プロジェクト１・2 最終報告書

タイトル

容量制約付き配送計画問題を用いた旅行プラン最適化

チーム名

TripY（トリッピー）

メンバー

◎平松勇紀
〇宮澤航
〇小野寺太郎

プロジェクトの目的

充実した旅行プランを立てるにも，予算や目的地，移動手段や何をするか決めるなど沢山のステップを踏む必要がある．その中で，旅行の計画を立てるのが面倒な人や目的地だけ決めてあとは自由という人が一定数いる．しかし行先によっては，コロナウイルスの５類化により観光地が混んでしまい計画通りに楽しむことが出来ない，実際は無理な計画だったという問題が生まれる．また，Dockpit より「旅行」の検索ユーザーを見ると 2023 年 1 月は前年同月比 118%，2023 年 2 月は前年同月比 145%と増加が顕著であり，今まで我慢していた旅行欲が再燃していることが分かる．この問題を解決するには，制約内で目的地をどれほど回ることが出来るかを可視化，利益を最大化するようなプログラムを構築し，旅行へのハードル，面倒くささ，失敗を減らすことが求められ，ここに意義があると考える．本研究では，目的地の滞在時間をノード，移動時間を重みに置く，***容量制約付き配送計画問題を用いた旅行プランを作成する最適化アルゴリズムの構築***を目指す．本研究の特色は数理工学の観点から「旅行の満足度」を高めることにある．

また、最近はフッ軽という言葉がある．フッ軽とは「フットワークが軽い」という意味で用いられることのある省略表現．おおむね，遊びの誘いの連絡にすぐ反応するようなさまを指して用いられる．こういった場面で旅行に行く場合に行きたいところを指定するだけで日程を出力してくれるため，旅行を楽に，身近にしたいと考える．

今年度の目標

・配送計画問題とは何か理解する
・Python を用いた旅行プラン最適化のプログラムの作成
・上記をパッケージ化し、誰でも pip を通してインストールし使用できるようにする
・上記を PySimpleGUI を使用しアプリ化し、.exe 化し Python 環境を構築しなくとも誰でも利用できるようにする。

→　.exe は Mac 対応していないため streamlit を用いて web アプリケーションを作成することに変更

方法

記号の説明

$G=(V,E)$: 行きたい観光地の場所及び経路を表すグラフ (無向グラフ)

$V=\{0,1,...,n\}$: ノードの集合。ホテルを表すノード 0 と各観光地の場所を表すノード $\{1,...,n\}$ から成る

E : 各ノード間を結ぶエッジ $e_{ij}=(i,j)$ の集合

$K=\{1,2,...,|K|\}$: 車両の集合

$v_{ij}(\geq 0)$: 各観光地 i に滞在する時間

$Q(\geq 0)$: 一日に使える時間

c_{ij} : 観光地 i と j 間の移動時間

決定変数

決定変数は次のように定義する

k : 日付, e_{ij} : 経路

$$x_{ij}^{k}=\begin{cases}1 & (\text{日付 } k \text{ が経路 } e_{ij} \text{を通るとき})\\ 0 & (\text{それ以外})\end{cases}$$

$$Min\sum_{k\in K}^{\infty}\sum_{(i,j)\in E}^{\infty}(v_{ij}x_{ij}^{k}+c_{ij}x_{ij}^{k}) \tag{1}$$

subject to

$$\sum_{k\in K}^{\infty}\sum_{i\in V,i\neq j}^{\infty}x_{ij}^{k}=1 \qquad \forall j\in V\backslash\{0\} \tag{2}$$

$$\sum_{j\in V\backslash\{0\}}^{\infty}x_{0j}^{k}=1 \qquad \forall k\in K \tag{3}$$

$$\sum_{i\in V,i\neq j}^{\infty}x_{ij}^{k}-\sum_{i\in V}^{\infty}x_{ij}^{k}=0 \qquad \forall j\in V,\forall k\in K \tag{4}$$

