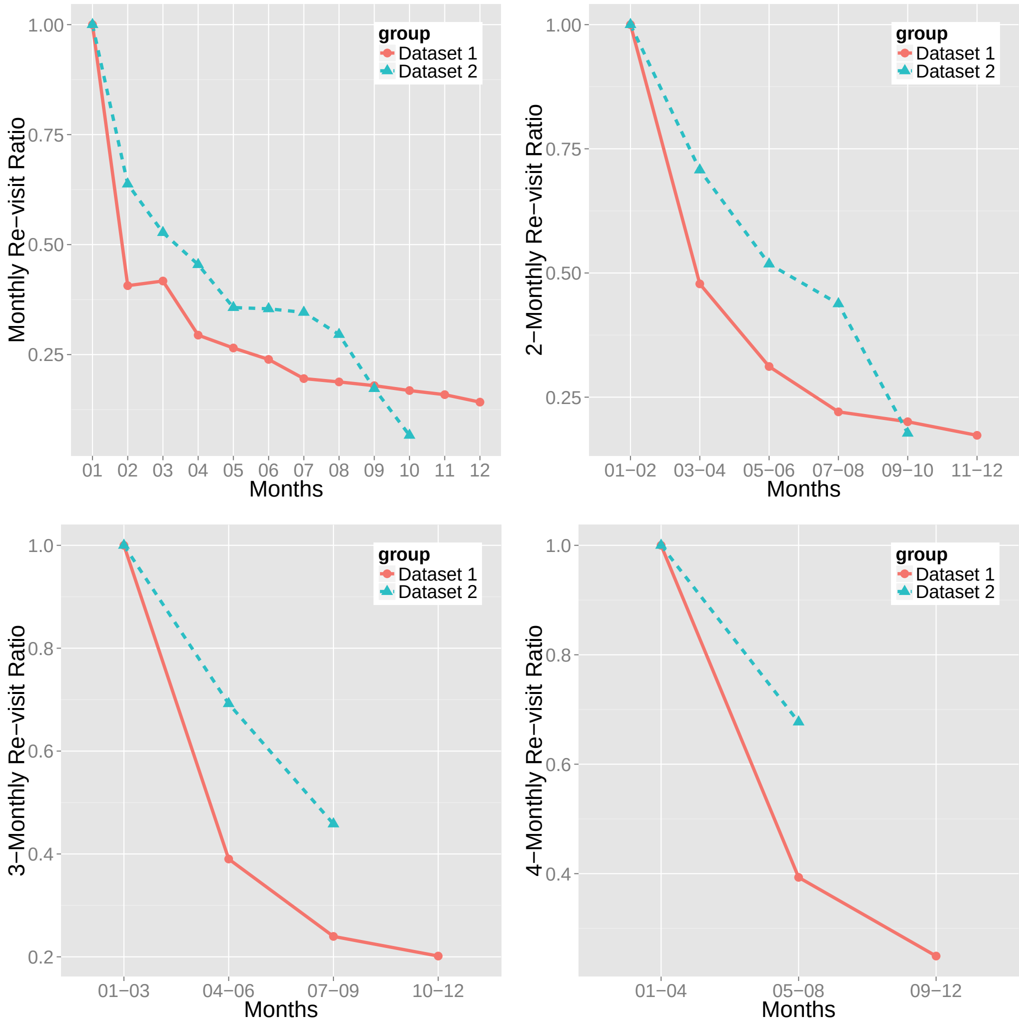


Figure B.1: Re-visit Ratio over Time Periods of 1 to 4 Months

compute the user's re-visit ratio based on how many of these unique places the same user has re-visited in the subsequent time periods. A re-visit ratio of 1 thus indicates that a user re-visits all of his/her formerly visited places, while a value of 0 indicates otherwise.

Fig. B.1 shows the average re-visit ratio for all users over the entire timespan of our two Foursquare datasets based on a time period of one to four months.[3] The results show a general downward trend of the re-visit ratio for both datasets at all time periods, indi-

[3]As dataset 2 is of a shorter duration than dataset 1, the last time period of dataset 2 is not plotted in Fig. B.1.

cating that users tend not to re-visit old places over time. Based on monthly re-visits, users only re-visit approximately half of these old places after three months, and re-visit less than one-fifth of these places after nine months. This observation indicates that LBSN users are less likely to re-visit a place which they have visited long ago (i.e., a short-term check-in trend). Our proposed HM-SR further exploits this short-term trend by implementing a strict recency constraint where we only use check-ins within one month to train our model. This short-term trend has also been exploited in other work such as [92] that uses check-in recency for identifying location-specific domain experts, and [123] that uses song recency for personalized music recommendation.

Building on the results in Fig. B.1, we have experimented with various values of check-in recency and found that a value of one month works the best for the HM-SR. Using a higher value of recency reverts the HM-SR to the original HM, while a lower value results in insufficient training data to provide accurate predictions. Thus, for the rest of the paper, we employ a check-in recency of one month for the HM-SR and SHM-PLSR. As future work, one possible extension is to further modify the HM-SR such that the recency value is optimized to individual users instead of using a common value for all users.

Evaluation of Historical Model. We now compare our proposed HM-SR to the original HM in terms of average prediction accuracy. Table B.1 shows that our proposed HM-SR provides an improvement of 2.6% and 8% over the original HM for datasets 1 and 2 respectively.

Table B.1: Prediction Accuracy for Historical Models

	Dataset 1		**Dataset 2**	
Model	**Accy.**	**Impv.**	**Accy.**	**Impv.**
HM	0.2741	-	0.2397	-
HM-SR	***0.2812***	2.6%	***0.2589***	8.0%

While our HM-SR offers a modest improvement over the HM, the lack of a bigger improvement is because the original HM is using the HPY process, which gives a heavier emphasis to check-ins that are performed more frequently and recently. However, a place that has been frequently visited in the past may be incorrectly emphasized, especially if the user no longer visits that place (e.g., due to a change of work, school or home). Our HM-SR achieves a further improvement over HM by further constraining the HPY process to a much smaller set of recent check-ins. This strict recency constraint proves to be effective as users only re-visit as few as 40% of the places they have visited in the previous month, as shown in Fig. B.1.

