**TRƯỜNG ĐẠI HỌC PHÚ YÊN**

**KHOA KỸ THUẬT – CÔNG NGHỆ**

**Đề tài**

TÌM HIỂU THUẬT TOÁN TỐI ƯU HÓA ĐƯỜNG ĐI CỦA GOOGLE MAP

Sinh viên chịu trách nhiệm : Lê Đức Thịnh

Lớp : DC18CTT01

Giảng viên hướng dẫn : Võ Thị Hồng Loan

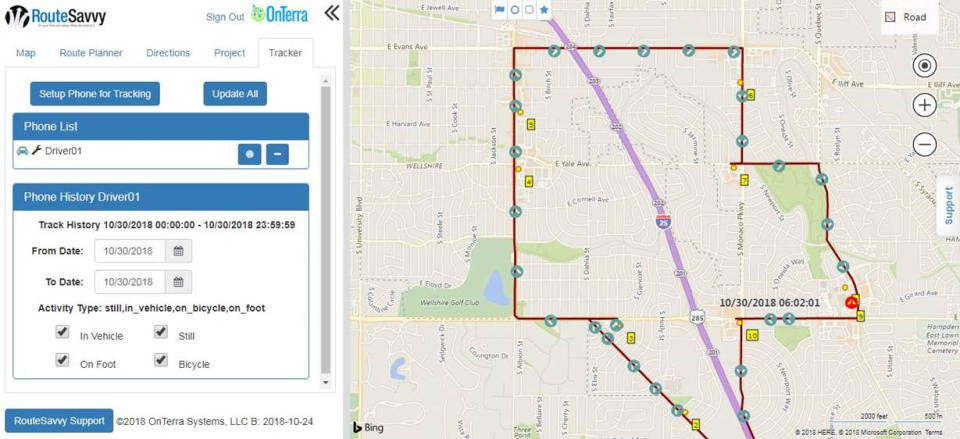
*Phú Yên, 2020*

**MỞ ĐẦU**

1. **Lý do chọn đề tài**

Google Maps là một dịch vụ lập bản đồ web do Google phát triển. Nó cung cấp hình ảnh vệ tinh, chụp ảnh từ trên không, bản đồ đường phố, chế độ xem toàn cảnh tương tác 360 °Của các đường phố (Google Street View), điều kiện giao thông thời gian thực và lập kế hoạch tuyến đường dành cho người đi bộ, ô tô, xe đạp, đường hàng không (trong phiên bản beta) và phương tiện giao thông công cộng. Vào năm 2020, Google Maps đã được hơn 1 tỷ người sử dụng mỗi tháng.

Trong thực tế cuộc sống của chúng ta cần phải giải quyết bài toán phức tạp là làm thế nào để tìm đường đi tối ưu nhất từ đa điểm lựa chọn xuất phát, đa điểm lựa chọn đích đến cho công việc của người đưa thư, người giao hàng hay đi du lịch hàng ngày, ... là vấn đề đang được quan tâm nghiên cứu.



1. **Tổng quan tình hình nghiên cứu**

Tối ưu hóa tuyến đường (Route planner optimizer) là công việc giúp các nhà vận hành tiết kiệm chi phí vận hành Logistic của mình. Vấn đề tìm đường đi trong tập các nút để cho đường đi đơn ngắn nhất giữa 2 nút trong đồ thị đã được giải quyết bằng một số thuật toán như Dijkstra, Hueristic, Floyd.

Nghiên cứu này đề xuất thuật toán tối ưu dựa trên tiếp cận giải thuật di truyền để giải quyết bài toán tìm đường đi qua đa điểm, thuộc lớp bài toán đa nguồn đi, đa đích đến, từ đó đề xuất một ứng dụng tối ưu hóa chi phí đi lại dựa trên dữ liệu Google Maps.

Kết quả thí nghiệm chỉ ra rằng phương pháp đề xuất tối ưu hơn về đường đi và thời gian khi so sánh với các ứng dụng phổ biến như Google Maps, Địa điểm và thuật toán Dijkstra.

Năm 2014, một nghiên cứu về việc vận chuyển sản phẩm hữu cơ của 62 nhà cung cấp tại Istanbul, Thổ Nhĩ Kì cho thấy, giải pháp hoạch định tuyến đường giúp giảm đến 50% chi phí vận chuyển

Kết quả tương tự cũng được tìm thấy tại Quebec, Canada khi Ameublement Tanguay Inc - một công ty hoạt động trong lĩnh vực điện tử dân dụng áp dụng giải pháp hoạch định tuyến đường. Lộ trình giao hàng rút ngắn đi đáng kể, 500,000CAD chi phí được tiết kiệm trong khi chất lượng giao hàng tăng lên đến 23.7%.

1. **Mục tiêu nghiên cứu**

* Tìm ra tuyến đường ngắn nhất từ giữa 2 điểm A và B trong GooGle Map.
* Giới thiệu tổng quan về tuyến đường ngắn nhất để đề xuất vào thuật toán tối ưu hóa.

