Touristic Routes Recommendation Focused on User Experience and the Environment

Lucas Gabriel da Silva Félix

February 7, 2024

1 Introduction

Before the 2020 pandemic, tourism was one of the fastest growing markets in the world [7] and was vastly impacted due to the travel ban restrictions [12]. Nevertheless, is not the only issue that can directly impact tourism. Environmental issues, especially climate change, are expected to have a significant impact in the next few years [12], in special, eco-touristic cities.
However, when traveling, most tourists do not focus on their environmental impact, due to the fact that planning a trip can be a complex and time-consuming process [8].
To facilitate the process of planning a trip, several works in the literature focus on the task of automating trip planning. The approaches available in the literature classify the task of selecting and touring between places as a hard-to-solve computational problem, that can be mapped into the Orienteering Problem (OP) [5, 15]. As stated in [16], this problem is a combination of the traveling salesman and the knapsack problem, where the goal is to find a path in a graph that maximizes a utility function while respecting a distance constraint. It is noteworthy that in the literature a wide variety of restrictions have been taken into consideration [8] for this problem. Below, we list some of the constraints considered in this problem:

- Places working hours
- Time budget
- Transport selection
- Daily distance
- Max. amount of daily places

- Monetary cost
- Climate data
- Single/Multiple Days
- Category diversity
- Hotel Selection
- Mandatory Places

It is possible to see that some of the constraints considered in the literature have an impact on the environment, such as transport selection, daily distance, and hotel selection. In these cases, if the selected transport is not environment-friendly, this could have a significant impact, especially if the user is traveling long distances to visit other Points-of-Interest (POI), and/or the users' hotel is a long way from the POIs that will be visited.
It is noteworthy, however, that the main objective of these works is not to recommend the route with less amount of impact on the environment, but to recommend the route that maximizes user satisfaction. Thus, most of the works consider that as bigger the amount of places visited, the better the route, hence, to maximize the amount of places, the user must travel from one place to another as fast as the solution can recommend. In scenarios where the POIs in the route are close, this is not a problem, given that the user can reach the next POI on foot, nevertheless, in sparse/big cities, where the POIs are far from each other, the approaches will recommend for the user the fastest transportation type, hence, increasing the tourist carbon footprint.
More recently, papers in the literature, have proposed strategies that focus on both maximizing user satisfaction and minimizing the $C0_2$ emission by tourists [12, 10, 11, 17, 13], enhance reducing the impact of tourists on the environment.
Given these facts, our aim with this programming assignment is to recommend routes to tourists, focusing on their impact on the environment. In this work, we intend to propose a route recommender focused on eco-tourism, recommending near places that can maximize user satisfaction and minimize the traveling dependence on high-emission vehicles (e.g. personal cars).
To accomplish so, initially, we intend to perform evaluations on a Brazilian tourist city, Rio de Janeiro. The first city is Rio de Janeiro is characterized as having different clusters of places spread through a larger area [2].
As for the user, in the proposed application, we need as input: the maximum monetary budget, amount of days traveling, and the maximum amount of POIs from the same category that can be visited consecutively. These inputs will enable our system to enhance users' personalization in the recommended route. For example, the category input enables us to design a constraint that permits the user to do specific tours (e.g. gastronomic route or museum route), but also to have a break within the visits, to rest or visit places from another category.
As for the main metrics of this project, our main goal is to maximize user satisfaction, which is given by each places' rating. Second, our goal is to reduce $C0_2$ emissions made by the

tourist, hence, our tour will take as the main transportation foot, which does not depend on any vehicle. Thus, our major difficulty is to select for tourists a hotel that is near many POIs hence, reducing the amount of kilometers that a user is walking, and maximizing the amount of places walked.
The rest of this work is organized as follows, first, we introduce everything that an application like this must have to properly work. We do this by introducing the User Requirements [2], Data Requirements [3], and the System Requirements [4]. Then, we show our proposal [5] showing what will be done in the limited time we have. Then we show the Results [6], and Conclusion [7].

2 User Requirements

When thinking about an application to suggest for users fully personalized trips, first we must think what are the users' preferences that this software would have. Compliance with these requirements is essential for creating a valuable and engaging user experience. So, below we enumerate two different types of requirements needed for software like this, first, the user requirements for the system, thus, indispensable features, for two different types of users: place owners, and then tourists. Second, the constraints that a user has in relation to the trip. In this second scenario, these constraints are important to provide the user with a deep personalization of the travel. These constraints are integrated with the recommender system and the optimization model.

2.1 Users' system requirements

2.1.1 Place Owners

To accommodate place owners (such as hotels, restaurants, tour operators, etc.) who want to use the platform to update information, provide discounts, interact with clients, and manage their listings. Since this is a business view, we believe that a web (e.g. desktop) web application would facilitate the visualization, in special of dashboard with the analytics from the business. For such application, the following features and requirements are needed:

- **Registration and Verification:** Place owners should be able to register as businesses on the platform. Verification mechanisms (e.g., email verification, and phone confirmation) should ensure the legitimacy of the business.
- **Dashboard for Place Management:** Provide a user-friendly dashboard where place owners can manage their business listings, information, and analytics information extracted from the users' opinions about their place.
- **Business Profile Management:** Allow place owners to update and maintain their business profiles, including photos, descriptions, contact information, operating hours, and characteristics (e.g. type of food served, kid-friendly).

- **Discount and Promotion Management:** Enable place owners to create and manage discounts, promotions, and special offers for travelers using the platform. This step is very important for the business model, given that to engage the user we must offer more than the information about the places (as Google places, or TripAdvisor does).
- **Booking and Availability Management:** Allow place owners to update real-time availability and prices for accommodations, tours, or other services. Provide tools for confirming, managing, and tracking bookings made through the platform. This type of feature will help both place owners and users. For place owners, this feature will allow them to estimate the amount of visitors. Users will guarantee that a place will be held for them when visiting such a place.
- **Customer Interaction:** Implement a messaging system that enables place owners to communicate directly with clients to answer inquiries, provide assistance, and offer guidance.
- **Review and Rating Management:** Enable place owners to respond to user reviews and ratings, fostering engagement with customers. Allow for the reporting of inappropriate or false reviews.
- **Integration with Online Payment Systems:** Integrate secure payment gateways to facilitate transactions, including accepting deposits and payments from travelers.
- **Analytics and Reporting:** Provide analytics tools that allow place owners to track the performance of their listings, including occupancy rates, conversion rates, and customer demographics.
- **Content Management System (CMS):** Offer a CMS for place owners to update and add content, such as images, descriptions, and amenities, to their business listings.
- **Customization Options:** Allow place owners to customize the appearance and branding of their listings to match their business identity.
- **Multi-Language Support:** Provide multilingual support for business owners to reach international travelers.
- **Booking Confirmation and Notifications:** Send booking confirmations and notifications to place owners, including real-time booking alerts and updates.

