GroupTour : A Framework for Group Travel Route Recommendation

Chenxiao Yang (517021910540, cxyang1@outlook.com)
Jiaqi Zeng (517021910882, Gabyyyyyy@sjtu.edu.cn)
Xu Huan (I517021910724, huanxu@sjtu.edu.cn)

Department of Computer Science,
Shanghai Jiao Tong University, Shanghai, China

Abstract. For a group of tourists with different personal tastes coming to a new city, it's an important problem to design a travel route. Tour design problem is challenging since we have large number of POIs (Point of Interest) to choose from and every member has his own different preference. In this article, we formulate group tour recommendation problem as a variant of orienteering problem and analyze the hardness. We define popularity of a POI based on its attributes, and we extract personal interest from their geo-tagged photos. Also, we address the algorithmic implication of the above problem by presenting several algorithms based on the idea of greedy and local search. The balance between satisfaction and fairness is carefully considered. Our model is implemented on the given datasets and is shown to be efficient.

Keywords: Group tour recommendation, Greedy Search, Orienteering Problem, Best-First.

1 Introduction

For tourists coming to a city, it's a troublesome work to plan the whole trip. They often have to search online and gather enough information to plan for a travel route consisting of POIs they are interested in, which wastes a lot of time. For a group of tourists, this task becomes even harder [4]. Hence, tour recommendation becomes an important subject in the field of recommendation system.

The tourist trip design problems (TTDP) [5] refer to a route-planning for tourists who are interested in visiting multiple POIs. This problem has many possible constraints, including weather condition, time window, money budget, time budget. In our article, we only consider time budget.

We present an efficient framework for recommending a travel route for group. And the trip time should be less or equal to a given time budget. For this problem, we formulate it as a variant of orienteering problem [3], and present three algorithms to solve it. The difficulty of this problem lies on how to satisfy group's overall taste and make a trade-off between efficiency and fairness. To solve this problem, we ensure the route meets most members' taste by maximizing the overall satisfaction, and keep the difference between group members small by minimizing the satisfaction variance.

Our article is organized as follows. Section 2 defines the problem, analyze the difficulty and formulates it into an ILP problem. Section 3 presents three algorithms to solve the problem. Section 4 conducts experiment on those algorithms to test their efficiency.

2 Problem Statement

In this section we formulate the problem as a variant of orienteering problem, judge the difficulty, and show that we can convert our problem as an ILP.

2.1 Preliminaries

Given an undirected graph $G = (V, E)$ with edge weights $w(u, v)$, a group of tourists $T = \{t_1, t_2, \cdots, t_m\}$ and a time budget B, our goal is to recommend a travel route whose time cost does not exceed B, maximizing the overall satisfaction while at the same time minimizing the variance.

Suppose there are n POIs in the city $P = \{p_1, p_2, \cdots, p_n\}$, and each POI belongs to a category $c_i \in C$. The attributes of POI is defined as:

Table 1. Attributes of a POI p.

Symbol	Attribute
$\#review(p)$	the number of the reviewer
$Score(p)$	the score of the POI, from 0 to 5
$Prof(p)$	profit of the destination POI, based on number of POI visits
C_p	the category of the destination POI

Definition 1. *The **Popularity** of a POI is defined based on those POI attributes.*

$$Pop(p) = \alpha \cdot \frac{Prof(p)}{max_P Prof(p_i)} + \beta \cdot \frac{Score(p)}{5} + \gamma \cdot (1 - \frac{1}{\#reviews(p)}) + \delta \cdot (1 - \frac{rank}{N}) \tag{1}$$

rank is the POI's rank of POI visits among all POIs with the same category in this city.We use N to represent the number of these POIs. Four parameters are set by users with the constraint $\alpha+\beta+\gamma+\delta = 1$. This limits the value field of $Pop(p)$ to $[0, 1]$

We define $Cost(p_u, p_v)$ as the traveling time a group of tourists cost from POI p_u to POI p_v. Traveling time is dependent on the distance between POI p_u and POI p_v, and the traveling speed of tourists. For convenience, we set travel speed as 4km/h, which is a normal walking speed.