1. **Đối tượng, phương pháp nghiên cứu**
2. Đối tượng nghiên cứu

* Bản đồ Google Map
* Tối ưu hóa đường đi ngắn nhất

1. Phương pháp nghiên cứu

* Nghiên cứu tài liệu : Nghiên cứu các tài liệu liên quan đến Tối ưu hóa đường đi ngắn nhất

1. **Nội dung nghiên cứu**

**CHƯƠNG 1**: Giới thiệu về Google Map

Sơ lược về Google Map

**CHƯƠNG 2**: Phân tích thuật toán tối ưu hóa đường đi của Google Map

2.1 Tổng quan về thuật toán

2.2 Bài toán tìm đường đi ngắn nhất

2.3 Phương pháp sử dụng thuật toán tới ưu hóa đường đi của Google Map

**CHƯƠNG 3:** Kết luận và hướng phát triển

3.1 Kết luận

3.2 Hướng phát triển

**CHƯƠNG 1:**

**GIỚI THIỆU VỀ GOOGLE MAP**

1. Sơ lược về Google Map

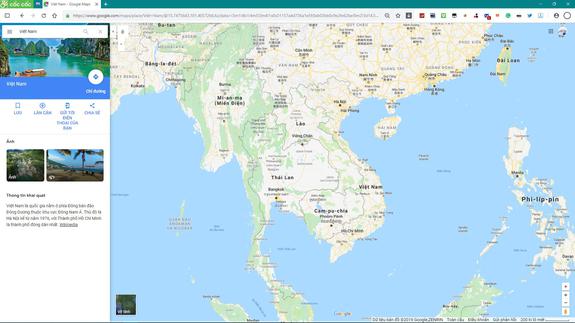
Google Maps là một dịch vụ lập bản đồ web do Google phát triển. Nó cung cấp hình ảnh vệ tinh, chụp ảnh từ trên không, bản đồ đường phố, chế độ xem toàn cảnh tương tác 360 °Của các đường phố (Google Street View), điều kiện giao thông thời gian thực và lập kế hoạch tuyến đường dành cho người đi bộ, ô tô, xe đạp, đường hàng không (trong phiên bản beta) và phương tiện giao thông công cộng. Vào năm 2020, Google Maps đã được hơn 1 tỷ người sử dụng mỗi tháng[2].

Google Maps được khởi đầu như một chương trình desktop viết bằng C ++ tại Where 2 Technologies. Vào tháng 10 năm 2004, khi công ty này được Google mua lại, công ty đã chuyển đổi nó thành một ứng dụng web. Sau khi mua lại một công ty trực quan hóa dữ liệu không gian địa lý và một công ty phân tích lưu lượng truy cập theo thời gian thực, Google Maps đã được ra mắt vào tháng 2 năm 2005 [3]. Phần front end sử dụng Javascript, XML, và Ajax. Google Maps cung cấp một API cho phép nhúng bản đồ trên các trang web của bên thứ ba, và cung cấp công cụ định vị cho các doanh nghiệp và tổ chức khác ở nhiều quốc gia trên thế giới.

Google Map Maker cho phép người dùng cộng tác với nhau để mở rộng và cập nhật bản đồ của dịch vụ trên toàn thế giới nhưng đã ngừng hoạt động từ tháng 3 năm 2017. Tuy nhiên, các đóng góp của nguồn lực cộng đồng cho Google Maps không bị ngừng vì công ty đã thông báo rằng các tính năng đó sẽ được chuyển sang chương trình Google Local Guides.

Chế độ xem vệ tinh của Google Maps là chế độ xem "từ trên xuống" hoặc chế độ xem toàn cảnh; hầu hết các hình ảnh có độ phân giải cao của các thành phố là không ảnh chụp từ máy bay bay ở độ cao 800 đến 1.500 foot (240 đến 460 m), trong khi hầu hết các hình ảnh khác là từ vệ tinh.[5] Phần lớn hình ảnh vệ tinh được chụp trong phạm vi ba năm gần nhất và được cập nhật thường xuyên. Google Maps trước đây đã sử dụng một biến thể của phép chiếu Mercator và do đó không thể hiển thị chính xác các khu vực xung quanh các cực. Vào tháng 8 năm 2018, phiên bản Google Maps dành cho máy tính để bàn đã được cập nhật để hiển thị hình ảnh địa cầu 3D, và có thể chuyển về bản đồ 2D trong phần cài đặt.

Google Maps cho Android và iOS thiết bị được phát hành vào tháng năm 2008 và tính năng dẫn đường chi tiết đến từng ngã tư dùng GPS cùng với tính năng hỗ trợ đậu xe. Vào tháng 8 năm 2013, nó được xác định là ứng dụng phổ biến nhất thế giới dành cho điện thoại thông minh, với hơn 54% chủ sở hữu điện thoại thông minh toàn cầu sử dụng nó ít nhất một lần. Vào năm 2012, Google báo cáo có hơn 7.100 nhân viên và nhà thầu trực tiếp làm việc trong lĩnh vực lập bản đồ. Vào tháng 5 năm 2017, ứng dụng đã được báo cáo có 2 tỷ người dùng trên Android, cùng với một số dịch vụ khác của Google bao gồm YouTube, Chrome, Gmail, Google Search và Google Play, Google Maps đạt hơn 1 tỷ người dùng hàng tháng.