2.1.2 Travelers

When thinking about the user, we must offer an application that allows the user to have all the information needed for travel planning on their hands. Thus, even though we could offer a web application for the user, we know that a mobile application available to all platforms is more than needed. For such application, the following features and requirements are needed:

- **User Profiles**: Allow users to create and manage detailed profiles, including personal information, travel history, and preferences. The more information the system can gather, the more personalized the recommendations can be.
- **Users' constraints**: Users should be able to specify their constraints. For instance, the duration of their trip and their budget will help the system recommend appropriate destinations and activities. We discuss this further in the next section.
- **Interest and Activity Preferences**: Users can select their interests and preferences, such as cultural experiences, adventure activities, relaxation, or culinary experiences. The system can then suggest destinations and activities that align with these preferences.
- **Travel Restrictions and Special Requirements**: Users can provide information about any restrictions or special requirements, such as dietary restrictions, mobility issues, or family-specific needs. The system should take these into account when making recommendations.
- **Itinerary Customization**: Users should have the flexibility to customize the automatically generated itineraries. They can add, remove, or rearrange activities and destinations to create a truly personalized plan.
- **Real-Time Cost Estimation**: The system should provide real-time cost estimates for the entire trip, taking into account flights, accommodations, activities, and meals. This helps users stay within their budget.
- **Interactive Maps**: Users can view their itinerary on an interactive map, making it easy to visualize the locations of attractions, accommodations, and transportation options.
- **Recommendation Algorithms**: Implement intelligent recommendations based on the users' content algorithms, that consider user preferences, current location, and other factors to suggest destinations, activities, and accommodations. This can be especially used when due to any factor the user is late for some activity.
- **Booking and Reservations**: Allow users to book accommodations, flights, tours, and tickets for attractions directly through the platform, simplifying the planning process.
- **Weather and Event Information**: Provide users with real-time weather updates for their destination and information about local events or festivals during their travel dates.
- **Travel companion:** Allow users to set their companion when making a trip. This would allow the system to provide better recommendations when different types of trips are made (e.g. romantic, friends, family, alone).

- **Trip Sharing**: Enable users to share their itineraries with friends and family, facilitating group travel planning. We also must allow them to edit this itinerary as a family.
- **Push Notifications**: Send push notifications to users to remind them of upcoming activities, changes to their itinerary, or important travel updates.
- **Review and Feedback System**: After the trip, users can rate and review destinations, accommodations, and activities, helping other travelers make informed choices.
- **Offline Access**: Allow users to access their itineraries and important travel information offline, especially useful when traveling to areas with limited internet connectivity.
- **Language and Currency Conversion**: Provide tools for language translation and currency conversion to help users navigate in foreign countries.
- **Emergency Information**: Include emergency contact information, local emergency services, and guidelines for handling unexpected situations during the trip.

2.2 Users' constraints

When considering the tourist trip design problem, it's essential to account for various constraints that can significantly impact the user's travel experience. These constraints should be taken into consideration when creating personalized itineraries and recommendations. When modeling such constraints, usually, we use them on the optimization model, however, some of them can be pre-processed on a step before, avoiding unnecessary evaluation on the optimization step. Here are some of the main constraints to consider for users:

- **Budget Constraints:** The user may have a limited budget for the trip, which should be considered when suggesting accommodations, activities, and transportation options.
- **Time Constraints:** Users may have a fixed duration for their trip, and their itinerary should be designed to fit within this timeframe.
- **Geographical Constraints:** Some users may have restrictions on where they can travel due to visa requirements, travel bans, or personal preferences
- **Distance Constraints:** The users can limit the number of kilometers that they can walk during the day.
- **Transport Constraints:** The users can put constraints on different types of transportation due to accessibility issues, or due to the fact that these users want to avoid one type of transportation (e.g. avoid cars to reduce pollution).
- **Category Constraints:** The users can limit sequential visits to different categories (e.g. a user does not want to visit any museums or a user that wants to do a museum tour).

- **Place Constraints:** The users can limit places that they do not want to visit on their travel. Or users want to specify places that are mandatory for visiting.
- **Physical Constraints:** Users with mobility issues or health concerns may have limitations on the types of activities and accommodations they can choose.
- **Dietary Constraints:** Users with dietary restrictions or preferences (e.g., vegetarian, vegan, gluten-free) may require specific dining options.
- **Language Constraints:** Language proficiency can be a constraint, so providing information and recommendations in the user's preferred language is essential.
- **Cultural Constraints:** Users may have cultural or religious preferences or restrictions that affect their travel choices, such as observing certain traditions or avoiding specific activities.
- **Weather Constraints:** The weather at the destination can impact the user's experience, so it's important to consider seasonal weather conditions and offer alternatives when necessary.
- **Safety and Health Constraints:** Safety concerns and health advisories related to the destination should be factored into travel recommendations.
- **Accessibility Constraints:** Considerations for users with disabilities, such as wheelchair accessibility and special accommodations, should be taken into account.
- **Group Travel Constraints:** For users traveling with a group, the system should accommodate the preferences and constraints of multiple travelers.
- **Family Constraints:** Users traveling with children may have specific needs, such as family-friendly accommodations and activities.
- **Local Event Constraints:** Users may have specific dates in mind for their trip to attend local events or festivals, which should be incorporated into the itinerary. This is also known as a fixed-mandatory place constraint. In this scenario, the whole users' trip is made around that event.
- **Custom Preferences:** Users may have custom preferences that don't fit into standard categories, and the system should allow for the inclusion of these unique constraints.