B.1.5 Experiments on Social Model

Comparison of Place-links and Social Links. As our SM-PL uses place-links instead of social links, we now investigate the effectiveness of place-links over social links for location prediction in terms of the check-in similarity of users. For each user i and his/her set of friends $j \in F$ (based on link type L), we define their check-in similarity as:

$$S_L = \frac{1}{|F|} \sum_{j \in F} sim(i, j) \tag{B.4}$$

Thus, S_P and S_S refers to the check-in similarity of users based on place-links and social links respectively (similarly calculated based on Eqn. B.4). We then compute S_P and S_S using the two Foursquare datasets described earlier in Section B.1.3, and present the results in Fig. B.2.

Next, we conduct a two sample t-test with null hypothesis H_0: $S_P \leq S_S$ and alternative hypothesis H_1: $S_P > S_S$. We obtained p-values of $<$ 2.2e-16 for both datasets 1 and 2, and reject the null hypothesis. This result shows that users connected by place-links share more common check-ins than users connected by social links, thus motivating the use of place-links in our proposed SM-PL, which we evaluate next.

Evaluation of Social Model. Comparing our proposed SM-PL to the original SM, we observe that our proposed SM-PL improves the average prediction accuracy by 30.2% and 20.4% over the original SM for datasets 1 and 2 respectively, as shown in Table B.2.

While the original SM utilizes a similarity weighting based on the common check-ins between a user and his/her friends, the SM does not consider the temporal aspects of this check-in, e.g., a common check-in between a user and his/her friend can be days

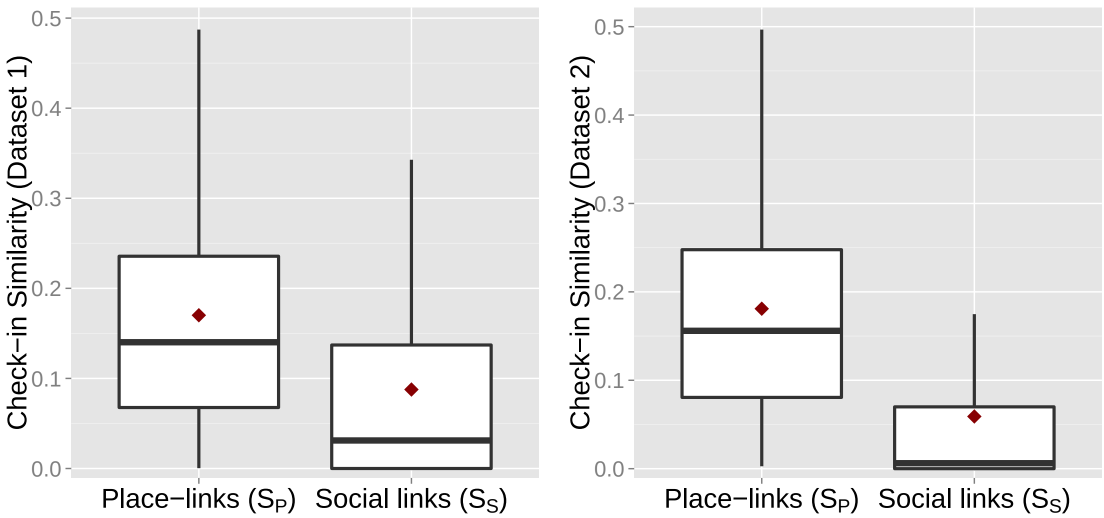


Figure B.2: Check-in similarity for users based on place-links (S_P) and social links (S_S)

Table B.2: Prediction Accuracy for Social Models

	Dataset 1		Dataset 2	
Model	**Accy.**	**Impv.**	**Accy.**	**Impv.**
SM	0.1242	-	0.0663	-
SM-PL	***0.1617***	30.2%	***0.0798***	20.4%

or months apart but are still given the same weighting. Our introduction of place-links enforces a temporal criterion on these social links, ensuring that we only consider friends who are similar in terms of a common check-in that is performed within a common time (within a day).

Prior work has also shown that spatial-social links (based on friends with a common check-in regardless of the check-in time) result in place-bound communities comprising users who frequently visit the same venues [25], which is also supported by [97,99] who observed that like-minded communities tend to consist of users that reside in the same city. Our work extends upon the concept of spatial-social links in [25] and uses place-links that are spatial-social links with a temporal constraint (i.e., friends who share a common check-in performed within a certain time) for location prediction.

B.1.6 Overall Evaluation of Prediction Models

The evaluation thus far shows that HM-SR and SM-PL out-performs their original counterparts, the HM and SM respectively. Moving on, we now evaluate the performance of SHM-PLSR against original SHM, in order to determine the overall improvement of our proposed modifications. In addition, we also compare our SHM-PLSR to various baseline prediction models that were used in [55, 94], namely:

- Most Frequent Check-in (MFC): Predicts next check-in as the most frequently visited location of the user, based on his/her previous check-ins.
- Most Frequent Check-in, Temporal-based (MFC-T): Same as MFC, but also considers the time (hour) of the day when the check-in is performed.
- Most Frequent Check-in, All Users (MFC-A): Same as MFC, but uses the most frequently visited location of all users, instead of a single user.
- Random Check-in Selection (RAND): Randomly select a location that a user has previously visited as his/her predicted next check-in location.

