**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

**======\*\*\*======**

****

**BÁO CÁO BÀI TẬP LỚN THUỘC HỌC PHẦN:**

**THỰC TẬP CƠ SỞ NGÀNH**

**ỨNG DỤNG THUẬT TOÁN GREEDY**

**TỐI ƯU HOÁ HỆ THỐNG ĐỊNH TUYẾN XE BUÝT**

|  |  |
| --- | --- |
| GVHD: | TS.Nguyễn Thị Mỹ Bình |
| Nhóm:  Lớp: | 8  20222IT6040001 |
| Thành viên: | Đặng Đức Mạnh - 2020604935 |
|  | Vũ Mạnh - 2021605274  Đoàn Khắc Minh - 2021601536  Lê Lệnh Nguyên - 2021604114  Cao Đình Nhật - 2021606232 |
|  |  |
|  |  |

Hà Nội, năm 2023

MỤC LỤC

[Chương 1. Giới thiệu chung 5](#_Toc137396381)

[1.1 Bài toán tối ưu hóa hệ thống định tuyến xe buýt 5](#_Toc137396382)

[1.2 Mục tiêu nghiên cứu 5](#_Toc137396383)

[1.2.1 Mục tiêu tổng quát 5](#_Toc137396384)

[1.2.2 Mục tiêu cụ thể 5](#_Toc137396385)

[1.3 Đối tượng và phạm vi nghiên cứu 6](#_Toc137396386)

[1.3.1 Đối tượng nghiên cứu 6](#_Toc137396387)

[1.3.2 Phạm vi nghiên cứu 6](#_Toc137396388)

[1.4 Công cụ sử dụng 7](#_Toc137396389)

[Chương 2. Cơ sở lý thuyết 8](#_Toc137396390)

[2.1 Bài toán ứng dụng thuật toán Greedy để tối ưu hóa hệ thống định tuyến xe buýt 8](#_Toc137396391)

[2.1.1 Lý thuyết về tối ưu hóa 8](#_Toc137396392)

[2.1.2 Mô tả bài toán 8](#_Toc137396393)

[2.2 Thuật toán Greedy (Tham lam) 10](#_Toc137396394)

[2.2.1 Nguồn gốc thuật toán 10](#_Toc137396395)

[2.2.2 Định nghĩa thuật toán Greedy 10](#_Toc137396396)

[2.2.3 Thành phần của thuật toán 10](#_Toc137396397)

[2.2.4 Phân tích thuật toán 11](#_Toc137396398)

[2.2.5 Ưu điểm và nhược điểm 12](#_Toc137396399)

[2.2.6 Ứng dụng của thuật toán vào các bài toán 13](#_Toc137396400)

[Chương 3. Cài đặt và kiểm thử 14](#_Toc137396401)

[3.1 Phát biểu bài toán 14](#_Toc137396402)

[3.1.1 Phát biểu bài toán 14](#_Toc137396403)

[3.1.2 Xác định bài toán 14](#_Toc137396404)

[3.2 Thiết kế thuật toán 15](#_Toc137396405)

[3.2.1 Xác định thuật toán 15](#_Toc137396406)

[3.2.2 Lưu đồ thuật toán 16](#_Toc137396407)

[3.3 Cài đặt chương trình 17](#_Toc137396408)

[3.4 Kiểm thử chương trình 19](#_Toc137396409)

[Chương 4. Kết luận 23](#_Toc137396410)

[4.1 Nội dung đã thực hiện 23](#_Toc137396411)

[4.2 Kỹ năng học được 24](#_Toc137396412)

[4.3 Bài học rút ra 24](#_Toc137396413)

[Chương 5. Tài liệu tham khảo 25](#_Toc137396414)

**Lời Mở Đầu**

Với tốc độ tăng trưởng chóng mặt như hiện nay, các ngành cũng phát triển với quy mô rộng lớn, đặc biệt đối với ngành dịch vụ. Một nền kinh tế năng động phải có một ngành dịch vụ phát triển với nhiều loại hình. Nhu cầu con người tăng cao, xuất hiện các loại hình về sức khoẻ, du lịch, chăm sóc,… Điều đó đi đôi với việc phải tính đến nhu cầu đi lại, tắc đường luôn là vấn đề gian nan trong việc phát triển và vận hành hệ thống giao thông công cộng. Chính vì vậy để giải quyết tắc đường các kĩ sư cần giải quyết các bài toán tối ưu số lượng xe buýt cần thiết

Việc tối ưu hoá số lượng xe buýt cần thiết để phục vụ một tuyến đường có thể giảm chi phí và tăng hiệu quả trong việc vận hành hệ thống giao thông công cộng.

Chính vì sự phát triển của giao thông công cộng và những lợi ích mà nó mang lại, dưới sự hướng dẫn của giảng viên Nguyễn Thị Mỹ Bình, nhóm 8 chúng em quyết định chọn đề tài: ”**Bài toán ứng dụng thuật toán Greedy để tối ưu hoá hệ thống định tuyến xe buýt**” để thực hiện bài tập lớn. Đề tài này phân tích và đặc tả thuật toán Greedy ứng dụng vào bài toán tối ưu hoá định tuyến xe buýt dựa trên việc khảo sát yêu cầu từ thực tiễn, giữa công ty vận tải và hành khách nhằm mục đích nâng cấp hoặc xây dựng mới hệ thống để giúp phát triển và tối ưu trong ngành giao thông vận tải công cộng.

Trong quá trình tìm hiểu và thực hiện báo cáo bài tập lớn, nhóm chúng em không thể tránh khỏi những thiếu sót, vì vậy, nhóm rất mong được nghe sự nhận xét và đánh giá của cô dành cho nhóm.

Nhóm chúng em xin chân thành cảm ơn cô!

**Nhóm sinh viên thực hiện đề tài**

# Giới thiệu chung

## Bài toán tối ưu hoá hệ thống định tuyến xe buýt

Ngày nay, do sự phát triển đô thị cũng như dân số. Tình hình giao thông nước ta ngày càng đi vào ùn tắc, chính vì vậy nhu cầu sử dụng phương tiện công cộng ngày càng cao. Điển hình là sử dụng xe buýt công cộng. Các công ty vận tải công cộng cần phải nghiên cứu để vận hành mô hình xe buýt. Từ đó bài toán tối ưu hoá hệ thống định tuyến xe buýt được phát triển lên. Đó là một bài toán tối ưu trong lĩnh vực vận tải. Liên quan đến việc xác định các tuyến đường và lịch trình tối ưu cho các xe buýt trong một mạng lưới vận chuyển cố định. Mục tiêu chính là phân bổ các xe buýt một các hiệu quả để đáp ứng nhu cầu khách hàng và làm giảm thiểu chi phí và cải thiện chất lượng dịch vụ.