For tourists, we have their visits to various POIs in eight cities. The user-POI visits are determined based on geo-tagged Flickr photos. According to the geo-tagged photos they have taken, we can estimate the visiting duration of a POI p and extract user's corresponding preferences.

Definition 2. *The **Cost** of a POI p represents the estimated normal visiting duration in POI p, which is based on the ratio of average number of photos each user has taken in POI p and average number of photos each user has taken in all POIs*

$$Cost(p) = BaseCost + \lambda \cdot \frac{\#AvgPhoto(p) \cdot |P|}{\sum_{p_i \in P} \#AvgPhoto(p_i)} \tag{2}$$

$\#AvgPhoto(p)$ is the average number of photos a user has taken in POI p (we only consider those users who have been to POI p). BaseCost is the minimal estimated visiting duration, and λ is the unit visiting duration. The more photos user takes in that POI on average, the higher value of visiting duration we estimate.

Definition 3. ***User Interest** of a tourist t for a Category c is defined to be:*

$$I(c, t) = \frac{\#AvgPhoto(c, t)}{\#AvgPhoto(c)} \tag{3}$$

$\#AvgPhoto(c, t)$ is the average number of photos a user t has taken in POI whose category is c, $\#AvgPhoto(c)$ is the average number of photos all users have taken in POI whose category is c.

We consider the average number of photos a user has taken in POIs of a certain category rather than a specific POI. The reason is that the user has never been to the target city before and we compute the user's interest based only on his historical visits in other cities. The main idea is the more photo he takes of a certain category compared with others, the more he prefers POIs of this category.

For those who have never taken any POI of a certain category, we can't get his interest of that category. In this case, we assume he has average taste and set $I(c, t) = 1$.

Definition 4. ***Satisfaction** of a user t for a POI p is defined as the weighted sum of popularity and user interest:*

$$Sat(p, t) = \eta \cdot Pop(p) + (1 - \eta) \cdot I(c_p, t) \tag{4}$$

η is a parameter, which is dependent on whether user count on his own interest or popularity of target POI.

Definition 5. *The **Weight** of an edge $e(p_u, p_v)$ is defined as:*

$$w(p_u, p_v) = Cost(p_u, p_v) + \frac{1}{2}Cost(p_u) + \frac{1}{2}Cost(p_v) \tag{5}$$

To generate a route from s to t within the time limit, we only have to choose a path from s to t, whose sum of weight is less or equal to $B - \frac{1}{2}Cost(s) - \frac{1}{2}Cost(t)$.

2.2 Problem Definition

We formally define the problem as follows: Given a weighted undirected POI Graph $G = (V, E, w)$, two nodes $s, t \in V$, a value $B \in \mathbb{R}^+$, compute a path $T = \{s, v_1, v_2, \cdots, v_l, t\}$, $v_i \in V$ such that $cost(T) \leq B$ and $\Phi(t_1(T), t_2(T), \cdots, t_m(T))$ is maximized, where $cost(T) = w(s, v_1) + w(v_1, v_2) + \cdots + w(v_l, t)$ and $t_i(T)$ is the route satisfaction for tourist t_i. To maximize the total satisfaction while at the same time decreasing the variance, we let:

$$\Phi(t_1(T), t_2(T), \cdots, t_m(T)) = \sum_{i=1}^{m} t_i(T) - \omega \cdot std_{i=1}^{m}(t_i(T)) \tag{6}$$

ω is a parameter depending on how much users care about the fairness of the optimization result in comparison with the sum of personal satisfaction.

2.3 Difficulty of the Problem

As for the difficulty of the problem, we claim that GroupTour is in NP-Complete.

Proof. We know that TSP is in NPC, so we will conduct our proof from TSP. Let's first define TSP problem and write GroupTour in another form:

Definition 6. ***TSP:***
Instance: a complete graph $G = (V, E)$, each edge e has a weight $w(e)$, the start point s
Solution: a cycle C starting and ending at s which passes through all the vertices and passes through each vertex only once
Measure: $\sum_{e \in C} w(e)$
Goal: Min

Definition 7. ***GroupTour:***
Instance: a complete graph $G = (V, E)$, each edge e has a weight $w(e)$, each vertex u has a value $v(u)$, the start point s, the end point t, a boundary cost b
Solution: a path p starting at s and ending at t which satisfies $\sum_{e \in p} w(e) \leq b$ and passes through each vertex only once
Measure: $\sum_{u \in p} v(u)$
Goal: Max

First, it is easily to show that GroupTour is NP because we can certificate it in $O(n)$. Next, we claim that TSP $\leq_p$ GroupTour.