Table B.3: Prediction Accuracy for Various Location Prediction Models

	Prediction Accuracy	
Model	**Dataset 1**	**Dataset 2**
MFC	0.2693	0.2039
MFC-T	0.1449	0.1357
MFC-A	0.0383	0.0064
RAND	0.0299	0.0472
SHM	0.2767	0.2395
SHM-PLSR	***0.2855***	***0.2619***

Table B.3 shows that our proposed SHM-PLSR out-performs all baselines (MFC, MFC-T, MFC-A, RAND), with improvements in prediction accuracy for all cases. In addition, our proposed SHM-PLSR offers an improvement of 3.2% and 9.4% over original SHM for datasets 1 and 2 respectively.

As noted in [55], historical check-ins play a bigger role in location prediction than social links, thus the HM component heavily influences the results produced by the SHM. Similarly, this trend is also reflected in our SHM-PLSR with an overall improvement of 3.2% and 9.4% (for datasets 1 and 2), despite a greater improvement in its SM-PL component of up to 30.2%. While the improvements to the original SHM is modest, our SHM-PLSR has been shown to out-perform various other baselines by large margins. More importantly, we believe that our findings offer some insight into user behavior on LBSN and provide future opportunities for new location prediction algorithms using strict recency and place-links.

B.1.7 Discussion

We first examined the recency effect where users exhibit a short-term check-in trend and are less likely to re-visit the same place over longer periods of time. Using this observation, we then applied a strict recency criterion (using only the most recent one month of check-ins) to the Historical Model, which improved its prediction accuracy by up to 8.0%. Thereafter, we introduced place-links, which are essentially social links embedded with spatial and temporal aspects (i.e., a link connecting two friends who check-in to the same place on the same day). Next, we modified the Social Model to use place-links (instead of social links) and succeeded in improving its prediction accuracy by up to 30.2%. Finally, we show how adding the concepts of strict recency and place-links to the Social Historical Model improves its prediction accuracy by up to 9.4%. Future directions include adopting a weighted version of place-links using a time-based decay function of the common daily check-ins.

Bibliography

[1] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "NP-hardness of Euclidean sum-of-squares clustering," *Machine Learning*, vol. 75, no. 2, pp. 245–248, 2009.

[2] S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu, "Group recommendation: Semantics and efficiency," in *Proceedings of the 35th International Conference on Very Large Data Bases (VLDB'09)*, 2009, pp. 754–765.

[3] A. Anagnostopoulos, R. Atassi, L. Becchetti, A. Fazzone, and F. Silvestri, "Tour recommendation for groups," *Data Mining and Knowledge Discovery*, pp. 1–32, 2016.

[4] L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso, "Intrigue: personalized recommendation of tourist attractions for desktop and hand held devices," *Applied Artificial Intelligence*, vol. 17, no. 8-9, pp. 687–714, 2003.

[5] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2-3, pp. 235–256, 2002.

[6] R. Baeza-Yates and B. Ribeiro-Neto, *Modern information retrieval*. Addison-Wesley, 1999.

[7] N. Banerjee, D. Chakraborty, K. Dasgupta, S. Mittal, A. Joshi, S. Nagar, A. Rai, and S. Madan, "User interests in social media sites: an exploration with micro-blogs," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM'09)*, 2009, pp. 1823–1826.

[8] J. Bao, D. W. Yu Zheng, and M. Mokbel, "Recommendations in location-based social networks: A survey," *GeoInformatica*, vol. 19, no. 3, pp. 525–565, 2015.

[9] J. Bao, Y. Zheng, and M. F. Mokbel, "Location-based and preference-aware recommendation using sparse geo-social networking data," in *Proceedings of the 20th International Conference on Advances in Geographic Information Systems (SIGSPATIAL'12)*, 2012, pp. 199–208.

[10] R. Baraglia, C. I. Muntean, F. M. Nardini, and F. Silvestri, "Learnext: learning to predict tourists movements," in *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13)*, 2013, pp. 751–756.

[11] I. Benouaret and D. Lenne, "A composite recommendation system for planning tourist visits," in *Proceedings of the 2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI'16)*, 2016, pp. 626–631.

[12] I. Benouaret and D. Lenne, "A package recommendation framework for trip planning activities," in *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys'16)*, 2016, pp. 203–206.

[13] M. Berkelaar, K. Eikland, and P. Notebaert, "lpsolve: Open source (mixed-integer) linear programming system," 2004, http://lpsolve.sourceforge.net/.

[14] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[15] A. Blum, S. Chawla, D. R. Karger, T. Lane, A. Meyerson, and M. Minkoff, "Approximation algorithms for orienteering and discounted-reward tsp," *SIAM Journal on Computing*, vol. 37, no. 2, pp. 653–670, 2007.

[16] P. Bolzoni, S. Helmer, K. Wellenzohn, J. Gamper, and P. Andritsos, "Efficient itinerary planning with category constraints," in *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL'14)*, 2014, pp. 203–212.

[17] L. Boratto and S. Carta, "State-of-the-art in group recommendation and new approaches for automatic identification of groups," in *Information Retrieval and Mining in Distributed Environments*. Springer Berlin Heidelberg, 2011, pp. 1–20.

[18] J. Borràs, A. Moreno, and A. Valls, "Intelligent tourism recommender systems: A survey," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7370–7389, 2014.

[19] A. Botea and S. Braghin, "Contingent versus deterministic plans in multi-modal journey planning," in *Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS'15)*, 2015, pp. 268–272.

[20] A. Botea, E. Nikolova, and M. Berlingerio, "Multi-modal journey planning in the presence of uncertainty," in *Proceedings of the 23rd International Conference on Automated Planning and Scheduling (ICAPS'13)*, 2013, pp. 20–28.

[21] O. Braysy and M. Gendreau, "Vehicle routing problem with time windows," *Transportation Science*, vol. 39, no. 1, pp. 104–118, 2005.