## Mục tiêu nghiên cứu

### Mục tiêu tổng quát

Mục tiêu tổng quát của nghiên cứu là tối ưu hoá các bố trí tuyến đường, thời gian di chuyển của một xe buýt và số lượng xe buýt cần dùng trong một lộ trình cố định nhằm giải quyết hoặc giảm bớt các vấn đề của các tuyến xe buýt hiện tại

### Mục tiêu cụ thể

* **Tối ưu hoá tuyến đường:** Cần tìm ra các tuyến đường ngắn nhất và hiệu quả nhất để đưa hành khách từ điểm A đến điểm B. Việc này đòi hỏi phải xem xét các yếu tố như thời gian di chuyển, mật độ giao thông, nhu cầu vận chuyển, cấu trúc địa hình, v.v.
* **Tối ưu hoá lịch trình:** Cần phân bổ các chuyến xe buýt sao cho đảm bảo đủ lượng xe phục vụ đúng giờ cao điểm và giảm thiểu thời gian chờ đợi của hành khách. Điều này đòi hỏi phải tính toán các thông số như số lượng chuyến xe, thời gian di chuyển trung bình, mật độ hành khách trên từng tuyến đường, v.v.
* **Tối ưu hoá số lượng xe:** Cần tính toán số lượng xe buýt cần thiết để đảm bảo đáp ứng nhu cầu của hành khách trong suốt thời gian hoạt động của hệ thống. Việc này đòi hỏi phải xem xét các yếu tố như mật độ giao thông, số lượng hành khách, tần suất chuyến xe, v.v.
* **Tối ưu hoá quản lý và bảo trì**: Cần quản lý và bảo trì các xe buýt sao cho đảm bảo chất lượng hoạt động và sự an toàn cho hành khách. Việc này đòi hỏi phải tính toán các thông số như số lượng xe đang hoạt động, tần suất bảo trì, chi phí bảo trì, v.v

## Đối tượng và phạm vi nghiên cứu

### Đối tượng nghiên cứu

Đối tượng nghiên cứu của bài toán này là thuật toán Greedy(Tham lam) và các mô hình đồ thị, các dữ liệu thực tế liên quan tới việc giải bài toán nhằm tối ưu hoá thời gian đi lại của khách hàng và tối ưu hoá hệ thống chi phí của công ty vận tải giao thông công cộng.

### Phạm vi nghiên cứu

Phạm vi nghiên cứu của bài toán ứng dụng thuật toán Greedy để tối ưu hoá hệ thống định tuyến xe buýt có thể bao gồm các khía cạnh sau:

* Định nghĩa và mô hình hoá bài toán: Xác định rõ ràng định nghĩa của bài toán định tuyến xe buýt và xây dựng mô hình toán học cho nó. Điều này bao gồm định nghĩa các biến, ràng buộc và hàm mục tiêu cần được tối ưu.
* Phát triển thuật toán Greedy: Nghiên cứu và phát triển thuật toán Greedy để áp dụng vào bài toán định tuyến xe buýt. Thuật toán Greedy sẽ dựa vào quyết định tại thời điểm hiện tại để chọn tuyến đường và lịch trình tối ưu. Cần xác định các quy tắc và tiêu chí quyết định cụ thể cho việc lựa chọn tuyến đường và lịch trình.
* Tối ưu hoá các mục tiêu cụ thể: Nghiên cứu các mục tiêu cụ thể trong bài toán định tuyến xe buýt và tìm cách tối ưu hoá chúng bằng thuật toán Greedy. Các mục tiêu có thể bao gồm tối thiểu hoá thời gian di chuyển, số lượng xe buýt, chi phí vận hành và sử dụng tài nguyên.
* Xác định giới hạn và ràng buộc: Xem xét các giới hạn và ràng buộc của bài toán định tuyến xe buýt, chẳng hạn như giới hạn thời gian hoạt động, quy định về số lượng xe buýt, tần suất dừng, và sức chứa của xe buýt. Đảm bảo rằng các giải pháp đáp ứng được các yêu cầu và hạn chế cụ thể.
* Đánh giá và so sánh kết quả: Đánh giá hiệu suất của thuật toán Greedy khi áp dụng vào bài toán định tuyến xe buýt. So sánh kết quả thu được với các phương pháp khác để đánh giá tính hiệu quả của thuật toán Greedy và khả năng tối ưu hoá của nó.
* Nghiên cứu mở rộng: Khám phá và nghiên cứu các phương pháp mở rộng hoặc cải tiến thuật toán.

## Công cụ sử dụng

Với bài toán "ứng dụng thuật toán Greedy để tối ưu hoá hệ thống định tuyến xe buýt" nhóm chúng em xin dùng những công cụ sau để hoàn thiện sản phẩm:

* Công cụ Office 365 (Word, PowerPoint, Excel)
* Nền tảng họp trực tuyến Google meet.
* Ngôn ngữ sử dụng để mô tả thuật toán: Java, python,…
* Môi trường cài đặt thuật toán: Eclipse IDE, Netbeans,..

……

# Cơ sở lý thuyết

## Bài toán ứng dụng thuật toán Greedy để tối ưu hoá hệ thống định tuyến xe buýt

### Lý thuyết về tối ưu hoá

* Khái niệm tối ưu hoá:
* Tối ưu hoá: là lĩnh vực nghiên cứu lý thuyết và thuật toán giải các bài toán cực trị. Nghĩa là, tìm trạng thái tối ưu của một hệ thống bị ràng buộc, sao cho đạt được mục tiêu mong muốn về chất lượng
* Tối ưu hoá trong lĩnh vực giao thông:
* Tối ưu hoá trong lĩnh vực giao thông công cộng là quá trình nghiên cứu và áp dụng các phương pháp và công nghệ để cải thiện hiệu suất, hiệu quả và chất lượng của hệ thống giao thông công cộng. Mục tiêu chính của tối ưu hoá trong lĩnh vực này là tạo ra một hệ thống vận chuyển công cộng thông minh, linh hoạt và bền vững, đáp ứng nhu cầu di chuyển của người dân và đồng thời giảm thiểu ùn tắc giao thông, lưu lượng khí thải và tiêu thụ năng lượng

### Mô tả bài toán

Bài toán tối ưu hoá hệ thống định tuyến xe buýt là một bài toán quan trọng trong lĩnh vực giao thông và vận tải. Nó liên quan đến việc xác định tuyến đường tối ưu cho các xe buýt để đáp ứng nhu cầu vận chuyển của hành khách và tối đa hoá hiệu quả hoạt động của hệ thống.

