Generating Multi-Day Round Trip Itineraries for Tourists

Elif Erbil
elif.erbil@tum.de
Technical University of Munich
Garching bei München, Germany

Wolfgang Wörndl
woerndl@in.tum.de
Technical University of Munich
Garching bei München, Germany

ABSTRACT

Recommender systems facilitate the decision making process for users by selecting and ranking items according to users' interests. Travellers can highly benefit from recommender systems due to vast amount of information available regarding places to visit and many different factors that affect the travel plans. Travel recommender systems tackle both the issues of recommending relevant places and also sequencing them in a feasible order. However there are many different constraints that affect the recommendations for travellers such as travel dates, weather and companions, which makes the recommendation system more complex. In this paper, we present a novel approach to create multi-day trips that start and end at the accommodation of the user. We apply different clustering algorithms to tackle the issue of creating multi-day trips with balanced itineraries and conduct a user study to understand how our approach performed against baseline methods. Our results show that our algorithm performs better than other selected methods to recommend interesting points-of-interests to users and create appealing itineraries.

CCS CONCEPTS

• **Information systems** → **Users and interactive retrieval**.

KEYWORDS

recommender system, tourist trip design problem, points-of-interests, clustering, user study

1 INTRODUCTION

With the growing amount of data collected and presented to the user, finding relevant information gets harder every day. One approach to solve this problem and filter through the vast amount of information is using recommender systems. Recommender systems are tools and techniques to support users in finding products, services or information that are relevant according to the users' query, profile and context, and present them in an efficient way. One of the areas that can highly benefit from recommender systems is the travel industry [13], e.g. in recommending travel plans or routes composed of multiple points-of-interests (POIs). This problem is formulated as the Tourist Trip Design Problem (TTDP) [8], which aims at combining interesting POIs along a route to maximize the value for travellers.

TTDP addresses this issue in two parts: ranking and selection of POIs that might interest the user, and creating a feasible route using the selected POIs [16]. Recommender systems can use routing algorithms to create routes that include highly rated POIs while minimizing the distances in a suitable way for the user to follow. The route creation process gets more complex in a scenario with multiple travel days. In this paper, we investigate a novel approach for recommending routes with sequences of POIs for multi-day trips. Our approach uses the hotel location of the traveller as a start and end point for the generated routes, and groups the most interesting POIs for each day of the trip to create balanced routes with optimized walking distances.

The paper is structured as follows: Section 2 discusses related work in travel recommender systems and route creation algorithms, Section 3 explains the methodology followed to create multi-day, round trip itineraries using recommender systems and different algorithms used for comparison, Section 4 discusses results from an offline study for the mentioned algorithms according to certain criteria, whereas Section 5 extends these results with a user study to evaluate the success of each algorithm. Section 6 concludes the paper with the possible future work and final remarks.

2 RELATED WORK

Different approaches for TTDPs have been proposed to create feasible sequences of POIs. Gavalas et al. [8] proposes different algorithmic approaches for different cases of TTDPs. Similar to Souffriau et al. [14], by modelling TTDPs as a variant of the Orienteering Problem (OP), the authors extend these to different cases such as single day tours using the Travelling Salesman Problem with Profits. The authors propose algorithmic approaches to solve further constraints by using different variants of the OP. For example, the system models travelling multiple days as Team OP (TOP) and formulates working hours of POIs as OP with Time Windows (OPTW). Another research by Garcia et al. [6] focuses on recommending trips to users while bringing together different constraints as an instance of the Time Dependent Team Orienteering Problem with Time Windows (TDTOPTW) and solves the problem in real time using heuristics.

Kurata and Hara [11] use the Selective Travelling Salesman Problem as a model to create single day route recommendations to users. Research by Wörndl et al. [15] uses a variant of Dijkstra's shortest path problem to recommend single day walking tours for users with the given start and end points. The mobile single day route recommender by Anacleto et al. [2] uses a lightweight solution based on decision trees for a route creation process on mobile devices. Deitch et al. [3] models the problem of creating itineraries with attractive routes between POIs to be visited as bus-touring problem (BTP). The BTP is similar to the orienteering tour problem and extends the OP to have the same start and end points, therefore allowing round trips. Gavalas et al. [7] tackles the same issue by modelling the problem as mixed team orienteering problem with time windows and solving it using iterated local search heuristics.

3 CREATING MULTI-DAY ITINERARIES

In this section we discuss our novel approach for creating multi-day round trip itineraries. We propose to first cluster POIs that are within the walking distance to the hotel for different days and then run a round trip routing algorithm to create the itineraries for each day. We compare our approach against a simple baseline algorithm.