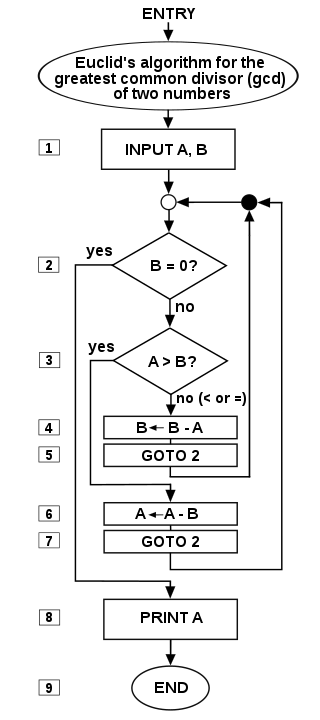
**CHƯƠNG 2:**

**PHÂN TÍCH THUẬT TOÁN TỐI ƯU HÓA ĐƯỜNG ĐI CỦA GOOGLE MAP**

1. Tổng quan về thuật toán

Trong toán học và khoa học máy tính, một thuật toán, còn gọi là giải thuật, là một tập hợp hữu hạn các hướng dẫn được xác định rõ ràng, có thể thực hiện được bằng máy tính, thường để giải quyết một lớp vấn đề hoặc để thực hiện một phép tính. Các thuật toán luôn rõ ràng và được sử dụng chỉ rõ việc thực hiện các phép tính, xử lý dữ liệu, suy luận tự động và các tác vụ khác.

Là một phương pháp hiệu quả, một thuật toán có thể được biểu diễn trong một khoảng không gian và thời gian hữu hạnvà bằng một ngôn ngữ hình thức được xác định rõ ràng để tính toán một hàm số. Bắt đầu từ trạng thái ban đầu và đầu vào ban đầu (có thể trống), các hướng dẫn mô tả một phép tính, khi được thực thi, sẽ tiến hành qua một số hữu hạn các trạng thái kế tiếp được xác định rõ, cuối cùng tạo ra "đầu ra" và chấm dứt ở trạng thái kết thúc cuối cùng. Sự chuyển đổi từ trạng thái này sang trạng thái tiếp theo không nhất thiết phải mang tính xác định; một số thuật toán, được gọi là thuật toán ngẫu nhiên, kết hợp đầu vào ngẫu nhiên

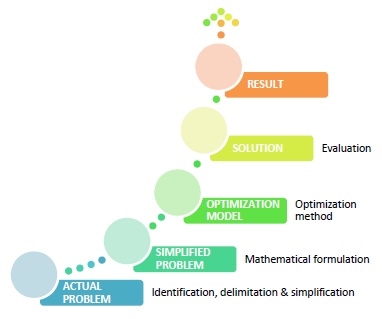


Lưu đồ thuật toán (thuật toán Euclid) để tính ước số chung lớn nhất (ưcln) của hai số a và b ở các vị trí có tên A và B. Thuật toán tiến hành bằng các phép trừ liên tiếp trong hai vòng lặp: NẾU phép thử B ≥ A cho kết quả "đúng" (chính xác hơn, số b ở vị trí B lớn hơn hoặc bằng số a ở vị trí A) thì thuật toán chỉ định B ← B - A (nghĩa là số b - a thay thế b cũ). Tương tự, IF A> B, THEN A ← A - B. Quá trình kết thúc khi (nội dung của) B bằng 0, tạo ra ưcln trong A

1. Tổng quan về tối ưu hóa đường đi ngắn nhất
2. Bài toán tối ưu hóa

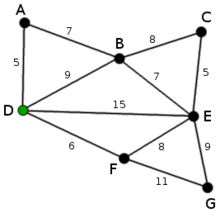
Bài toán tối ưu hóa NP (NPO-"NP optimization") là bài toán tối ưu hóa tổ hợp với các điều kiện bổ sung sau. Lưu ý rằng các đa thức dưới đây là các hàm kích thước của đầu vào của các hàm tương ứng, không phải là kích thước của một số tập hợp ẩn của các trường hợp đầu vào.

Trong khoa học máy tính, các bài toán tối ưu hóa thú vị thường có những đặc tính trên và cho nên đó là những bài toán NPO. Một bài toán ngoài ra còn được gọi là một bài toán tối ưu hóa-P (PO), nếu có tồn tại một thuật toán mà tìm các lời giải tối ưu trong thời gian đa thức. Thông thường, khi đối phó với lớp NPO, thứ được quan tâm trong các bài toán tối ưu hóa mà các phiên bản quyết định là NP-đầy đủ. Lưu ý rằng các quan hệ độ cứng luôn đối với một số phép suy giảm nào đó. Do sự kết hợp giữa các thuật toán xấp xỉ và các bài toán tối ưu hóa máy tính, các suy giảm duy trì xấp xỉ trong một số khía cạnh là dành cho đối tượng này được ưu tiên hơn so với mức giảm Turing và Karp thông thường.