By taking these constraints into account, the travel planning system can generate personalized itineraries that respect the user's individual needs and preferences while ensuring a safe and enjoyable travel experience. It's important to provide users with the flexibility to input and adjust these constraints as they plan their trips.

3 Data Requirements

To develop travel planning software that can efficiently manage user data, itinerary information, recommendations, and other essential details, you would typically need several databases to store different types of information. Here are the needed databases for such software:

- **User:** This database stores user profiles and account information, including names, contact details, travel preferences, and past travel history. It also includes user authentication data for secure login and access control. This will be generated within our system and will be provided by the user.
- **User Travel Preferences:** Stores user preferences, constraints, and settings, including budget, trip duration, interests, and dietary restrictions. This dataset will be filled by the user.
- **Itinerary:** This database is crucial for storing user-generated itineraries, including the list of activities, destinations, dates, and custom notes. Itineraries should be associated with specific user profiles. This will be generated within our system, will take as input the users' preferences for that itinerary, and will be generated by the recommender system and optimization model.
- **Destination:** Contains information about various travel destinations, including names, descriptions, geographic coordinates, and images. Information about local attractions, accommodations, and restaurants can be linked to destination data. This database will be provided by users and place owners. Initially, we can scrap such information from open datasets as OpenStreet Maps and Wikipedia.
- **Accommodation:** Contains data related to accommodations, including hotels, Airbnb listings, and other lodging options. Information includes property names, descriptions, photos, prices, availability, and user reviews. To fill this database, we must think of a way to attract place owners to fill such information.
- **Booking:** Stores booking and reservation data, including booking confirmations, dates, prices, and reservation status. Links to user profiles and specific itineraries. This database will be filled by the users, and will be used by both users and place owners.
- **Review:** Holds user-generated reviews and ratings for destinations, activities, and accommodations. Data includes user comments, star ratings, and date of submission. This database will be filled by users making reviews of places.
- **Weather:** Stores historical and real-time weather data for various destinations. Includes temperature, precipitation, and climate information for different times of the year. For this dataset, we will need an API from some weather platform.

- **Language and Currency:** Contains language translation and currency conversion data for multilingual and multi-currency support. For this dataset, we will need an API from a translation platform (e.g. Google Translate or ChatGPT) and a currency platform.

- **Emergency Information:** Stores emergency contact information, local healthcare facilities, police stations, and other critical safety-related details. This dataset can be filled with governmental data.

- **Notification:** Records notifications and alerts that need to be sent to users, including reminders, updates, and information about changes to their itineraries. This database will be filled with system information.

- **Event and Festival:** Contains information about local events, festivals, and cultural celebrations taking place in various destinations. This database can be filled with information from social networks and also can be informed by city administrators, users, and place owners.

- **Transportation:** Contains information about car rentals, flights, and public transportation. To fill such a database we can use governmental information, and also allow these companies (e.g. travel companies, car rental companies) to fill such information.

4 System Requirements

The requirements for the software and interface of a travel planning application should be designed to provide a user-friendly, visually appealing, and efficient experience. Here are some key software and interface requirements:

4.1 Software Requirements

- **User Authentication and Security:** Implement secure user authentication and authorization mechanisms to protect user data. Use encryption to secure user information and transactions.

- **User Profile Management:** Allow users to create, edit, and manage their profiles with detailed information about preferences and travel history. Provide options for users to set privacy and data-sharing preferences.

- **Recommendation Engine:** Implement recommendation algorithms that consider user preferences, constraints, and past travel history. Continuously update recommendations based on user interactions.

- **Database Management:** Set up a database system to store user profiles, itineraries, bookings, and reviews. Ensure efficient data retrieval and management.

- **Integration with Third-Party APIs:** Integrate with travel-related APIs for real-time data on flights, accommodations, attractions, weather, and more. Ensure smooth communication with external services for booking and reservation.
- **Notification System:** Implement a notification system to send reminders and updates to users via email, push notifications, or SMS. Allow users to set notification preferences.
- **Responsive Design:** Ensure that the software is responsive and accessible on various devices, including desktops, tablets, and smartphones.
- **Offline Access:** Provide an offline mode for accessing itineraries and essential travel information without an internet connection.
- **Multi-Language Support:** Support multiple languages to cater to an international user base. Enable language selection for the user interface.
- **Currency Conversion:** Include a currency conversion tool to help users understand and manage expenses in foreign currencies.
- **Robust Search Functionality:** Implement a powerful search engine to help users discover destinations, activities, and accommodations.

4.2 Interface Requirements

- **Intuitive User Interface:** Design an intuitive and user-friendly interface with easy navigation and clear information hierarchy.
- **Personalized Dashboards:** Create personalized dashboards for users to view their profiles, itineraries, and recommendations.
- **Interactive Maps:** Incorporate interactive maps for visualizing itineraries and points of interest. Include options for route planning and map exploration.
- **Customization Features:** Provide tools for users to customize itineraries, including adding or removing activities, destinations, and notes.
- **Booking and Reservation Forms:** Design clear and user-friendly forms for booking accommodations, flights, tours, and tickets directly through the application.
- **Rating and Review System:** Implement a rating and review system for users to share feedback on destinations, accommodations, and activities. Display user-generated content to help others make informed choices.
- **Visual Design and Branding:** Create an aesthetically pleasing design that aligns with the application's branding. Use a consistent color scheme and typography.

- **Mobile Responsiveness:** Ensure that the interface is fully functional and visually appealing on mobile devices.
- **User Support and Help Center:** Include a user support section with FAQs and guides to assist users in using the application.
- **Accessibility:** Ensure that the application is accessible to users with disabilities, adhering to web accessibility guidelines.
- **Feedback Mechanism:** Provide an easy way for users to provide feedback and report issues directly through the interface.
- **Emergency Information:** Include an easily accessible section with emergency contact information and guidelines for handling travel emergencies.
- **Customer Support:** Offer customer support for place owners, including assistance with technical issues, account management, and listing optimization.
- **Compliance and Guidelines:** Set clear guidelines for place owners regarding ethical practices, content quality, and user engagement. Enforce policies to ensure the quality and integrity of the platform.
- **Training and Resources:** Provide educational materials and resources to help place owners optimize their listings and improve their business performance.
- **Security and Data Protection:** Ensure that sensitive business and customer data is secure and compliant with data protection regulations.
- **Feedback Mechanism:** Allow place owners to provide feedback and suggestions to improve the platform's functionality and usability.

