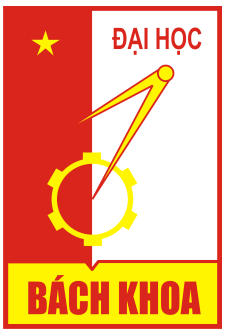
HÀ QUANG HANH

BÀI TOÁN ĐỊNH TUYẾN PHƯƠNG TIỆN VÀ ỨNG DỤNG TRONG HỆ THỐNG ĐIỀU PHỐI XE TRUNG CHUYỂN

HÀ NỘI - 2024

ĐẠI HỌC BÁCH KHOA HÀ NỘI
KHOA TOÁN - TIN



HÀ QUANG HANH

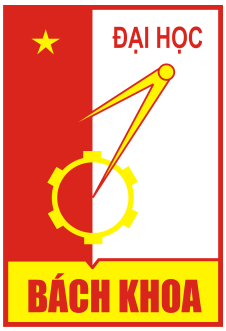
BÀI TOÁN ĐỊNH TUYẾN PHƯƠNG TIỆN VÀ ỨNG DỤNG TRONG HỆ THỐNG ĐIỀU PHỐI XE TRUNG CHUYỂN

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

Chuyên ngành: TOÁN TIN

HÀ NỘI - 2024

ĐẠI HỌC BÁCH KHOA HÀ NỘI
KHOA TOÁN - TIN



BÀI TOÁN ĐỊNH TUYẾN PHƯƠNG TIỆN VÀ ỨNG DỤNG TRONG HỆ THỐNG ĐIỀU PHỐI XE TRUNG CHUYỂN

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

Chuyên ngành: TOÁN TIN

Giảng viên hướng dẫn: TS. LÊ HẢI HÀ
Sinh viên thực hiện: HÀ QUANG HANH
Mã sinh viên: 20195870

HÀ NỘI - 2024

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

1. Mục tiêu và nội dung của đồ án:

2. Kết quả đạt được:

3. Ý thức làm việc của sinh viên:

Hà Nội, ngày . . . tháng . . . năm

Giảng viên hướng dẫn

Mục lục

Mở đầu **7**

Chương 1. Bài toán định tuyến phương tiện **8**

1.1 Giới thiệu chung 8

1.2 Phát biểu bài toán 10

1.3 Mô hình toán học 13

1.4 Một số biến thể của VRP 17

1.4.1 Ràng buộc về khả năng xử lý 18

1.4.2 Ràng buộc về độ dài lộ trình 18

1.4.3 Ràng buộc về backhaul 19

1.4.4 Ràng buộc về thời gian 19

1.4.5 Ràng buộc về lấy và giao hàng 20

1.5 Một số phương pháp giải 21

1.5.1 Thuật toán Metaheuristic 21

1.5.2 Thuật toán Local Search 22

Chương 2. Hệ thống điều phối xe trung chuyển **24**

2.1 Đặt vấn đề 24

2.2 Phân tích bài toán 26

2.2.1 Bài toán trả khách 26

2.2.2 Bài toán đón khách 27

2.3 Nhận định và hướng giải pháp 29

2.3.1 Nhận định 29

2.3.2 Hướng giải pháp 29

Chương 3. Giải pháp và cài đặt **31**
3.1 Công cụ OR-Tools 31
3.1.1 Giới thiệu OR-Tools 31
3.1.2 Cách cài đặt OR-Tools 32
3.1.3 Bài toán định tuyến phương tiện với OR-Tools 32
3.2 Thu thập dữ liệu đầu vào 35
3.2.1 Giới thiệu Geoapify 35
3.2.2 Lấy và xử lý dữ liệu từ Geoapify 35

Chương 4. Thử nghiệm và kết quả **40**
4.1 Thử nghiệm 1 40
4.1.1 Mô tả dữ liệu 40
4.1.2 Kết quả 41
4.2 Thử nghiệm 2 43
4.2.1 Mô tả dữ liệu 43
4.2.2 Kết quả 45
4.3 Thử nghiệm 3 46
4.3.1 Mô tả dữ liệu 47
4.3.2 Kết quả 48

Kết luận và hướng phát triển **50**

Tài liệu tham khảo **51**

Danh sách hình vẽ

1.1 Bài toán Người du lịch (TSP) 11
1.2 Bài toán Định tuyến phương tiện (VRP) 11
1.3 Ví dụ minh hoạ cho hàm mục tiêu 14
1.4 Kết quả minh hoạ cho hàm mục tiêu 15
1.5 Bài toán VRP cơ bản và các biến thể 17

2.1 Sơ đồ quy trình trả khách 26
2.2 Sơ đồ quy trình trả khách tình huống 1 27
2.3 Sơ đồ quy trình đón khách 28
2.4 Sơ đồ quy trình đón khách tình huống 1 28

3.1 Tạo ma trận khoảng cách/thời gian từ Geoapify 36

4.1 Hình ảnh mô tả dữ liệu thử nghiệm 1 41
4.2 Hình ảnh mô tả dữ liệu thử nghiệm 2 44
4.3 Hình ảnh mô tả dữ liệu thử nghiệm 3 47

Danh mục viết tắt

API	Application Programing Interface
CVRP	Capacitated Vehicle Routing Problem
DCVRP	Distance-Constrained Vehicle Routing Problem
TSP	Travelling Salesman Problem
VRP	Vehicle Routing Problem
VRPB	Vehicle Routing Problem with Backhaul
VRPTW	Vehicle Routing Problem with Time Windows
VRPPD	Vehicle Routing with Pickups and Deliveries

Mở đầu

Trong lĩnh vực logistics, vận tải là một yếu tố then chốt và quan trọng trong dây chuyền, chất lượng của dịch vụ logistics phụ thuộc không nhỏ vào chất lượng của dịch vụ vận tải. Khi hoạt động vận tải được tổ chức thực hiện một cách tối ưu, chất lượng tốt sẽ góp phần đáng kể nâng cao chất lượng dịch vụ logistics. Việc tối ưu trong vận tải chính là việc điều phối các phương tiện vận chuyển hàng hoá sao cho tối ưu chi phí vận chuyển hay còn gọi là bài toán định tuyến phương tiện.

Bài toán định tuyến phương tiện (Vehicle Routing Problem) là bài toán xác định các lộ trình tối ưu cho một tập xe để vận chuyển nhằm phục vụ các khách hàng ở các vị trí khác nhau. Đây là bài toán có nhiều ứng dụng trong thực tế từ việc cung cấp nguyên liệu đến việc phân phối sản phẩm trong các nhà máy sản xuất, các công ty dịch vụ vận chuyển như hàng hoá, bưu phẩm, khách hàng. Lời giải của bài toán này là các lộ trình tối ưu của các xe giúp chi phí vận chuyển sẽ tối ưu khi di chuyển theo lộ trình này.

Đồ án này nghiên cứu về bài toán định tuyến phương tiện, trình bày về khái niệm, mô hình bài toán; các phương pháp và một số công cụ để giải bài toán này. Qua đó, phân tích và sử dụng các giải pháp để thực hiện giải quyết một bài toán định tuyến xe trung chuyển thực tế của một công ty trung chuyển khách.

Tại đây, em xin cảm ơn thầy Lê Hải Hà đã hướng dẫn, giúp đỡ em hoàn thành bài báo cáo này.

Em xin chân thành cảm ơn!

Chương 1

Bài toán định tuyến phương tiện

1.1 Giới thiệu chung

Bài toán định tuyến phương tiện (Vehicle Routing Problem) là một bài toán quan trọng trong lĩnh vực quản lý chuỗi cung ứng và là một dạng bài toán tối ưu hóa. Bài toán này liên quan đến việc tìm lộ trình hiệu quả nhất cho một tập hợp các phương tiện di chuyển để phục vụ một số khách hàng hay địa điểm cụ thể và đáp ứng nhu cầu cung ứng hoặc dịch vụ.

Cụ thể, bài toán yêu cầu xác định cách phân phối một tập hợp các khách hàng và hàng hóa từ một hoặc nhiều điểm xuất phát đến một hoặc nhiều điểm đích, sử dụng một hoặc nhiều phương tiện di chuyển. Mục tiêu là tối ưu hóa một số tiêu chí nhất định, như chi phí vận chuyển, thời gian di chuyển, hoặc số lượng phương tiện sử dụng.

Các biến thể của bài toán có thể bao gồm các yếu tố như thời gian cửa hàng, hạn chế tải trọng và dung lượng của phương tiện, ưu tiên ưu đãi cho một số điểm cụ thể và các ràng buộc về quy định lịch trình.

Giải quyết bài toán giúp tăng cường hiệu suất và giảm thiểu chi phí trong quá trình

vận chuyển hàng hóa, điều này quan trọng đặc biệt đối với các doanh nghiệp hoạt động trong lĩnh vực logistics và vận tải. Các phương pháp giải quyết Vehicle Routing Problem thường sử dụng các thuật toán tối ưu hóa và kỹ thuật heuristics để tìm kiếm giải pháp tối ưu hoặc gần tối ưu.

Một số ứng dụng:

Vehicle Routing Problem (VRP) có nhiều ứng dụng trong thực tế, đặc biệt là trong lĩnh vực quản lý chuỗi cung ứng, logistics và vận tải. Dưới đây là một số ứng dụng phổ biến của VRP:

1. Quản lý điều phối và giao hàng:
    - Giao hàng hàng ngày: Đối với các doanh nghiệp bán lẻ, dịch vụ giao hàng hàng ngày, VRP giúp tối ưu hóa lịch trình vận chuyển để giảm chi phí và thời gian giao hàng.
    - Dịch vụ giao hàng nhanh: Các dịch vụ giao hàng nhanh cần quy hoạch hành trình linh hoạt để đáp ứng yêu cầu người tiêu dùng và giảm thời gian chờ đợi.
2. Dịch vụ vận tải công cộng:
    - Giao thức Uber for Business: Các nền tảng giao thông công cộng như Uber for Business cũng sử dụng giải pháp VRP để tối ưu hóa các chuyến đi của xe để đáp ứng nhu cầu người dùng.
3. Quản lý dịch vụ tổ chức và hội nghị:
    - Giao hàng và dịch vụ tổ chức sự kiện: Trong ngành tổ chức sự kiện, VRP có thể được sử dụng để tối ưu hóa quá trình giao hàng và dịch vụ trong các sự kiện lớn.
4. Thu gom rác và dịch vụ điều phối:
    - Thu gom rác: Các doanh nghiệp quản lý rác thải sử dụng VRP để tối ưu hóa lịch trình của xe thu gom rác và giảm thiểu quãng đường di chuyển.

- Dịch vụ điều phối dựa trên nhu cầu: Các dịch vụ cung ứng đồ ăn, thực phẩm, hoặc đồ đạc dựa trên đơn đặt hàng cũng có thể sử dụng VRP để tối ưu hóa quá trình vận chuyển.

5. Quản lý hệ thống cung ứng:

   - Quản lý cung ứng trong ngành bán lẻ: Trong ngành bán lẻ, VRP giúp tối ưu hóa lịch trình giao hàng từ các nhà cung ứng đến các cửa hàng bán lẻ.