$$\sum_{i\in V}^{\infty}\sum_{j\in V\backslash 0}^{\infty}(v_{ij}x_{ij}^{k}+c_{ij}x_{ij}^{k})\leq Q \qquad \forall k\in K \tag{5}$$

$$\sum_{k\in V}^{\infty}\sum_{(i,j)\in E}^{\infty}x_{ij}^{k}\leq |S|-1 \qquad S\subseteq V\backslash\{0\} \tag{6}$$

$$x_{ij}^{k}\in\{0,1\} \qquad \forall k\in K,\forall (i,j)\in E \tag{7}$$

宮澤作成（LaTeX）

(1)式は，目的関数である全日程の移動コストを最小化することを表している．

(2)式は，各観光地の場所に訪れるのは全日程で 1 度であるという制約.
(3)式は，ホテルから出発するという制約.
(4)式は，1 観光地に対し訪れる回数と離れる回数を同じにする制約.
(5)式は，各日程において容量が超えないようにする制約.
(6)式は，ホテルから繋がれていない経路を除去するための制約（部分巡回路除去制約）.
(7)式は，決定変数の制約.

プログラムの実装方法として，(使用したパッケージ)

```
import streamlit as st
import numpy as np
import pandas as pd
import pulp
import itertools
import folium
import openrouteservice
from branca.element import Figure
from openrouteservice import convert
import sys
import time
import urllib.parse
```

を使用した（コードをそのままコピーしたため import 文）.
また，パッケージ化，アプリ化するにあたり，用途に合わせた（スケジュールのみを表示したい，ルートのみを表示したい，または両方）対応をさせるため，クラスかをした.
上記の制約条件，目的関数を用いて最適化をし，スケジュールを出力（Dataframe 形式），ルートの可視化(Folium を使用して HTML 形式で表示)をしている.

パッケージ化階層（https://github.com/Yuki1005/TripOpt）

```
TripOpt
├── TripOpt
│   ├── __init__.py（main.py 内のクラスを直接呼び出せるようにするもの）
│   └── main.py(TripOpt のクラスを格納)
├── LICENSE(パッケージ化する際に必要なもの(形上これがないとパッケージ化できない))
├── REMIND.md（使用方法を GitHub 上に明記するため）
└── setup.py(バージョンを更新する際に変更，使用するために必要なモジュールの
             インストールをさせる文)
```

アプリ化階層（https://github.com/Yuki1005/TripOpt_App）

```
TripOpt_App
├── REMIND.md（使用方法を GitHub 上に明記するため）
├── requirements.txt(Streamlit にてアプリ化する際，パッケージに
│                    インストールするべきモジュールを書くもの(setup.py だとダメ))
└── main.py(TripOpt_App のクラスを格納)
```

Streamlit を GitHub でログインし，Create new app with GitHub Codespaces より
Streamlit のプロンプト上でプッシュされることで初めてアプリ化を実現させる.

活動経過

4,5 月

・方向性決定

・配送計画問題について先行研究を読む

6 月

・目標物に対しての参考となるプログラムや資料を集める

・上記を参考に計画を練る

・プログラム作成

7 月

・発表資料作成

・プログラム作成

9,10 月

・プログラム作成

11 月

・パッケージ化手法探索

・アプリ化手法探索

・発表を聞いて改善出来そうなところがあればする

12 月

・最終報告に向けての資料作り

・パッケージ化

・アプリ化

1 月

・成果発表

結果と観察

GitHub （https://github.com/Yuki1005/TripOpt_App ） に仕様が掲載してあり，OpenRouteService に登録してもらうことで利用することが可能である.

Yuki1005 / TripOpt_App

Code Issues Pull requests Actions Projects Security

Don't get locked out of your account. Download your recovery codes or add a passkey so you don't lose access when you get a new device.