3.1 Creating Round Trips

In order to create round trips starting and ending at the hotel, the model is defined as a travelling salesmen problem with profits model [4]. The traveller has a start location and each POI that is added to the route brings a certain amount of profit, whereas walking between POIs and visiting them has a cost. So the profit must be maximized in order to have the most efficient route with high rated POIs. Since TSP with Profits is NP-hard and it is computationally inefficient to find an exact solution, we use the algorithm proposed by [12] to approximate a solution for the given problem. [12] states that the problem can be solved by selecting the most profitable POI in a greedy fashion by selecting the POI with highest $score/distance$ ratio towards the two ends of the route and adding them accordingly. We have modified the following algorithm in three ways:

- changing the profit criteria to increase the weight of the profit of the score to ensure that the algorithm will prefer the higher rated POIs instead of lower rated but closer ones
- adding the visiting times of each POI to the cost of the POI as well as the time needed to travel to reach to the POI
- re-adjusting the sequence of POIs using the 2-opt algorithm for further optimization [5]

The 2-opt algorithm is used to optimize the solution after each POI is selected so that it creates a shorter route that includes the same POIs, if there is a better route sequence that can be created by eliminating crossing routes (see Figures 1 and 2 for an example). The idea behind 2-opt is to switch places of POIs if the total distance of the new alignment is less than the original one. For each POI, POIs i and j are swapped if:

$$dist(i, i+1) + dist(j, j+1) > dist(i, j) + dist(i+1, j+1)$$

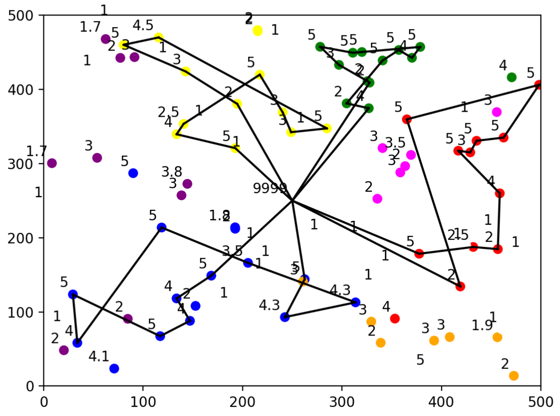


Figure 1: Initial Route Created by the Greedy Algorithm

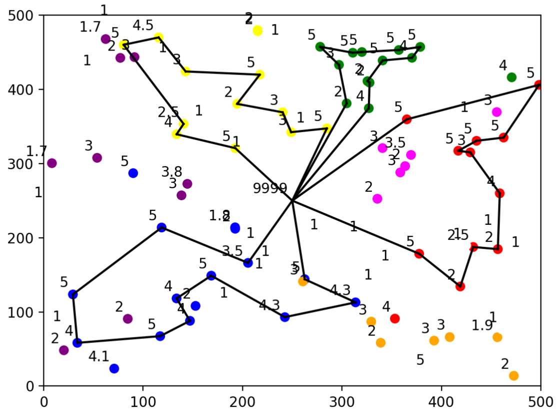


Figure 2: Routes in Figure 1 Optimized by 2-opt Algorithm

3.2 Creating Multi-Day Trips

In order to create multi-day trips, the POIs must be distributed among each day so that the routes created will have unique POIs that are selected to create routes with most profit. Therefore while creating POI lists for each day we need to take the following into account:

- each day must have enough POIs to fill the time limit that is set by the traveller,
- each day must have POIs that are rated highly by the traveller,
- the POIs must be clustered in a way that closer POIs recommended must be in the same day.

3.2.1 Baseline algorithm. While creating multi-day trips, the baseline algorithm that is selected to create multiple round trips is running the algorithm specified for creating round trips n times, n being the number of days. For each run, the POIs that are used in the previous days will be removed from the POI list. This allows to get a baseline since this way the algorithm is able to take all POIs into account and therefore create itineraries that have closer POIs together and allows to create itineraries with multiple tours to the same location if the high rated POIs are clustered in the same location within the city. Although the itineraries are expected to include many POIs and high ratings, there are two disadvantages of the baseline approach that is expected:

- The baseline algorithm takes into account all of the POIs rated by the user and if a POI with a lower score is much closer than a distant high rated POI, a selection of lower rated POIs might be favored over higher rated ones.
- The baseline algorithm needs to go over all of the POIs to decide the highest profit ones, therefore running the round trip algorithm n times will have a longer runtime which is not efficient for the mobile users with lower computation power and limited battery. Consequently, the baseline algorithm

will be compared to certain clustering algorithms in terms of both runtime and itinerary success criteria.