6. Quản lý dịch vụ bảo dưỡng và sửa chữa:

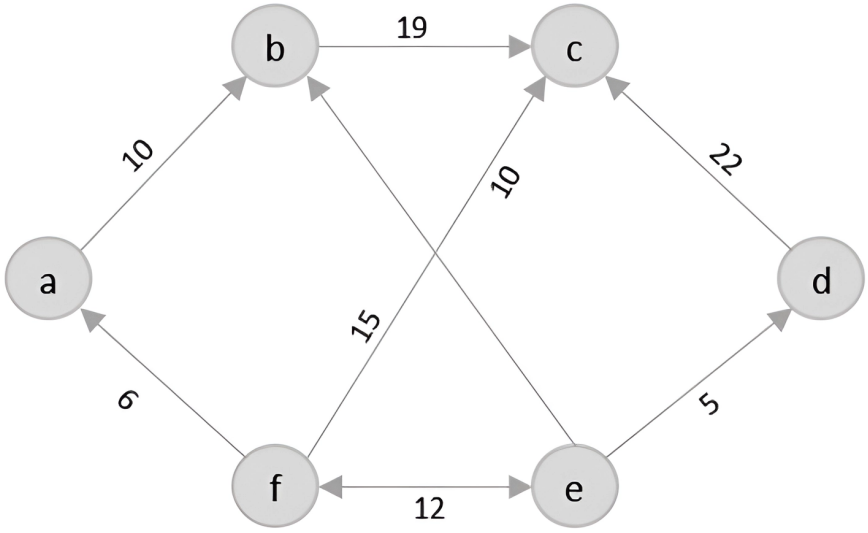
   - Dịch vụ bảo dưỡng và sửa chữa công nghiệp: Các doanh nghiệp chuyên cung cấp dịch vụ bảo dưỡng và sửa chữa có thể sử dụng VRP để lên lịch trình các cuộc gặp khách hàng và đảm bảo hiệu suất cao.

7. Quản lý dịch vụ y tế:

   - Giao dịch dược phẩm: Trong ngành y tế, VRP có thể được áp dụng để quản lý và tối ưu hóa quá trình giao dịch dược phẩm từ nhà thuốc đến bệnh nhân.

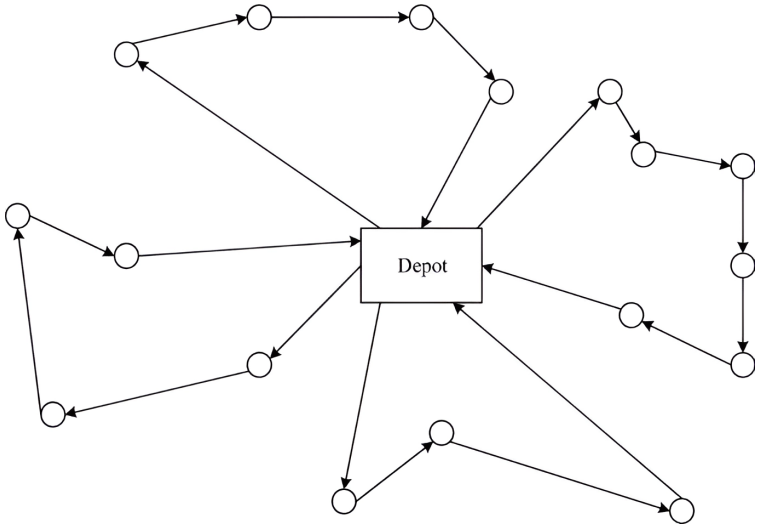
1.2 Phát biểu bài toán

Bài toán định tuyến phương tiện (Vehicle Routing Problem - VRP) dựa trên một bài toán tối ưu rất đơn giản được gọi là bài toán người du dịch hay còn gọi là bài toán người bán hàng (Travelling Salesman Problem - TSP).



Hình 1.1: Bài toán Người du lịch (TSP)

Bài toán TSP là trường hợp đơn giản nhất của bài toán VRP với một người bán hàng hay một xe giao hàng duy nhất. Trong bài toán này, có một số lượng đã cho các thành phố. Xuất phát từ một thành phố, người bán hàng sẽ đi giao hàng tại tất cả các thành phố và quay trở về thành phố ban đầu với yêu cầu của bài toán là phải tìm một lộ trình tối ưu nhất (tổng chi phí di chuyển là nhỏ nhất) để người bán hàng đi giao hàng cho tất cả thành phố, mỗi thành phố được ghé thăm duy nhất một lần hay là tìm chu trình ngắn nhất để thăm mỗi thành phố đúng một lần.



Hình 1.2: Bài toán Định tuyến phương tiện (VRP)

Trên cơ sở mở rộng bài toán TSP, bài toán VRP cơ bản bao gồm một tập các xe được tập kết tại kho hàng (nơi xuất phát) và mỗi khách hàng có các yêu cầu vận chuyển khác nhau. Vấn đề đặt ra là phải tìm cách định tuyến cho tập các xe phục vụ được tất cả khách hàng với chi phí vận chuyển là nhỏ nhất (chi phí vận chuyển có thể là độ dài quãng đường, thời gian,...).

Bài toán VRP là một bài toán tối ưu tổ hợp, ở dạng cơ bản nhất của VRP thì bài toán giống với bài toán TSP chỉ khác về số lượng xe. Nhưng sự khác biệt lớn nhất giữa VRP và TSP là VRP là một bài toán tối ưu đa ràng buộc có nghĩa là nhiều hơn một ràng buộc cần phải được đáp ứng (ví dụ như khoảng cách, thời gian, sức chứa của xe,...).

Các khái niệm được sử dụng trong bài toán VRP bao gồm:

- Phương tiện (Vehicle): Một tập hợp các phương tiện di chuyển có sẵn để vận chuyển hàng hóa. Các phương tiện này có thể có các hạn chế về dung lượng, tải trọng, thời gian hoạt động, và các ràng buộc khác.
- Điểm xuất phát (Depot): Điểm xuất phát là nơi mà các phương tiện bắt đầu và kết thúc hành trình của mình. Tại đây, hàng hóa được nạp lên phương tiện và sau đó được phân phối đến các địa điểm khác.
- Khách hàng (Customer): Các điểm cần được đến để phục vụ yêu cầu vận chuyển. Mỗi điểm đích có thể có một lượng cố định của hàng hóa cần được giao đến.
- Lộ trình (Route): Một lộ trình là lịch trình di chuyển của một phương tiện từ điểm xuất phát, đi qua các điểm đích, và quay lại điểm xuất phát. Mục tiêu là tối ưu hóa hành trình để giảm chi phí hoặc thời gian vận chuyển.
- Các ràng buộc (Constraints): Các ràng buộc như thời gian làm việc của phương tiện, giới hạn dung lượng và tải trọng, thời gian đến cửa hàng, và các yếu tố khác cũng được xem xét trong quá trình giải quyết VRP.

$\Rightarrow$ Có thể phát biểu bài toán VRP cơ bản một cách đơn giản như sau:

Có một tập hợp V phương tiện giống nhau cùng xuất phát tại một kho hàng đi làm nhiệm vụ giao hàng cho tập C khách hàng, mỗi khách hàng cung cấp một lượng hàng nhất định. Yêu cầu đặt ra của bài toán là tìm đường đi ngắn nhất cho V phương tiện phục vụ được tất cả các yêu cầu của khách hàng (các ràng buộc bài toán).

1.3 Mô hình toán học

Bài toán VRP được biểu diễn bởi một tập V các xe vận chuyển, một tập $C = \{1, 2, \ldots, n\}$ các khách hàng và một đồ thị có hướng G. Đồ thị G có $|C| + 2$ đỉnh, các khách hàng là các đỉnh từ $1, 2, \ldots, n$; đỉnh 0 biểu thị nơi xuất phát và đỉnh $n + 1$ biểu diễn cho nơi kết thúc. Tập A gồm các cung đồ thị biểu diễn kết nối các điểm xuất phát, kết thúc với khách hàng và giữa các khách hàng với nhau. Mỗi cung (i, j) có giá trị chi phí c_{ij}. Một lộ trình bắt đầu từ nơi xuất phát, đi qua một số khách hàng và kết thúc tại nơi kết thúc, mỗi khách hàng được một xe phục vụ. Mục tiêu của bài toán là tìm một phương án điều đội xe hợp lý để giao hàng tới tất cả các khách hàng sao cho tổng chi phí là nhỏ nhất.
Mô hình toán học của một bài toán VRP cơ bản như sau, giả sử:

- n: tổng số khách hàng cần phục vụ;
- K: tổng số chuyến xe;
- $N = \{0, 1, \ldots, n + 1\}$: tập các đỉnh của đồ thị;
- $V = \{0, 1, \ldots, K\}$: tập các xe vận chuyển;
- $C = \{1, 2, \ldots, n\}$: tập các khách hàng;
- c_{ij}: chi phí đi từ i đến j;

Khi đó ta có các biến trong mô hình:

x_{ijk}: biểu diễn xem một xe thứ k có đi từ khách hàng i đến khách hàng j không với

$$x_{ijk} = \begin{cases} 1 & \text{nếu xe thứ } k \text{ đi từ } i \text{ tới } j \\ 0 & \text{với các trường hợp còn lại} \end{cases} \quad \text{(trong đó } i, j \in N; k \in V\text{).}$$

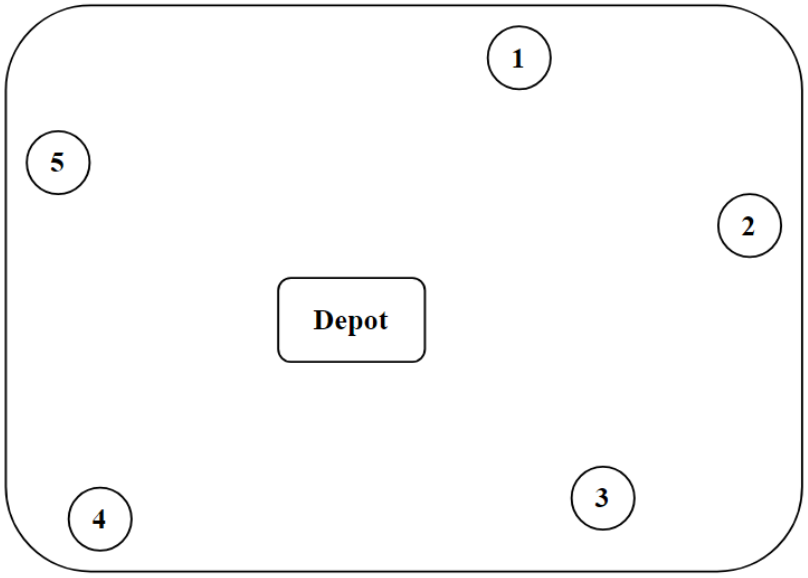
y_{ik}: biểu diễn khách hàng i có được xe k phục vụ không với

$$y_{ik} = \begin{cases} 1 & \text{nếu xe thứ } k \text{ đi đến } i \\ 0 & \text{với các trường hợp còn lại} \end{cases} \quad \text{(trong đó } i \in N; k \in V\text{).}$$

Hàm mục tiêu của bài toán:

$$\min \sum_{i \in N} \sum_{j \in N} \sum_{k \in V} c_{ij} x_{ijk}$$

Ví dụ minh hoạ:



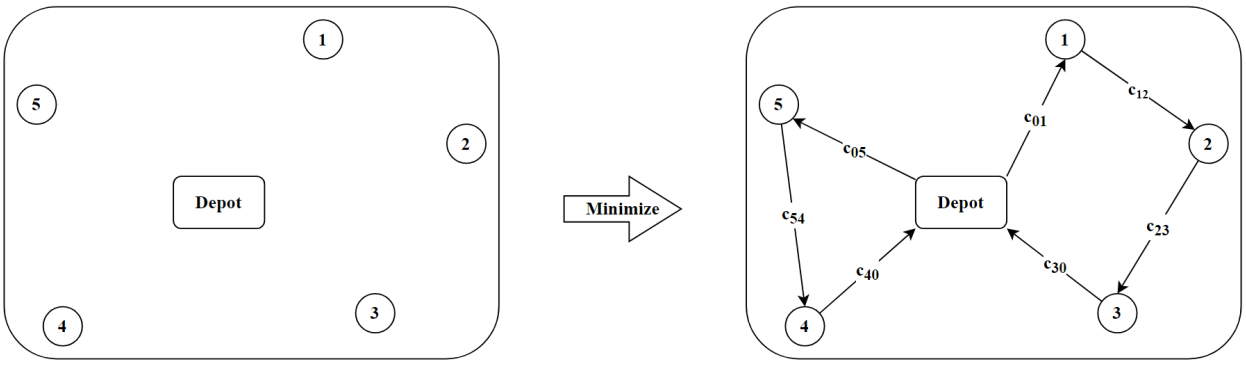
Hình 1.3: Ví dụ minh hoạ cho hàm mục tiêu

Ta có như trên Hình 1.3, có 2 xe ($V = \{0, 1\}$) đang ở tại kho (Depot) và 5 khách hàng ở năm vị trí khách hàng khác nhau ($N = \{0, 1, \ldots, 5\}$), ở đây ta có nơi xuất phát (đỉnh 0) và nơi kết thúc (đỉnh $n + 1$) cùng là tại Depot, ta có ma trận chi phí c_{ij} như sau:

Vị trí	0	1	2	3	4	5
0	0	4	6	3	3	4
1	4	0	3	7	10	8
2	6	3	0	4	11	12
3	3	7	4	0	8	10
4	3	10	11	8	0	4
5	4	8	12	10	4	0

Bảng 1.1: Ma trận chi phí c_{ij}

Sau khi tính toán các giá trị hàm mục tiêu, ta tìm được kết quả minimize của hàm mục tiêu là: $(c_{05} + c_{54} + c_{40}) + (c_{01} + c_{12} + c_{23} + c_{30}) = 11 + 14 = 25$. Trong đó, 11 là chi phí cho xe thứ nhất và 14 là chi phí cho xe thứ hai.