These software and interface requirements are essential for creating a comprehensive travel planning application that meets the needs of users and enhances their overall travel experience.

5 Methodology

In this section, we present our proposal for solving the tourist problem. Given that this problem has high practical applicability and can be incorporated into applications that require real-time user responses (e.g., social networks), the development of techniques that generate good solutions but fall short in terms of computational time does not meet user demands. Users are willing to wait a maximum of 1 second for a response [6]. Given these factors, our objective in proposing this methodology is not only to recommend routes that satisfy user constraints but also to have a methodology with a maximum response time within the ideal limits for the user. However, even though we describe the ideal application for recommending users' trips, we still have a limited scope. Thus, for this work, we only make a proof of concept (POC), focused on solving optimization problems under 1 second, with transportation selection, and carbon emission reduction. First, we introduce the constraints 5.1 that we consider in this POC, then we introduce the heuristic developed to solve this problem, and the dataset used to test our proposal 5.3. Lastly, we introduce the interface 5.4 that we developed for the problem.

5.1 Constraints

For this work, we have the objective function to maximize the user ratings, hence the quality of the places visited. The constraints we considered are the monetary budget, daily distance, category, maximum amount of places to be visited per day, multiple visits, and hotel constraints. Below, we provide the mathematical modeling for the objective function and all constraints considered in this work.
Consider a tourist (user) traveling through a city for p days with the goal of visiting a maximum of m tourist attractions every day. The user's objective is to create daily tours to optional locations represented by the set $\mathcal{O}$ that can be visited during the p days. Each location i in $\mathcal{O}$ is associated with a utility $R_i \in \mathbb{R}_+$. Our aim is to find a subset of $\mathcal{O}$ that maximizes the utility of the visited locations.
Let $G = (V, E)$ be a complete-directed graph representing the city. Let $V = v_0, v_1, \ldots, v_n$ represent the set of vertices corresponding to places available to be visited, where each place $v \in V$ is associated with a utility R_v, a category $\mathcal{C}v$, and a cost $Cv \in \mathbb{R}_+$. The first h nodes of this graph are hotels, and n is the number of nodes in G. Additionally, $\mathcal{V}$ defines the set of all places on the route, and $\mathcal{V}c$ is the set of places excluding the starting and ending points. Each edge connecting points of interest has a corresponding weight $D = Di, j, i \neq j | D_{i,j} \to \mathbb{R}_+$, denoting the distance between two places. Our objective can be adapted on this graph to find a cycle (e.g., the path that starts and ends at the same node) in G that maximizes the utility R while respecting real-world constraint sets. It is essential to note that for each of the attributes of a place defined above, there is a corresponding user-defined constraint value. Below, we present all these user-defined constraints and inputs.
Recalling that we define the cost (e.g., entrance fees, food prices) of visiting each place i as C_i. In this scenario, users define the total budget $b \in \mathbb{R}_+$ they are willing to spend on the entire trip. The cost constraint is defined by Equation 1:

$$\sum_{d=0}^{p} \sum_{i=0}^{n} \sum_{j=0}^{n} X_{i,j}^{d} C_i \leq b \tag{1}$$

Users also define the amount $w \in \mathbb{R}_+$ of distance they are willing to walk per day. Equation 2 presents the daily distance constraint:

$$\sum_{i=0}^{n} \sum_{j=0}^{n} X_{i,j}^{d} D_{i,j} \leq w, \forall d \in 1, 2, \ldots, p \tag{2}$$

In addition to being daily limited by the maximum distance w, the users' route is also daily limited by the maximum number of places $m \in \mathbb{Z}_+ | m > 0$ that can be visited. The daily activity constraint is presented in Equation 3:

$$\sum_{i=0}^{n} \sum_{j=0}^{n} X_{i,j}^{d} \leq m + 1, \forall d \in 1, 2, \ldots, p \tag{3}$$

Equation 4 ensures the connectivity of the solution and deals with symmetric solutions.

$$X^{d}i, j + X^{d}j, i \leq 1, \forall d \in 1, 2, \ldots, p, \forall i \in \mathcal{V}c, \forall j \in \mathcal{V}c \tag{4}$$

Furthermore, Equation 5 defines the multiple visits constraints, avoiding revisits on the same day:

$$\sum_{i=0}^{n} X_{i,k}^{d} = \sum_{j=1}^{n} X_{k,j}^{d} \leq 1, \forall d \in 1, 2, \ldots, p, \forall k \in \mathcal{V} \tag{5}$$

Recalling that in our model, each place is also associated with a category, such as hotels, churches, restaurants, and museums. Let $\mathcal{C}$ denote the set of categories $\mathcal{C} = 0, 1, \ldots, n_c$, where category $\mathcal{C}0$ defines a hotel. Let a binary matrix Ki, c denote whether a point of interest i belongs to category $c \in \mathcal{C}$. In this scenario, we define two category-based constraints. First, a value $q^d \in \mathbb{Z}_+ | q^d > 0$, defined by the user, restricts the number of places of the same category that can be visited in a day. The category constraint is presented in Equation 6:

$$\sum_{i=0}^{n} \sum_{j=0}^{n} K_{i,c} X_{i,j}^{d} \leq q^{d}, \forall d \in 1, 2, \ldots, p, \ \forall c \in \mathcal{C} \backslash 0 \tag{6}$$

Finally, the hotel constraint defines that only the first and last places visited in a day can be a hotel, as defined by Equation 7:

$$\sum_{i=h}^{n} \sum_{j=h}^{n} K_{i,c} X_{i,j}^{d} = 0, \forall d \in 1, 2, \ldots, p, \ c \in \mathcal{C}, \ c = 0 \tag{7}$$

It's worth noting that ideally, for our proposal to have more real-world applications, time window constraints should be implemented. However, given the limited scope of this work, we leave such an evaluation for future research.