Bài toán tối ưu hoá hệ thống định tuyến xe buýt có thể được mô tả như sau:

* Đầu vào:

+) Bản đồ đường phố: xác định mạng lưới tuyến đường, các điểm và đón trả khách

+)Dữ liệu hành khách: Bao gồm thông tin lưu lượng khách, các điểm xuất phát và điểm đến, thời gian và mật độ hành khách trên đường.

* Mục tiêu:

+) Tối ưu hoá thời gian di chuyển : Xác định các tuyến đường cho xe buýt sao cho thời gian di chuyển của hành khách được tối thiểu hoá. Điều này đảm bảo rằng hành khách có thể di chuyển từ điểm xuất phát đến điểm đến một cách nhanh chóng và thuận tiện.

+) Tối ưu hoá sử dụng tài nguyên: Xác định số lượng và lộ trình của các xe buýt sao cho tối ưu hoá việc sử dụng tài nguyên như nhiên liệu, nhân lực và phương tiện.

* Các ràng buộc:

+) Ràng buộc về đường phố: Xe buýt chỉ có thể di chuyển trên các tuyến đường hợp lệ và tuân thủ các ràng buộc về chiều, loại phương tiện và các giới hạn tốc độ.

+) Ràng buộc về mật độ hành khách: Mỗi xe buýt có thể chở được một số hành khách giới hạn. Điều này đảm bảo rằng hành khách có đủ chỗ ngồi và không gây quá tải trên các tuyến đường.

+) Ràng buộc về thời gian: Hệ thống định tuyến xe buýt phải tuân thủ các ràng buộc thời gian, bao gồm thời gian khởi hành, thời gian đến điểm dừng và thời gian di chuyển giữa các điểm dừng. Điều này đảm bảo rằng các xe buýt hoạt động đúng giờ và phục vụ hành khách theo lịch trình.

+) Ràng buộc về tài nguyên: Hệ thống định tuyến cần xem xét các ràng buộc về tài nguyên như số lượng xe buýt có sẵn, nhiên liệu, nhân lực và nguồn lực khác. Giải pháp tối ưu phải đảm bảo tận dụng tài nguyên hiệu quả và giảm thiểu sự lãng phí.

## Thuật toán Greedy (Tham lam)

### Nguồn gốc thuật toán

Nguồn gốc của thuật toán Greedy có thể được truy ngược lại đến những thuật toán phân tích và thiết kế thuật toán truyền thống trong khoa học máy tính, đặc biệt là các thuật toán tìm kiếm. Thuật toán Greedy đã được sử dụng từ rất sớm trong lịch sử của khoa học máy tính, trong đó có thuật toán Dijkstra để tìm đường đi ngắn nhất trong đồ thị, hoặc thuật toán Huffman để mã hoá thông tin.

### Định nghĩa thuật toán Greedy

Thuật toán tham lam là một chiến lược thuật toán đưa ra lựa chọn tối ưu tốt nhất ở từng giai đoạn nhỏ với mục tiêu cuối cùng là đưa ra giải pháp tối ưu toàn cục. Điều này có nghĩa là thuật toán chọn giải pháp tốt nhất tại thời điểm đó mà không quan tâm đến hậu quả. Nó chọn đầu ra tốt nhất ngay lập tức, nhưng không xem xét bức tranh toàn cảnh, do đó nó được coi là tham lam.

### Thành phần của thuật toán

* Thuật toán Greedy bao gồm 5 thành phần:
* Một tập hợp các ứng viên (candidate), để từ đó tạo ra lời giải
* Một hàm lựa chọn, để theo đó lựa chọn ứng viên tốt nhất để bổ sung vào lời giải
* Một hàm khả thi (feasibility), dùng để quyết định nếu một ứng viên có thể được dùng đê xây dựng lời giải
* Một hàm mục tiêu, ấn định giá trị của lời giải hoặc một lời giải chưa hoàn chỉnh
* Một hàm đánh giá, chỉ ra khi nào ta tìm ra một lời giải hoàn chỉnh
* Trong đó có hai thành phần quyết định đến quyết định tham lam:
* Tính chất lựa chọn tham lam:

Chúng ta có thể lựa chọn giải pháp nào được cho là tốt nhất ở thời điểm hiện tại và sau đó giải bài toán con nảy sinh từ việc thực hiện lựa chọn vừa rồi. Lựa chọn của thuật toán tham lam có thể phụ thuộc vào các lựa chọn trước đó. Nhưng nó không thể phụ thuộc vào một lựa chọn nào trong tương lai hay phụ thuộc vào lời giải của các bài toán con. Thuật toán tiến triển theo kiểu thực hiện các chọn lựa theo một vòng lặp, cùng lúc đó thu nhỏ bài toán đã cho về một bài toán con nhỏ hơn. Đấy là khác biệt giữa thuật toán này và giải thuật Quy Hoạnh Động. Giải thuật quy hoạch động duyệt hết và luôn đảm bảo tìm thấy lời giải. Tại mỗi bước của thuật toán, quy hoạch động đưa ra quyết định dựa trên các quyết định của bước trước, và có thể xét lại đường đi của bước trước hướng tới lời giải. Giải thuật tham lam quyết định sớm và thay đổi đường đi thuật toán theo quyết định đó, và không bao giờ xét lại các quyết định cũ. Đối với một số bài toán, đây có thể là một thuật toán không chính xác.

* Cấu trúc con tối ưu:

Một bài toán được gọi là “có cấu trúc tối ưu”, nếu một lời giải tối ưu của bài toán con chứa lời giải tối ưu của bài toán lớn hơn.