Hình 1.4: Kết quả minh hoạ cho hàm mục tiêu

Các ràng buộc của bài toán:

$$\sum_{i \in N} x_{ijk} = y_{jk} \quad \forall j \in N, k \in V \tag{1.1}$$

$$\sum_{i \in N} \sum_{k \in K} x_{ijk} = 1 \quad \forall j \in C \tag{1.2}$$

$$\sum_{i \in N} x_{ijk} - \sum_{i \in N} x_{jik} = 0 \quad \forall j \in C, \quad k \in V \tag{1.3}$$

$$\sum_{j \in N} x_{0jk} = 1 \quad \forall k \in V \tag{1.4}$$

$$\sum_{i \in N} x_{i0k} = 0 \quad \forall k \in V \tag{1.5}$$

$$\sum_{i \in N} x_{i(n+1)k} = 1 \quad \forall k \in V \tag{1.6}$$

$$\sum_{j \in N} x_{(n+1)jk} = 0 \quad \forall k \in V \tag{1.7}$$

Trong đó:

- Ràng buộc (1.1) biểu diễn mối liên hệ giữa biến x và biến y.
  Nói cách khác, tổng của các giá trị x_{ijk} cho mỗi khách hàng i đối với mọi j và k phải bằng giá trị của y_{jk}. Điều này có thể hiểu là nếu khách hàng j được phục vụ bởi xe k ($y_{jk} = 1$), thì sẽ có một chuyến đi từ khách hàng i đến điểm j được thực hiện bởi xe k ($x_{ijk} = 1$). Ngược lại, nếu khách hàng j không được phục vụ bởi xe k ($y_{jk} = 0$), thì không có chuyến đi nào từ khách hàng i đến bất kỳ điểm đến j nào được thực hiện bởi xe k ($x_{ijk} = 0$).
  Ví dụ trong kết quả Hình 1.4: Khách hàng thứ 2 ($j = 2$) được xe thứ 2 phục vụ ($k = 1$) thì $\sum_{i \in N} x_{i21} = x_{021} + x_{121} + ... + x_{621} = 0 + 1 + 0 + 0 + 0 + 0 + 0 = 1 = y_{21}$

- Ràng buộc (1.2), (1.3) đảm bảo mỗi khách chỉ được một xe phục vụ.
  Ràng buộc (1.2) thể hiện tổng của các giá trị x_{ijk} theo i và k cho mỗi khách hàng j phải bằng 1. Điều này đảm bảo rằng mỗi khách hàng chỉ được phục vụ

bởi một xe duy nhất.

Ví dụ trong kết quả Hình 1.4: Với khách hàng thứ 3 $j = 3$ thì $\sum_{k \in K} x_{i3k} = x_{231} = 1$

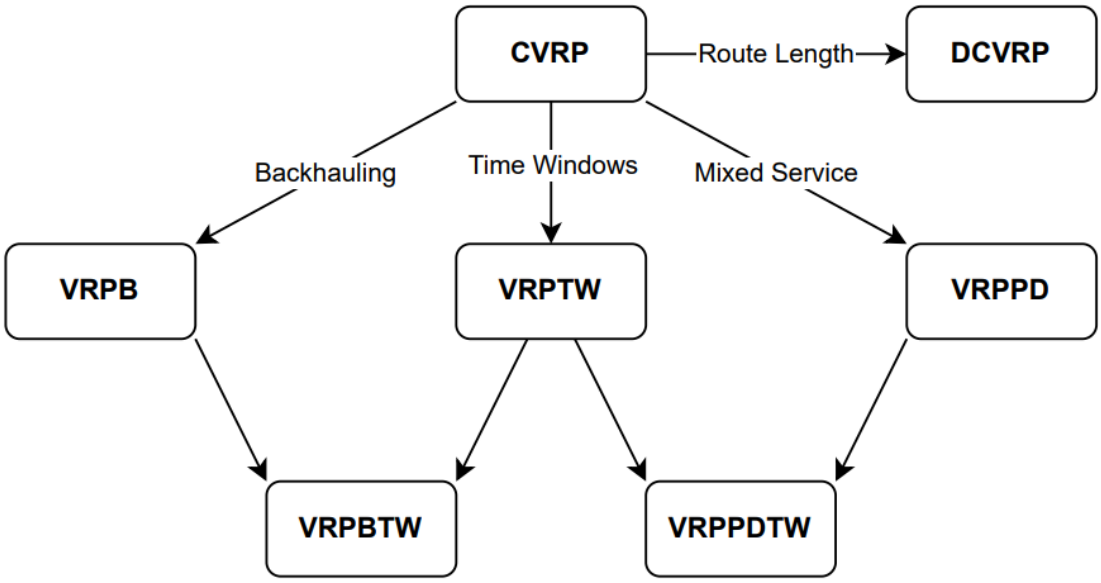
Ràng buộc (1.3) đảm bảo rằng nếu xe đi từ một khách hàng i đến khách hàng j thì lần di chuyển tiếp theo phải bắt đầu từ khách hàng j và tổng số chuyến đi từ khách hàng i đến khách hàng j bằng tổng số chuyến đi từ khách hàng j đến khách hàng khác.

- Ràng buộc (1.4), (1.5), (1.6), (1.7) đảm bảo mọi lộ trình đều bắt đầu và kết thúc tại kho.

  Nếu nơi kết thúc cũng là nơi bắt đầu thì các ràng buộc này có thể thay bằng hai ràng buộc: $\sum_{j \in N} x_{0jk} = 1 \quad \forall k \in V$ và $\sum_{i \in N} x_{i0k} = 1 \quad \forall k \in V$.

1.4 Một số biến thể của VRP

Bài toán VRP có rất nhiều biến thể dựa trên các yêu cầu vận chuyển cụ thể của các bài toán thực tế.



Hình 1.5: Bài toán VRP cơ bản và các biến thể

1.4.1 Ràng buộc về khả năng xử lý

Bài toán VRP với ràng buộc về khả năng xử lý (Capacitated Vehicle Routing Problem - CVRP): Bài toán bao gồm một tập xe và một tập yêu cầu của khách hàng, mỗi xe đều có một giới hạn về sức chứa, mục tiêu của bài toán này là phải tìm đường đi cho tất cả các xe phục vụ được hết các yêu cầu của khách hàng sao cho tổng lượng hàng hóa mà xe phải chở trong một lộ trình không được vượt quá sức chứa của xe.

Ta có, với N là tập các đỉnh của đồ thị và V là tập các xe (như trong phần 1.3):

- Mỗi khách hàng i có nhu cầu $d_i \quad (\forall i \in N)$.
- q_k: sức chứa của xe $k \quad (\forall k \in V)$.
- y_{ik}: biểu diễn khách hàng i có được xe k phục vụ không.

ràng buộc sau đảm bảo không vượt quá sức chứa của xe:

$$\sum_{i \in N} y_{ik} d_i \leq q_k \quad \forall k \in V \tag{1.8}$$

1.4.2 Ràng buộc về độ dài lộ trình

Bài toán VRP với ràng buộc độ dài quãng đường đi tối đa của mỗi xe (Distance Constrained - DCVRP): trong bài toán ứng với mỗi loại xe có một tham số để giới hạn độ dài quãng đường tối đa mà xe được phép đi.

Mục tiêu của bài toán này là định tuyến cho tập các xe đi làm nhiệm vụ sao cho tổng quãng đường mà mỗi xe phải đi không được vượt quá tham số giới hạn đã đặt ra.

Ta có, với:

- c_{ij}: chi phí (độ dài quãng đường) đi từ i đến j;
- x_{ijk}: biến quyết định biểu diễn xem một xe thứ k có đi từ khách hàng i đến khách hàng j không;

- S_k: độ dài quãng đường tối đa cho xe k.

ràng buộc được biểu diễn như sau:

$$\sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk} \leq S_k \quad \forall k \in V \tag{1.9}$$

1.4.3 Ràng buộc về backhaul

Bài toán VRP với ràng buộc về Backhaul (Vehicle Routing Problem with Backhauls - VRPB) là bài toán mở rộng của CVRP trong đó tập khách hàng C được phân chia thành hai tập con. Tập con đầu tiên $L = 1, \ldots, z$ chứa z khách hàng Linehaul, mỗi khách hàng yêu cầu một lượng hàng hoá nhất định phải được giao. Tập con thứ hai $B = z + 1, \ldots, z + t$ chứa t khách hàng Backhaul, nơi mà các hàng hoá được yêu cầu giao trước đưa giao đến.

Trong VRPB, có một ràng buộc giữa các khách hàng Linehaul và Backhaul là: khi một tuyến đường phục vụ cả hai loại khách hàng, tất cả khách hàng Linehaul phải được phục vụ trước khi bất kỳ khách hàng Backhaul nào được phục vụ.

Với $C = \{1, 2, \ldots, n\}$ là tập các khách hàng với $z + t = n$ và x_{ijk} là biến quyết định, ràng buộc được biểu diễn như sau:

$$\sum_{i \in B} x_{ijk} = 0 \quad \forall j \in L, \quad k \in V \tag{1.10}$$

1.4.4 Ràng buộc về thời gian

Bài toán định tuyến phương tiện với ràng buộc về thời gian (Vehicle Routing Problem with Time Windows - VRPTW) là bài toán mở rộng của bài toán CVRP, ở đây mỗi khách hàng sẽ chỉ cho phép xe đến giao hoặc nhận hàng trong khoảng thời gian nhất định. Mỗi khách hàng i được gán với một khoảng thời gian $[a_i, b_i]$ (gọi là cửa sổ thời gian - time windows). Khách hàng i chỉ cho phép xe đến giao/nhận hàng trong khoảng thời gian $[a_i, b_i]$, trường hợp nếu xe đến giao/nhận hàng cho khách hàng i

vào trước thời điểm a_i, xe sẽ phải đứng chờ cho đến thời điểm a_i mới được giao/nhận hàng và phải kết thúc việc giao/nhận hàng trước thời điểm b_i.

Tùy theo giá trị a_i, b_i trong cửa sổ thời gian $[a_i, b_i]$, có một số dạng cửa sổ thời gian sau:

- Cửa sổ thời gian hai bên nếu $0 < a_i \leq b_i < \infty$
- Cửa sổ thời gian một bên nếu $a_i = \infty$ hoặc $b_i = \infty$

Ở biến thể này, ta có thêm 2 giá trị sau:

- s_{ik}: là thời điểm xe k bắt đầu phục vụ khách hàng i.
- t_{ij}: là chi phí thời gian đi từ khách hàng i đến khách hàng j.