Given an instance x of TSP: a complete graph $G = (V, E)$, each edge e has a weight $w(e)$, the start point s, we construct an instance y of GroupTour. y: graph $G' = (V', E')$, each edge e has a weight $w(e)$, each vertex u has a value $v(u)$, the start point s', the end point t', a boundary cost b. The steps of constructing y:

1. let $G' = G$
2. For each $(v_1, v_2) \in E$, construct an vertex u between them, construct edges (v_1, u) and (u, v_2) and delete edge (v_1, v_2)
3. For all vertices u, $v(u) = -w(v_1, v_2)$ and for all vertices v_1, v_2, $v(v_1) = v(v_2) = 0$
4. For each edge $e \in E'$, $w(e) = -1$
5. $b = -2|V|$
6. $s' = t' = s$

First we prove if x is yes then y is yes : If we find the cycle C of TSP, we claim that $p = C + \{u | u$ is between (v_1, v_2) and $(v_1, v_2) \in C\}$ makes y yes. c passes through all vertices so #edges in $p = 2|V|$ and $\sum_{e \in p} w(e) \leq b$. C minimizes $\sum_{e \in C} w(e)$ which means p maximizes $\sum_{u \in p} v(u)$. Therefore, y is yes.

Next, we prove if y is yes then x is yes: Assume p is the path, we claim that $C = p - \{u | u$ is between (v_1, v_2) and $(v_1, v_2) \in C \backslash p\}$ makes x yes. p satisfies $\sum_{e \in p} w(e) \leq b$ and passes through all vertices only once. This means that c contains all $v \in V$ and passes through all vertices only once. p maximizes $\sum_{u \in p} v(u)$ which means C minimizes $\sum_{e \in C} w(e)$. Therefore, x is yes.

In conclusion, we have proved that GroupTour is in NPC.

2.4 Convert to ILP

Further, using the definitions given above, we can formulate our optimization problem into an ILP problem. The number of edges in our graph is n^2 since every pair of nodes is connected. We let $x_{ij} = 1$ if we choose the v_j after choosing v_i, otherwise we let $x_{ij} = 0$. Assume we start at v_1 and end at v_n:

$$max\ \Phi(t_1(T), t_2(T), \cdots, t_m(T))$$

$$s.t. \quad \sum_{i,j} w(v_i, v_j) \cdot x_{ij} + \frac{1}{2}Cost(v_1) + \frac{1}{2}Cost(v_n) \leq B$$

$$\sum_{i=2}^{n} x_{1i} = 1$$

$$\sum_{j=1}^{n-1} x_{jn} = 1$$

$$\sum_{i=1}^{n} x_{ik} = \sum_{j=1}^{n} x_{kj} \leq 1, j \in \{2, 3, \cdots, n-1\}$$

$$x_{ij} = 0\ or\ 1, \quad i, j \in \{1, 2, \cdots, n\}$$

3 Algorithms

In this section, we present several algorithms to solve group tour optimization problem.

3.1 Enumeration

For enumeration algorithm, we enumerate every possible route for the group, check the feasibility of the route and choose one with the best result. Obviously, the running time grow exponentially as input size increases. Time complexity in the worst case can grow up to $O(n^n)$, which is intolerable in application. Hence, we use enumeration as one of our baseline algorithm to check for the best result.

Algorithm 1: Enumeration Algorithm

```
Max = 0;
TU = 2^{P\{s,t}};
for TS ∈ TU do
    for T ∈ Permutation(TS) do
        if Φ(T) > Max and Cost(T) < B then
            Max = Φ(T);
            T' = T;
        end
    end
end
return T';
```

3.2 Greedy Search

Greedy search algorithm is a heuristic approach. The main idea of this algorithm is to continuously appending a POI before the destination POI t and meanwhile replacing the POI in the route with another POI.