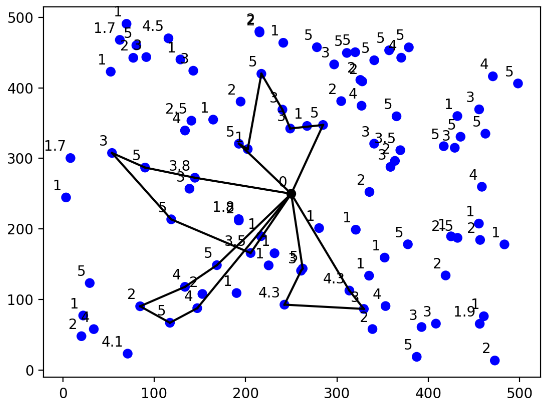


Figure 3: Sample 4 Day Trip with Random Data Points using Baseline Approach

3.2.2 *k-means Clustering.*

Secondly, we used the k-means algorithm to cluster these items. k-means allows to cluster POIs that are closer to each other by starting with k random clusters (where k is the number of days for the trip) and then iteratively reassigning each POI to the cluster that is the closest one [10]. By this approach, the round trip algorithm will be run on each of the clusters created which is expected to decrease the runtime. However, k-means algorithm might not behave optimally in the following cases:

- k-means clusters POIs according to their distances, therefore for the cities where most of the POIs are gathered in a close perimeter and a few are outside this perimeter, the clusters might not have equal distribution of POIs for each day.
- Higher rated POIs that might be preferred over lower rated ones might be bundled into one cluster, therefore the ratings of itineraries might be lower.

3.2.3 *Time-Limit Approach.*

In order to eliminate the problems that might be encountered with the above approaches, we have developed a novel approach called time-limit approach. Our novel approach uses agglomerative clustering as a starting point since it similarly follows a bottom-up fashion to bring closer POIs together in the same clusters [1]. Agglomerative clustering yields a similar solution as k-means since it creates clusters with a single POI and merges these clusters that are closest to each other together until there is expected number of clusters [9]. However, these clustering algorithms create a fixed number of clusters rather than dividing elements in a balanced way, therefore the number of items in each cluster may not be equal. In our system, we aim to have balanced clusters so that the users can visit more POIs each day. Therefore, instead of fixing the number of clusters, our approach has a constraint on each cluster where the sum of the visiting durations of the POIs within that cluster does not exceed the visiting duration assigned to a day. So the POIs closer to each other are added to a

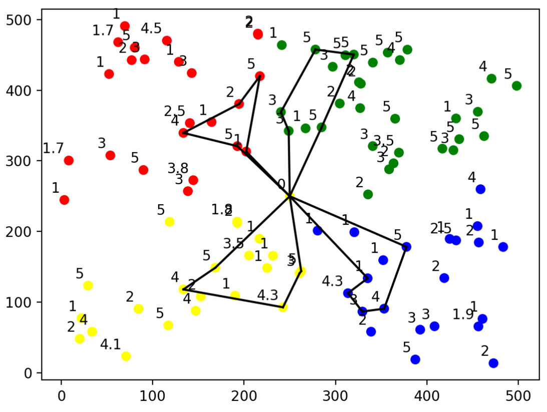


Figure 4: Sample 4 Day Trip with Random Data Points using k-means Approach

cluster until the time limit is reached at that cluster or the number of clusters created is equal to the number of days (which might be the case if there are not enough POIs or the number of days are too high). Since the user is expected to spend a certain time at each POI, it can be assumed that the POIs in the cluster will be sufficient for a day trip, as there is also the travelling time between these POIs that must be accounted for. By this way, each cluster will have POIs that are close to each other and each day will have roughly the same number of POIs assigned. This idea is followed by the assumption that in most of the cities, POIs are clustered densely at the city centers and it might not be feasible to visit all the POIs in the center in one day. Therefore, this approach allows to divide these POIs in multiple days so that the traveller would not miss them if they are preferred over POIs that are outside the city, which would be in a different cluster due to their distance.