Ràng buộc được thể hiện như sau:

$$x_{ijk}(s_{ik} + t_{ij} - s_{jk}) \leq 0 \quad \forall i, j \in N, \quad k \in V \tag{1.11}$$

$$a_i \leq s_{ik} \leq b_i \quad \forall i \in N, \quad k \in V \tag{1.12}$$

Trong đó:

- Ràng buộc (1.11) đảm bảo rằng nếu xe k đi từ khách hàng i đến khách hàng j với chi phí thời gian t_{ij}, thì thời điểm bắt đầu phục vụ khách hàng j phải lớn hơn hoặc bằng thời điểm rời khách hàng i cộng với thời gian đi từ i đến j.
- Ràng buộc (1.12) đảm bảo thời điểm xe k đến khách hàng i phải nằm trong khoảng cửa sổ thời gian của khách hàng i.

1.4.5 Ràng buộc về lấy và giao hàng

Bài toán VRP với ràng buộc về lấy và giao hàng (Vehicle Routing Problem with Pickup and Delivery - VRPPD): trong bài toán này, tập các khách hàng $C = 1, \ldots, n$ sẽ có một danh sách các cặp khách hàng $[a, b]$, trong đó khách hàng a là nơi lấy hàng

và khách hàng b là nơi để giao hàng. Trong mỗi lộ trình đi của xe nếu phục vụ khách hàng a thì sẽ phục vụ khách hàng b.

$$\sum_{k \in V} x_{abk} = 1 \quad a, b \in C \tag{1.13}$$

1.5 Một số phương pháp giải

Bài toán định tuyến xe và các biến thể của nó được coi là bài toán tối ưu tổ hợp có độ phức tạp tính toán cao và được phân loại thuộc lớp NP- khó. Các phương pháp giải cho bài toán này được chia làm 2 loại chính: các phương pháp chính xác và các phương pháp gần đúng.

Các phương pháp chính xác thường được áp dụng để đưa ra lời giải tối ưu cho bài toán. Các phương pháp này chủ yếu áp dụng để giải quyết các bài toán VRP có kích thước nhỏ và số ràng buộc ít do bị hạn chế về thời gian tìm kiếm lời giải. Phần lớn các thuật toán chính xác giải bài toán VRP được phát triển từ các thuật toán chính xác giải bài toán TSP, bao gồm: thuật toán nhánh cận (Brand and Bound), thuật toán sinh cột (Column Generation Algorithm), thuật toán quy hoạch động (Dynamic Programming), ...

Các phương pháp gần đúng gồm các thuật giải cho chất lượng lời giải gần với lời giải tối ưu như nhóm các giải thuật heuristic cổ điển, nhóm các giải thuật tìm kiếm cục bộ và nhóm các giải thuật metaheuristic.

1.5.1 Thuật toán Metaheuristic

Nhóm các giải thuật metaheuristic được phát triển từ năm 1990 và được xem là nhóm giải thuật có triển vọng nhất hiện nay, được sử dụng rộng rãi để giải quyết các bài toán tối ưu, đặc biệt là trong các bài toán có không gian tìm kiếm lớn. Các giải thuật này thường đem lại lời giải tương đối tốt trong khoảng thời gian chấp nhận được mà các giải thuật chính xác là không khả thi trong các trường hợp này. Dưới

đây là một số giải thuật metaheuristic quan trọng trong nhóm này:

- Giải thuật luyện kim (Simulated Annealing): đến từ quá trình làm mát dần dần trong luyện kim thực tế, nơi kim loại được làm nóng và sau đó được làm mát dần dần để giảm căng và tạo ra cấu trúc tinh thể ổn định..
- Giải thuật dựa vào miền láng giềng: Variable Neighborhood Search (VNS), Large Neighborhood Search và Adaptive Large Neighborhood Search.
- Nhóm các giải thuật dựa trên quần thể: Giải thuật di truyền (Genetic Algorithm), giải thuật tối ưu hóa đàn kiến (Ant Colony Algorithm).

Sơ đồ chung của một thuật toán metaheuristic có thể được biểu diễn như sau:

Algorithm 1: Thuật toán metaheuristic

```
Khởi tạo lời giải ban đầu;
Định nghĩa các chiến lược tăng cường hóa;
Định nghĩa các chiến lược đa dạng hóa;
Tính độ thích nghi cho mỗi lời giải;
while (điều kiện dừng chưa thỏa mãn) do
    Thực hiện các chiến lược tăng cường hóa;
    Thực hiện các chiến lược đa dạng hóa;
    Cập nhật lời giải tốt nhất cho đến thời điểm hiện tại;
end while
Trả về lời giải tốt nhất tìm được;
```

1.5.2 Thuật toán Local Search

Từ một giải pháp hiện tại được chọn là s, thuật toán local search sẽ tìm một giải pháp lân cận s' của s; nếu s' tốt hơn s thì s' sẽ trở thành giải pháp hiện tại. Quá trình này sẽ dừng khi thuật toán lặp đến một số lần định trước hoặc khi giải pháp tốt nhất không được cải thiện qua một số lần lặp xác định.

Local search là thuật toán quan trọng của dạng thuật toán metaheuristic, là nền tảng của nhiều thuật toán metaheuristic khác.

Sơ đồ thuật toán Local Search:

Algorithm 2: Thuật toán Local Search

Gọi S là toàn bộ (hoặc một phần) không gian lời giải của bài toán;
Khởi tạo ngẫu nhiên giải pháp s thuộc S;
while (*điều kiện dừng chưa thỏa mãn*) **do**
    Tìm một giải pháp lân cận s' của s với s' thuộc S;
    if s' tốt hơn s **then**
        $s \leftarrow s'$;
    end if
end while
s là lời giải tìm được của bài toán;

Chương 2

Hệ thống điều phối xe trung chuyển

2.1 Đặt vấn đề

Một công ty cung cấp dịch vụ trung chuyển khách đang thực hiện việc điều phối xe trung chuyển và sắp xếp các khách hàng vào các xe.

Xe trung chuyển là loại xe có nhiệm vụ đón khách và trả khách tại các điểm có trong vùng phục vụ đến các hub, để từ hub lên các xe dịch vụ khác (liên tỉnh, xe sân bay) theo nhu cầu của khách hàng.

- Nhiệm vụ trả khách: Xe đưa khách từ điểm trung chuyển (hub) đến các điểm họ cần đến.
- Nhiệm vụ đón khách: Xe đến đón khách tại các điểm và đưa về điểm trung chuyển (hub).

Hiện tại việc thực hiện điều hành trung chuyển của công ty đang thực hiện theo các bước sau:

- Khách hàng thực hiện book vé xe tuyến sẽ cung cấp thông tin vị trí điểm đón, trả và thời gian di chuyển.

- Các đối tác vận hành trung chuyển được thiết lập có khả năng trung chuyển tại các vùng trung chuyển theo quan hệ nhiều nhiều (n-n). Có nghĩa là mỗi đối tác vận hành trung chuyển có thể có khả năng trung chuyển tại nhiều vùng trung chuyển khác nhau, và mỗi vùng trung chuyển cũng có thể có nhiều đối tác vận hành trung chuyển khác nhau. Các đối tác trung chuyển có các loại xe phục vụ được khai báo trước.

- Với các điều kiện như trên khách hàng khi book vé xe tuyến sẽ được tự động phân bổ về các đối tác vận hành trung chuyển.

Sau khi phân bổ về các đối tác trung chuyển, bài toán chia chọn sẽ được tiếp tục phát triển để tối ưu hoạt động điều hành. Hiện tại điều hành trung chuyển dựa trên danh sách khách hàng cần trung chuyển và các thông tin có thể lọc theo nhu cầu, dựa vào kinh nghiệm cá nhân để gom khách vào các chuyến đi.

$\Rightarrow$ Công ty muốn thay thế việc điều hành thủ công bằng một hệ thống chia chọn sử dụng các thuật toán chia chọn tự động hiệu quả hơn.

Bài toán được phát biểu như sau:

- Có N khách hàng ở các vị trí khác nhau có định vị điểm đón và định vị điểm đến, có thời gian cần xuất phát hoặc thời gian cần đến đích.

- Công ty gồm 4 loại xe trung chuyển khác nhau gồm: xe 5 chỗ (dung lượng 4 khách), xe 7 chỗ (dung lượng 6 khách) và xe 16 chỗ (dung lượng 15 khách).

- Sau mỗi lần gom được đủ số lượng khách (theo nhu cầu của công ty), người dùng sẽ đưa dữ liệu về khách hàng, phương tiện (bao gồm: vị trí, thời gian, số lượng, ...) vào hệ thống và nhận kết quả điều phối cho các xe của công ty.

Các ràng buộc chung của hai bài toán:

- Tối ưu về tổng khoảng cách di chuyển của các xe (chi phí đi lại, thời gian,...)

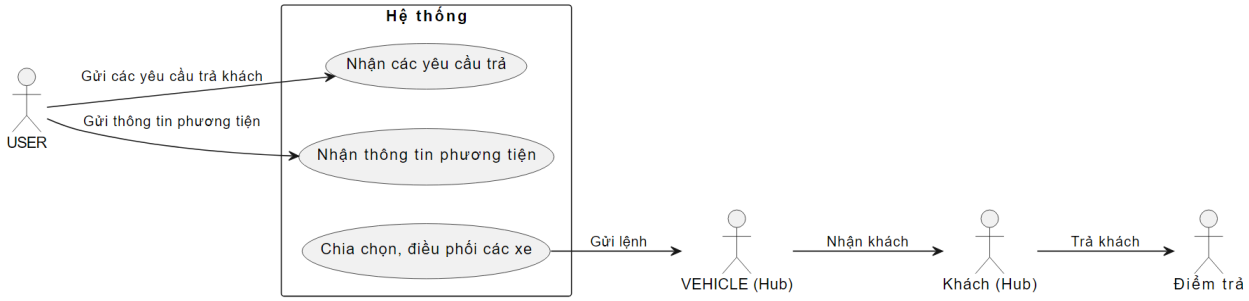
- Số lần xe dừng lại đón khách/trả khách không quá số lần cho phép của khách.

- Xe không chở quá khả năng/số chỗ của xe.
- Không để khách chờ quá khoảng thời gian cho phép của khách (nếu có).
- Giới hạn số khoảng cách/thời gian di chuyển tối đa của mỗi xe.

2.2 Phân tích bài toán

2.2.1 Bài toán trả khách

Xe đưa khách từ điểm trung chuyển (hub) đến các điểm họ cần đến.
Giả thiết: Các xe đang ở tại nơi trung chuyển (hub).
⇒ Tiêu chí không để khách chờ quá khoảng thời gian cho phép được đảm bảo.
Sơ đồ use case:



Hình 2.1: Sơ đồ quy trình trả khách

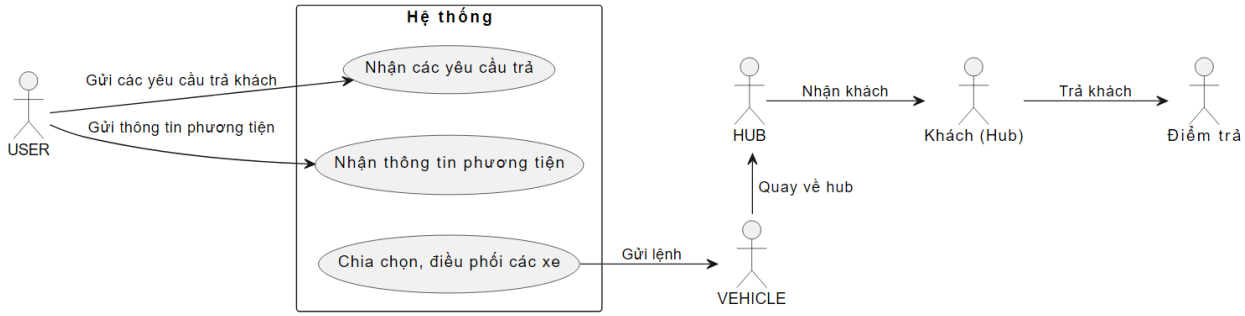
Một số các tình huống nghiệp vụ của bài toán:

Tình huống 1:

- Bỏ giả thiết các xe đang ở hub, các xe bây giờ sẽ xuất phát tại các vị trí khác nhau.
- Sau khi nhận nhiệm vụ trả khách, các xe sẽ từ các điểm hiện tại quay lại hub lấy khách rồi trả khách đến các điểm trả, nhiệm vụ kết thúc sau khi trả khách cuối cùng.