[22] I. Brilhante, J. A. Macedo, F. M. Nardini, R. Perego, and C. Renso, "Where shall we go today? Planning touristic tours with TripBuilder," in *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13)*, 2013, pp. 757–762.

[23] I. Brilhante, J. A. Macedo, F. M. Nardini, R. Perego, and C. Renso, "TripBuilder: A tool for recommending sightseeing tours," in *Proceedings of the 36th European Conference on Information Retrieval (ECIR'14)*, 2014, pp. 771–774.

[24] I. R. Brilhante, J. A. Macedo, F. M. Nardini, R. Perego, and C. Renso, "On planning sightseeing tours with TripBuilder," *Information Processing & Management*, vol. 51, no. 2, pp. 1–15, 2015.

[25] C. Brown, V. Nicosia, S. Scellato, A. Noulas, and C. Mascolo, "The importance of being placefriends: discovering location-focused online communities," in *Proceedings of the 2012 ACM Workshop on Online Social Networks (WOSN'12)*, 2012, pp. 31–36.

[26] C. Brown, A. Noulas, C. Mascolo, and V. Blondel, "A place-focused model for social networks in cities," in *Proceedings of the 2013 International Conference on Social Computing (SocialCom'13)*, 2013, pp. 75–80.

[27] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.

[28] R. C. Browning, E. A. Baker, J. A. Herron, and R. Kram, "Effects of obesity and sex on the energetic cost and preferred speed of walking," *Journal of Applied Physiology*, vol. 100, no. 2, pp. 390–398, 2006.

[29] R. E. Burkard, "Quadratic assignment problems," *European Journal of Operational Research*, vol. 15, no. 3, pp. 283–289, 1984.

[30] R. Burke, "Hybrid web recommender systems," in *The Adaptive Web*. Springer Berlin/Heidelberg, 2007, pp. 377–408.

[31] L. Castillo, E. Armengol, E. Onaindía, L. Sebastiá, J. González-Boticario, A. Rodríguez, S. Fernández, J. D. Arias, and D. Borrajo, "SAMAP: An user-oriented adaptive system for planning tourist visits," *Expert Systems with Applications*, vol. 34, no. 2, pp. 1318–1332, 2008.

[32] C. Chekuri and M. Pal, "A recursive greedy algorithm for walks in directed graphs," in *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, 2005, pp. 245–253.

[33] C. Chen, X. Chen, Z. Wang, Y. Wang, and D. Zhang, "Scenicplanner: planning scenic travel routes leveraging heterogeneous user-generated digital footprints," *Frontiers of Computer Science*, vol. 11, no. 1, p. 6174, 2017.

[34] C. Chen, D. Zhang, B. Guo, X. Ma, G. Pan, and Z. Wu, "TripPlanner: Personalized trip planning leveraging heterogeneous crowdsourced digital footprints,"

IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 3, pp. 1259–1273, 2015.

[35] D. Chen, C. S. Ong, and L. Xie, "Learning points and routes to recommend trajectories," in *Proceedings of the 25th ACM International Conference on Information and Knowledge Management (CIKM'16)*, 2016, pp. 2227–2232.

[36] Y.-Y. Chen, A.-J. Cheng, and W. H. Hsu, "Travel recommendation by mining people attributes and travel group types from community-contributed photos," *IEEE Transactions on Multimedia*, vol. 15, no. 6, pp. 1283–1295, 2013.

[37] Z. Chen, H. T. Shen, and X. Zhou, "Discovering popular routes from trajectories," in *Proceedings of the IEEE 27th International Conference on Data Engineering (ICDE'11)*, 2011, pp. 900–911.

[38] A.-J. Cheng, Y.-Y. Chen, Y.-T. Huang, W. H. Hsu, and H.-Y. M. Liao, "Personalized travel recommendation by mining people attributes from community-contributed photos," in *Proceedings of the 19th ACM International Conference on Multimedia (MM'11)*, 2011, pp. 83–92.

[39] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11)*, 2011, pp. 1082–1090.

[40] M. D. Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel, and C. Yu, "Automatic construction of travel itineraries using social breadcrumbs," in *Proceedings of the 21st ACM Conference on Hypertext and Hypermedia (HT'10)*, 2010, pp. 35–44.

[41] M. D. Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel, and C. Yu, "Constructing travel itineraries from tagged geo-temporal breadcrumbs," in *Proceedings of the 19th International Conference on World Wide Web (WWW'10)*, 2010, pp. 1083–1084.

[42] D. Cogswell, "Why are customized tour packages the hot travel trend?" 2014, http://www.travelpulse.com/news/vacation-packages/why-are-customized-tour-packages-the-hot-travel-trend.html.

[43] R. Cohen and L. Katzir, "The generalized maximum coverage problem," *Information Processing Letters*, vol. 108, no. 1, pp. 15–22, 2008.

[44] V. Conitzer and T. Sandholm, "Complexity of mechanism design," in *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI'02)*, 2002, pp. 103–110.

[45] R. Coulom, "Efficient selectivity and backup operators in monte-carlo tree search," in *Proceedings of the 2016 International Conference on Computers and Games (CCG'06)*, 2006, pp. 72–83.

[46] D. J. Crandall, L. Backstrom, D. Cosley, S. Suri, D. Huttenlocher, and J. Kleinberg, "Inferring social ties from geographic coincidences," *Proceedings of the National Academy of Sciences*, vol. 107, no. 52, pp. 22 436–22 441, 2010.

[47] S. Dhar and U. Varshney, "Challenges and business models for mobile location-based services and advertising," *Communications of the ACM*, vol. 54, no. 5, pp. 121–128, 2011.

[48] M. Dorigo, M. Birattari, and T. Stützle, "Ant colony optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[49] D. Eppstein, "Finding the k shortest paths," *SIAM Journal on Computing*, vol. 28, no. 2, pp. 652–673, 1998.