### Phân tích thuật toán

* Phương pháp cài đặt

Điều kiện để bài toán có thể giải bằng phương pháp Tham lam đó là chúng thuộc lớp các bài toán tối ưu tổ hợp là một trường hợp riêng của bài toán tối ưu:

Các bài toán tối ưu có dạng tổng quát như sau:

* Hàm f(X) được gọi là hàm mục tiêu,xác định trên một tập hữu hạn các phần tử D.
* Mỗi phần tử X thuộc D có dạng X =() được gọi là một phương án.
* Tìm một phương án thuộc D sao cho f(X) đạt max hoặc min trên D thì được gọi là phương pháp tối ưu.
* Tập D được gọi là tập các phương án của bài toán.
* Cấu trúc chung của một thuật toán tham lam có thể được tóm tắt trong các bước sau:

1. Xác định vấn đề là một vấn đề tối ưu hóa trong đó chúng ta cần tìm giải pháp tốt nhất trong số các giải pháp khả thi.
2. Xác định tập nghiệm khả thi cho bài toán.
3. Xác định cấu trúc con tối ưu của bài toán, nghĩa là lời giải tối ưu cho bài toán có thể được xây dựng từ các lời giải tối ưu của các bài toán con của nó.
4. Xây dựng chiến lược tham lam để từng bước xây dựng lời giải khả thi, đưa ra lựa chọn tối ưu cục bộ tại mỗi bước.

Chứng minh tính đúng đắn của thuật toán bằng cách chỉ ra rằng các lựa chọn tối ưu cục bộ ở mỗi bước dẫn đến một giải pháp tối ưu toàn cục.

### Ưu điểm và nhược điểm

* Ưu điểm:
* **Tính đơn giản** – Các thuật toán tham lam thường đơn giản và dễ hiểu, khiến chúng có thể tiếp cận được với các lập trình viên ở mọi cấp độ kỹ năng. Chúng liên quan đến việc đưa ra các lựa chọn tối ưu cục bộ ở mỗi bước, thay vì cố gắng tìm ra giải pháp tối ưu toàn cầu.
* **Hiệu quả** – Các thuật toán tham lam có thể rất hiệu quả xét về độ phức tạp của không gian và thời gian. Chúng thường liên quan đến việc lặp lại thông qua các đầu vào của vấn đề một lần và các giải pháp của chúng thường có độ phức tạp thời gian tuyến tính hoặc logarit.
* **Tính linh hoạt** – Các thuật toán tham lam có thể được áp dụng cho nhiều vấn đề khác nhau, từ lập biểu đến duyệt đồ thị đến nén dữ liệu. Chúng là một công cụ linh hoạt có thể được sử dụng trong nhiều ngữ cảnh khác nhau.
* Nhược điểm:
* **Tối ưu cục bộ** – Các thuật toán tham lam đưa ra các lựa chọn tối ưu cục bộ ở mỗi bước mà không xem xét bối cảnh lớn hơn. Điều này có nghĩa là chúng không phải lúc nào cũng dẫn đến giải pháp tối ưu toàn cầu
* **Khả năng ứng dụng hạn chế** – Mặc dù các thuật toán tham lam có thể được sử dụng để giải quyết nhiều loại vấn đề, nhưng có một số loại vấn đề không phù hợp với phương pháp này. Đặc biệt, các vấn đề liên quan đến nhiều ràng buộc hoặc các mục tiêu xung đột có thể yêu cầu các kỹ thuật tinh vi hơn.
* **Khó khăn trong phân tích** – Phân tích tính chính xác và hiệu quả của các thuật toán tham lam có thể là một thách thức, đòi hỏi sự hiểu biết sâu sắc về vấn đề và bản thân thuật toán. Điều này có thể gây khó khăn cho việc tối ưu hóa thuật toán hoặc xác định những cạm bẫy tiềm ẩn

### Ứng dụng của thuật toán vào các bài toán

**Bài toán:** Bài toán xếp ba lô

Ta có một ba lô có trọng lượng là m và n loại đồ vật với trọng lượng và giá trị tương ứng, yêu cầu ở đây là bạn sẽ phải chọn tối đa số lượng đồ vật để vừa phù hợp với trọng lượng của ba lô mà giá trị lấy được là lớn nhất.

**Bài toán:** Bài toán người du lịch

Một người du lịch muốn tham quan n thành phố từ T1,.., Tn.Xuất phát từ một thành phố nào đó, người du lịch muốn đi qua tất cả các thành phố còn lại, mỗi thành thành phố đúng 1 lần rồi quay trở lại thành phố xuất phát. Gọi Cij là chi phí đi từ thành phố Ti đến Tj. Hãy tìm một hành trình thỏa mãn yêu cầu bài toán sao cho chi phí là nhỏ nhất.

# Cài đặt và kiểm thử

## Phát biểu bài toán

### Phát biểu bài toán

Một công ty cần phát triển một hệ thống xe buýt công cộng tại một thành phố. Xuất phát từ bến A, đi qua n điểm dừng xe buýt để đón trả khách rồi kết thúc tại bến B.Trong một vòng chạy xe buýt sẽ được đi trong một thời gian t nhất định. Trong một ngày xe buýt đó có thể đi qua các điểm dừng m lần và khoảng các giữa các điểm dừng là tương đối bằng nhau và xác định được bởi đơn vị độ dài (m, km,…) hoặc thời gian di chuyển (phút, giờ, …). Hãy tìm một lộ trình đường đi khép kín sao cho tổng độ dài từ bến A đến bến B là nhỏ nhất nhưng vẫn đảm bảo có n điểm dừng xe buýt và thời gian t là nhỏ nhất, từ đó đưa ra được số lượng xe cần dùng trong một tuyến chạy. Bài toán này gọi là tối ưu hóa hệ thống định tuyến xe buýt.

### Xác định bài toán

Input:

* Đầu vào là một số nguyên dương n (n > 0) tương ứng với n điểm dừng xe buýt và ma trận khoảng cách giữa các điểm dừng a[i][j] (i, j >0).
* Điểm bắt đầu beginID, điểm kết thúc stopID
* Số lượng khách hàng num\_passengers

Output:

* Đầu ra là một danh sách điểm dừng sao cho tối ưu hóa lộ trình theo yêu cầu bài toán.
* Thời gian di chuyển nhỏ nhất minTime.
* Số lượng xe buýt cần sử dụng numberBus.