Ví dụ 1: Có 10 khách đang ở bến xe Mỹ Đình trong đó 8 khách cần về Văn Quán Hà Đông, 1 khách cần về Đại Cồ Việt, 1 khách cần về Văn Cao. Có 3 xe trung chuyển trong đó 1 xe 5 chỗ đang ở tại Nguyễn Chí Thanh, 2 xe 7 chỗ đang ở tại Điện Biên Phủ.

Sơ đồ use case:



Hình 2.2: Sơ đồ quy trình trả khách tình huống 1

Tình huống 2:

- Thêm dữ kiện khoảng thời gian khách cần xuống điểm trả.

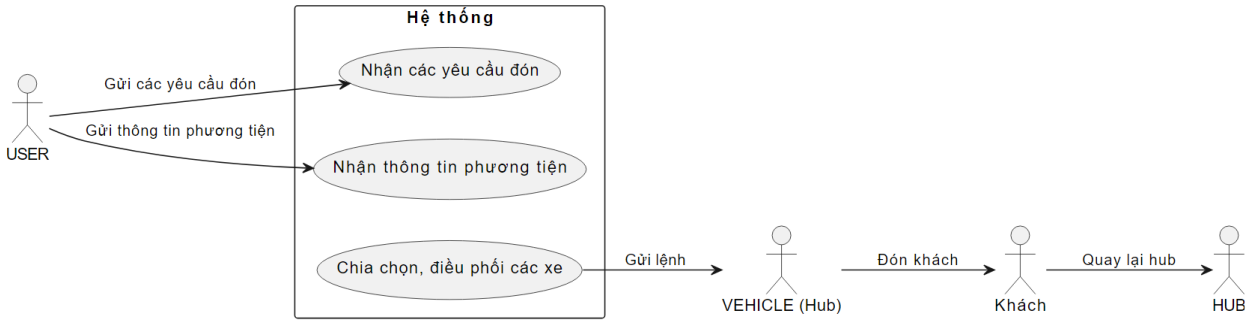
Ví dụ 2: Hiện giờ là 8:30 sáng có 10 khách đang ở tại hub ở bến xe Mỹ Đình. Các khách này cần về các địa điểm như trong **Ví dụ 1** và 3 xe trung chuyển hiện đang ở các vị trí cũng như trong **Ví dụ 1**. Nhưng trong 10 khách có 5 khách cần xuống xe trong khoảng từ 10:00 đến 10:30; 5 khách cần xuống xe trong khoảng từ 10:45 đến 11:15.

2.2.2 Bài toán đón khách

Xe đến đón khách tại các điểm và đưa về điểm trung chuyển (hub).

Giả thiết: Các xe đang ở tại nơi trung chuyển (hub).

Sơ đồ use case:



Hình 2.3: Sơ đồ quy trình đón khách

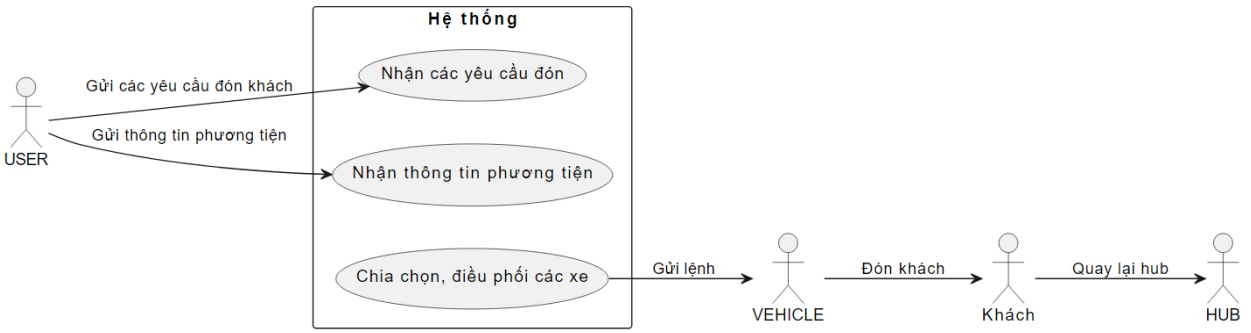
Một số các tình huống nghiệp vụ của bài toán:

Tình huống 1:

- Bỏ giả thiết các xe đang ở hub và không có ràng buộc quá thời gian khách phải chờ, các xe bây giờ sẽ xuất phát tại các vị trí khác nhau.
- Sau khi nhận nhiệm vụ đón khách, các xe sẽ từ các điểm hiện tại sẽ tới thẳng các điểm đón khách và trở về hub sau khi đón khách cuối cùng, nhiệm vụ kết thúc sau khi đón khách cuối cùng về hub.

Ví dụ 3: Có 10 khách đang ở các điểm khác nhau cần đón về hub (tại số 1 Cổ Linh). Có 3 xe trong đó 1 xe 5 chỗ đang ở tại số 1 Đại Cồ Việt, 1 xe 7 chỗ ở số 1 Xuân Thủy, xe 7 chỗ còn lại đang ở hub.

Sơ đồ use case:



Hình 2.4: Sơ đồ quy trình đón khách tình huống 1

Tình huống 2:

- Thêm dữ kiện khoảng thời gian khách có thể lên xe và khoảng thời gian khách cần đến hub.

Ví dụ 4: Hiện giờ là 8:30 sáng. Số lượng khách, vị trí khách và xe trung chuyển như trong **Ví dụ 3**. Nhưng trong 10 khách có 5 người có thể lên xe từ 8:45 đến 9:15; 5 người có thể lên xe từ 9:00 đến 9:30 và cần đến hub không quá lúc 10:00.

2.3 Nhận định và hướng giải pháp

2.3.1 Nhận định

Bài toán được đưa ra là một bài toán tối ưu, thuộc lớp bài toán định tuyến phương tiện (VRP).

- ✓ Công ty có 3 loại xe có kích cỡ khác nhau nên cả hai bài toán đón khách và trả khách đều là biến thể bài toán VRP có ràng buộc về khả năng xử lý (CVRP).
- ✓ Ở tình huống 1, cả hai bài toán vẫn là biến thể bài toán VRP có ràng buộc về khả năng xử lý nhưng ta cần thay đổi điểm xuất phát của từng xe.
- ✓ Ở tình huống 2, cả hai bài toán có thêm dữ kiện về khoảng thời gian của từng khách hàng nên 2 bài toán là biến thể bài toán VRP có ràng buộc về khả năng xử lý và ràng buộc về thời gian (CVRPTW).

2.3.2 Hướng giải pháp

Phần dữ liệu đầu vào của hệ thống:
Về phần khách hàng:

- Các vị trí toạ độ điểm cần đón và điểm trả của khách

- Số lượng khách từng điểm
- Khoảng thời gian khách cần đón và cần xuống xe
- Một số yêu cầu hay ràng buộc khác nếu có

Về phần phương tiện:

- Các vị trí toạ độ của phương tiện
- Số chỗ ngồi của xe

Phần giải thuật:

Như đã trình bày ở phần **(1.5)** trên, các giải pháp cho bài toán tối ưu được chia làm hai lớp chính: các phương pháp chính xác và các phương pháp gần đúng.

- ✓ Các phương pháp giải chính xác đảm bảo lời giải tối ưu sẽ được tìm thấy trong một khoảng thời gian thời gian hữu hạn. Tuy nhiên, thời gian chạy của nó trong trường hợp tốt nhất là rất lớn, do đó lớp thuật toán này chỉ giải được những bài toán có kích thước nhỏ hoặc vừa.
- ✓ Các phương pháp gần đúng gồm các giải thuật cho chất lượng lời giải gần với lời giải tối ưu như nhóm các giải thuật heuristic cổ điển, nhóm các giải thuật tìm kiếm cục bộ và nhóm các giải thuật metaheuristic.

$\Rightarrow$ Vì tập dữ liệu đầu vào của bài toán này có thể khá lớn nên ta sử dụng các phương pháp gần đúng như nhóm các giải thuật metaheuristic để tìm lời giải tối ưu cho bài toán này.

Chương 3

Giải pháp và cài đặt

3.1 Công cụ OR-Tools

3.1.1 Giới thiệu OR-Tools

OR-Tools là 1 phần mềm mã nguồn mở do Google phát triển, dùng để giải quyết các bài toán tối ưu hóa tổ hợp, nhằm tìm kiếm lời giải tốt nhất cho 1 bài toán trong số rất nhiều lời giải khả thi.

Một số bài toán mà OR-Tools giải quyết:

- Định tuyến xe (Vehicle routing): Tìm đường đi tối ưu cho xe nhận và giao các đơn hàng có các ràng buộc nhất định.
- Lập lịch (Scheduling): Tìm lịch tối ưu cho 1 tập hợp các nhiệm vụ phức tạp, 1 số công việc cần được thực hiện trước những công việc khác, trên một tập hợp cố định các tài nguyên khác.
- Đóng thùng (Bin packing): Đóng gói các thùng hàng có kích thước khác nhau với số lượng càng nhiều càng tốt vào 1 thùng xe có sức chứa cố định.

Trong hầu hết các trường hợp, những vấn đề như vậy sẽ có rất nhiều giải pháp khả thi hay cần quá nhiều máy tính để có thể tìm kiếm tất cả. Để khắc phục điều này,

OR-Tools sử dụng các thuật toán hiện đại để thu hẹp tập hợp tìm kiếm nhằm tìm ra giải pháp tối ưu (hoặc gần với mức tối ưu).

OR-Tools được viết bằng C++, nhưng được hỗ trợ cho cả Python, Java và C# (trên nền tảng .NET).

Công cụ này chứa một số trình giải, cụ thể là:

- Trình giải lập trình ràng buộc
- Trình giải quy hoạch tuyến tính
- Trình bao bọc cho các bộ giải thương mại (như Gurobi hoặc CPLEX) và các bộ giải mã nguồn mở khác (SCIP, GLPK, v.v.)
- Trình giải định tuyến

3.1.2 Cách cài đặt OR-Tools

Cách nhanh nhất để cài đặt OR-Tools là cài đặt cho phiên bản nhị phân Python. Nếu đã có Python (phiên bản 3.8 trở lên trên Linux, macOS hoặc Windows) và PIP trình quản lý gói Python, ta có thể cài đặt OR-Tools như sau:

```
python -m pip install --upgrade --user ortools
```

Hoặc có thể tải và cài đặt trực tiếp công cụ từ trang chủ của Google OR-Tools đối với các ngôn ngữ lập trình khác.

