International Journal of Data mining Management Systems (IJDMS) Vol.1, No.4, October 2022.

TRAVEL ITINERARY CREATION BASED ON DATA TECHNIQUES

Divya Mereddy

Department of Computer Science, University Of Cincinnati

ABSTRACT

An optimized itinerary plan for visiting n number of points in a tourist trip can be obtained by developing the Hamilton clustering system on top of N points by considering K numbers of NN with calculating the reachability distance between points. Please find Rab!=Rba. Which decides which point should come first. This model is developed based on the Hamilton graph/ Salesman travel problem algorithm, and HDBSCAN clustering along with considering tourist constraints like limited total spent time per day, estimated time to visit the place, etc. While this algorithm is great for a multi-day trip, it is especially quite useful and makes the process easy, if someone wants to cover a great number of points in a trip in less number of days or when someone is trying to go for a world tour / big tour with a considerably high number of visiting points.

KEYWORDS

Hamilton Graphs, Clustering, DBSCAN, Traveling Systems, Itinerary

1. INTRODUCTION

Tourism is an industry that drives people to travel for recreation and leisure. Tourists are people who travel away from their homes for pleasure. Tourism is one of the fastest-growing industries in the world and it generates a lot of jobs.

In 2013, a total of 1.087 billion people travel to another country as tourists. This has increased from just 25 million in 1950. Tourism is improving day to day throw-out the world. people are showing more interest in exploring places and entertainment than before. With this increasing trend, the demand definitely increased for tourist guides and apps. Tourism is highly labour-intensive and provides a high volume of jobs for low-skilled workers, together with higher-skilled jobs.

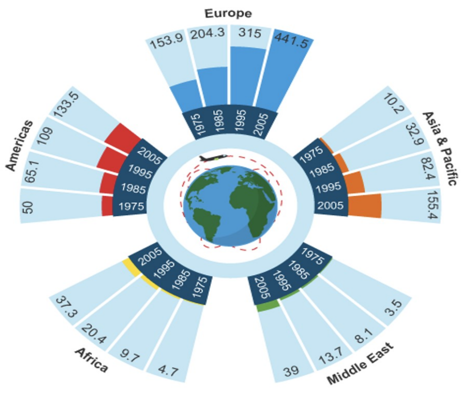

Fig. 1 Increase in world tourism 1975 -2005 (millions)

While the industry is growing very drastically, manual labor services or technology services have not grown at that speed especially, tourist guide services and technical resources like itinerary creation, automatic travel bookings etc. Still, tourists are depending on travel agencies which are very expensive or manual plan creation, doing everything on their own by spending an enormous amount of time and effort researching about tourist places and travel arrangements along with the chance of getting misled because of lack of information available about expected visiting places, etc.

Also, when a person wants to visit a place or travel to a country. Tourist companies offer packages to show multiple famous places and take care of transportation. Some organizations also provide adjusted services for activities as well. But most organizations, plan basic visiting places or activities list irrespective of the tourist interest variation. None of the services are customized for different groups of tourists. So they cannot have a customer experience. Most of the time, tourists only have limited knowledge and low awareness on the destinations they visit. They do have different needs and characteristics. Developing crowd-sourced applications by using tourists' input could give valuable insight to destinations in capturing tourists' demand and tourist complaints in a timely manner. Our system helps the customer to choose the visiting places based on their interest among the total famous options available which makes the plan customized for the tourist. In addition, as discussed while the number of tourists is increasing drastically, tourist labor is not increasing at that rate, tourism is becoming costly and less comfortable for people. Our aim is to resolve these issues and create a hassle-free customized self-guided travel experience for the 21 th century tourists using data techniques.

In our work, we utilized advanced data collection ML techniques and advanced data sources for the specific APIs for data gathering. Our work is novel in multiple ways. First of all, developing a complete automatic data-based itinerary is quite new. Also, while the first method is developed using new techniques based on a combination of sale man traveling solutions with clustering techniques, the second method is based on utilizing Hamilton graphs in an innovative way. The last method is a completely different type of clustering method based on DBSCAN, HDBSCAN, and Hamilton graphs.

2. Research In Industry

Some organizations are already researching this topic. But no one has developed a system to give a full-fledged itinerary plan for a visiting place. As a part of our research, we have developed an app to provides complete itinerary for visiting famous places in a location based on your interest and other circumstances like weather and scheduled days of the visit.

Existing researches:

1. Google has the famous google distance API to provide the time required to travel through multiple locations. But this system still has limitations to combining famous places with travel time and estimated visiting time of places.