[50] J. E. Flórez, A. T. A. de Reyna, C. L. L. Javier García, A. G. Olaya, and D. Borrajo, "Planning multi-modal transportation problems," in *Proceedings of the 21st International Conference on Automated Planning and Scheduling (ICAPS'11)*, 2011, pp. 66–73.

[51] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.

[52] C.-Y. Fu, M.-C. Hu, J.-H. Lai, H. Wang, and J.-L. Wu, "Travelbuddy: Interactive travel route recommendation with a visual scene interface," in *Proceedings of the 20th International Conference on Multimedia Modeling (MMM'14)*, 2014, pp. 219–230.

[53] E. Galbrun, K. Pelechrinis, and E. Terzi, "Safe navigation in urban environments," in *Proceedings of the 3rd International SIGKDD Workshop on Urban Computing (UrbComp'14)*, 2014.

[54] E. Galbrun, K. Pelechrinis, and E. Terzi, "Urban navigation beyond shortest route: The case of safe paths," *Information Systems*, vol. 57, p. 160171, 2016.

[55] H. Gao, J. Tang, and H. Liu, "Exploring social-historical ties on location-based social networks," in *Proceedings of the 6th International AAAI Conference on Weblogs and Social Media (ICWSM'12)*, 2012, pp. 114–121.

[56] H. Gao, J. Tang, and H. Liu, "gSCorr: modeling geo-social correlations for new check-ins on location-based social networks," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM'12)*, 2012, pp. 1582–1586.

[57] I. Garcia, L. Sebastia, and E. Onaindia, "On the design of individual and group recommender systems for tourism," *Expert Systems with Applications*, vol. 38, no. 6, pp. 7683–7692, 2011.

[58] I. Garcia, L. Sebastia, E. Onaindia, and C. Guzman, "A group recommender system for tourist activities," in *Proceedings of the 10th International Conference on E-Commerce and Web Technologies (EC-WEB'09)*, 2009, pp. 26–37.

[59] D. Gavalas, V. Kasapakis, C. Konstantopoulos, G. Pantziou, N. Vathis, and C. Zaroliagis, "The ecompass multimodal tourist tour planner," *Expert Systems with Applications*, vol. 42, no. 21, pp. 7303–7316, 2015.

[60] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou, "A survey on algorithmic approaches for solving tourist trip design problems," *Journal of Heuristics*, vol. 20, no. 3, pp. 291–328, 2014.

[61] D. Gildea and T. Hofmann, "Topic-based language models using em," in *Proceedings of 6th European Conference on Speech Communication and Technology (EuroSpeech'99)*, 1999.

[62] A. Gionis, T. Lappas, K. Pelechrinis, and E. Terzi, "Customized tour recommendations in urban areas," in *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM'14)*, 2014, pp. 313–322.

[63] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Naval Research Logistics*, vol. 34, no. 3, pp. 307–318, 1987.

[64] Google, "Google maps API," 2016, https://developers.google.com/maps/.

[65] A. Gunawan, H. C. Lau, and P. Vansteenwegen, "Orienteering problem: A survey of recent variants, solution approaches and applications," *European Journal of Operational Research*, vol. 255, no. 2, pp. 315–332, 2016.

[66] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. C. Hsu, "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth," in *Proceedings of the 17th International Conference on Data Engineering (ICDE'01)*, 2001, pp. 215–224.

[67] U. Hanani, B. Shapira, and P. Shoval, "Information filtering: Overview of issues, research and systems," *User Modeling and User-Adapted Interaction*, vol. 11, no. 3, pp. 203–259, 2001.

[68] J. A. Hartigan and M. A. Wong, "A k-means clustering algorithm," *Applied statistics*, vol. 28, no. 1, pp. 100–108, 1979.

[69] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, 1999, pp. 50–57.

[70] B. Hu, M. Jamali, and M. Ester, "Spatio-temporal topic modeling in mobile social media for location recommendation," in *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM'13)*, 2013, pp. 1073–1078.

[71] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and W. Cao, "Deep modeling of group preferences for group-based recommendation," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI'14)*, 2014, pp. 1861–1867.

[72] A. Jameson, S. Baldes, and T. Kleinbauer, "Enhancing mutual awareness in group recommender systems," in *Proceedings of the IJCAI Workshop on Intelligent Techniques for Web Personalization*, 2003.

[73] R. Ji, X. Xie, H. Yao, and W.-Y. Ma, "Mining city landmarks from blogs by graph modeling," in *Proceedings of the 17th ACM International Conference on Multimedia (MM'09)*, 2009, pp. 105–114.

[74] S. Jiang, X. Qian, T. Mei, and Y. Fu, "Personalized travel sequence recommendation on multi-source big social media," *IEEE Transactions on Big Data*, vol. 2, no. 1, pp. 43–56, 2016.

[75] T. Joachims, "Training linear SVMs in linear time," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'06)*, 2006, pp. 217–226.

[76] A. Keler and J. D. Mazimpaka, "Safety-aware routing for motorised tourists based on open data and vgi," *Journal of Location Based Services*, vol. 10, no. 1, pp. 64–77, 2016.

[77] A. S. Kenyon and D. P. Morton, "Stochastic vehicle routing with random travel times," *Transportation Science*, vol. 37, no. 1, pp. 69–8, 2003.

[78] J. Kiseleva, M. J. Mueller, L. Bernardi, C. Davis, I. Kovacek, M. S. Einarsen, J. Kamps, A. Tuzhilin, and D. Hiemstra, "Where to go on your next trip?: Optimizing travel destinations based on user preferences," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'15)*, 2015, pp. 1097–1100.