The second aim is to have higher rated POIs added to each cluster. In order to create balanced itineraries for each day, we perform pre-processing on the POI data to select the higher rated POIs over the ones that are closer to the clusters. In order to do the pre-processing, the recommender systems first predict ratings for each POI by matching the user profile and information collected about POIs. One way the recommender system can predict ratings for the user is proposed by [15], which assigns venues to different categories and predicts a score by using the ratings on Foursquare. The recommender system then eliminates POIs that have a low score according to other users' ratings as well as traveller's category preferences. After the ratings are obtained, each POI is assigned to three groups according to their ratings: must-visit, can-visit or don't-visit. Since each user might have a different rating scale, instead of assigning fixed ratings to separate POIs into these groups a different approach is followed. Must-visits are POIs that have ratings one standard deviation above the mean of the predicted ratings of the user for all POIs. These POIs are added to the clusters first. Don't-visits are POIs that have ratings one standard deviation below the mean rating of the user for all POIs and these POIs are discarded by the system before the clustering algorithm. This ensures lower rated POIs that are close to others are not preferred over higher rated ones. The rest of the POIs are marked as can-visits, which are added to the itinerary if there is time left after adding all the must-visits. Therefore, the lowest rated POIs are eliminated and there is sufficient amount of must-visits that can be added to the itineraries.

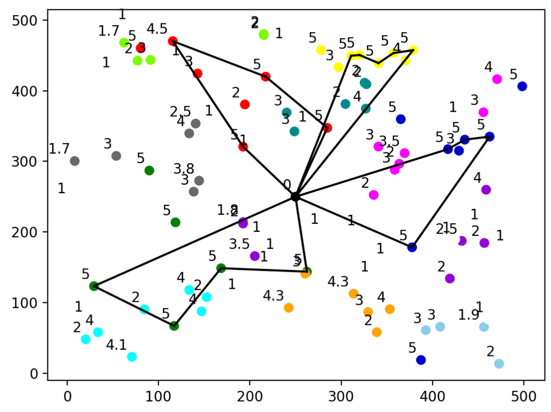


Figure 5: Sample 4 Day Trip with Random Data Points using Time Limit Approach

After grouping the POIs according to their relevance to the user, the clustering algorithm is first run on must-visit POIs to create clusters. Then the algorithm is re-run using the clusters created in first step and the can-visit POIs to ensure that each cluster have enough POIs to create routes while having higher rated ones preferred over others while increasing the diversity of the tour and keep the distances between them as low as possible. After creating the clusters, the top n clusters with highest average rating is selected to create round trips.

Figures 3,4 and 5 show 4-day itineraries created with the baseline, k-means and time-limit approaches respectively. It can be seen that time-limit approach has more clusters and less data points assigned to clusters as low rated POIs are discarded beforehand.

4 OFFLINE EVALUATION

In order to evaluate the performance of each approach, we first compare the algorithms in an offline analysis. To do so, we generate routes with the three algorithms using a dataset with 160 POIs and hotels in Istanbul. The hotels and POIs were selected from the attractions that are listed on Tripadvisor. We use different metrics to better understand the characteristics of the routes created by each algorithm: mean of average ratings, coefficient of variation of average rating and runtime of the algorithms.

4.1 Mean of Average Ratings

For each number of days, we calculate the mean of the average ratings. For a trip with n days, where each day i has k_i POIs selected, we sum up the ratings of the POIs of a day, divide the number by k_i and compute the overall mean. Mean of average ratings ensures that the ratings of the POIs of each proposed route are as high as possible.

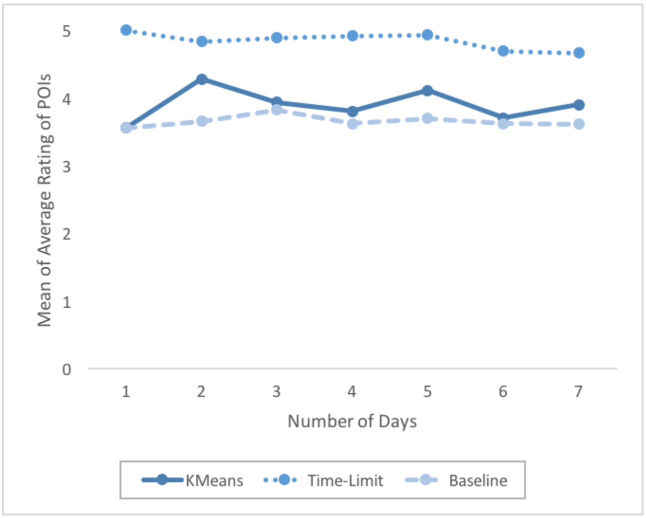


Figure 6: Mean of Average Rating of Routes for Different Number of Days per Route

The results presented in Figure 6 show that the time-limit approach has routes with the highest number of average ratings per day compared to the baseline and k-means approaches. This is due to the fact that the POIs are first grouped to create clusters that maximize user satisfaction by pre-filtering for highly rated POIs, which is absent in the other approaches. Therefore by the time-limit approach, POIs that have lower scores but which were closer to other POIs were favored over the ones that are higher rated but farther away during the route creation process to ensure more POIs can be added. The results are consistent with increasing number of days. The means are decreasing with more days, which is expected because more POIs with lower ratings have to be included in the routes.