1. Bài toán tìm đường đi ngắn nhất

Trong lý thuyết đồ thị, bài toán đường đi ngắn nhất nguồn đơn là bài toán tìm một đường đi giữa hai đỉnh sao cho tổng các trọng số của các cạnh tạo nên đường đi đó là nhỏ nhất. Định nghĩa một cách hình thức, cho trước một đồ thị có trọng số (nghĩa là một tập đỉnh V, một tập cạnh E, và một hàm trong số có giá trị thực f : E → R), cho trước một đỉnh v thuộc V, tìm một đường đi P từ v tới mỗi đỉnh v' thuộc V sao cho là nhỏ nhất trong tất cả các đường nối từ v tới v' . Bài toán đường đi ngắn nhất giữa mọi cặp đỉnh là một bài toán tương tự, trong đó ta phải tìm các đường đi ngắn nhất cho mọi cặp đỉnh v và v'.



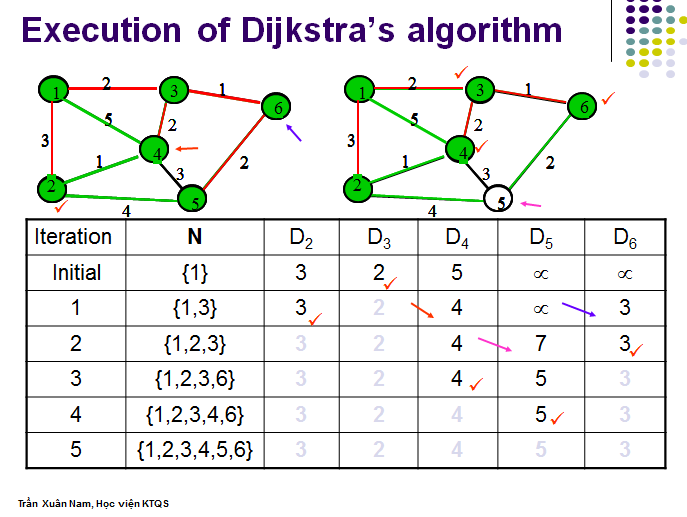
Các thuật toán quan trọng nhất giải quyết bài toán này là:

* Thuật toán Dijkstra — giải bài toán nguồn đơn nếu tất cả các trọng số đều không âm. Thuật toán này có thể tính toán tất cả các đường đi ngắn nhất từ một đỉnh xuất phát cho trước s tới mọi đỉnh khác mà không làm tăng thời gian chạy.
* Thuật toán Bellman-Ford — giải bài toán nguồn đơn trong trường hợp trọng số có thể có giá trị âm.
* Giải thuật tìm kiếm A\* - giải bài toán nguồn đơn sử dụng heuristics để tăng tốc độ tìm kiếm
* Thuật toán Floyd-Warshall — giải bài toán đường đi ngắn nhất cho mọi cặp đỉnh.
* Thuật toán Johnson — giải bài toán đường đi ngắn nhất cho mọi cặp đỉnh, có thể nhanh hơn thuật toán Floyd-Warshall trên các đồ thị thưa.

Trong tư duy của ngành mạng hay viễn thông, bài toán đường đi ngắn nhất đôi khi được gọi là bài toán đường đi có độ trễ nhỏ nhất (min-delay path problem) và thường được gắn với một bài toán đường đi rộng nhất (widest path problem). ví dụ đường đi rộng nhất trong các đường đi ngắn nhất (độ trễ nhỏ nhất) hay đường đi ngắn nhất trong các đường đi rộng nhất.

1. Thuật toán Dijkstra

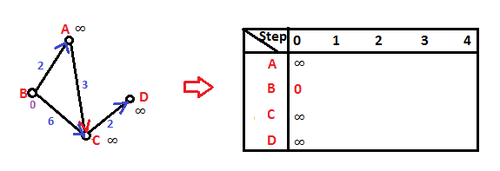
Thuật toán Dijkstra, mang tên của nhà khoa học máy tính người Hà Lan Edsger Dijkstra vào năm 1956 và ấn bản năm 1959, là một thuật toán giải quyết bài toán đường đi ngắn nhất nguồn đơn trong một đồ thị có hướng không có cạnh mang trọng số không âm. Thuật toán thường được sử dụng trong định tuyến với một chương trình con trong các thuật toán đồ thị hay trong công nghệ Hệ thống định vị toàn cầu (GPS).