Ràng buộc:

* Tốc độ di chuyển của xe buýt Bus\_speed.
* Sức chứa giới hạn của xe buýt Bus\_capacity.

Trong bài toán này, ta sử dụng một số kỹ thuật lập trình như khởi tạo một ma trần khoảng cách giữa các điểm dừng, sử dụng mảng và vòng lặp để lưu trữ và xử lý dữ liệu , sử dụng giải thuật Greedy để tìm đường đi ngắn nhất và sử dụng phương thức để tính toán thời gian cần thiết để di chuyển giữa hai điểm dừng trên tuyến đường.

## Thiết kế thuật toán

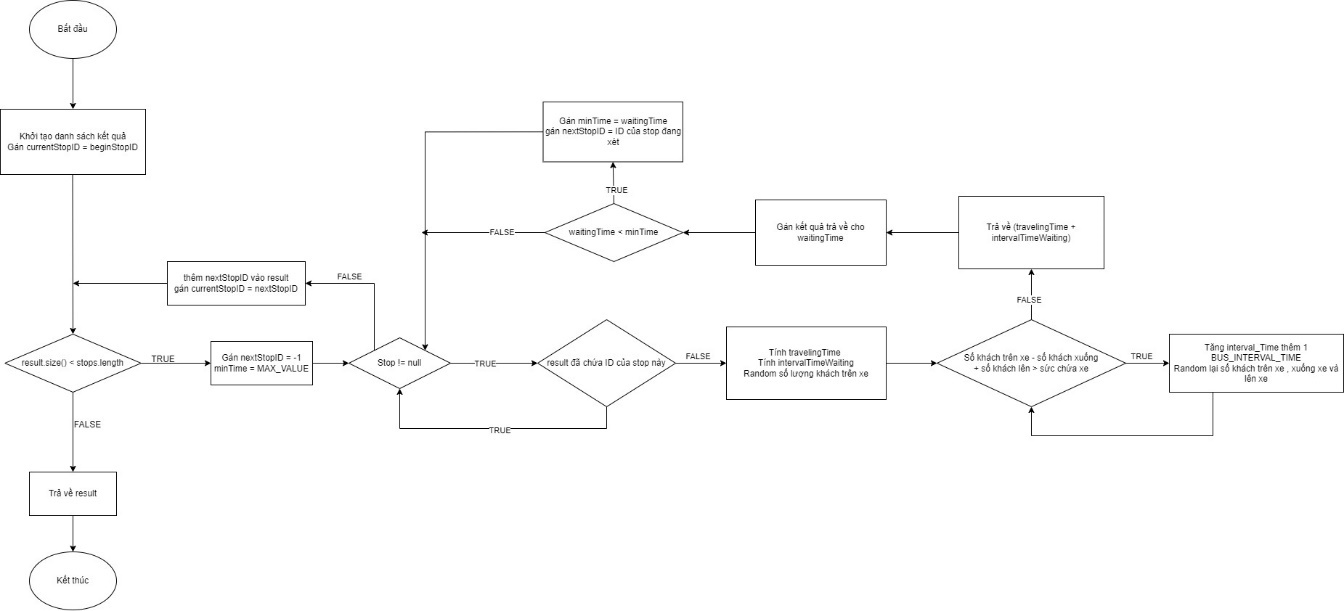
### Xác định thuật toán

Thuật toán sẽ bắt đầu chạy khi bộ dữ liệu đầu vào được đưa vào và bắt đầu tại điểm xuất phát là 1 điểm dừng.

* Ta sẽ duyệt qua tất cả các điểm dừng ở trong danh sách các điểm dừng cần đi qua và sẽ bỏ qua những điểm dừng đã đi qua. Tại mỗi điểm dừng chưa được đi qua , ta tiến hành tính toán tổng thời gian di chuyển tới điểm dừng này và chờ 1 xe buýt cập bến tại điểm dừng đó , để đi đến các bến kế tiếp.
* Cụ thể , ta cần tính thời gian di chuyển giữa điểm A (điểm hiện tại) tới điểm B (điểm đang xét) , gọi thời gian đi từ điểm A tới B là T1. Giả sử, khi ta tới điểm B , ta cần phải đi từ điểm B đến điểm C kế tiếp , tuy nhiên khi ta xuống điểm B , chưa chắc đã có bus ngay để đi mà cần phải chờ xe buýt vì sau 1 thời gian nhất định thì sẽ có 1 xe buýt cập bến , ta gọi đây là thời gian chờ xe buýt T2. Khi xe buýt cập bến , ta cần phải xem xét xem số lượng hành khách của chuyến xe bus này , nếu như số khách trên xe – số khách xuống + số khách lên (tại bến đang xét) lớn hơn sức chứa của xe , ta cần phải chờ đến khi có chuyến xe còn chỗ để đi (số lượng khách sẽ được random ngẫu nhiên) , điều này dẫn đến thời gian chờ bus T2 tăng lên . Như vậy , tổng thời gian đi từ điểm A tới B và chờ xe buýt để đi từ B tới C (T1 + T2) sẽ là tiêu chí lựa chọn điểm đến kế tiếp một cách tối ưu.
* Sau khi xét hết các điểm dừng , ta sẽ chọn được điểm đến có thời gian lý tưởng nhất . Ta tiến hành thêm điểm đến này vào danh sách đồng thời cập nhật điểm dừng hiện tại và thực hiện cho tới khi thăm hết các điểm dừng có trong danh sách và trả về kết quả là danh sách thứ tự các điểm dừng.