For the first step, we assume T is a feasible tour ($T = (s, t)$ from the very beginning). Consider POIs $p \notin T$, we try to append a POI that maximize the ratio of our optimization goal and total cost, which should not exceed the given time budget B.

For the second step, when no more POI can be appended to T and the final result can't improve any more, we then consider replacing a POI $p \in T$ with a POI $isp \in P \backslash T$ if this operation can make the

result better. Naturally, in this step, the total traveling time may decrease, which makes it possible for appending a new POI. So after each replacement, we go back to the first step to check if we can append a new POI. The algorithm terminates when we can neither append nor replace POIs.

Algorithm 2: Greedy Search (G,s,t,B)

```
T =< s, t >;
R = P\T;
Cost = 0;
// First Step;
while R is not empty do
    for p_i ∈ R do
        T' = append(T, p_i);
        Max = 0;
        if Φ(T')/Cost(T') > Max and newCost < B then
            T = T';
            Cost = newCost;
            tempP = p_i;
            Max = Φ(T')/Cost(T');
        end
    end
    if tempP exist then
        R = R\{p_i};
    end
    else
        break;
    end
end
// Second Step;
for p_i ∈ R and p_j ∈ T do
    T' = Swap(p_i, p_j);
    if Φ(T')/Cost(T') > Φ(T)/Cost(T) then
        T = T';
        cost = newCost;
        run Step1;
    end
end
return T;
```

Since local optimal doesn't necessarily mean the global optimal, this algorithm can't guarantee to achieve the best result. Since we append at most $(n-2)$ POIs in the route and each appending searches no more than n POIs, the time complexity of the first step is at most $O(n^2)$. For the second step, each replacement operation takes $O(n^2)$ on average. Because each valid replacement operation is followed by an appending operation, there are at most n replacements. Correspondingly, the time complexity of the second step is at most $O(n^3)$. Therefore, the total time complexity of greedy search algorithm is $O(n^3)$.

3.3 Best-First Algorithm

Best-First algorithm is based on the implementation of greedy algorithm described in last subsection. For this algorithm, we compute the best route for each member of the group by running greedy search on them. The greedy search algorithm we use here is the same as in last subsection, except that we change the optimization goal to the personal satisfaction. In this way, we will get m different routes.

Then we evaluate those routes for the group, which considers both sum and variance of satisfaction. Finally we choose the route with the maximal optimization result for the group. Finally, we run second step in greedy search to improve the route.

Algorithm 3: Best-First Algorithm(G,s,t,B)

```
Max = 0;
for p_i ∈ P do
    T' = GreedySearch(G, s, t, B, p_i);
    if Φ(T') > Max then
        Max = Φ(T');
        T = T';
    end
end
run step 2 in Greedy Search on T;
return T;
```

The time complexity for recommending for each member is $O(mn^3)$ and the evaluation is $O(mn)$. Therefore, the total time complexity is $O(mn^3)$.

4 Experiments

In this section, we will describe the detailed experimental settings and give analysis of the final result.

4.1 Datasets

Our datasets are provided by [1,2]. It contains POIs in eight cities, namely Buda, Delh, Edin, Glas, Osak, Pert, Toro and Vien, along with geographical information. Also, data extracted from TripAdivsor and user visits history are provided in the form of geo-tagged photos. The detail of the datasets is shown in table 2

Table 2. Dataset description.

City	#Users	#POI Visits	# Travle Sequences
Toronto	1395	39419	6059
Osaka	450	7747	1115
Glasgow	601	11434	2227
Edinburgh	1454	33944	5028
Perth	159	3643	716
Budapest	935	18513	2361
Delhi	279	3993	489
Vienna	1155	34515	3193

In our experiment, we only use the data in Dudapest, Osaka, Toronto and Vinenna since they are more complete than other cities.

4.2 Baseline Algorithm

We choose enumeration as our baseline algorithm, which returns the optimal solution but the running time is exponential. This is for checking how good the other algorithms perform. As for Greedy Search and Best-First algorithm, we test them and compare their result and running time.