5.2 Heuristic

Population-based techniques are approaches that delve into intensity and diversity aspects at the expense of computational performance, despite generating good results for the tourist problem [9, 16, 19, 18], they do not generally meet the necessary requirements for real-time applications. In this case, techniques based on single solutions such as local search, iterated local search, tabu search, simulated annealing, and GRASP have better computational performance [14], although they may lack diversity in the generated solutions. However, among

all these single-solution techniques, GRASP stands out due to its random nature, which allows for deeper exploration and consequently generates solutions with more diversity than other single-solution techniques that lack any stochastic principle. Therefore, considering these factors, in this work, we use GRASP to solve the tourist problem.
GRASP is a metaheuristic proposed by [4], which combines elements of greedy construction with local search to solve combinatorial optimization problems [14]. Figure 1 illustrates how each of the solutions generated by the heuristics is represented, with the hotel fixed in the first position and the other locations to be visited in subsequent positions.

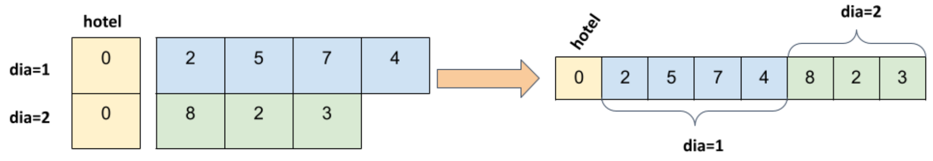


Figure 1: Solution representation

A pseudo-code for the operation of GRASP is presented in Algorithm [1], and its general idea is as follows:

1. **Greedy-Randomized Construction**: Starts with an empty solution and, in a greedy manner, adds elements to the solution set, but with a degree of randomness controlled by the parameter *alpha*. This allows for the exploration of different initial solutions. In each iteration, the elements that can be included in the partial solution are sorted in a list (in descending order of value) using a local heuristic. From this list, a subset is generated to represent the restricted candidate list (RCL).

2. **Local Search**: After construction, local search is applied to improve the solution found in the previous step. Local search explores neighboring solutions in search of improvements.

3. **Iterations**: The steps of construction and local search are repeated for a defined number of iterations, maintaining the best solution found.

Despite the application of GRASP in other literature proposals [1, 3, 11], all of them base their selection policy in the RCL on their objective function, guided by the ratings given to each of the visited locations. However, in some scenarios, this may not be the best solution, as we not only need to address the solution's fitness but also respect constraints. Especially in real-world scenarios in large cities like Rio de Janeiro, the distance between points of interest can be significant, causing heuristics, in general, to generate infeasible candidate solutions. To avoid this problem, in this work, we propose sorting the RCL based on the distance traveled for each rating that the location has, calculated by the formula $c_j = \frac{D_{i,j}}{R_j}$, where the values i and j represent available locations to be visited. In this scenario, locations that are closer and have higher ratings have a better chance of being chosen within the solution, consequently making a value-based selection, which is given by the formula [8], where α is

Algorithm 1 *GRASP - Greedy Randomized Adaptive Search Procedure*

```
procedure GRASP($iterations, alpha$)
    $best\_solution \leftarrow$ initialize_solution()
    for $i \leftarrow 1$ to $iterations$ do
        $candidate\_solutions \leftarrow \emptyset$
        for $j \leftarrow 1$ to $N$ do
            $solution \leftarrow$ construct_greedy_randomized_solution($alpha$)
            $solution \leftarrow$ local_search($solution$)
            $candidate\_solutions \leftarrow solution$
        end for
        $current\_solution \leftarrow$ select_best_solution($candidate\_solutions$)
        if $current\_solution$ is better than $best\_solution$ then
            $best\_solution \leftarrow current\_solution$
        end if
    end for
    return $best\_solution$
end procedure
```

calculated adaptively following the proposal presented in [14]. In general, the value-based selection picks candidates within the range [c^{min}, β], with β calculated by the formula below:

$$\beta = c^{min} + \alpha(c^{max} - c^{min}) \tag{8}$$

In the local search phase, we employ operations based on the knapsack problem, aiming to replace locations with lower ratings. In this case, for locations with a rating lower than five, we check locations of the same category within the same region that have a better rating, thus ensuring the viability of the generated solution.

5.3 Experimental Procedures

For the experimental evaluation of our work, we considered a real-world scenario. The target city for our data collection was Rio de Janeiro, a large city with many tourist attractions. To collect this data, we developed a crawler responsible for automatically navigating the TripAdvisor website, gathering all the content available to users and the Points of Interest (POIs). The collected data was initially in an unstructured format (e.g., HTML). We developed a parser to extract the content from each page, preprocess it, and store it in a semi-structured format (e.g., CSV). From the data collected from TripAdvisor, we used attributes of the places such as location, category, and price.

From the collected data, we have 648 hotels, 564 restaurants, and 959 attractions such as beaches (e.g., Copacabana and Ipanema), theaters, and points of interest (e.g., Morro da Urca and Corcovado). In total, the super category of attractions is formed by 110 different subcategories. To calculate the distance between all places, we used the OpenStreet Maps library to calculate the distance along real routes, rather than geodesic distance. This processing step is done offline. Figure 2 shows the spatial distribution of all the collected points.