### Lưu đồ thuật toán

Từ xác định thuật toán chúng ta có được lưu đồ thuật toán để giải quyết bài toán như sau :

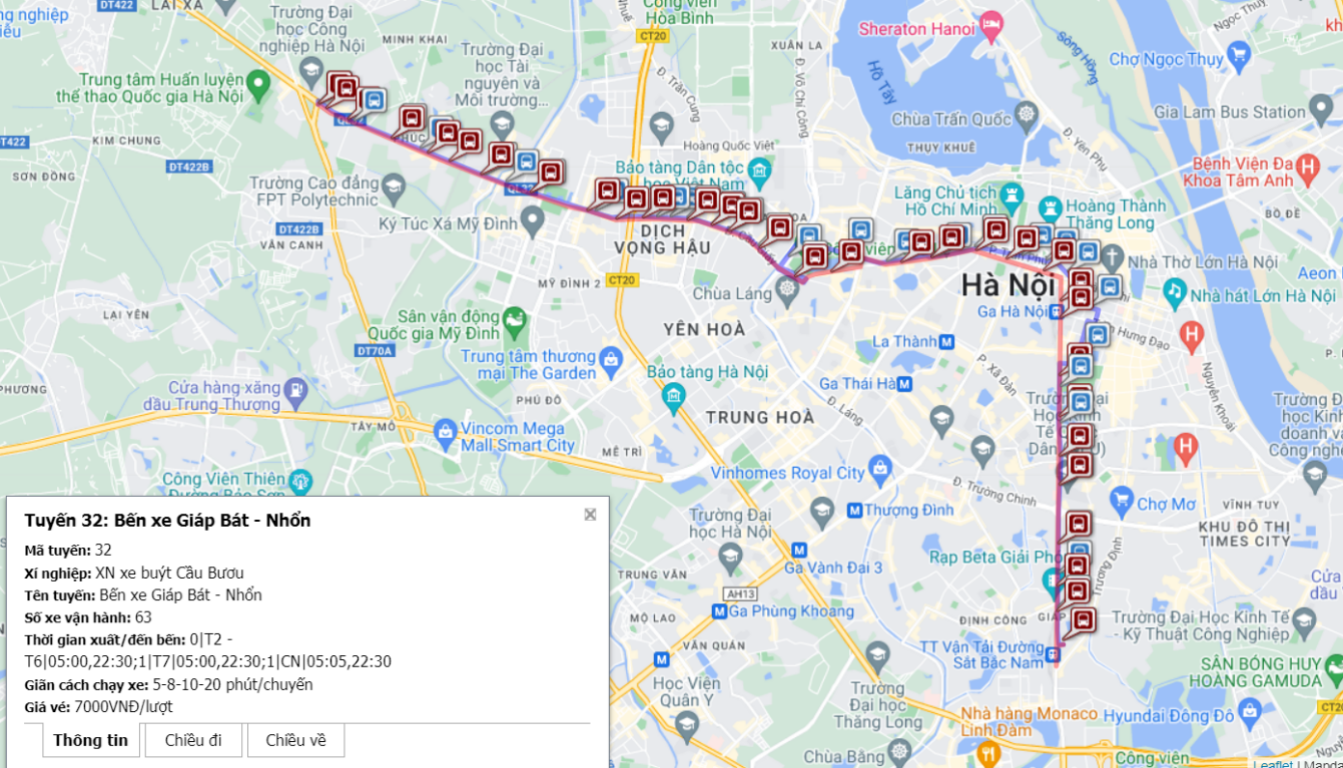
­­­­­­­­­­­

## Cài đặt chương trình

public class BusRoutingOptimization {  
  
 private static final int BUS\_SPEED = 20;  
 private static final int BUS\_CAPACITY = 80;  
 private static final int BUS\_INTERVAL\_TIME = 15; // (minute)  
 private static final int MIN\_WAITING\_PASSENGERS = 20;  
 private static final int MAX\_WAITING\_PASSENGERS = 50;  
 private static final int DEFAULT\_NUM\_PASSENGERS\_BUS = 60;  
 private static final int MAX\_NUM\_PASSENGERS\_OUT = 10;  
  
 public static class Stop {  
 public int ID;  
 public int numWaitingPassengers;  
 public int numOutPassengers;  
  
 public Stop() {  
  
 }  
  
 public Stop(int ID) {  
 this.ID = ID;  
 randomPassengers(this);  
 }  
 }  
  
 public BusRoutingOptimization() {  
  
 }  
  
 public static void main(String[] args) {  
  
 Stop[] stops = new Stop[10];  
 int[][] stopsDistance = createDummyData(stops);  
  
 System.out.println(optimizing(stopsDistance, stops, 1).toString());  
 }  
  
 public static int[][] createDummyData(Stop[] stops) {  
 Stop s;  
 for (int i = 0; i < 10; i++) {  
 s = new Stop(i + 1);  
 stops[i] = s;  
 }  
  
 int[][] stopsDistance =   
 { 0, 5, 17, 9, 12, 2, 8, 15, 3, 10 },  
 { 5, 0, 6, 4, 13, 7, 15, 14, 22, 18 },  
 { 17, 6, 0, 9, 2, 11, 5, 8, 14, 11 },  
 { 9, 4, 9, 0, 22, 16, 7, 11, 5, 10 },  
 { 12, 13, 2, 22, 0, 10, 3, 8, 4, 6 },  
 { 2, 7, 11, 16, 10, 0, 13, 8, 3, 1 },  
 { 8, 15, 5, 7, 3, 13, 0, 10, 6, 2 },  
 { 15, 14, 8, 11, 8, 8, 10, 0, 6, 3 },  
 { 3, 22, 14, 5, 4, 3, 6, 6, 0, 15 },  
 { 10, 18, 11, 10, 6, 1, 2, 3, 15, 0 } };  
  
 return stopsDistance;  
 }  
  
 public static void randomPassengers(Stop stop) {  
 stop.numWaitingPassengers = MIN\_WAITING\_PASSENGERS  
 + (int) (Math.random() \* (MAX\_WAITING\_PASSENGERS - MIN\_WAITING\_PASSENGERS + 1));  
 stop.numOutPassengers = (new Random()).nextInt(MAX\_NUM\_PASSENGERS\_OUT + 1);  
 }  
  
 public static int measureTime(int[][] stopsDistanceMap, Stop[] stops, int currentStopID, int nextStopID) {  
  
 // calculate the time to go from current stop to next stop  
 int travelingTime = (int) ((1.00f \* stopsDistanceMap[currentStopID - 1][nextStopID - 1] / BUS\_SPEED) \* 60);  
  
 // measure the minimum time to wait a bus come to the next stop   
 int intervalTimeWaiting = (travelingTime % BUS\_INTERVAL\_TIME == 0) ? 0  
 : BUS\_INTERVAL\_TIME - (travelingTime % BUS\_INTERVAL\_TIME);  
  