2. Google and trip advisor also have famous places data and they are providing it to users through their portals and APIs. But most organizations don't have data about estimated visiting time for all tourist points.

3. Google is also providing similar traveling between locations problems but for a different use case like a salesman travel problem. Which doesn't solve the customized travel itinerary creation problem perfectly.

4. Some organization has developed a machine learning model to extract famous visiting places based on the most number of pictures taken. Trip Advisor and some other organizations are providing similar services or partial services like providing a list of visiting places etc.

In our research work, we tried to create a full-fledged automatic system using ML techniques like extraction of famous places, finding the travel time, finding the estimated time to visit the place based on user type, and optimizing the travel itinerary. This paper concentrates on the itinerary creation part using different data sources.

3. Data Collection

Below is a high-level description of our data sources.

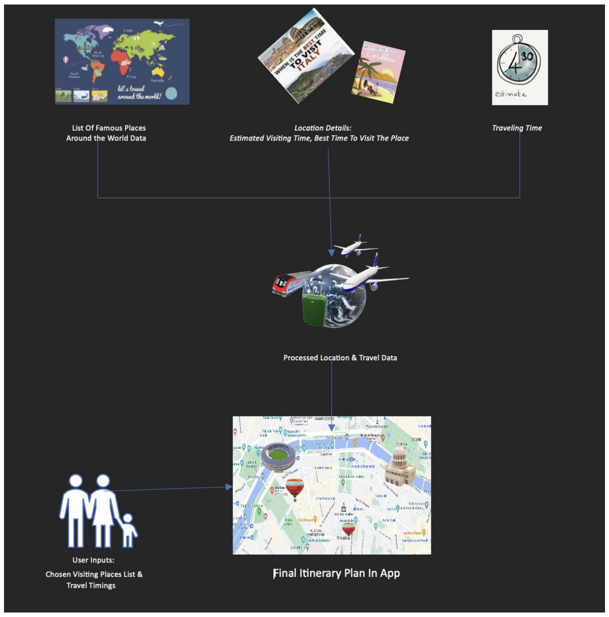
List Of Famous Places Around the World Data
Location Details: Estimated Visiting Time, Best Time To Visit The Place
Traveling Time
Processed Location & Travel Data
User Inputs: Chosen Visiting Places List & Travel Timings
Final Itinerary Plan In App

Fig. 2 Data Sources And Process Flow

3.1. Find Famous Places Around The World

Publicly available photos from different viewpoints and places with geoinformation were used to find famous places based on the paper [1]. For the places with less data, we tried to extract data from different APIs like google things to do, trip advisor API, etc. Through our app, we are also tracking the location data through traveler's routes and famous places visited by the tourist along with time spent by the people.

2.2. Location Details: Estimated Visiting Time, Best Time To Visit The Place :

We tried to extract the best time to visit the places, open hours, and estimated time to visit the places using different APIs like google business hours of an organization, and Google & trip advisor's expected time details. Our team tour guides have also done manual research on this topic. We are also collecting this information from our app users' activity.

2.3. Get traveling time on that particular day

We used google maps to extract the place's location and traveling time between locations for the current time and planned travel time for that location. For example, if location A is an evening visiting time point, then we predict the travel time to a place in the evening time.

3. EXPERIMENTATION WITH DIFFERENT METHODS

Using different data sources(mentioned above), we developed multiple systems to create an itinerary based on different techniques. we provide all the suggested places based on the reviews count and ranking we created in our app as the first step. once the tourist chooses the places they want to visit among the listed famous places, we finalize that as the list of places to visit in our algorithm.

We tried to utilize the existing systems combination methods and new clustering techniques based on HDBSCAN and Hamilton graphs systems. Below are the results we got from them and the methodologies.

In every method, our goal is to reduce the traveling distance between these places provided every one of a cluster's expected visiting time and travel time is less than K hours (K - customer-provided visiting time per day). Our work optimization function is d1V1+d2V2+d3V3.. dnVn= K. here d1,d2,d3… dn are distances that have to be travelled between locations, and V1, V2,.. Vn are expected visiting time of places. Among all the points we tried to start our visit and end our visit from end points which are predicted based on below logic.

Choosing endpoints:

The point of the travel path is the one whose distance from other points' cumulative sum follows an increasing trend.

Isolated points:

Who distance from other points is always random and comparatively high than any other point distance. Handle isolated points at the end individually.

3.1. Random Optimization

We developed a random optimization process for finding the best route just like a salesman travel problem. But the results are not great.