[79] S. Kisilevich, F. Mansmann, and D. Keim, "P-dbscan: a density based clustering algorithm for exploration and analysis of attractive areas using collections of geo-

tagged photos," in *Proceedings of the 1st International Conference and Exhibition on Computing for Geospatial Research and Application (COM.Geo'10)*, 2010, p. 38.

[80] L. Kocsis and C. Szepesvári, "Bandit based monte-carlo planning," in *Proceedings of the 17th European Conference on Machine Learning (ECML'06)*, 2006, pp. 282–293.

[81] L. Kocsis, C. Szepesvári, and J. Willemson, "Improved monte-carlo search," University of Tartu, Institute of Computer Science, Tech. Rep., 2006.

[82] C. Kofler, L. Caballero, M. Menendez, V. Occhialini, and M. Larson, "Near2me: An authentic and personalized social media-based recommender for travel destinations," in *Proceedings of the 3rd ACM SIGMM International Workshop on Social Media (WSM'11)*, 2011, pp. 47–52.

[83] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, 1995, pp. 1137–1145.

[84] R. Kohavi, A. Deng, B. Frasca, T. Walker, Y. Xu, and N. Pohlmann, "Online controlled experiments at large scale," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13)*, Aug 2013, pp. 1168–1176.

[85] S. Kotiloglu, T. Lappas, K. Pelechrinis, and P. Repoussis, "Personalized multiperiod tour recommendations," *Tourism Management*, vol. 62, pp. 76–88, 2017.

[86] T. Kurashima, T. Iwata, G. Irie, and K. Fujimura, "Travel route recommendation using geotags in photo sharing sites," in *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM'10)*, 2010, pp. 579–588.

[87] T. Kurashima, T. Iwata, G. Irie, and K. Fujimura, "Travel route recommendation using geotagged photos," *Knowledge and Information Systems*, vol. 37, no. 1, pp. 37–60, 2013.

[88] K. W.-T. Leung, D. L. Lee, and W.-C. Lee, "Clr: a collaborative location recommendation framework based on co-clustering," in *Proceedings of the 34th Interna-*

tional ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'11), 2011, pp. 305–314.

[89] R. A. Lewis, J. M. Rao, and D. H. Reiley, "Here, there, and everywhere: Correlated online behaviors can lead to overestimates of the effects of advertising," in *Proceedings of the 20th International Conference on World Wide Web (WWW'11)*, Mar 2011, pp. 157–166.

[90] J. Li, X. Qian, Y. Y. Tang, L. Yang, and T. Mei, "Gps estimation for places of interest from social users' uploaded photos," *IEEE Transactions on Multimedia*, vol. 15, no. 8, pp. 2058–2071, 2013.

[91] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and W.-Y. Ma, "Mining user similarity based on location history," in *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL'08)*, 2008, p. 34.

[92] W. Li, C. Eickhoff, and A. P. de Vries, "Geo-spatial domain expertise in microblogs," in *Proceedings of the 36th European Conference on Information Retrieval (ECIR'14)*, 2014, pp. 487–492.

[93] X. Li, G. Cong, X.-L. Li, T.-A. N. Pham, and S. Krishnaswamy, "Rank-geofm: a ranking based geographical factorization method for point of interest recommendation," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'15)*, 2015, pp. 433–442.

[94] D. Lian, V. W. Zheng, and X. Xie, "Collaborative filtering meets next check-in location prediction," in *Proceedings of the 22nd International Conference on World Wide Web Companion (WWW'13)*, 2013, pp. 231–232.

[95] T. Liebig, N. Piatkowski, C. Bockermann, and K. Morik, "Predictive trip planning-smart routing in smart cities," in *Proceedings of EDBT/ICDT Workshop on Mining Urban Data (MUD'14)*, 2014, pp. 331–338.

[96] T. Liebig, N. Piatkowski, C. Bockermann, and K. Morik, "Dynamic route planning with real-time traffic predictions," *Information Systems*, vol. 64, pp. 258–265, 2017.

[97] K. H. Lim and A. Datta, "Tweets beget propinquity: Detecting highly interactive communities on twitter using tweeting links," in *Proceedings of the 2012 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT'12)*, 2012, pp. 214–221.

[98] K. H. Lim and A. Datta, "A topological approach for detecting twitter communities with common interests," in *Ubiquitous Social Media Analysis*. Springer Berlin Heidelberg, 2013, pp. 23–43.

[99] K. H. Lim and A. Datta, "An interaction-based approach to detecting highly interactive twitter communities using tweeting links," *Web Intelligence*, vol. 14, no. 1, pp. 1–15, 2016.

[100] K. H. Lim, E.-P. Lim, B. Jiang, and P. Achananuparp, "Using online controlled experiments to examine authority effects on user behavior in email campaigns," in *Proceedings of the 27th ACM Conference on Hypertext and Social Media (HT'16)*, 2016, pp. 255–260.

[101] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: A recurrent model with spatial and temporal contexts," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*, 2016, pp. 194–200.

[102] X. Liu, Y. Liu, K. Aberer, and C. Miao, "Personalized point-of-interest recommendation by mining users' preference transition," in *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13)*, 2013, pp. 733–738.

[103] E. M. Loiolaa, N. M. M. de Abreua, P. O. Boaventura-Nettoa, P. Hahnb, and T. Queridoc, "A survey for the quadratic assignment problem," *European Journal of Operational Research*, vol. 176, no. 2, pp. 657–690, 2007.

[104] X. Lu, C. Wang, J.-M. Yang, Y. Pang, and L. Zhang, "Photo2trip: generating travel routes from geo-tagged photos for trip planning," in *Proceedings of the 18th ACM International Conference on Multimedia (MM'10)*, 2010, pp. 143–152.