4.3 Experiment Setup

There are several parameters users can choose. α, β, γ, δ determines the weight of POI's attributes on its popularity. η determines how much popularity and personal interest affects travel satisfaction. ω reflects how users balance overall satisfaction and fairness. $BaseCost$ and λ reflects how we estimate the visiting duration of a certain POI.

In our experiment, we choose those parameters as tabel 3 shows.

Also, we set total time budget as 6 hours as required.

Table 3. Parameters setup in our experiment.

α	β	γ	δ	η	ω	$BaseCost$	λ
0.25	0.25	0.25	0.25	0.5	0.2	0.5 hour	0.5 hour

4.4 Result and Analysis

The algorithm performance on our target city is shown in Fig.1, Fig.2 and Fig.3. From these figures, we prove that our algorithm performs well in comparison with the result of *Enumeration*. In most cases, *GreedySearch* and *BestFirst* can achieve results which is very close to the best result. And in general, the result of *BestFirst* algorithm is better than *GreedySearch*.

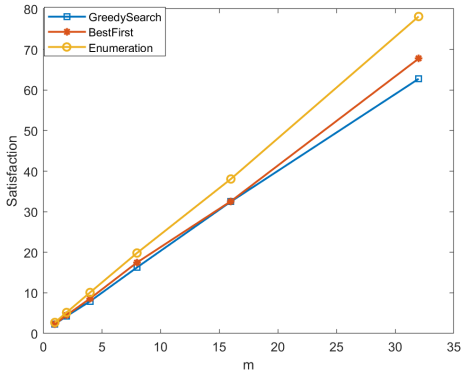

Fig. 1. Algo performance comparison in Buda

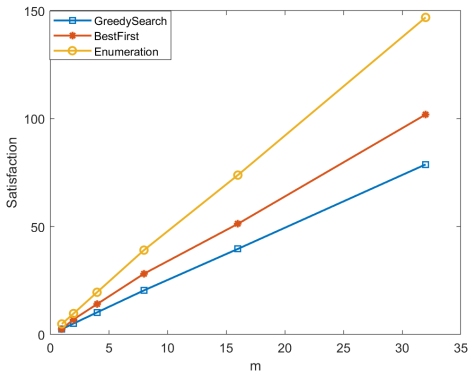

Fig. 2. Algo performance comparison in Toro

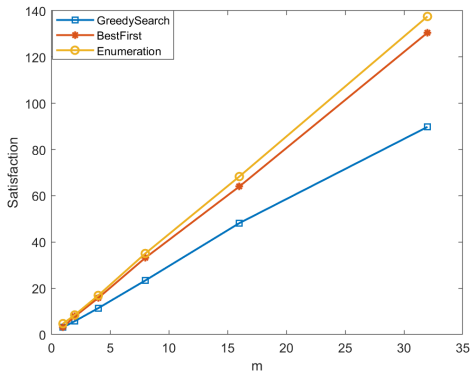

Fig. 3. Algo performance comparison in Vien

The running time of our algorithms is shown in Fig.4, Fig.6, Fig.7. For this particular dataset, the running time of *Enumeration* is not very large, which is not what we expected. In theory, the time complexity of *Enumeration* is n^n. We believe if we increase the number of POIs, the running time of this algorithm will be larger. For *GreedySearch* and *BestFirst*, the theoretical time complexity is $O(n^3)$ and $O(mn^3)$. The result also shows *BestFirst* is more time-expensive than *GreedySearch* when m is large enough, which grows linearly as m increases.