3.2. Pure Hamilton Graph

We tried to develop the Hamilton graph for all the points and then divide the travel path into multiple sub-parts such that every sub-part can be covered in a day. As this system is just based on the distance traveled by tourists doesn't include the estimated time. It didn't give us great results.

3.3. Hamilton Graphs and HDBSCAN Clustering System

We have created initial clusters using HDBSCAN based on Geospatial distances. Within clusters, we developed Morose Travelling Salesman problem algorithm to find the optimized path. This optimized path per cluster is considered as the travel plan per day. We tried to divide the cluster if the total time taken (both traveling time and time spent at visiting point) is greater than 7 hours. Once we got data for every cluster (/everyday itinerary), we created the final itinerary as a combination of single-day itineraries created.

In our method, overall below is our optimization logic to divide the clusters further into small clsuters. d1V1+d2V2+d3V3.. dnVn= K. here d1,d2,d3… dn are distances has to be traveled between locations, and V1,V2,..Vn are expected visiting times of places. We tried to develop our clusters such that the visiting points under one cluster are close enough to each other and travel time can be as less as possible. Through our cluster division, we also made sure total travel time and visiting time will be approximately equal to the number of hours the tourist wants to spend per day(7 hours as per our example).

3.4. Hamilton-Based DBSCAN CLUSTERING.

Based on HDBSCAN clustering algorithm development, we developed a system to divide the connected graphs into multiple clusters. While HDBSCAN utilized the minimum spanning tree, we developed this algorithm on the bases of Hamilton graphs as in our algorithm more than the distance between one point to other points in the cluster, we care more about the only incoming path distance(traveling to the visiting spot) and outgoing path distance(traveling from visiting spot to another spot).

3.4.1 Find Nearest Neighbours

Taking the HDBSCAN as a reference, let’s use a k value (example five). Considering every point as a cluster, then for a given point, we can try to expand this cluster till we get the minimum points. We construct a graph between all these points for every cluster. We are doing this to eliminate checking multiple comparisons with every point in the database. In our distance array, we consider the distance between the centre of the point and non-neighbour points as infinity, and a connection is not possible.

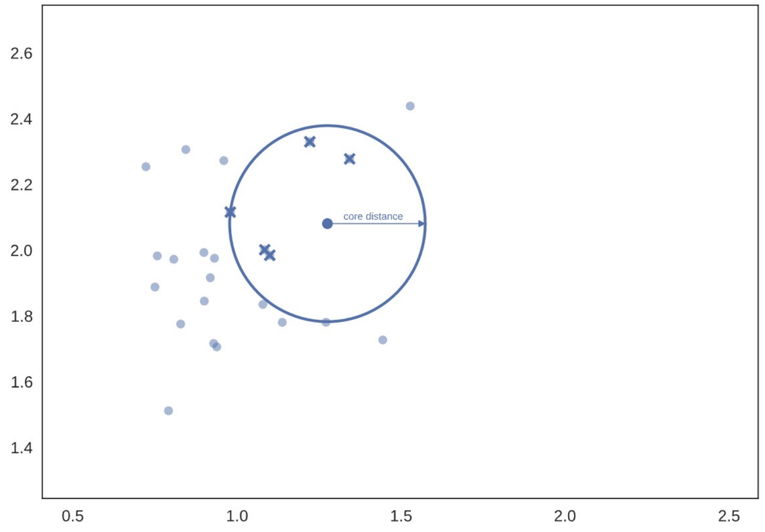

Fig. 3 Finding Nearest Neighbours

3.4.2 Develop The Hamilton Tree

We have the geo special distance between each point with other n-1 points. With them, we created a complete matrix. As we start exploring all the n2 edges and test all possible combinations of edges to make a path including all the tourist points. This will be very

expensive. there are n^2 edges and we don't want to have to run a connected components algorithm that many times. Here the path we want to find is nothing but the Hamilton graphs, we can utilize some of the data structures principles here and can optimize our process.
Find a minimal set of edges such that dropping any edge from the set divides the graph into subgraphs and there is only one in and one out of the edge.