[105] C. Lucchese, R. Perego, F. Silvestri, H. Vahabi, and R. Venturini, "How random walks can help tourism," in *Proceedings of the 34th European Conference on Information Retrieval (ECIR'12)*, 2012, pp. 195–206.

[106] A. Majid, L. Chen, H. T. Mirza, I. Hussain, and G. Chen, "A system for mining interesting tourist locations and travel sequences from public geo-tagged photos," *Data & Knowledge Engineering*, vol. 95, pp. 66–86, 2015.

[107] I. Memon, L. Chen, A. Majid, M. Lv, I. Hussain, and G. Chen, "Travel recommendation using geo-tagged photos in social media for tourist," *Wireless Personal Communications*, vol. 80, no. 4, pp. 1347–1362, 2015.

[108] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *Journal of the ACM*, vol. 7, no. 4, pp. 326–329, 1960.

[109] J.-P. Onnela, S. Arbesman, M. C. González, A.-L. Barabási, and N. A. Christakis, "Geographic constraints on social network groups," *PLoS one*, vol. 6, no. 4, p. e16939, 2011.

[110] J. Pennebaker, *The Secret Life of Pronouns: What Our Words Say About Us*. Bloomsbury, 2013.

[111] T.-A. N. Pham, X. Li, and G. Cong, "A general model for out-of-town region recommendation," in *Proceedings of the 26th International Conference on World Wide Web (WWW'17)*, 2017, pp. 401–410.

[112] A. Popescu, G. Grefenstette, and P.-A. Moëllic, "Mining tourist information from user-supplied collections," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM'09)*, 2009, pp. 1713–1716.

[113] E. J. Powley, D. Whitehouse, and P. I. Cowling, "Monte carlo tree search with macro-actions and heuristic route planning for the multiobjective physical travelling salesman problem," in *Proceedings of the 2013 IEEE Conference on Computational Intelligence in Games (CIG'13)*, 2013, pp. 1–8.

[114] D. Quercia, R. Schifanella, and L. M. Aiello, "The shortest path to happiness: Recommending beautiful, quiet, and happy routes in the city," in *Proceedings of the 25th ACM Conference on Hypertext and Social Media (HT'14)*, 2014, pp. 116–125.

[115] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical Review E*, vol. 76, no. 3, p. 036106, 2007.

[116] I. Refanidis and A. Alexiadis, "Deployment and evaluation of selfplanner, an automated individual task management system," *Computational Intelligence*, vol. 27, no. 1, pp. 41–59, 2011.

[117] I. Refanidis, C. Emmanouilidis, I. Sakellariou, A. Alexiadis, R.-A. Koutsiamanis, K. Agnantis, A. Tasidou, F. Kokkoras, and P. S. Efraimidis, "myVisitPlanner GR: Personalized itinerary planning system for tourism," in *Proceedings of the 8th Hellenic Conference on Artificial Intelligence (SETN'14)*, 2014, pp. 615–629.

[118] I. Refanidis and N. Yorke-Smith, "A constraint-based approach to scheduling an individual's activities," *ACM Transactions on Intelligent Systems and Technology*, vol. 1, no. 2, p. 12, 2010.

[119] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work (CSCW'94)*, 1994, pp. 175–186.

[120] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proceedings of the National Academy of Science*, vol. 105, no. 4, pp. 1118–1123, 2008.

[121] S. B. Roy, L. V. Lakshmanan, and R. Liu, "From group recommendations to group formation," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (SIGMOD'15)*, 2015, pp. 1603–1616.

[122] R. Schaller, M. Harvey, and D. Elsweiler, "Recsys for distributed events: investigating the influence of recommendations on visitor plans," in *Proceedings of the 36th*

international ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'13), 2013, pp. 953–956.

[123] M. Schedl, D. Hauger, and D. Schnitzer, "A model for serendipitous music retrieval," in *Proceedings of the 2nd Workshop on Context-awareness in Retrieval and Recommendation (CaRR'12)*, 2012, pp. 10–13.

[124] Y. Shi, P. Serdyukov, A. Hanjalic, and M. Larson, "Personalized landmark recommendation based on geotags from photo sharing sites," in *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM'11)*, 2011, pp. 622–625.

[125] R. W. Sinnott, "Virtues of the Haversine," *Sky and Telescope*, vol. 68, no. 158, 1984.

[126] W. Souffriau and P. Vansteenwegen, "Tourist trip planning functionalities: State-of-the-art and future," in *Proceedings of the 10th International Conference on Web Engineering (ICWE'10)*, 2010, pp. 474–485.

[127] W. Souffriau, P. Vansteenwegen, J. Vertommen, G. V. Berghe, and D. V. Oudheusden, "A personalized tourist trip design algorithm for mobile tourist guides," *Applied Artificial Intelligence*, vol. 22, no. 10, pp. 964–985, 2008.

[128] E. Spyrou and P. Mylonas, "A survey on flickr multimedia research challenges," *Engineering Applications of Artificial Intelligence*, vol. 51, pp. 71–91, 2016.

[129] D. Tang, A. Agarwal, D. O'Brien, and M. Meyer, "Overlapping experiment infrastructure: more, better, faster experimentation," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'10)*, Jul 2010, pp. 17–26.

[130] Y. W. Teh, "A hierarchical Bayesian language model based on Pitman-Yor processes," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL'06)*, 2006, pp. 985–992.

[131] The Verge, "The man behind Flickr on making the service 'awesome again'," 2013, http://www.theverge.com/2013/3/20/4121574/flickr-chief-markus-spiering-talks-photos-and-marissa-mayer.

[132] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li, "The new data and new challenges in multimedia research," in *arXiv:1503.01817*, 2015.

[133] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li, "YFCC100M: The new data in multimedia research," *Communications of the ACM*, vol. 59, no. 2, pp. 64–73, 2016.