4.2 Coefficient of Variation of Average Ratings

For each number of days, we calculate the coefficient of variation of the average ratings. For a trip with n days, where each day i has k_i POIs selected, the coefficient of variation CV of average ratings is calculated as:

$$CV = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\bar{x_i}-\bar{x})^2}}{\bar{x}}, \ where\ \bar{x_i} = \frac{\sum_{p=1}^{k_i} rating_p}{k_i}$$

This measure ensures that each day will have a similar distribution of highly rated POIs. The coefficient of variation is selected instead of standard deviation so that the amount of change can be compared across different algorithms with different means.

The coefficient of variation of average ratings of POIs presented in Figure 7 shows that for most of the cases, the time-limit approach had a lower variance in average rating than the baseline and k-means approaches. This can be interpreted as that the average ratings of each day in a route is similar and POIs that are more preferred by the user are distributed among different days since each cluster is limited with an upper limit. Therefore higher rated POIs are added to different clusters when the upper limit for a cluster is reached and a more even distribution is obtained. However, the coefficient of variation is relatively low for all samples.

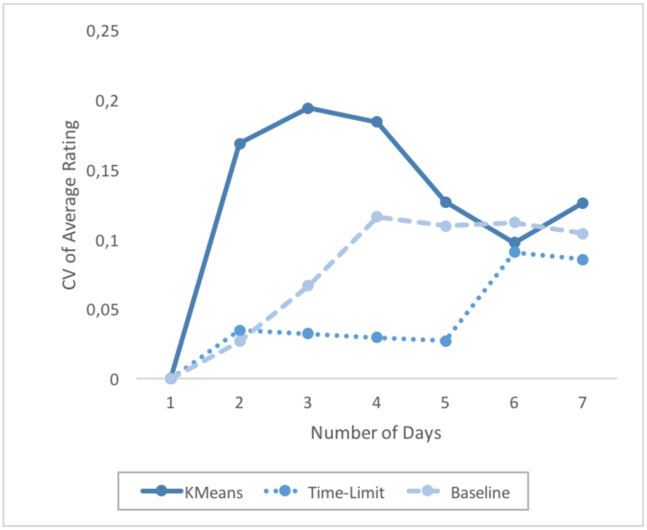


Figure 7: Coefficient of Variation of Average Rating of Routes for Different Number of Days per Route

4.3 Runtime

We also investigated the runtime of each clustering algorithm as well as the route creation algorithm. We used an Intel Core i7 (3.1 GHz) processor for the performance analysis. In order to get reliable results, the runtime was calculated as the average of 1000 runs.

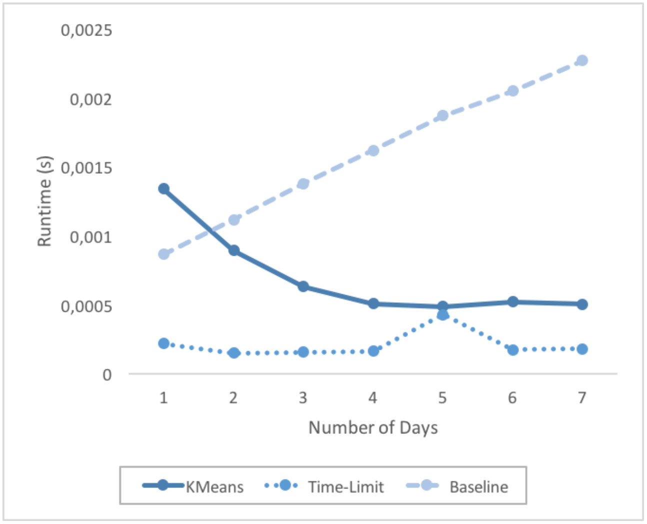
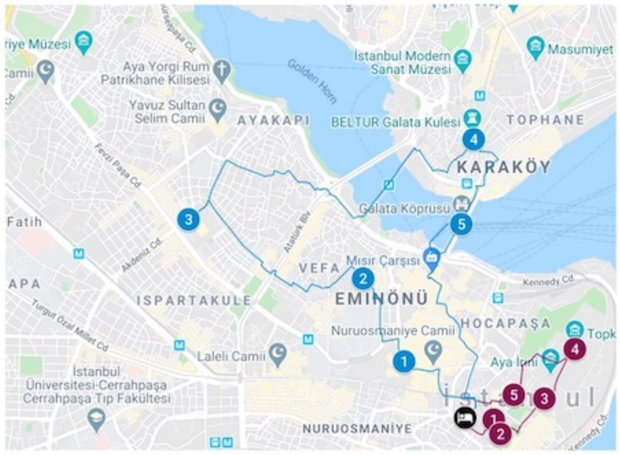
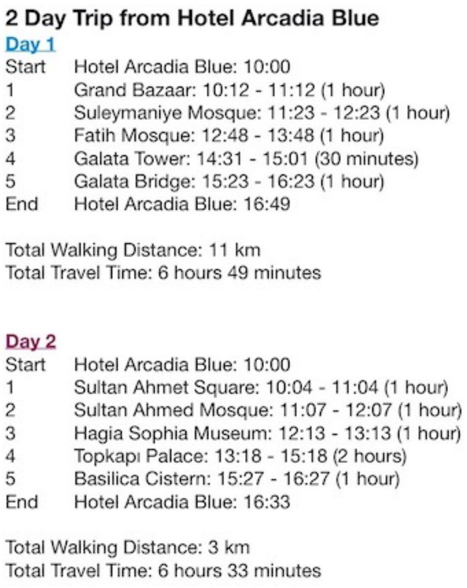