 // random the passengers on the bus  
 int passengersOnBus = (new Random()).nextInt(DEFAULT\_NUM\_PASSENGERS\_BUS + 1); // random so luong hanh khach  
  
 // checking the bus is full or not , we have to wait the next bus and the waiting time increase a BUS\_INTERVAL\_TIME  
 while (passengersOnBus - stops[nextStopID - 1].numOutPassengers  
 + stops[nextStopID - 1].numWaitingPassengers > BUS\_CAPACITY) {  
 intervalTimeWaiting += BUS\_INTERVAL\_TIME;  
 randomPassengers(stops[nextStopID - 1]);  
 passengersOnBus = (new Random()).nextInt(DEFAULT\_NUM\_PASSENGERS\_BUS + 1);  
 }  
 // sum of the time we need to go this stop and wait for the bus to go travel to next stop  
 return travelingTime + intervalTimeWaiting;  
 }  
  
 public static ArrayList<Integer> optimizing(int[][] stopsDistanceMap, Stop[] stops, int beginStopID) {  
 ArrayList<Integer> result = new ArrayList<Integer>();  
 result.add(beginStopID);  
 int currentStopID = beginStopID;  
 do {  
 int nextStopID = -1;  
 int minTime = Integer.MAX\_VALUE;  
 for (Stop stop : stops) {  
  
 // ignore this stop if it was traveled  
 if (result.contains(stop.ID)) {  
 continue;  
 }  
  
 // get the time we need to go this stop  
 int waitingTime = measureTime(stopsDistanceMap, stops, currentStopID, stop.ID);  
 if (waitingTime < minTime) {  
 minTime = waitingTime;  
 nextStopID = stop.ID;  
 }  
 }  
 // add the stop that we need minimum time to go in the journey  
 result.add(nextStopID);  
 // apdate current stop to the next stop that have already added above  
 currentStopID = nextStopID;  
 } while (result.size() < stops.length);  
 return result;  
 }  
  
}

## Kiểm thử chương trình

Bài toán tối ưu đường đi xe bus:



Ví dụ tuyến bus 32

Xác định đầu vào bài toán

BUS\_SPEED  // Tốc độ xe bus

BUS\_CAPACITY // Sức chứa xe bus

BUS\_INTERVAL\_TIME // thời gian nghỉ

MIN\_WAITING\_PASSENGERS // thời gian chờ tối thiểu

MAX\_WAITING\_PASSENGERS // thời gian chờ tối đa

DEFAULT\_NUM\_PASSENGERS\_BUS  // số khách mặc định

MAX\_NUM\_PASSENGERS\_OUT // số khách xuống tối đa

stopsDistance  // Mảng 2 chiều thể hiện khoảng cách giữa các trạm bus

Ví dụ một stopsDistance

int[][] stopsDistance =

                { { 0, 5, 17, 9, 12, 2, 8, 15, 3, 10 },

                { 5, 0, 6, 4, 13, 7, 15, 14, 22, 18 },

                { 17, 6, 0, 9, 2, 11, 5, 8, 14, 11 },

                { 9, 4, 9, 0, 22, 16, 7, 11, 5, 10 },

                { 12, 13, 2, 22, 0, 10, 3, 8, 4, 6 },

                { 2, 7, 11, 16, 10, 0, 13, 8, 3, 1 },

                { 8, 15, 5, 7, 3, 13, 0, 10, 6, 2 },

                { 15, 14, 8, 11, 8, 8, 10, 0, 6, 3 },

                { 3, 22, 14, 5, 4, 3, 6, 6, 0, 15 },

                { 10, 18, 11, 10, 6, 1, 2, 3, 15, 0 } };

Tạo bộ dữ liệu kiểm tra bài toán sử dụng ngôn ngữ lập trình Python:

*import random*

*class Stop:*

*def \_\_init\_\_(self, ID):*

*self.ID = ID*

*self.numWaitingPassengers = 0*

*self.numOutPassengers = 0*

*def random\_passengers(stop):*

*stop.numWaitingPassengers = random.randint(MIN\_WAITING\_PASSENGERS, MAX\_WAITING\_PASSENGERS)*

*stop.numOutPassengers = random.randint(0, MAX\_NUM\_PASSENGERS\_OUT)*

*def measure\_time(stops\_distance\_map, stops, current\_stop\_id, next\_stop\_id):*

*traveling\_time = int((1.00 \* stops\_distance\_map[current\_stop\_id - 1][next\_stop\_id - 1] / BUS\_SPEED) \* 60)*

*interval\_time\_waiting = 0 if traveling\_time % BUS\_INTERVAL\_TIME == 0 else BUS\_INTERVAL\_TIME - (traveling\_time % BUS\_INTERVAL\_TIME)*

*passengers\_on\_bus = random.randint(0, DEFAULT\_NUM\_PASSENGERS\_BUS)*

*while passengers\_on\_bus - stops[next\_stop\_id - 1].numOutPassengers + stops[next\_stop\_id - 1].numWaitingPassengers > BUS\_CAPACITY:*

*interval\_time\_waiting += BUS\_INTERVAL\_TIME*

*random\_passengers(stops[next\_stop\_id - 1])*

*passengers\_on\_bus = random.randint(0, DEFAULT\_NUM\_PASSENGERS\_BUS)*

*return traveling\_time + interval\_time\_waiting*

*def optimizing(stops\_distance\_map, stops, begin\_stop\_id):*

*result = [begin\_stop\_id]*

*current\_stop\_id = begin\_stop\_id*

*while len(result) < len(stops):*

*next\_stop\_id = -1*

*min\_time = float('inf')*

*for stop in stops:*

*if stop.ID in result:*

*continue*

*waiting\_time = measure\_time(stops\_distance\_map, stops, current\_stop\_id, stop.ID)*