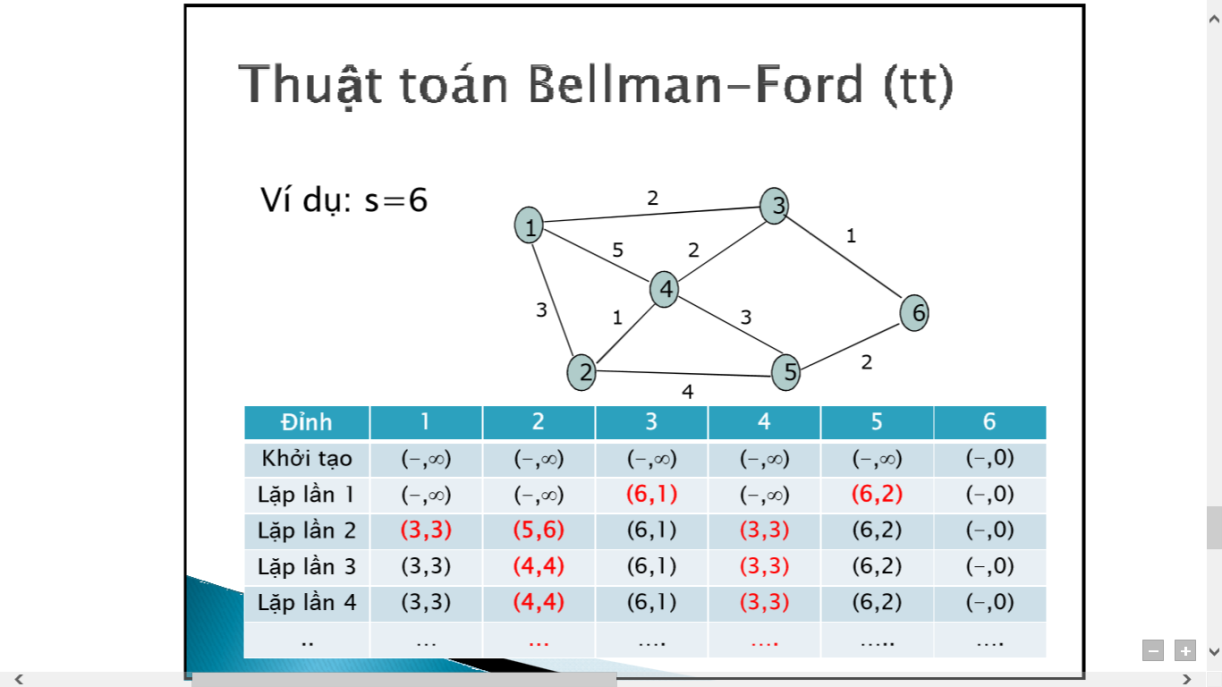


1. Thuật toán Bellman-Ford

Thuật toán Bellman-Ford là một thuật toán tính các đường đi ngắn nhất nguồn đơn trong một đồ thị có hướng có trọng số (trong đó một số cung có thể có trọng số âm). Thuật toán Dijkstra giải cùng bài toán này tuy nhiên Dijkstra có thời gian chạy nhanh hơn, đơn giản là đòi hỏi trọng số của các cung phải có giá trị không âm.

Thuật toán Bellman-Ford là một thuật toán tính các đường đi ngắn nhất nguồn đơn trong một đồ thị có hướng có trọng số (trong đó một số cung có thể có trọng số âm). Thuật toán Dijkstra giải cùng bài toán này tuy nhiên Dijkstra có thời gian chạy nhanh hơn, đơn giản là đòi hỏi trọng số của các cung phải có giá trị không âm.



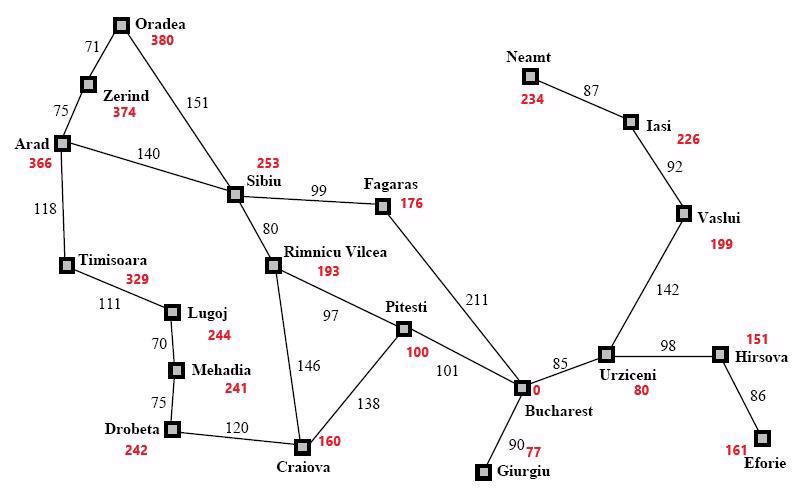


1. Thuật toán giải thuật tìm kiếm A\*

Trong khoa học máy tính, A\* (đọc là A sao) là thuật toán tìm kiếm trong đồ thị. Thuật toán này tìm một đường đi từ một nút khởi đầu tới một nút đích cho trước (hoặc tới một nút thỏa mãn một điều kiện đích). Thuật toán này sử dụng một "đánh giá heuristic" để xếp loại từng nút theo ước lượng về tuyến đường tốt nhất đi qua nút đó.

Thuật toán A\* được mô tả lần đầu vào năm 1968 bởi Peter Hart, Nils Nilsson, và Bertram Raphael. Trong bài báo của họ, thuật toán được gọi là thuật toán A; khi sử dụng thuật toán này với một đánh giá heuristic thích hợp sẽ thu được hoạt động tối ưu, do đó mà có tên A\*.