Figure 8: Runtime of Each Algorithm for Different Number of Days per Route

As stated before, one of the main drawbacks of the baseline algorithm is its runtime, as it executes the greedy approach on the whole dataset for each day of the route. The results in Figure 8 show that as the number of days increases, the number of comparisons and therefore the runtime increases linearly for the baseline algorithm. However, the k-means and time-limit approaches use clustering first and then run the greedy algorithm on these smaller clusters. The time-limit approach constrains the size of the clusters, therefore the input for the route creation algorithm was limited to a smaller number of POIs for each day. For the k-means algorithm, the runtime decreases as the number of days increases, because the number of POIs in each cluster decreases with more days and therefore the route creation algorithm takes less time overall.




2 Day Trip from Hotel Arcadia Blue

Day 1

Start	Hotel Arcadia Blue: 10:00
1	Grand Bazaar: 10:12 - 11:12 (1 hour)
2	Suleymaniye Mosque: 11:23 - 12:23 (1 hour)
3	Fatih Mosque: 12:48 - 13:48 (1 hour)
4	Galata Tower: 14:31 - 15:01 (30 minutes)
5	Galata Bridge: 15:23 - 16:23 (1 hour)
End	Hotel Arcadia Blue: 16:49

Total Walking Distance: 11 km
Total Travel Time: 6 hours 49 minutes

Day 2

Start	Hotel Arcadia Blue: 10:00
1	Sultan Ahmet Square: 10:04 - 11:04 (1 hour)
2	Sultan Ahmed Mosque: 11:07 - 12:07 (1 hour)
3	Hagia Sophia Museum: 12:13 - 13:13 (1 hour)
4	Topkapı Palace: 13:18 - 15:18 (2 hours)
5	Basilica Cistern: 15:27 - 16:27 (1 hour)
End	Hotel Arcadia Blue: 16:33

Total Walking Distance: 3 km
Total Travel Time: 6 hours 33 minutes

Figure 9: Sample Two-Day Trip Created with Istanbul Dataset from the Questionnaire

5 USER STUDY

The offline results give an insight about how each algorithm behaves to create routes with preferable POIs. In order to evaluate the different algorithms from a user's perspective, we conducted a preliminary user study with the Istanbul dataset. For mitigating the effect of different user preferences and limitations of the dataset, each user is presented with predefined routes where each POI category was given a static rating. POI categories are created similarly to the categories proposed in [15]. POIs that are in the top 20 recommended attractions in Tripadvisor for Istanbul were given higher ratings than all other POIs. A sample route created using the algorithms can been seen in Figure 9.

To evaluate the behavior of different algorithms with a different number of days, each algorithm is used to create a two-day and three-day trip from selected hotels. To decrease the effect of the hotel locations on efficiency of routes, we selected two different hotels in different locations. In total, each user was presented with 12 routes. For each route, the user was asked to answer the following questions by rating each option ranging on a Likert scale ranging from strongly disagree (1) to strongly agree (5):

(1) There is an adequate number of points-of-interests for each day
(2) Points-of-interests are distributed evenly among each day
(3) An adequate amount of time is spent on visiting points-of-interests rather than travelling between them
(4) The points-of-interests recommended are interesting
(5) I'm satisfied with the overall recommendation

In total, the questionnaire was filled out by 22 people (50% female, 50% male). The user study participants were identified by sharing the questionnaire on social media and among acquaintances. In order to understand the success of the routes, participants were chosen among people who have lived in Istanbul and who thus have knowledge about the city and the presented POIs. The distribution of the ages of the participants were 0-20 (13.6%), 21-30 (59%), 31-40 (4.5%), 41-50 (4.5%), 51-60 (9%) and 61-70 (4.5%). Figures 10 and 11 show the results of each algorithm for each question for two-day and three-day trips respectively. For two-day trips the time-limit approach performed better than both other algorithms in overall recommendations (ø: 3.70, σ: 1.06), POIs are distributed evenly among days (ø: 3.80, σ: 1.16) and recommended POIs are interesting (ø: 4.32, σ: 0.79). For three-day trips, the time-limit approach performed better than other algorithms in all five categories.