3.1.3 Bài toán định tuyến phương tiện với OR-Tools

OR-Tools có thể giải quyết nhiều biến thể bài toán VRP, bao gồm:

- Bài toán Người bán hàng (TSP) (Trường hợp đơn giản nhất của VRP gồm 1 xe)
- Bài toán Định tuyến xe cơ bản (VRP)
- Bài toán Định tuyến xe có ràng buộc sức chứa (CVRP)

- Bài toán Định tuyến xe có ràng buộc thời gian (VRPTW)
- Bài toán Định tuyến xe có ràng buộc lấy và giao hàng (VRPPD)

Dữ liệu đầu vào

Dữ liệu đầu vào của các bài toán VRP trong OR-Tools (tuỳ theo từng biến thể của bài toán VRP) là:

- Ma trận khoảng cách giữa kho hàng hay điểm xuất phát của các xe với các điểm khách hàng (VRP).
- Danh sách sức chứa của từng xe (CVRP).
- Ma trận thời gian, cửa sổ thời gian của từng khách hàng (VRPTW).
- Danh sách các cặp vị trí lấy và giao hàng (VRPPD).

Giải thuật sử dụng

1. Thuật toán tìm kiếm:

Thuật toán OR-Tools sử dụng để giải quyết bài toán VRP là thuật toán tìm kiếm cục bộ Local search (nhóm giải thuật metaheuristic), ngoài ra ta có thể lựa chọn một số các giải thuật khác có trong OR-Tools để thay đổi cải thiện giải pháp:

- `Greedy_Descent`: Kết hợp giữa giải thuật tham lam và phương pháp giảm thiểu.
- `Guided_Local_Search`: Tìm kiếm cục bộ có hướng dẫn.
- `Simulated_Annealing`: Giải thuật luyện kim.
- `Tabu_Search`: Giải thuật tìm kiếm Tabu Search.[1]
- `Generic_Tabu_Search`: Giải thuật tìm kiếm Generic Tabu Search.

2. Giải pháp ban đầu:

Bước chung đầu tiên của thuật toán metaheuristic đều bắt đầu từ một giải pháp ban đầu từ đó tìm kiếm các giải pháp tốt hơn, trong OR-Tools ta có thể lựa chọn một trong các thuật toán tìm giải pháp ban đầu sau để cải thiện kết quả bài toán:

- `Path_Cheapest_Arc`: Kết nối nút bắt đầu của tuyến đường với nút mà tạo ra đoạn đường rẻ nhất. Mở rộng tuyến đường bằng cách lặp lại trên nút được thêm vào tuyến đường.
- `Path_Most_Constrained_Arc`: Tương tự `Path_Cheapest_Arc` nhưng ưu tiên đoạn đường bị ràng buộc nhiều nhất trước. Các đoạn đường được đánh giá lựa chọn bằng một bộ chọn dựa trên so sánh.
- `Evaluator_Strategy`: Tương tự `Path_Cheapest_Arc`, ở đây chi phí đoạn đường được đánh giá.
- `Savings`: Thuật toán Savings (Clarke & Wright).
- `Sweep`: Thuật toán Sweep (Wren & Holliday).
- `Christofides`: Thuật toán Christofides (Nicos Christofides).
- `All_Unperformed`: Đặt tất cả các nút không hoạt động. Chỉ tìm giải pháp nếu các nút không bắt buộc (là phần tử của một quy tắc ràng buộc với một chi phí hình phạt hữu hạn).
- `Best_Insertion`: Xây dựng lại một giải pháp bằng cách chèn nút rẻ nhất ở vị trí rẻ nhất. Chi phí chèn dựa trên hàm chi phí toàn cục của mô hình định tuyến.
- `Parallel_Cheapest_Insertion`: Xây dựng lại một giải pháp bằng cách chèn nút rẻ nhất ở vị trí rẻ nhất; chi phí chèn dựa trên hàm chi phí của cạnh. Nhanh hơn `Best_Insertion`.

- `Local_Cheapest_Insertion`: Xây dựng lại một giải pháp bằng cách chèn nút rẻ nhất ở vị trí rẻ nhất; chi phí chèn dựa trên hàm chi phí của cạnh. Khác với `Parallel_Cheapest_Insertion` là theo nút được chọn để chèn; ở đây các nút được xem xét theo thứ tự tạo. Nhanh hơn `Parallel_Cheapest_Insertion`.
- `Global_Cheapest_Arc`: Kết nối hai nút lặp đi lặp lại để tạo ra đoạn đường rẻ nhất.
- `Local_Cheapest_Arc`: Chọn nút đầu tiên có thành phần kế tiếp không bị ràng buộc và kết nối nút đó với nút tạo ra đoạn đường rẻ nhất.
- `First_Unbound_Min_Value`: Chọn nút đầu tiên được kế nhiệm chưa được liên kết và kết nối nó với nút có sẵn đầu tiên.

3.2 Thu thập dữ liệu đầu vào

3.2.1 Giới thiệu Geoapify

Geoapify là một công ty cung cấp dịch vụ địa lý (geospatial) và các công cụ liên quan. Họ chủ yếu tập trung vào phát triển các API (Giao diện lập trình ứng dụng) và công nghệ địa lý để giúp các nhà phát triển xây dựng và tích hợp các tính năng địa lý vào ứng dụng và trang web của họ.

Các dịch vụ của Geoapify có thể bao gồm việc xử lý địa lý, tìm kiếm địa lý, định vị địa lý, và nhiều tính năng khác liên quan đến xử lý thông tin địa lý và địa điểm. Điều này có thể hỗ trợ trong việc xây dựng ứng dụng như bản đồ, ứng dụng giao thông, và các ứng dụng khác có liên quan đến vị trí.

3.2.2 Lấy và xử lý dữ liệu từ Geoapify

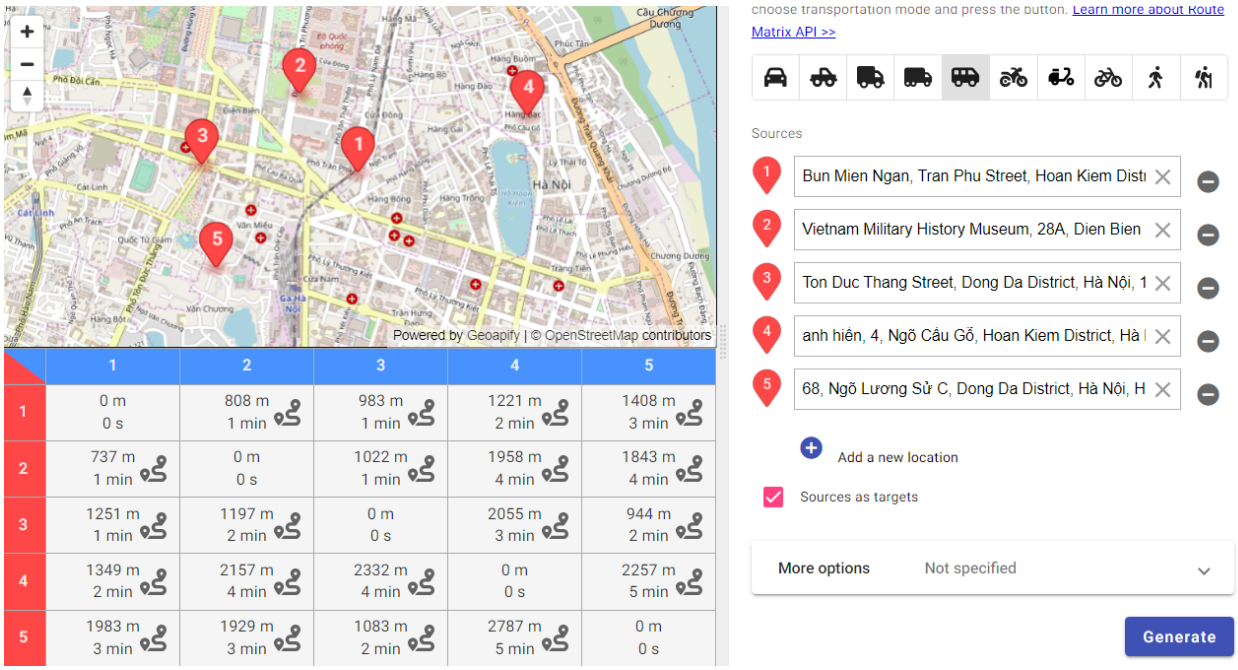
Geoapify cung cấp một dịch vụ cho phép ta nhận được một ma trận khoảng cách và thời gian giữa các vị trí, toạ độ mà ta đưa vào.

Lấy dữ liệu:

Có hai cách lấy dữ liệu từ Geoapify:

1. Lấy dữ liệu trực tiếp trên web của nhà phát hành.

Ta vào trang web (`https://apidocs.geoapify.com/playground/route-matri`
Nhập các điểm cần tạo ma trận khoảng cách bằng cách ghi tên địa chỉ có sẵn trên bản đồ vào các thanh ngang bên phải hoặc nhấn trực tiếp vào vị trí đó trên bản đồ.



	1	2	3	4	5
1	0 m 0 s	808 m 1 min	983 m 1 min	1221 m 2 min	1408 m 3 min
2	737 m 1 min	0 m 0 s	1022 m 1 min	1958 m 4 min	1843 m 4 min
3	1251 m 1 min	1197 m 2 min	0 m 0 s	2055 m 3 min	944 m 2 min
4	1349 m 2 min	2157 m 4 min	2332 m 4 min	0 m 0 s	2257 m 5 min
5	1983 m 3 min	1929 m 3 min	1083 m 2 min	2787 m 5 min	0 m 0 s

Hình 3.1: Tạo ma trận khoảng cách/thời gian từ Geoapify

2. Sử dụng API của Geoapify để request dữ liệu.

Ta có thể request dữ liệu qua API của Geoapify. Dữ liệu đưa vào request là một định dạng Json gồm một mảng các toạ độ của các vị trí ta cần.

Ví dụ:

Request:

```
{"mode":"drive",
 "sources":
   [{"location":[8.73784862216246,48.543061473317266]},
```

```
   {"location":[9.305536080205002,48.56743450655594]},
   {"location":[9.182792846033067,48.09414029055267]}],
 "targets":
  [{"location":[8.73784862216246,48.543061473317266]},
   {"location":[9.305536080205002,48.56743450655594]},
   {"location":[9.182792846033067,48.09414029055267]}]
}
```


Result:


```
{"sources":
  [{"origin_loc":[8.73784862216246,48.543061473317266],
      "location":[8.737849,48.543061]},
   {"origin_loc":[9.305536080205002,48.56743450655594],
      "location":[9.305536,48.567435]}
   {"origin_loc":[9.182792846033067,48.09414029055267],
      "location"[9.182793,48.09414]}],
 "targets":
  [{"origin_loc":[8.73784862216246,48.543061473317266],
      "location":[8.737849,48.543061]},
   {"origin_loc":[9.305536080205002,48.56743450655594],
      "location":[9.305536,48.567435]},
   {"origin_loc":[9.182792846033067,48.09414029055267],
      "location":[9.182793,48.09414]}],
 "sources_to_targets":
 [[{"distance":0,"time":0,"source_index":0,
     "target_index":0},
  {"distance":56706,"time":2969,"source_index":0,
     "target_index":1},
  {"distance":88645,"time":4325,"source_index":0,
     "target_index":2}],
  [{"distance":56143,"time":2952,"source_index":1,
     "target_index":0},
```

```
  {"distance":0,"time":0,"source_index":1,
     "target_index":1},
  {"distance":72676,"time":3815,"source_index":1,
     "target_index":2}],
 [{"distance":88267,"time":4323,"source_index":2,
     "target_index":0},
  {"distance":72794,"time":3857,"source_index":2,
     "target_index":1},
  {"distance":0,"time":0,"source_index":2,
     "target_index":2}]],
 "units":"metric",
 "distance_units":"meters",
 "mode":"drive"
}
```


Xử lý dữ liệu:

Kết quả sau khi request ma trận khoảng cách qua API của Geoapify là một dạng Json, ta có thể biến đổi kết quả request thành 2 ma trận khoảng cách và thời gian để phù hợp với đầu vào của bài toán và chương trình.