Hamilton Tree:

There are n! different sequences of vertices that might be Hamiltonian paths in a given n-vertex graph (and are, in a complete graph), so a brute force search algorithm that tests all possible sequences would be very slow. An early exact algorithm for finding a Hamiltonian cycle on a directed graph was the enumerative algorithm of Martello.[3] A search procedure by Frank Rubin[4] divides the edges of the graph into three classes: those that must be in the path, those that cannot be in the path, and undecided. As the search proceeds, a set of decision rules classifies the undecided edges, and determines whether to halt or continue the search. The algorithm divides the graph into components that can be solved separately. Also, a dynamic programming algorithm of Bellman, Held, and Karp can be used to solve the problem in time O(n2 2n). In this method, one determines, for each set S of vertices and each vertex v in S, whether there is a path that covers exactly the vertices in S and ends at v. For each choice of S and v, a path exists for (S,v) if and only if v has a neighbor w such that a path exists for (S − v,w), which can be looked up from already-computed information in the dynamic program.

Andreas Björklund provided an alternative approach using the inclusion–exclusion principle to reduce the problem of counting the number of Hamiltonian cycles to a simpler counting problem, of counting cycle covers, which can be solved by computing certain matrix determinants. Using this method, he showed how to solve the Hamiltonian cycle problem in arbitrary n-vertex graphs by a Monte Carlo algorithm in time O(1.657n); for bipartite graphs this algorithm can be further improved to time o(1.415n).

We utilized the below strategy to develop Hamilton graph.

Create an nxn dimensional distance matrix such that every edge to every edge distance can be captured. start connecting the graphs from low distance values to high values. Once a path is made to a point, the rest of the connections to that particular point will not be valid and they were be dropped. Following this process to continue to choose next lowest edges and drop any other edges leading to that destination point, then continue to add the lowest weighted edges and drop not needed edges. Thus we can build the Hamilton tree very efficiently. In this case, we took the k value as 5. Once we have completely connected the Hamilton graph, we can start clustering the graph.

3.4.3 Cluster Building

Given the Hamilton tree, the next step is to convert that into the hierarchy of connected components. This is most easily done in the reverse order: sort the edges of the tree by distance (in increasing order) and then iterate through, creating a new merged cluster for each edge. This brings us to the point where robust single linkage stops. We want more though; a cluster hierarchy is good, but we really want a set of flat clusters. We could do that by choosing a part of the hierarchy. While HDBSCAN does that by corresponding cluster stability and DBSCAN effectively does this by declaring any singleton clusters at the cut level as noise. In our algorithm as we care more about aggregated time per cluster (a combination of travel time and total time spent at the visiting points) along with the closeness of the points. The challenge here is dividing this complete Hamilton graph into subgraphs. Ideally, we can divide the tree at different places to select our clusters but optimization of travel time and time spent are also what we want to achieve. The next steps explain how the Hamilton graph can be divided such that every cluster can be covered in a day time along with making sure the travel time is minimized and time spent at the visiting points is maximized.

3.4.4 Condense The Cluster Tree

First of all, we need to condense the small splits in to single point / few points graph to a smaller graphs with a stable and decent number of points attached to each branch (or cluster). In this process, we can observe a few points(two or three or a few more) falling out of the tree. In HDBSCAN we let go of them considering them as noise. But in our process, as visiting every single point is important for a traveller, these points become a separate cluster or can become a part of another cluster but with less cohesion. If the split is large enough, we consider two divided child graphs as individual clusters and continue the splitting process. We get decent-sized clusters as a result of this process.

The first step in cluster extraction is condensing down the large and complicated cluster hierarchy into a smaller tree with a little more data attached to each node. In this process, it is often the case that a cluster split is one or two points splitting off from a cluster, but we cannot let go of those points, as the tourist would like to visit every single tourist point, but based on its distance and estimated time to visit the place, it becomes a separate cluster or can become a part of other cluster but with less cohesion. Based on the condensed graph, now we need to choose the finalized clusters. While a mother cluster or the child clusters can be chosen from the graph, we want to select the cluster based on the stability of those utilizing our optimization process as shown below.

3.4.5 Day Itinerary Division Process Based On Stability Function Validation

Intuitively we want the choose clusters that are stable (less travel time and more visiting time and the time spend total should be less than K hours in our case it's 5 hours). We will continue to divide the data into sub-cluster until we find the cluster time spent to be approximately equal to K hours. To make a flat clustering we will need to add a further requirement that, if you select a cluster, then you cannot select any cluster that is a descendant of it. We need a measure to calculate cluster stability. First, we need to have a measure to optimize our process that's nothing but minimizing the travel distance and maximizing the visiting time. For this algorithm, we also need to make sure the sum of time spent should be approximately equal to K hours. I choose the stability function or the optimization function as below.