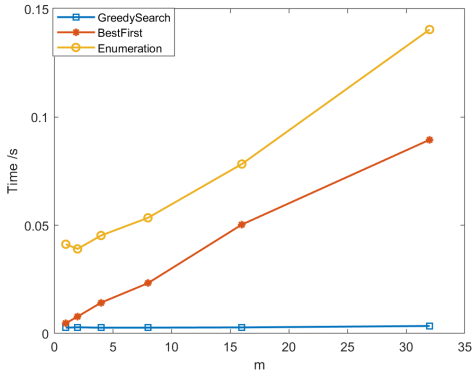

Fig. 4. Running time comparison in Buda

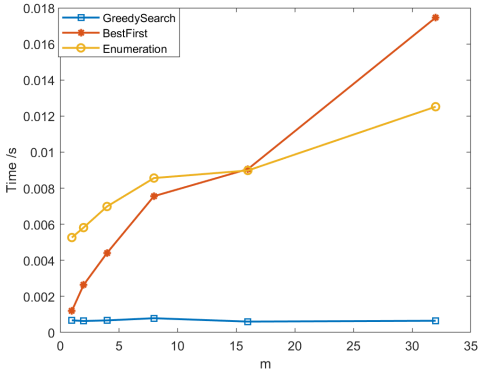

Fig. 5. Running time comparison in Toro

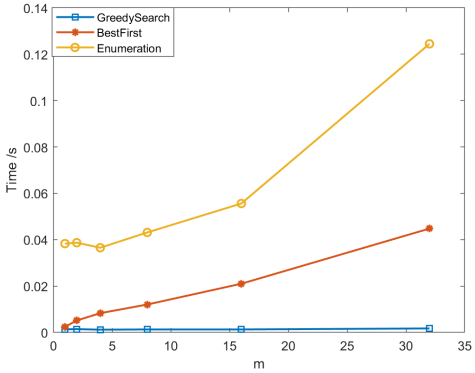

Fig. 6. Running time comparison in Vien

Also, we find the fact that as members number increases, average satisfaction suffers. The result is shown in Fig.8. The reason is that when there are too many tourists, it's harder to find a route satisfying each member and meanwhile keeping the variance low.

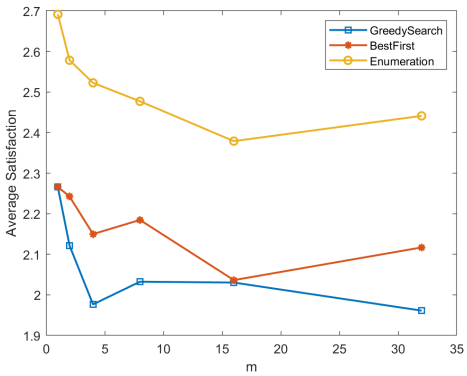

Fig. 7. Average satisfaction in Toro

5 Conclusion

In this article, we formulate the group tour recommendation problem as a variant of orienteering problem and prove that it's a NP-Complete problem. We estimate the POI visiting duration by using some POI attributes. And we get user's personal interest from historical geo-tagged photos. Then, we propose several algorithms, including baseline algorithms, Greedy Search and Best-First to solve the problem. Our experiment result shows the efficiency of our algorithms. And the recommendation result is very close to the optimal result. In the future, we can consider more constraints such as time window, weather condition, money budget and so on, which increases the hardness of this problem to a new level.

Acknowledgements

Group tour recommendation is a hard but interesting problem. Tour recommendation is a field that has been well studied by many data scientists. Our team read a lot of papers about tour recommendation and orienteering problem. Some of these papers are really helpful to us. We thank Kwan Hui Lim for providing us the data. Also, we thank TAs for instructing and giving us some useful advice in the wechat group. Finally, we thank Xiaofeng Gao for giving us the chance to learn.

References

1. Kwan Hui Lim, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Personalized Tour Recommendation based on User Interests and Points of Interest Visit Durations. In Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15). Pg 1778-1784. Jul 2015.
2. Kwan Hui Lim, Jeffrey Chan, Christopher Leckie and Shanika Karunasekera. Towards Next Generation Touring: Personalized Group Tours. In Proceedings of the 26th International Conference on Automated Planning and Scheduling (ICAPS'16). Pg 412-420. Jun 2016.
3. Pieter Vansteenwegen, Wouter Souffriau, Dirk Van Oudheusden. The orienteering problem: A survey. European Journal of Operational Research. Volume 209, Issue 1, Pages 1-10. Feb 2011.
4. Aris Anagnostopoulos, Reem Atassi, Luca Becchetti, Adriano Fazzone, Fabrizio Silvestri. Tour recommendation for groups. Data Mining and Knowledge Discovery (DMKD'17). Volume 31, Issue 5, Pages 1157-1188. Sep 2017.
5. Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, Grammati Pantziou. A Survey on Algorithmic Approaches for Solving Tourist Trip Design Problems. Journal of Heuristics. Volume 20, Issue 3, Pages 291-328. June 2014.