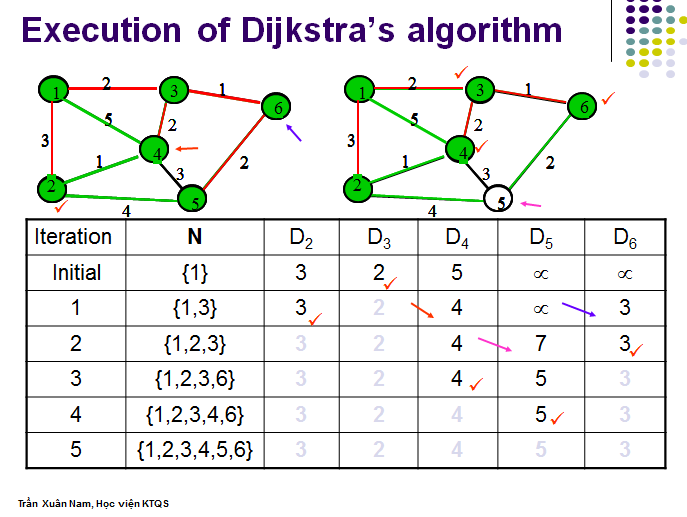
Năm 1964, Nils Nilsson phát minh ra một phương pháp tiếp cận dựa trên khám phá để tăng tốc độ của thuật toán Dijkstra. Thuật toán này được gọi là A1. Năm 1967 Bertram Raphael đã cải thiện đáng kể thuật toán này, nhưng không thể hiển thị tối ưu. Ông gọi thuật toán này là A2. Sau đó, trong năm 1968 Peter E. Hart đã giới thiệu một đối số chứng minh A2 là tối ưu khi sử dụng thuật toán này với một đánh giá heuristic thích hợp sẽ thu được hoạt động tối ưu. Do đó ông đặt tên cho thuật toán mới là A \*(A sao, A-star).



1. Thuật toán Floyd-Warshall

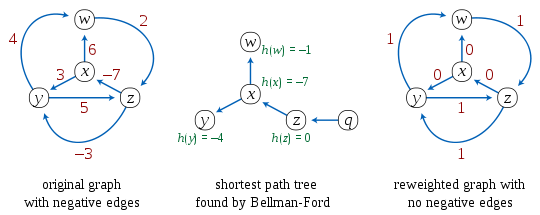
Thuật toán Floyd-Warshall còn được gọi là thuật toán Floyd được Robert Floyd tìm ra năm 1962.thuật toán Floyd là một thuật toán giải quyết bài toán đường đi ngắn nhất trong một đồ thị có hướng dựa trên khái niệm các Đỉnh Trung Gian.

Thuật toán Dijkstra bình thường có 2 vòng lặp lồng nhau sẽ có Độ phức tạp thuật toán là O(n2). Thuật toán Floyd-Warshall bình thường có 3 vòng lặp lồng nhau sẽ có Độ phức tạp thuật toán là O(n3).



1. Thuật toán Johnson

Thuật toán Johnson được Donald B. Johnson tìm ra năm 1977. Thuật toán Johnson là một thuật toán giải quyết bài toán đường đi ngắn nhất giữa mọi cặp đỉnh trong đồ thị có hướng, có trọng số và không có chu trình âm. Nó hoạt động bằng cách sử dụng thuật toán Bellman – Ford để tính toán một phép biến đổi của đồ thị đầu vào loại bỏ tất cả các trọng số âm, cho phép thuật toán Dijkstra được sử dụng trên đồ thị đã biến đổi.