Calculation Process For Optimization(O):

(Visiting Time* exponential(-abs(K-Total Time)))/Travel Time

Examples:

7 hours visiting time,3 hours travel time, K =9 hours

=> O = 7*e pow(-abs(9-10)) /3= 2.33/e = 0.857 – Decent Case

7 hours visiting time,3 hours travel time, K =10 hours

=> O = 7*e pow(-abs(10-10)) /3 = 2.33 – Good Case

9 hours visiting time, 3 hours travel time, K = 9 hours

=> O= 9*e pow(-abs(12-9)) /3 = 3/e pow 3 = 0.149 – Comparatively Bad Case

Here I choose to give balanced importance to all three factors travel time, visiting time, and extra/fewer hours spent per cluster than planned hours. We considered the less O value as less stable and the high O value as more stable. Based on the stability values calculated, divided the system into child clusters or single-day itineraries.

4. IMPACT OF THE MODEL

As mentioned before, our app can make the travel process more comfortable for travelers. People will be able to visit places on the go without any preparation but still be able to have great experiences. It will improve the number of travelers thereby the travel industry overall.

5. FUTURE SCOPE

we found this field is very vast and there is high scope for improvement and research. Below are some ideas.

- Improve our app by adding suggested time to visit for future based on current weather predictions.
- Instead of a batch process, Make the system online and create an itinerary based the places the tourist already visited and some places they skipped from the plan or some places they added manually. This gives more freedom for the tourist to plan their trip and also gives them more command and understanding about the city they are visiting.
- Create a more customized places itinerary to visiting points list based on customer passion whether they like outdoor activities, historical places, viewpoints or experiences, etc. While most of the lists or articles available on the internet or in tourist guides lists are mostly based on the place but they never consider tourist interests. Which can be best solved by this method
- For the everyday visit itinerary we also need to add the accommodation points as starting and end points, in our future work we are planning to suggest some optimized accommodation places to customers based on the tourist points they choose and create a full-fledged itinerary.
- We can also recommend food places around based on optimized routing options, based on the places chosen, we can update our itinerary and provide improved services. This can also help to high late the local food and local experiences. While helping the local industries to grow, this method provides a more customized experience to the users.

ACKNOWLEDGMENTS

I would like to acknowledge the support of my team who have supported me in data acquisition and website development.

REFERENCES

[1] Basic Usage of HDBSCAN* for Clustering. https://hdbscan.readthedocs.io/

[2] DBSCAN clustering. https://scikit-learn.org

[3] Ahmed Derdouri & Toshihiro Osaragi, A machine learning-based approach for classifying tourists and locals using geotagged photos: the case of Tokyo, https://link.springer.com/article/10.1007/s40558-021-00208-3

[4] Juan Carlos Cepeda-Pacheco & Mari Carmen Domingo, Deep learning and Internet of Things for tourist attraction recommendations in smart cities, https://link.springer.com/article/10.1007/s00521-021-06872-0

[5] Ahmed Gad(Jul 3, 2018), Introduction to Optimization with Genetic Algorithm. https://towardsdatascience.com/travel-time-optimization-with-machine-learning-and-genetic-algorithm-71b40a3a4c2

[6] Silvia Paldino, Iva Bojic, Stanislav Sobolevsky, Carlo Ratti and Marta C González, Urban magnetism through the lens of geo-tagged photography, EPJ Data Science

[7] Ricardo Fukasawa, Humberto Longo, Jens Lysgaard, Marcus Poggi de Aragão, Marcelo Reis, Eduardo Uchoa & Renato F. Werneck (2005), Robust Branch-and-Cut-and-Price for the Capacitated Vehicle Routing Problem, Mathematical Programming volume 106, pages491–511

[8] J. K. Lenstra & A. H. G. Rinnooy Kan, Complexity of vehicle routing and scheduling problems

[9] Roberto Baldacci & Aristide Mingozzi(2008), A unified exact method for solving different classes of vehicle routing problems

[10] Traveling Salesperson Problem, Google OR-Tools. https://developers.google.com/optimization/routing/tsp

[11] Genevieve Hayes, mlrose Documentation. https://readthedocs.org/projects/mlrose/downloads/pdf/stable/

[12] Hamilton Path, Wikipedia. https://en.wikipedia.org/wiki/Hamiltonian_path

[13] Buhalis D, Amaranggana A (2013) Smart Tourism Destinations. In: Xiang Z, Tussyadiah I (eds) Information and communication technologies in tourism 2014. Springer International Publishing, Cham, pp 553–564

Authors

Divya Mereddy

She is A data Scientist at 7-Eleven. She has completed her Masters at the University of Cincinnati. Her research goal is to make the world around her a better place to live. She is interested in Vision, Cognition, Learning, and Autonomy.