[134] H. Tong and C. Faloutsos, "Center-piece subgraphs: problem definition and fast solutions," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'06)*, 2006, pp. 404–413.

[135] TouringPlans.com, "Disney's hollywood studios attraction durations," 2016, https://touringplans.com/hollywood-studios/attractions/duration.

[136] T. Tsiligirides, "Heuristic methods applied to orienteering," *Journal of the Operational Research Society*, vol. 35, no. 9, pp. 797–809, 1984.

[137] G. Tumas and F. Ricci, "Personalized mobile city transport advisory system," in *Information and Communication Technologies in Tourism*. Springer, 2009, pp. 173–183.

[138] UNWTO, "United Nations World Tourism Organization (UNWTO) annual report 2015," 2016, http://www2.unwto.org/annual-reports.

[139] P. Vansteenwegen and D. V. Oudheusden, "The mobile tourist guide: An OR opportunity," *OR Insight*, vol. 20, no. 3, pp. 21–27, 2007.

[140] P. Vansteenwegen, W. Souffriau, G. V. Berghe, and D. V. Oudheusden, "Iterated local search for the team orienteering problem with time windows," *Computers & Operations Research*, vol. 36, no. 12, pp. 3281–3290, 2009.

[141] P. Vansteenwegen, W. Souffriau, G. V. Berghe, and D. V. Oudheusden, "The city trip planner: An expert system for tourists," *Expert Systems with Applications*, vol. 38, no. 6, pp. 6540–6546, 2011.

[142] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, no. 1, pp. 1–10, 2011.

[143] Victoria State Government, "Domestic visitation to regions of victoria," 2016, http://www.tourism.vic.gov.au/research/domestic-and-regional-research/domestic-visitation.html.

[144] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'01)*, 2001, pp. 511–518.

[145] X. Wang, C. Leckie, J. Chan, K. H. Lim, and T. Vaithianathan, "Improving personalized trip recommendation to avoid crowds using pedestrian sensor data," in *Proceedings of the 25th ACM International Conference on Information and Knowledge Management (CIKM'16)*, 2016, pp. 25–34.

[146] J. H. Ward, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.

[147] World Travel and Tourism Council, "2016 economic impact annual update summary," 2016, https://www.wttc.org/research/economic-research/economic-impact-analysis/.

[148] W. Wörndl and A. Hefele, "Generating paths through discovered places-of-interests for city trip planning," in *Information and Communication Technologies in Tourism*. Springer International Publishing, 2016, pp. 441–453.

[149] A. Yahi, A. Chassang, L. Raynaud, H. Duthil, and D. H. P. Chau, "Aurigo: An interactive tour planner for personalized itineraries," in *Proceedings of the 20th International Conference on Intelligent User Interfaces (IUI'15)*, 2015, pp. 275–285.

[150] Yahoo, "Flickr API," 2016, https://www.flickr.com/services/api/.

[151] Yahoo! Webscope, "Yahoo! Flickr Creative Commons 100M dataset (YFCC-100M)," 2014, http://webscope.sandbox.yahoo.com/catalog.php?datatype=i&did=67.

[152] T. Yamasaki, A. Gallagher, and T. Chen, "Personalized intra-and inter-city travel recommendation using large-scale geotags," in *Proceedings of the 2nd ACM International Workshop on Geotagging and Its Applications in Multimedia (GeoMM'13)*, 2013, pp. 25–30.

[153] L. Yao, Q. Z. Sheng, Y. Qin, X. Wang, A. Shemshadi, and Q. He, "Context-aware point-of-interest recommendation using tensor factorization with social regularization," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'15)*, 2015, pp. 1007–1010.

[154] M. Ye, P. Yin, W.-C. Lee, and D.-L. Lee, "Exploiting geographical influence for collaborative point-of-interest recommendation," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'11)*, 2011, pp. 325–334.

[155] P. Yin, M. Ye, W.-C. Lee, and Z. Li, "Mining gps data for trajectory recommendation," in *Proceedings of the 18th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD'14)*, 2014, pp. 50–61.

[156] Z. Yu, H. Xu, Z. Yang, and B. Guo, "Personalized travel package with multi-point-of-interest recommendation based on crowdsourced user footprints," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 1, pp. 151–158, 2016.

[157] Q. Yuan, G. Cong, and C.-Y. Lin, "COM: a generative model for group recommendation," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'14)*, 2014, pp. 163–172.

[158] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann, "Time-aware point-of-interest recommendation," in *Proceedings of the 36th International ACM SIGIR Con-*

ference on Research and Development in Information Retrieval (SIGIR'13), 2013, pp. 363–372.

[159] C. Zhang, H. Liang, and K. Wang, "Trip recommendation meets real-world constraints: Poi availability, diversity, and traveling time uncertainty," *ACM Transactions on Information Systems*, vol. 35, no. 1, p. 5, 2016.

[160] C. Zhang, H. Liang, K. Wang, and J. Sun, "Personalized trip recommendation with poi availability and uncertain traveling time," in *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM'15)*, 2015, pp. 911–920.

[161] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang, "Collaborative location and activity recommendations with gps history data," in *Proceedings of the 19th International Conference on World Wide Web (WWW'10)*, 2010, pp. 1029–1038.

[162] Y. Zheng and X. Xie, "Learning travel recommendations from user-generated gps traces," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 1, p. 2, 2011.

[163] Y. Zheng, L. Zhang, Z. Ma, X. Xie, and W.-Y. Ma, "Recommending friends and locations based on individual location history," *ACM Transactions on the Web*, vol. 5, no. 1, p. 5, 2011.

[164] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th International Conference on World Wide Web (WWW'09)*, 2009, pp. 791–800.

[165] Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun, "A general boosting method and its application to learning ranking functions for web search," in *Advances in Neural Information Processing Systems (NIPS'07)*, 2007, pp. 1697–1704.