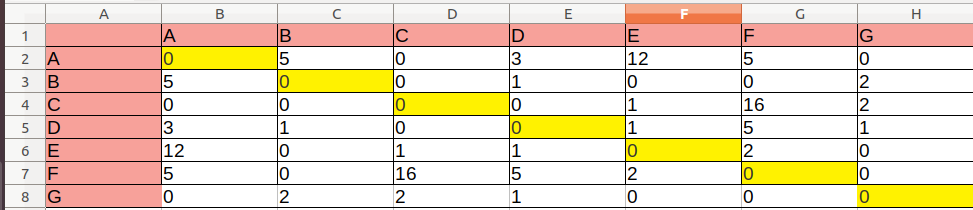


1. Phương pháp sử dụng thuật toán tối ưu hóa đường đi của Google Map

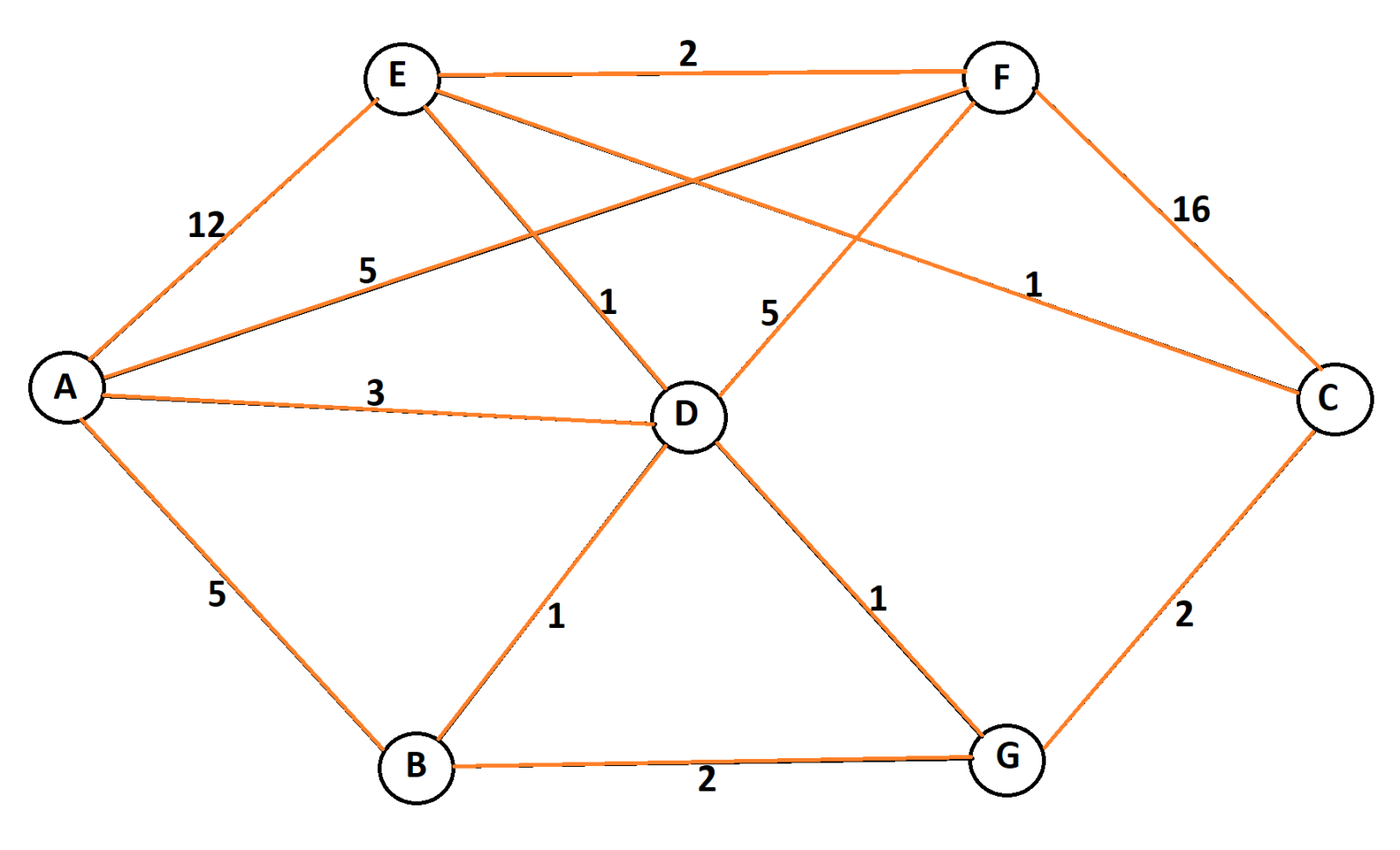
Vấn đề tối ưu quãng đường di chuyển là một trong những vấn đề quan trọng của Logistic, những công cụ như Badger Map ra đời chính là để giải quyết vấn đề này và đã rất thành công.

Hãy tưởng tượng bạn là tài xế lái xe du lịch, mỗi chuyến đi bạn phải xuất phát từ nhà xe và đi đến đón N vị khách tại nhà riêng của họ, bạn sẽ chọn đón ai trước? Hay công ty của bạn cung cấp nước đóng chai mỗi ngày phải giao hàng ở 100 cửa tiệm khác nhau trên địa bàn thành phố, bạn sẽ giao chúng như thế nào để tiết kiệm quãng đường cũng như thời gian di chuyển nhất :

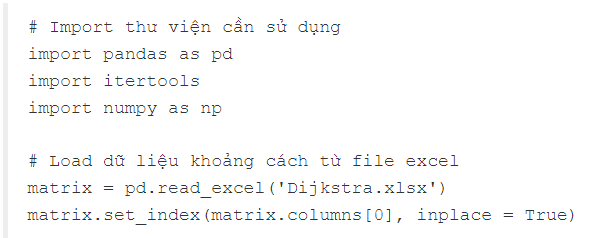
* Bước 1:
* Đầu tiên chúng ta có một ma trận khoảng cách đầu vào có dạng tương tự như trong hình, ở đây tôi giả sử có dữ liệu các địa điểm là A, B, C, D, E, F, G, với những ô có giá trị 0 là địa điểm trùng nhau hoặc là không có đường đi giữa hai điểm đó. Ví dụ: ô ở hàng A cột C có giá trị bằng 0 có nghĩa là không có đường đi từ A đến C. Trên thực tế nếu bạn dùng Google Map để lấy bản đồ thì hầu như là không có những ô có giá trị bằng 0, nhưng ở đây tôi vẫn để ô có giá trị 0 để tổng quát trường hợp nhất.