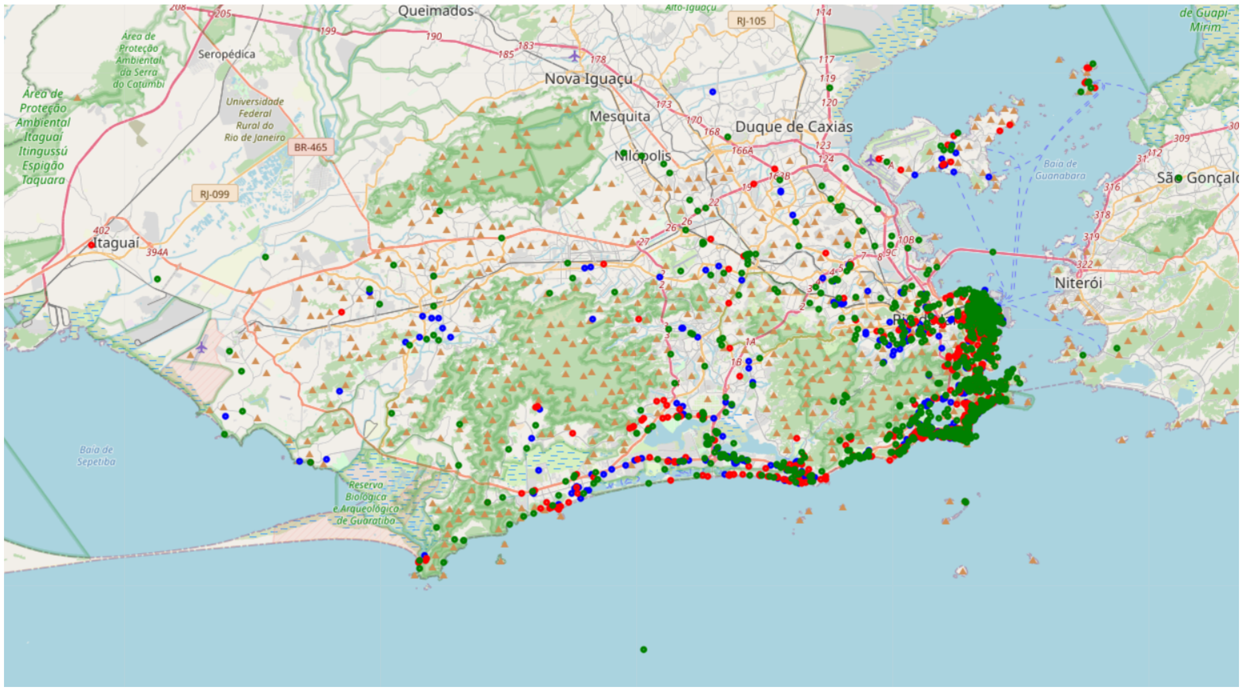


Figure 2: Map of Rio de Janeiro with points of visit. In green, we have attractions; in red, hotels; and in blue, restaurants.

Additionally, we used several complementary datasets to enhance our analysis. These datasets include information about user preferences, budget constraints, and time constraints. All of these data sources were used to create a comprehensive dataset for our experiments.

5.4 Interface

To provide personalized routes, our proposal is highly dependent on the users' inputs. Thus, there are mandatory parameters for the users. As mandatory inputs, there are the money budgets, the number of kilometers that the users plan to dislocate each day, the maximum amount of places to visit per day, and the maximum amount of places per category to be visited per day. The rating of each place is defined as the average rating of the place provided by TripAdvisor.
Besides the users' inputs, a database with information of each place is used, providing the location restrictions. As mentioned earlier, this data was collected from TripAdvisor, and each place has its geolocation, category, and price. To measure the actual distance between the places was used the OpenStreetMaps API. From OpenStreetMaps we also identify the

streets between the POIs, which were plotted using Folium library [1]. The front-end and design of the interface were implemented using Streamlit [2].

6 Results

In this section, we present the results achieved by our proposal. First, we introduce the different profiles with which we evaluated our method. Second, we present the main parameter of GRASP and its impact on the proposal. Third, we discuss the impacts of these profiles on the utility (fitness) of our results, and also in the execution time. Lastly, we introduce the proposed interface for real users to test our proposal.

6.1 Experimental evaluation profiles

For the experimental evaluation, we considered a tourist traveling to Rio de Janeiro city. For this purpose, we considered different combinations of parameters that we called profiles. The parameters can be categorized into 2 types: fixed and variable. The fixed ones include the duration of the trip (2 days), the maximum number of daily locations (5 locations), and consecutive visits per category (3 visits). The variables are the kilometers per day and the budget, creating different user profiles, which are presented in Table 1.

Profile	**Monetary Budget**	**Daily walk**	**Description**
A	500[3]	10	Low spender, walks little
B	1000	10	Average spender, walks little
C	2500	10	High spender, walks little
D	500	20	Low spender, walks a lot
E	1000	20	Average spender, walks a lot
F	2500	20	High spender, walks a lot

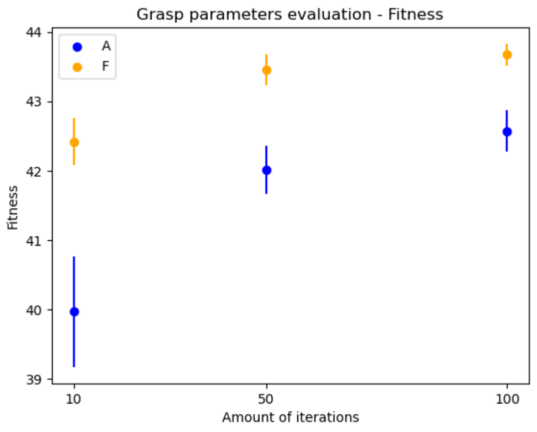
Table 1: Different profiles evaluated

6.2 Parameters selection

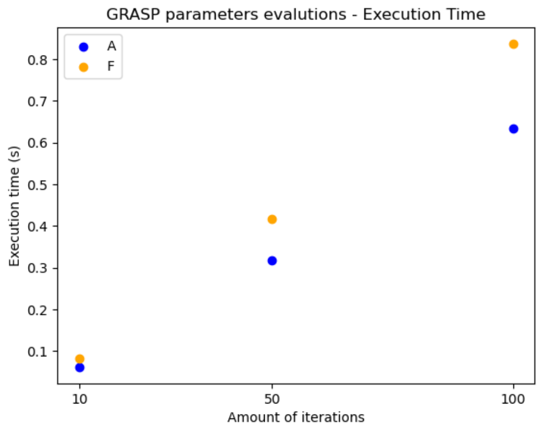
For GRASP we present and discuss the parametrization in terms of utility and time for Profiles A and F since they are the two most distinct profiles among those evaluated.