In order to understand if the time-limit algorithm is significantly better than other algorithms, we performed a one-way ANOVA test. The results of the ANOVA test with the confidence interval of $\alpha = 0.05$ shows that for two-day trips the POIs recommended are significantly more interesting for the time-limit approach than both the baseline and k-means approaches, whereas the rest didn't yield results significant enough for comparison. For three-day trips the ANOVA results shows that both the POIs are significantly more interesting and also the overall recommendations are significantly more appealing for time-limit approach than both the baseline and k-means approaches. Also for the three-day trips, the POIs are distributed more evenly among each day by the time-limit approach compared to k-means approach but not significant enough to compare to the baseline algorithm.

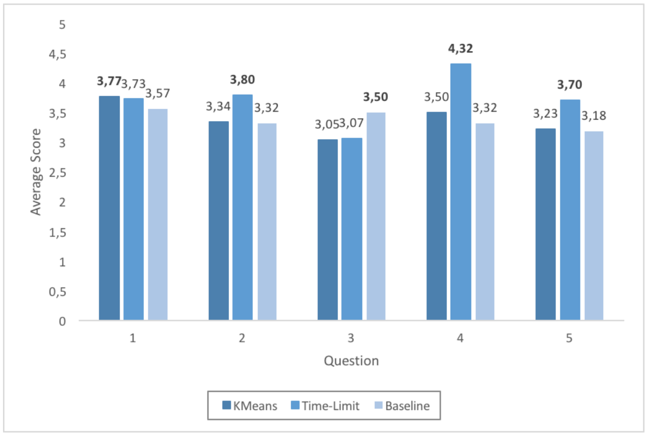


Figure 10: Questionnaire Results for Two-day Trips

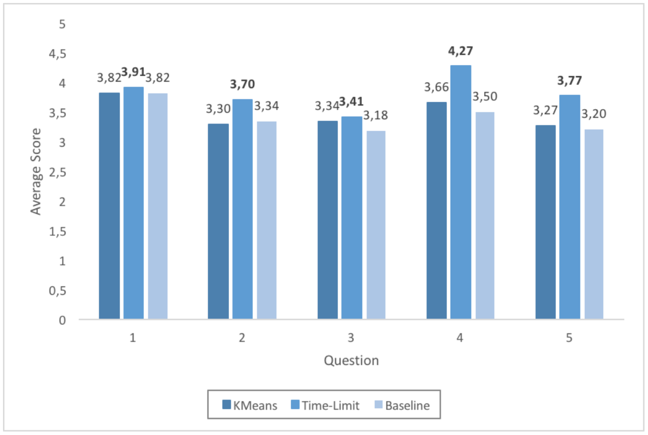


Figure 11: Questionnaire Results for Three-day Trips

6 CONCLUSION AND FUTURE WORK

In this research, we propose a new route creation process for multi-day round trips, starting and ending at the accommodation of the user. Our approach tags each POI according to their importance for the user, then clusters the POIs to days by limiting each cluster with the maximum amount of time the user spends on travelling per day, starting from the POIs with the highest importance to the user. By doing so, the top clusters that include high importance POIs that are closer to each other are selected. One assumption of the algorithms is that there are enough places highly rated by the user to create routes for the given number of days for the route. By creating clusters and running the round trip algorithm, which is a greedy approach that gives an approximate solution for the orienteering problem, both the runtime is decreased and POIs with higher importance are prioritized over POIs closer to the accommodation. The proposed solution allows parallelization of the route creation process, which can further increase the performance of the algorithm.

According to the offline results and the user study, the selection of recommended POIs for the route and the overall recommendation quality of the proposed time-limit approach is better than the recommendations given by the baseline algorithm and k-means clustering. However, the current approach follows a very simple method in terms of rating and ranking POIs and was not personalized to user interests, which is also a factor that needs to be taken into account. For the future work, the route creation algorithm can be combined with a recommendation algorithm to create personalized scores for the POIs. By this way the algorithms can be tested by a more extensive user study with personalized routes.