旅行の予定を openrouteservice を用いて Python から最適化するためのアプリケーションです。

0 stars 0 forks 1 watching 1 Branch 0 Tags Activity

Public repository

main Go to file Code

Yuki1005 a last week

README.md	a	last month
logo.png	a	last month
main.py	a	last week
openroutesurvice.png	a	last month
requirements.txt	a	last month

README

Itinerary Optomization package for Python

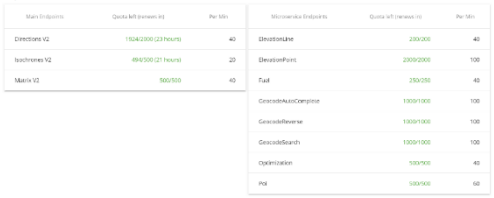
旅行の予定を openrouteservice を用いて Python から最適化するためのアプリケーションです。

使い方

準備

はじめに、openrouteserviceのページ からアカウントを作成してAPIキーを取得します。
APIキーを取得すると，下の図のように各機能が利用できる回数が表示されています。

例えば経路検索に使うDirections V2 は，1日で無料の範囲で2000回 (1分で40回まで) 利用できることが分かります。



次に行きたい場所のtextファイルを用意します。
GoogleMapから取得したURLと滞在時間を2行で1セットとします。
この時、1セット目にはホテルを入力し、滞在時間を0と入力してください。

例として

```
https://www.google.co.jp/maps/place/%E4%BA%AC%E9%83%BD%E9%A7%85/@34.985
0
https://www.google.co.jp/maps/place/%E8%83%80%E8%8C%82%E5%BE%A1%E7%A5%9
20
https://www.google.co.jp/maps/place/%E4%BC%8F%E8%A6%8B%E7%A8%B2%E8%8D%B
60
https://www.google.co.jp/maps/place/%E6%B8%85%E6%B0%B4%E5%AF%BA/@34.994
50
https://www.google.co.jp/maps/place/%E6%9D%B1%E5%B1%B1%E6%85%88%E7%85%A
40
https://www.google.co.jp/maps/place/%E4%BA%AC%E9%83%BD%E3%82%BF%E3%83%A
30
https://www.google.co.jp/maps/search/%E9%87%91%E9%96%A3%E5%AF%BA/@35.03
40
https://www.google.co.jp/maps/place/%E4%BB%81%E5%92%8C%E5%AF%BA/@35.028
50
https://www.google.co.jp/maps/place/%E5%BB%BA%E4%BB%81%E5%AF%BA/@35.000
50
```