Kết quả dữ liệu sau khi chuyển đổi từ result của ví dụ trên:

```
Distance Matrix:
[[    0 56706 88645]
 [56143     0 72676]
 [88267 72794     0]]

Time Matrix:
[[   0 2969 4325]
 [2952    0 3815]
 [4323 3857    0]]
```

Đơn vị của ma trận khoảng cách là mét (m) và đơn vị của ma trận thời gian là giây

(s).

Trong hai ma trận trên, phần tử (i, j) là khoảng cách/thời gian đi từ i tới j và (j, i) là từ j tới i. Ví dụ: Khoảng cách quãng đường từ 1 tới 3 là 88645m, từ 3 tới 1 là 88267m.

Chương 4

Thử nghiệm và kết quả

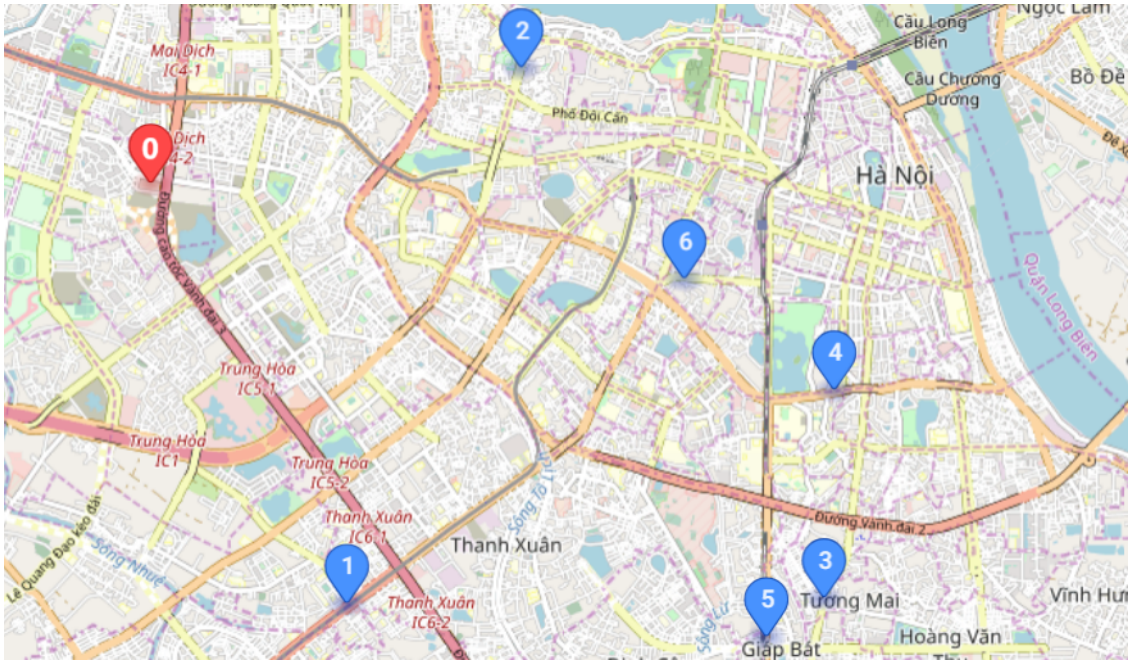
4.1 Thử nghiệm 1

Thử nghiệm 1: Có 10 khách và 2 xe gồm: 1 xe 5 chỗ và 1 xe 7 chỗ đang ở tại Hub ở bến xe Mỹ Đình có toạ độ $(21.028199, 105.778827)$, trong đó:

- 2 khách cần về Văn Quán có toạ độ $(20.987635, 105.797656)$.
- 2 khách cần về Văn Cao có toạ độ $(21.038563, 105.815278)$.
- 1 khách cần về Trương Định có toạ độ $(20.988512, 105.847116)$.
- 1 khách cần về Đại Cồ Việt có toạ độ $(21.008464, 105.848317)$.
- 2 khách cần về Giáp Bát có toạ độ $(20.983939, 105.841150)$.
- 2 khách cần về Khâm Thiên có toạ độ $(21.019356, 105.832763)$.

4.1.1 Mô tả dữ liệu

Mô tả bằng hình ảnh:



Hình 4.1: Hình ảnh mô tả dữ liệu thử nghiệm 1

Kết quả ma trận đầu vào (đơn vị là mét):

```
Distance Matrix:
[[   0 5652 7814 11319 11337 11073 9589]
 [ 9420   0 8369 7435 7068 7189 5705]
 [ 6170 7215  0 9073 6040 8827 4263]
 [13006 7051 9573  0 4146 1061 5517]
 [12916 6961 6188 3703 0 3487 2786]
 [12709 6754 8769 3261 3342 0 4713]
 [ 7011 5263 3984 5098 2913 4882 0]]
```

4.1.2 Kết quả

Kết quả hệ thống:

```
Objective: 2597610
Dropped nodes:
Route for vehicle 0:
```

```
 0 Load(0) -> 1 Load(2) -> 2 Load(4) -> 0 Load(4)
Distance of the route: 20191m
Load of the route: 4

Route for vehicle 1:
 0 Load(0) -> 3 Load(1) -> 5 Load(3) -> 4 Load(4) -> 6
    Load(6) -> 0 Load(6)
Distance of the route: 25519m
Load of the route: 6

Total distance of all routes: 45710m
Total load of all routes: 10
```

Giải thích kết quả:

- 0,. . . ,6 đại diện cho các vị trí theo thứ tự trong thử nghiệm (0 là Hub)
- Lộ trình cho xe 1 là: 0 → 1 → 2
  Hub → Văn Quán (Trả 2) → Văn Cao (Trả 2)
  Độ dài lộ trình 1: 20191m
  Số khách trả: 4
- Lộ trình cho xe 2 là: 0 → 3 → 5 → 4 → 6
  Hub → Trương Định (Trả 1) → Giáp Bát (Trả 2) → Đại Cồ Việt (Trả 1) → Khâm Thiên (Trả 2)
  Độ dài lộ trình 2: 25519m
  Số khách trả: 6
- Tổng độ dài quãng đường di chuyển của 2 xe: 45710m
- Tổng số khách trả: 10

4.2 Thử nghiệm 2

Thử nghiệm 2: Có 20 khách cần đón về Hub ở bến xe Mỹ Đình có toạ độ $(21.028199, 105.7788$
Có 3 xe trong đó:

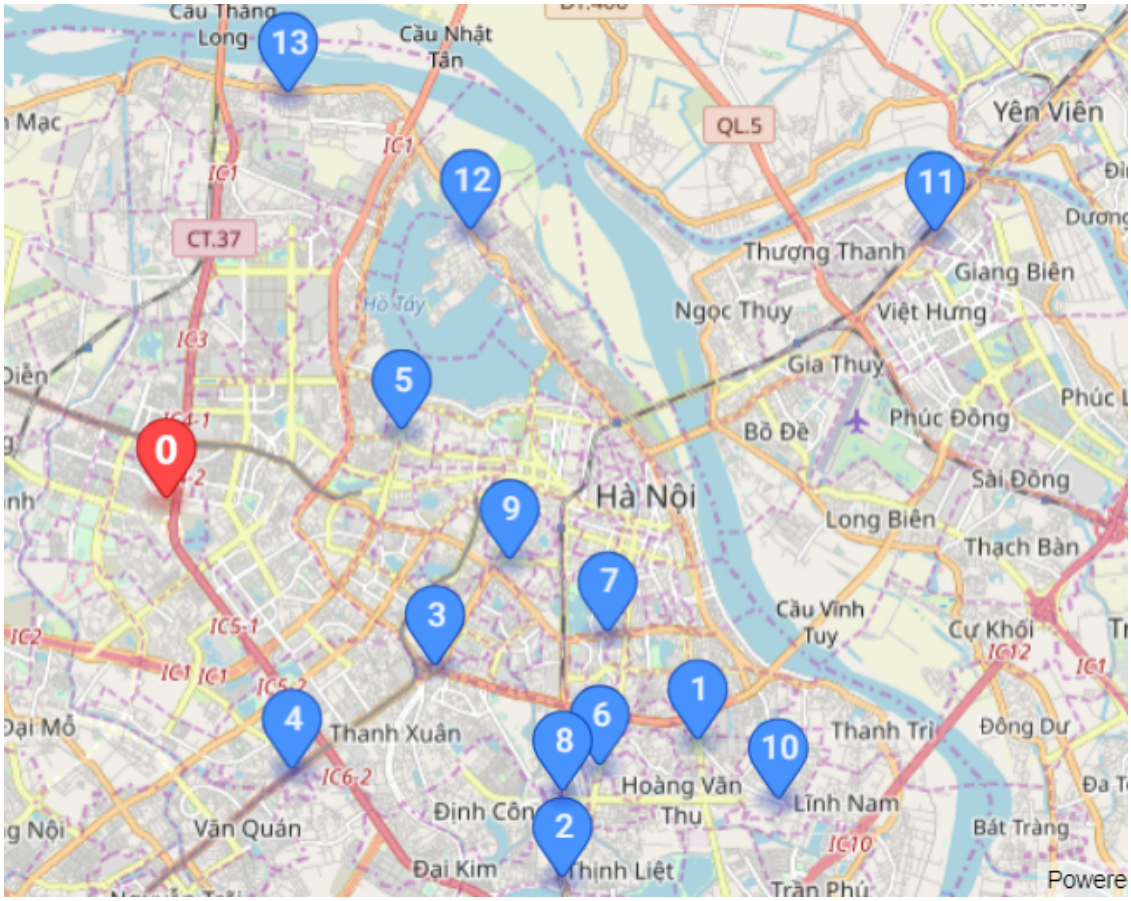
- 1 xe 5 chỗ đang tại Lĩnh Nam có toạ độ $(20.992273, 105.862981)$
- 1 xe 7 chỗ đang tại Giải Phóng có toạ độ $(20.971757, 105.840965)$
- 1 xe 16 chỗ đang tại Ngã Tư Sở có toạ độ $(21.003571, 105.820838)$

Có 20 khách gồm:

- 2 khách cần đón tại Văn Quán có toạ độ $(20.987635, 105.797656)$.
- 2 khách cần đón tại Văn Cao có toạ độ $(21.038563, 105.815278)$.
- 1 khách cần đón tại Trương Định có toạ độ $(20.988512, 105.847116)$.
- 1 khách cần đón tại Đại Cồ Việt có toạ độ $(21.008464, 105.848317)$.
- 2 khách cần đón tại Giáp Bát có toạ độ $(20.983939, 105.841150)$.
- 2 khách cần đón tại Khâm Thiên có toạ độ $(21.019356, 105.832763)$.
- 3 khách cần đón tại Lĩnh Nam có toạ độ $(20.983375, 105.875300)$.
- 3 khách cần đón tại Ngô Gia Tự có toạ độ $(21.068590, 105.900846)$.
- 2 khách cần đón tại Âu Cơ có toạ độ $(21.068112, 105.826671)$.
- 2 khách cần đón tại An Dương Vương có toạ độ $(21.089141, 105.797062)$.