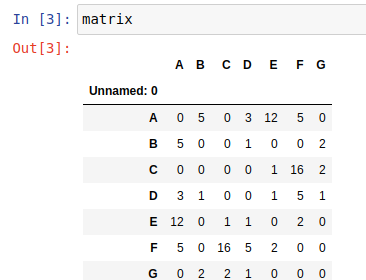
* Bạn có thể tưởng tượng ma trận này giống như một đồ hình (graph) thể hiện khoảng cách giữa các điểm:



* Bước 2:
* Dùng thư viện Pandas của Python để đọc dữ liệu khoảng cách trong file excel:



* Ta sẽ có được ma trận khoảng cách như sau:



* Nếu muốn lấy khoảng cách từ điểm A đến điểm B ta chỉ cần truy xuất đến vị trí A, B trên ma trận matrix.loc['A']['B']
* Bước 3:

Thuật toán tìm đường đi ngắn nhất qua các điểm:

* Ý tưởng của thuật toán này rất đơn giản: ta có một danh sách gồm điểm xuất phát (kho), những điểm cần đi (cửa hàng) và điểm đến cuối cùng (kho), ta sẽ tìm hết tất cả những tuyến đường khả dĩ từ điểm xuất phát đi qua tất cả những điểm cần đến và về điểm đến cuối cùng bằng cách sử dụng chỉnh hợp.
* Với mỗi tuyến đường khả dĩ ta sẽ tính tổng quảng đường của nó.
* Cuối cùng lấy ra tuyến đường có distance nhỏ nhất.

Giải thuật trên ngoài việc giải quyết bài toán tìm đường đi ngắn nhất qua tất cả các điểm ra, nó còn có thể mở rộng để giải những bài toán như tìm đường đi ngắn nhất đi qua số lượng điểm cụ thể (thay đổi các tham số: destination, number\_of\_node để thử).

**CHƯƠNG 3:**

**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

1. Kết luận

Trong bối cảnh ứng dụng công nghệ thông tin theo xu thế thời đại công nghiệp 4.0 đang diễn ra và hướng đến một cuộc cách mạng 5.0 sắp tới. Việc xây dựng các giải pháp tối ưu để giảm chi phí cho bài toán đi lại càng trở nên cần thiết nhằm góp phần phát triển kinh tế, xã hội của mỗi quốc gia trên thế giới. Trong bài báo này, các tác giả đã dựa trên tiếp cận của giải thuật di truyền để xây dựng một giải thuật nhằm tìm kiếm đường đi tối ưu trên cơ sở dữ liệu đa điểm của Google Maps. Giải thuật này được dùng để phát triển một ứng dụng tìm kiếm đường đi tối ưu cho bài toán trên bản đồ Google Maps.

1. Hướng phát triển

Kết quả so sánh cho thấy giải pháp đề xuất tối ưu hơn các ứng dụng và thuật toán được so sánh về chi phí khoảng cách và thời gian. Với kết quả này, các tác giả hy vọng giải pháp sẽ được sử dụng vào các lĩnh vực khác nhau trong giao thông, trong du lịch, trong vận hay các công việc khác. Trong nghiên cứu này, khi sử dụng ứng dụng được đề xuất để tìm đường đi với số điểm khá lớn, bước xử lý mảng và tính khoảng cách các điểm của thuật toán còn chậm. Do mất thời gian hàng đợi khi đọc và chuyển đổi dữ liệu thực phụ thuộc vào dữ liệu của Google Maps. Đây cũng là công việc mà nhóm tác giả sẽ nghiên cứu để cải tiến và đề xuất giải thuật tối ưu hơn trong tương lai. Hơn nữa, trên nền ứng dụng này, xây dựng các ứng dụng đi kèm để giải quyết các công việc khác nhằm áp dụng vào đời sống xã hội cũng là nhiệm vụ trong tương lai mà nhóm tác giả hướng đến.

**TÀI LIỆU THAM KHẢO**

[1]. Trang thông tin Google Map: <https://www.google.com/maps/>

[2]. “Google Maps now used by over 1 billion people every month”. PPC Land (bằng tiếng Anh). 15 tháng 2 năm 2020. Truy cập ngày 15 tháng 2 năm 2020: <https://newsletter.ppc.land/p/>

[3]. Google Maps đã được ra mắt vào tháng 2 năm 2005 <https://developers.google.com/maps/>

[4]. Vehicle Routing Solution Reduces Transportation Costs for Organic Farmers Serving the Domestic Market in Turkey”, A. Yonca Demir, 2014 <http://www.ijastnet.com/journals/Vol_4_No_6_November_2014/5.pdf>

[5]. “Solving A Real Vehicle Routing Problem In The Furniture And Electronics Industries”, Jean-Philippe Gagliardi, 2013.

<https://www.cirrelt.ca/DocumentsTravail/CIRRELT-FSA-2013-35.pdf>

[6]. Z. Liang and W. Wenjia, "A New Path Search Algorithm for Providing Paths among Multiple Origins and One Single Destination"International Journal of Computer Science and Application (IJCSA), vol. 3, pp. 5, 2014.

[7]. S. Yagvalkya, C. S. Subhash, and B. Manisha, "Comparison of Dijkstra’s Shortest Path Algorithm with Genetic Algorithm for Static and Dynamic Routing Network," International Journal of Electronics and Computer Science Engineering, vol. 1, pp. 10, 2016.