[1]http://python-visualization.github.io/folium/

[2]https://streamlit.io/



(a) Evaluation of the utility per parameter

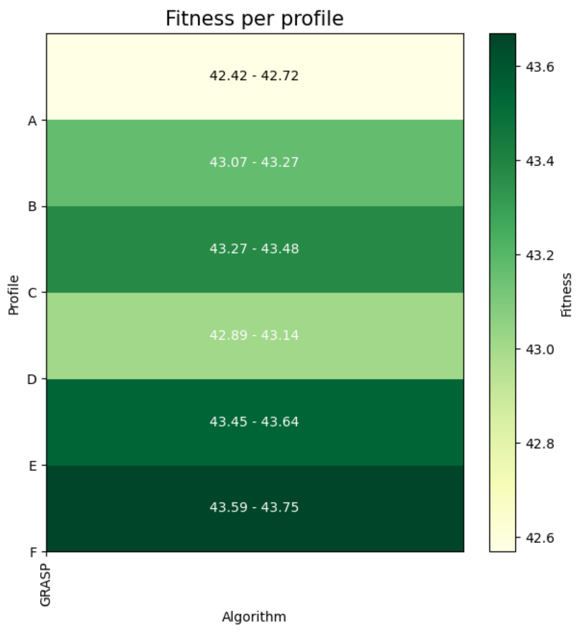


(b) Evaluation of execution time per parameter

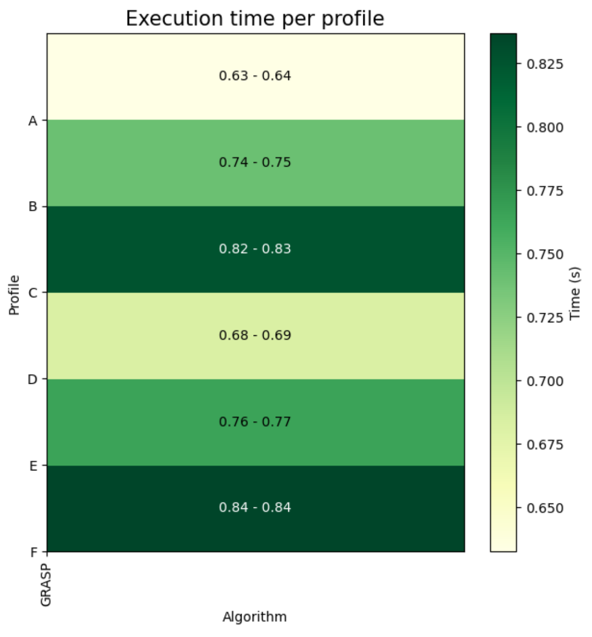
Figure 3: Time and utility of the tested parameters

The GRASP technique, following our proposal presented in Section 5, has parameters alpha and the number of iterations. Since alpha is automatically calculated, the number of iterations is the only parameter impacting our proposal. Figure 3 presents the impact of the number of iterations on the results achieved in Profiles A and F. It's noticeable that increasing the number of iterations leads to an increase in utility. In this scenario, this happens due to the multiple starts of GRASP, which with a higher number of iterations can better explore the search space and consequently find better solutions. As we will see in the next section, in Section 6.3, the GRASP results fall short of the ideal time (1 second) in all scenarios. Therefore, we use a parameter of 100 iterations in our upcoming evaluations.
However, it is arguable that the algorithm presents good enough results for parameters. So, in scenarios where the execution time is much longer, it is possible to decrease it by reducing the amount of iterations or parallelizing the multi-start search.

6.3 Profiles evaluation



(a) Evaluation of the utility per profile



(b) Evaluation of execution time per profile

Figure 4: Time and utility of the proposed solution

In Figure 4, it is presented the results of the proposed solution in terms of utility and execution time. For this graph, we present also the confidence interval of 95% confidence for 100 executions.

Utility: For evaluating the utility of our proposal, we introduce a theoretical baseline that allows us to calculate an upper limit for the quality of solutions, given by the formula [9]. In this formula, the theoretical limit for the generated route is determined if the user visited the maximum number of locations in a day throughout (m) all their days of travel (p), and if all the locations visited received the maximum rating (rating 5). This value is then increased by 5, which represents the rating of the hotel and is counted only once.

$$teo = (5 \times p \times m) + 5 \tag{9}$$

The result of Equation 9 is equal to 55 for our evaluation scenario. This is the value for a theoretically optimal solution, given that we do not know if there exists a solution that satisfies all constraints and has a rating equal to 5.
In this scenario, comparing GRASP with the theoretical value, it is possible to see that the GRASP can achieve good quality results with a small computational time. In the future, we intend to investigate other scenarios in which this heuristic could fail.

Execution time Figure 4b, presents the execution time and confidence interval for different profiles. One of the factors impacting longer execution time for GRASP is its multiple initialization technique, where in this scenario, GRASP evaluates 100 solutions. Additionally, we observe that as the number of kilometers traveled per day increases, the execution

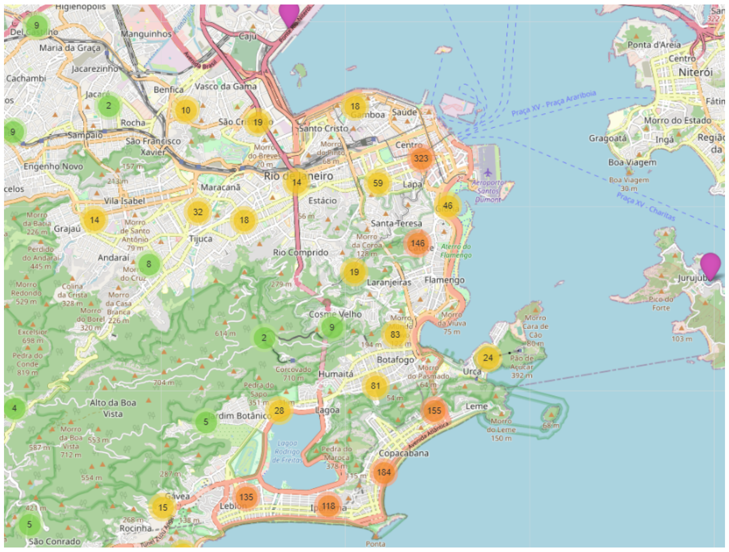


Figure 5: Presents the interface without the route generation.

time of our approach also rises. This happens because as we increase this characteristic of the profile, we consequently expand the list of candidate solutions, which impacts the computational cost of our technique. Finally, to avoid runtime issues in scenarios with an increase in the number of kilometers traveled per day, it would be possible to evaluate within the location selection criterion for LRC based on cost or some user preference factor.