4.2.1 Mô tả dữ liệu

Mô tả bằng hình ảnh:



Hình 4.2: Hình ảnh mô tả dữ liệu thử nghiệm 2

Kết quả ma trận khoảng cách đầu vào (đơn vị là mét):

0	16291	13740	7964	5652	7814	11319	11337	11073	9589	17477	29161	13911	9356
17914	0	6596	7607	9900	11937	2731	4485	3318	5856	1654	11910	11507	17206
16621	5496	0	5822	7822	10130	2052	4703	1361	6074	7150	16533	14588	23351
8952	6657	7584	0	2997	5382	5783	5801	5537	4053	10237	17692	12931	15682
9420	8309	9828	4080	0	8369	7435	7068	7189	5705	13565	19344	11978	16150
6170	9947	10874	5432	7215	0	9073	6040	8827	4263	13527	11892	8353	10225
13006	2674	3187	4758	7051	9573	0	4146	1061	5517	4328	13716	12465	18164
12916	3232	5534	4362	6961	6188	3703	0	3487	2786	5634	13354	9921	15620
12709	4135	2290	4461	6754	8769	3261	3342	0	4713	5789	15172	13227	19439
7011	5972	6929	2266	5263	3984	5098	2913	4882	0	7626	11850	7350	13814
19046	1654	7718	9261	13068	13591	4385	5635	9079	8421	0	12578	11806	17505
28274	11332	20396	16007	18300	12339	13107	11284	14776	11678	15588	0	11005	20343
13356	11110	12484	8622	18912	9184	12555	9971	10437	7354	11975	11686	0	5729
8851	15505	22495	15491	14407	9080	17337	14366	17091	11749	16370	20141	4571	0

4.2.2 Kết quả

Kết quả hệ thống:

```
Objective: 3519471
Route for vehicle 0:
 1 Load(0) -> 12 Load(2) -> 13 Load(4) -> 0 Load(4)
Distance of the route: 26087m
Load of the route: 4

Route for vehicle 1:
 2 Load(0) -> 6 Load(1) -> 11 Load(4) -> 5 Load(6) -> 0
    Load(6)
Distance of the route: 34277m
Load of the route: 6

Route for vehicle 2:
 3 Load(0) -> 4 Load(2) -> 8 Load(4) -> 10 Load(7) -> 7
    Load(8) -> 9 Load(10) -> 0 Load(10)
Distance of the route: 31407m
Load of the route: 10

Total distance of all routes: 91771m
Total load of all routes: 20
```

Giải thích kết quả:

- Lộ trình cho xe 1 là: 1 → 12 → 13 → 0
  Lĩnh Nam (Xuất phát) → Âu Cơ (Đón 2) → An Dương Vương (Đón 2) → Hub
  Độ dài lộ trình: 26087m
  Số khách đón: 4

- Lộ trình cho xe 2 là: 2 → 6 → 11 → 5 → 0
  Giải Phóng (Xuất phát) → Trương Định (Đón 1) → Ngô Gia Tự (Đón 3) → Văn

Cao (Đón 2) → Hub
Độ dài lộ trình: 34277m
Số khách đón: 6

- Lộ trình cho xe 3 là: 3 → 4 → 8 → 10 → 7 → 9 → 0
  Ngã Tư Sở (Xuất phát) → Văn Quán (Đón 2) → Giáp Bát (Đón 2) → Lĩnh Nam (Đón 3) → Đại Cồ Việt (Đón 1) → Khâm Thiên (Đón 2) → Hub
  Độ dài lộ trình: 31407m
  Số khách đón: 10

- Tổng độ dài quãng đường di chuyển của 3 xe: 91771m

- Tổng số khách đón của tất cả các lộ trình: 20

4.3 Thử nghiệm 3

Thử nghiệm 3: Hiện tại đang là 8:30. Có 10 khách cần đón về Hub và có 2 xe gồm: 1 xe 5 chỗ và 1 xe 7 chỗ đang ở tại Hub ở bến xe Mỹ Đình có toạ độ $(21.028199, 105.778827)$, trong đó:

- 2 khách cần đón tại Văn Quán có toạ độ $(20.987635, 105.797656)$ trong khoảng thời gian từ 8:15 đến 9:00.

- 2 khách cần đón tại Văn Cao có toạ độ $(21.038563, 105.815278)$ trong khoảng thời gian từ 8:40 đến 8:50.

- 1 khách cần đón tại Trương Định có toạ độ $(20.988512, 105.847116)$ trong khoảng thời gian từ 8:35 đến 9:00.

- 1 khách cần đón tại Đại Cồ Việt có toạ độ $(21.008464, 105.848317)$ trong khoảng thời gian từ 8:30 đến 8:45.

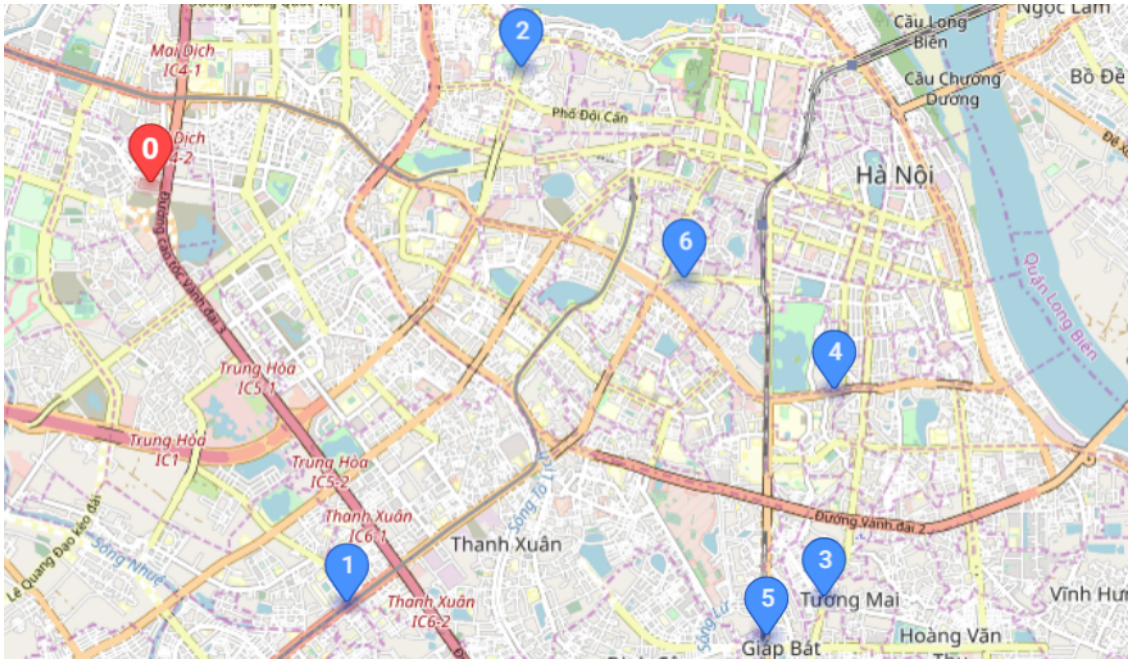
- 2 khách cần đón tại Giáp Bát có toạ độ $(20.983939, 105.841150)$trong khoảng thời gian từ 8:30 đến 8:45.

- 2 khách cần đón tại Khâm Thiên có toạ độ $(21.019356, 105.832763)$trong khoảng thời gian từ 8:45 đến 9:10.
- Tất cả các khách cần đến Hub trước lúc 9h15.

Ở thử nghiệm này, ta sử dụng các vị trí giống ở thử nghiệm 1 nhưng thêm ràng buộc về cửa sổ thời gian của các khách hàng nên kết quả định tuyến sẽ không giống với kết quả ở thử nghiệm 1.

4.3.1 Mô tả dữ liệu

Mô tả bằng hình ảnh:



Hình 4.3: Hình ảnh mô tả dữ liệu thử nghiệm 3

Kết quả ma trận thời gian đầu vào (đơn vị là giây):

0	344	550	718	720	684	624
504	0	554	463	492	429	369
420	451	0	593	484	559	368
759	464	670	0	292	78	387
740	444	435	268	0	236	193
717	421	595	215	216	0	312
476	336	279	366	218	334	0

4.3.2 Kết quả

Kết quả hệ thống:

```
Objective: 3119695
Route for vehicle 0:
0 Time(0,0) Distance:0 Load:0 -> 2 Time(600,600)
   Distance:7814 Load:0 -> 1 Time(1051,1051) Distance:15029
   Load:2 -> 0 Time(1555,1555) Distance:24449 Load:4)
Time of the route: 1555sec
Distance of the route: 24449m
Load of the route: 4

Route for vehicle 1:
0 Time(0,0) Distance:0 Load:0 -> 5 Time(684,684)
   Distance:11073 Load:0 -> 4 Time(900,900) Distance:14415
   Load:2 -> 3 Time(1168,1168) Distance:18118 Load:3 -> 6
   Time(1555,1555) Distance:23635 Load:4 -> 0
   Time(2031,2031) Distance:30646 Load:6)
Time of the route: 2031sec
Distance of the route: 30646m
Load of the route: 6

Total time of all routes: 3586sec
```

```
Total distance of all routes: 55095m
Total load of all routes: 10
```

Giải thích kết quả:

- Thời gian trong cửa sổ thời gian trong kết quả là thời gian mà xe đã di chuyển trong lộ trình đó (đơn vị là giây). Thời gian đến tại các điểm của khách bằng thời gian đã di chuyển cộng với thời gian bắt đầu xuất phát của xe.
  Ví dụ: *0 Time(0,0)* $\rightarrow$ *2 Time(600,600)* có nghĩa từ điểm xuất phát của xe đến điểm 2 mất 600s = 10 phút. Vậy xe đến điểm 2 lúc 8:40.

- Lộ trình cho xe 1 là: 0 $\rightarrow$ 2 $\rightarrow$ 1 $\rightarrow$ 0 Hub (8:30) $\rightarrow$ Văn Cao (8:40) $\rightarrow$ Văn Quán (8:48) $\rightarrow$ Hub (8:56)
  Độ dài lộ trình 1: 24449m
  Thời gian lộ trình 1: 1555s
  Số khách trả: 4

- Lộ trình cho xe 2 là: 0 $\rightarrow$ 5 $\rightarrow$ 4 $\rightarrow$ 3 $\rightarrow$ 6 $\rightarrow$ 0
  Hub (8:30) $\rightarrow$ Giáp Bát (8:41) $\rightarrow$ Trương Định (8:45) $\rightarrow$ Đại Cồ Việt (8:50) $\rightarrow$ Khâm Thiên (8:56) $\rightarrow$ Hub (9:04)
  Độ dài lộ trình 2: 30646m
  Thời gian lộ trình 2: 2031s
  Số khách trả: 6

- Tổng độ dài quãng đường di chuyển của 2 xe: 55095m

- Tổng thời gian di chuyển của 2 xe: 3586s

- Tổng số khách trả: 10

Kết luận và hướng phát triển

Tóm tắt nội dung chính

1. Trình bày định nghĩa, mô hình và một số phương pháp giải bài toán VRP.
2. Phân tích, nhận định và hướng giải pháp một bài toán VRP thực tế.
3. Áp dụng và thực hiện sử dụng một số công cụ hỗ trợ để giải quyết bài toán thực tế.

Hướng phát triển

1. Phân tích thêm các tình huống nghiệp vụ khác của bài toán thực tế.
2. Làm thêm phần giao diện (ứng dụng hoặc trang web) cho chương trình.

Tài liệu tham khảo

Tiếng Việt

1. Cao Ngọc Ánh, Trần Bích Thảo, "Ứng dụng giải thuật Tabu Search trong giải bài toán định tuyến xe", *Tạp chí Khoa học & Công nghệ*, (38-2023), 2022.

2. Nguyễn Việt Hân, *Thuật toán di truyền song song giải bài toán VRP (Vehicle Routing Problem) với hạn chế thời gian*, Luận văn Thạc sĩ Khoa học, Hà Nội, 2009.

3. Trần Lan Phương, *Một thuật toán tối ưu đàn kiến giải bài toán điều phối xe*, Luận văn Thạc sĩ, Hà Nội, 2019.

Tiếng Anh

4. Guy Desaulniers, Jacques Desrosiers, Marius M. Solomon, *Column Generation*, Chapter 3: Vehicle Routing Problem with Time Windows, 2005, [67-98].

5. Paolo Toth, Daniele Vigo, *The Vehicle Routing Problem*, Society for Industrial and Applied Mathematics Philadelphia, 2002.

6. Pedro Munari, Twan Dollevoet, Remy Spliet, "A generalized formulation for vehicle routing problems", *Optimization and Control (math.OC)*, (arXiv:1606.01935), 2016.