Currently, the algorithm only takes user ratings into account when clustering POIs and creating routes. In the next step, the working hours of the POIs can be incorporated into the routing algorithm as well the clustering algorithm, so the user can visit these POIs at the suggested times. Also this could be used to further customize trips such as adding restaurants to routes for lunch time or creating certain breaks for the users between POIs. Another possible addition is using the context information to rank POIs or create different routes according to different contextual factors.

REFERENCES

[1] Marcel R. Ackermann, Johannes Blömer, Daniel Kuntze, and Christian Sohler. 2012. Analysis of Agglomerative Clustering. *Algorithmica* 69, 1 (Dec 2012), 184–215. https://doi.org/10.1007/s00453-012-9717-4

[2] Ricardo Anacleto, Lino Figueiredo, Ana Almeida, and Paulo Novais. 2014. Mobile Application to Provide Personalized Sightseeing Tours. *Journal of Network and Computer Applications* 41 (May 2014), 56–64. https://doi.org/10.1016/j.jnca.2013.10.005

[3] Ray Deitch and Shaul P. Ladany. 2000. The one-period bus touring problem: Solved by an effective heuristic for the orienteering tour problem and improvement algorithm. *European Journal of Operational Research* 127, 1 (2000), 69–77. https://doi.org/10.1016/S0377-2217(99)00323-9

[4] Dominique Feillet, Pierre DEJAX, and Michel Gendreau. 2005. Traveling Salesman Problems With Profits. *Transportation Science* 39 (05 2005), 188–205. https://doi.org/10.1287/trsc.1030.0079

[5] Croes G. A. 1958. A Method for Solving Traveling-Salesman Problems. *Operations Research* 6, 6 (1958), 791. https://doi.org/10.1287/opre.6.6.791

[6] Ander Garcia, Olatz Arbelaitz, Maria Teresa Linaza, Pieter Vansteenwegen, and Wouter Souffriau. 2010. Personalized Tourist Route Generation. In *Current Trends in Web Engineering*, Florian Daniel and Federico Michele Facca (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 486–497. https://doi.org/10.1007/978-3-642-16985-4_47

[7] Damianos Gavalas, Vlasios Kasapakis, Charalampos Konstantopoulos, Grammati Pantziou, and Nikolaos Vathis. 2016. Scenic route planning for tourists. *Personal and Ubiquitous Computing* (10 2016). https://doi.org/10.1007/s00779-016-0971-3

[8] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, and Grammati Pantziou. 2014. A Survey on Algorithmic Approaches for Solving Tourist Trip Design Problems. *Journal of Heuristics* 291, 20 (2014). https://doi.org/10.1007/s10732-014-9242-5

[9] Anil Jain and Richard Dubes. 1988. *Algorithms for Clustering Data*. Vol. 32. https://doi.org/10.2307/1268876

[10] Xin Jin and Jiawei Han. 2010. *K-Means Clustering*. Springer US, Boston, MA, 563–564. https://doi.org/10.1007/978-0-387-30164-8_425

[11] Yohei Kurata and Tatsunori Hara. 2013. *CT-Planner4: Toward a More User-Friendly Interactive Day-Tour Planner*. 73–86. https://doi.org/10.1007/978-3-319-03973-2_6

[12] Gilbert Laporte and Silvano Martello. 1990. The selective travelling salesman problem. *Discrete Applied Mathematics* 26, 2 (1990), 193 – 207. https://doi.org/10.1016/0166-218X(90)90100-Q

[13] Francesco Ricci. 2002. Travel Recommender Systems. *IEEE Intelligent Systems* 17, 6 (Nov. 2002), 55–57.

[14] Wouter Souffriau, Pieter Vansteenwegen, Joris Vertommen, and Greet Vanden Berghe. 2008. A Personalized Tourist Trip Design Algorithm For Mobile Tourist Guides. *Applied Artificial Intelligence* 22 (10 2008), 964–985. https://doi.org/10.1080/08839510802379626

[15] Wolfgang Wörndl, Alexander Hefele, and Daniel Herzog. 2017. Recommending a Sequence of Interesting Places for Tourist Trips. *Information Technology & Tourism* 17, 1 (01 Mar 2017), 31–54. https://doi.org/10.1007/s40558-017-0076-5

[16] Wolfgang Wörndl. 2016. Solving Tourist Trip Design Problems from a User's Perspective. In *Mensch und Computer 2016 – Workshopband*, Benjamin Weyers and Anke Dittmar (Eds.). Gesellschaft für Informatik e.V., Aachen. https://doi.org/10.18420/muc2016-ws05-0003