6.4 Interface

The Figures 5, and 6, shows the application without and with the generated route. Currently, the application only has a few options for the users. The options are the parameters options of the model presented in Section 5. In the future, we expect to expand this interface to give users more personalization options like recommender systems and places' time windows.

7 Conclusions

In this work, we propose a metaheuristic-based technique for the Orienteering Problem, also known as the Tourist Problem. This problem combines elements of the knapsack problem and the traveling salesman problem and has been widely studied in the context of tourism. The objective is to find a route that maximizes user preferences while respecting location and user constraints. The study introduced the GRASP metaheuristic, utilizing real data from TripAdvisor for the city of Rio de Janeiro. We also propose an application for users to plan their travels.
Most of all, we also discuss different aspects that would be good to have in an application to recommend touristic routes to users. In this discussion, it is possible to notice different functionalities that are not available in our application but must be tackled in the future.

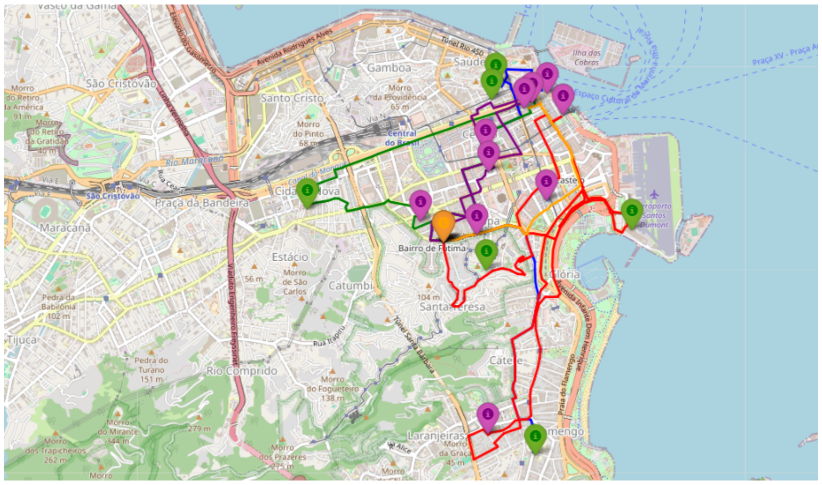


Figure 6: Presents the interface after the route is generated. Each colored edge is a daily route.

In the end, it is possible to notice that the proposed GRASP heuristic attends to some of the requirements, so it is possible to use it in real scenarios.

References

[1] Julio Brito, Airam Expíósito-Márquez, and José A Moreno. A fuzzy grasp algorithm for solving a tourist trip design problem. In *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6. IEEE, 2017.

[2] Lucas Gabriel da Silva Felix et al. Planet caravan: full trip planner. 2022.

[3] Airam Expósito, Simona Mancini, Julio Brito, and José A Moreno. A fuzzy grasp for the tourist trip design with clustered pois. *Expert Systems with Applications*, 127:210–227, 2019.

[4] Thomas A Feo and Mauricio GC Resende. Greedy randomized adaptive search procedures. *Journal of global optimization*, 6:109–133, 1995.

[5] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, and Grammati Pantziou. A survey on algorithmic approaches for solving tourist trip design problems. *Journal of Heuristics*, 20(3):291–328, 2014.

[6] Headspin. Why measuring and optimizing response time is critical for applications success. `https://shorturl.at/aTX56`, 2023. Accessed: 2023-10-17.

[7] Amir Khatibi, Fabiano Belem, Ana P Silva, Dennis Shasha, Marcos A Goncalves, et al. Improving tourism prediction models using climate and social media data: A fine-grained approach. In *Twelfth International AAAI Conference on Web and Social Media*, 2018.

[8] Serhan Kotiloglu, Theodoros Lappas, Konstantinos Pelechrinis, and PP Repoussis. Personalized multi-period tour recommendations. *Tourism Management*, 62:76–88, 2017.

[9] Zhixue Liao and Weimin Zheng. Using a heuristic algorithm to design a personalized day tour route in a time-dependent stochastic environment. *Tourism Management*, 68:284–300, 2018.

[10] José Ruiz-Meza, Julio Brito, and Jairo R Montoya-Torres. Multi-objective fuzzy tourist trip design problem with heterogeneous preferences and sustainable itineraries. *Sustainability*, 13(17):9771, 2021.

[11] José Ruiz-Meza, Julio Brito, and Jairo R Montoya-Torres. A grasp-vnd algorithm to solve the multi-objective fuzzy and sustainable tourist trip design problem for groups. *Applied Soft Computing*, 131:109716, 2022.

[12] José Ruiz-Meza and Jairo R Montoya-Torres. Tourist trip design with heterogeneous preferences, transport mode selection and environmental considerations. *Annals of Operations Research*, 305(1-2):227–249, 2021.

[13] Aries Susanty, Nia Budi Puspitasari, Singgih Saptadi, and Sidiq Prasetyo. Implementation of green tourism concept through a dynamic programming algorithm to select the best route of tourist travel. In *IOP conference series: Earth and environmental science*, volume 195, page 012035. IOP Publishing, 2018.

[14] El-Ghazali Talbi. *Metaheuristics: from design to implementation*. John Wiley & Sons, 2009.

[15] Pieter Vansteenwegen and Dirk Van Oudheusden. The mobile tourist guide: an or opportunity. *OR insight*, 20(3):21–27, 2007.

[16] Budhi S Wibowo and Monica Handayani. A genetic algorithm for generating travel itinerary recommendation with restaurant selection. In *2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 427–431. IEEE, 2018.

[17] Lunwen Wu, Tao Gu, Zhiyu Chen, Pan Zeng, and Zhixue Liao. Personalized day tour design for urban tourists with consideration to co2 emissions. *Chinese Journal of Population, Resources and Environment*, 20(3):237–244, 2022.

[18] Phatpicha Yochum, Liang Chang, Tianlong Gu, and Manli Zhu. An adaptive genetic algorithm for personalized itinerary planning. *IEEE Access*, 8:88147–88157, 2020.

[19] Weimin Zheng and Zhixue Liao. Using a heuristic approach to design personalized tour routes for heterogeneous tourist groups. *Tourism Management*, 72:313–325, 2019.