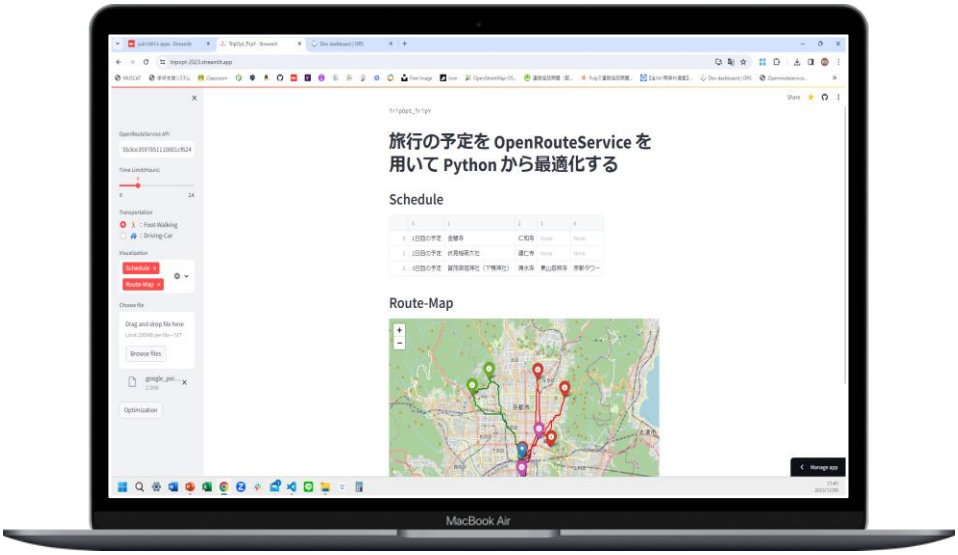
最適化

TripOpt_Appのページ

自分の行きたい地点の URL を Google Map から引っ張り，その地点の滞在時間を設定したテキストボックスを用意することにより，行きたい観光地に対し，一日に使用できる時間を考慮したうえで最適化した旅程を出力するとともに，どういったルートをたどるかを可視化することを可能とした．これを，TripOpt として pip にてインストールを可能に（https://github.com/Yuki1005/TripOpt），TripOpt_App としてアプリ化をし，実装をすることが出来た．（https://github.com/Yuki1005/TripOpt_App）

TripOpt_App(アプリ化したもの)の使用例：

PC 画面（使用例）



スマートフォン画面（使用例）

アプリを使用しているときに，右上に表示される RUNNING（最適化中）があとどれくらいかかるのかが分かるバーを実装できればより良いもの，ストレスフリーなアプリを実装できたと考える．

成果物

・最終的に使用しているコード（アプリ化、パッケージ化したものでない）
https://github.com/Yuki1005/Project2_Code_TripY

・過程で使用したコード
https://github.com/Yuki1005/Project2_Hozon

・パッケージ化（TripOpt、使用方法、コード）（Github）
https://github.com/Yuki1005/TripOpt

・アプリ化（TripOpt_app、使用方法、コード）（Github）
https://github.com/Yuki1005/TripOpt_App

・アプリ URL（Streamlit）
https://tripopt-2023.streamlit.app

・ハピネスオプティマイザー（最適化グループ）のアプリ化手伝い（平松）（GitHub）
https://github.com/Yuki1005/Project2_Riku_Ryo

残された課題

・OpenRouteService を使用して拠点間の移動時間を出力しているが、無料サービスが故、一日に使える回数が決まっているかつ、導出するまでの速度が遅い点を解決したいと考える。
　→利用制限のない GoogleAPI を使用することにより多少は解決するが、大きな解決にはからないと考える。
・最適化した日程が１９日を超えると，ルートを可視化するために必要な色がなくなってしまう（色コードで表示しているわけではないため，matplotlib.pyplot のサポートされている色のリストを作成したため）ため，改善していけたらよいと考えた．しかし最適化する際に pulp では計算できない量の拠点数になっている恐れがあるため(19 日も使用することは限りなくないかつ，pulp の性能があまりよくないため)改善しようがない可能性もある．

参考にした文献・資料・情報

先行研究名：配送計画問題に対する発見的解法
著者：橋本秀樹、胡艶楠

URL： https://www.jstage.jst.go.jp/article/isciesci/64/6/64_218/_article/-char/ja/

先行研究名：時間枠制約付き配送計画問題に対する局所探索法の適用について
著者：増田友泰、柳浦睦憲、茨木俊秀
URL： https://orsj.org/wp-content/or-archives50/pdf/bul/Vol.45_01_034.pdf

OpenRouteService,参照：2023/12/22
URL：https://openrouteservice.org/

“IMAX® Countdown (Cameras)”, YouTube, uploaded by IMAX
Jul, 11 Apr. 2017, https://www.youtube.com/watch?v=n5HbQ7vCvDY

Python+folium+openrouteservice を使う（経路・時間行列・等時線を例に）
参照：2023/12/22 , URL: https://zenn.dev/takilog/articles/2be029ccd35972

A faster way to build and share data apps, Streamlit,
参照：2023/12/22 , URL: https://streamlit.io/

運搬経路問題(VRP)を解く　混合整数計画編，Qiita，@shinji071(信二 岩城)
参照：2023/12/22 , URL: https://qiita.com/shinji071/items/2f593d2610cdaf2ca5e6

購入物品リスト

なし