*if waiting\_time < min\_time:*

*min\_time = waiting\_time*

*next\_stop\_id = stop.ID*

*result.append(next\_stop\_id)*

*current\_stop\_id = next\_stop\_id*

*return result*

*def create\_dummy\_data(stops):*

*for i in range(10):*

*s = Stop(i + 1)*

*stops.append(s)*

*stops\_distance = [*

*[0, 5, 17, 9, 12, 2, 8, 15, 3, 10],*

*[5, 0, 6, 4, 13, 7, 15, 14, 22, 18],*

*[17, 6, 0, 9, 2, 11, 5, 8, 14, 11],*

*[9, 4, 9, 0, 22, 16, 7, 11, 5, 10],*

*[12, 13, 2, 22, 0, 10, 3, 8, 4, 6],*

*[2, 7, 11, 16, 10, 0, 13, 8, 3, 1],*

*[8, 15, 5, 7, 3, 13, 0, 10, 6, 2],*

*[15, 14, 8, 11, 8, 8, 10, 0, 6, 3],*

*[3, 22, 14, 5, 4, 3, 6, 6, 0, 15],*

*[10, 18, 11, 10, 6, 1, 2, 3, 15, 0]*

*]*

*return stops\_distance*

*BUS\_SPEED = random.randint(10, 30)*

*BUS\_CAPACITY = random.randint(50, 100)*

*BUS\_INTERVAL\_TIME = random.randint(10, 30)*

*MIN\_WAITING\_PASSENGERS = random.randint(10, 30)*

*MAX\_WAITING\_PASSENGERS = random.randint(30, 60)*

*DEFAULT\_NUM\_PASSENGERS\_BUS = random.randint(40, 80)*

*MAX\_NUM\_PASSENGERS\_OUT = random.randint(5, 15)*

*stops = []*

*stops\_distance = create\_dummy\_data(stops)*

*begin\_stop\_id = random.randint(1, len(stops))*

*result = optimizing(stops\_distance, stops, begin\_stop\_id)*

*print(result)*

Phân tích các kết quả nhận được :

Từ phần xác định thuật toán , ta có thể thấy rằng , tiêu chí lựa chọn điểm đến lý tưởng cục bộ với chi phí thời gian là ngắn nhất bị chi phối bởi 2 yếu tố : yếu tố địa lý và số lượng người sử dụng phương tiện xe buýt . Ta không thể chỉ dựa vào khoảng cách địa lý để đánh giá xem đó có phải một lựa chọn tối ưu không vì điều này sẽ làm mất đi tính thực tế của bài toán . Dựa trên điều này , ta có thể đưa ra các trường hợp kết quả 1 cách tương đối như sau :

* Trường hợp tốt nhất (Best Case) : Tất cả các chuyến bus gặp tại các điểm dừng đều còn chỗ , ta sẽ không phải chờ đến chuyến sau mới được đi. Lúc này , ta chỉ cần chọn đi các điểm từ gần đến xa để tối ưu chi phí.
* Trường hợp thường (Normal Case) : Lưu lượng người tham gia xe bus tại các điểm này không quá đông , tuy vẫn sẽ có sự cản trở nhưng không hoàn toàn là tại tất cả các điểm dừng đều đông
* Trường hợp tệ nhất (Worst Case) : Điểm xuất phát ban đầu không thuận lợi , điểm gần nhất so với điểm xuất phát xa hơn khoảng cách giữa các điểm còn lại với nhau . Mật độ người tham gia đi xe bus quá đông đúc ảnh hưởng đến quá trình di chuyển.

# Kết luận

Vấn đề về tối ưu hóa giao thông nói chung và hệ thống xe buýt nói riêng là 1 vấn đề khá phức tạp. Trên thực tế , có rất nhiều các tác nhân ảnh hưởng đến việc chọn tuyến đường đi thuận lợi , chẳng hạn như ví trí địa lý giữa các điểm, hệ thống điểm dừng xe bus , mạng lưới xe buýt được chia thành các tuyến , mật độ tham gia giao thông , các yếu tố khác như thời tiết, tình trạng hỏng hóc của đường , mật độ tham gia giao thông trên đường , hay các sự cố dẫn đến tắc nghẽn ,…. Để giải quyết được vấn đề đó , ta cần nhiều hơn về kiến thức và nguồn lực. Chẳng hạn như Google , Grab, Uber , họ có những kỹ sư tài năng hàng đầu thế giới , có nguồn lực để thực hiện những vấn đề như vậy. Do đó , mục tiêu của đề tài là giới thiệu cho mọi người về vấn đề này một cách cơ bản và tổng quan , đồng thời lồng ghép 1 vài yếu tố thực tế để bài toán thêm sinh động và phản ánh được đầy đủ các tính chất của thuật toán Greedy. Từ đó nhóm 8 nghiên cứu và được ra được một số kết luận trong báo cáo như sau:

## Nội dung đã thực hiện

Tài liệu đã trình bày các nội dung như sau:

* Giới thiệu bài toán tối ưu hóa hệ thống định tuyến xe buýt;
* Định nghĩa bài toán tối ưu hóa;
* Giới thiệu và trình bày thuật toán Greedy ;
* Áp dụng thuật toán Greedy trong bài toán tối ưu hóa hệ thống định tuyến xe buýt;
* Cài đặt được chương trình tối ưu hóa cho hệ thống định tuyến xe buýt;
* Kiểm thử các trường hợp thực tế đối với chương trình trên.

## Kỹ năng học được

* Kỹ năng làm việc nhóm;
* Kỹ năng làm bài tập lớn;
* Kỹ năng quản lý tiến độ công việc;
* Kỹ năng tìm hiểu và học hỏi;
* Kỹ năng trình bày báo cáo bài tập lớn;
* Kỹ năng sử dụng các công cụ hỗ trợ như: Meet, Netbeans, Microsoft Word ,….

## Bài học rút ra

* Cần đặt ra các yêu cầu, quy tắc làm việc nhóm một cách cụ thể hơn;
* Cần chú trọng thời gian hoàn thành tiến độ bài tập lớn;
* Cần chọn lọc các tài liệu tham khảo;
* Cần đa dạng hóa các công cụ hỗ trợ trong quá trình làm bài tập lớn.

# Tài liệu tham khảo

Một số tài liệu tham khảo để hoàn thiện báo cáo bài tập lớn về đề tài Ứng dụng thuật toán Greedy để giải bài toán tối ưu hóa hệ thống định tuyến xe buýt như sau:

* <http://www.cs.toronto.edu/~lalla/373s16/notes/Dijkstra.pdf>
* <https://secon.utulsa.edu/cs2123/slides/greed2p.pdf>
* https://drive.google.com/file/d/1yD0CodAs1BETBj0ml5LcN1I1hcXpxDXJ/view?usp=drive\_link