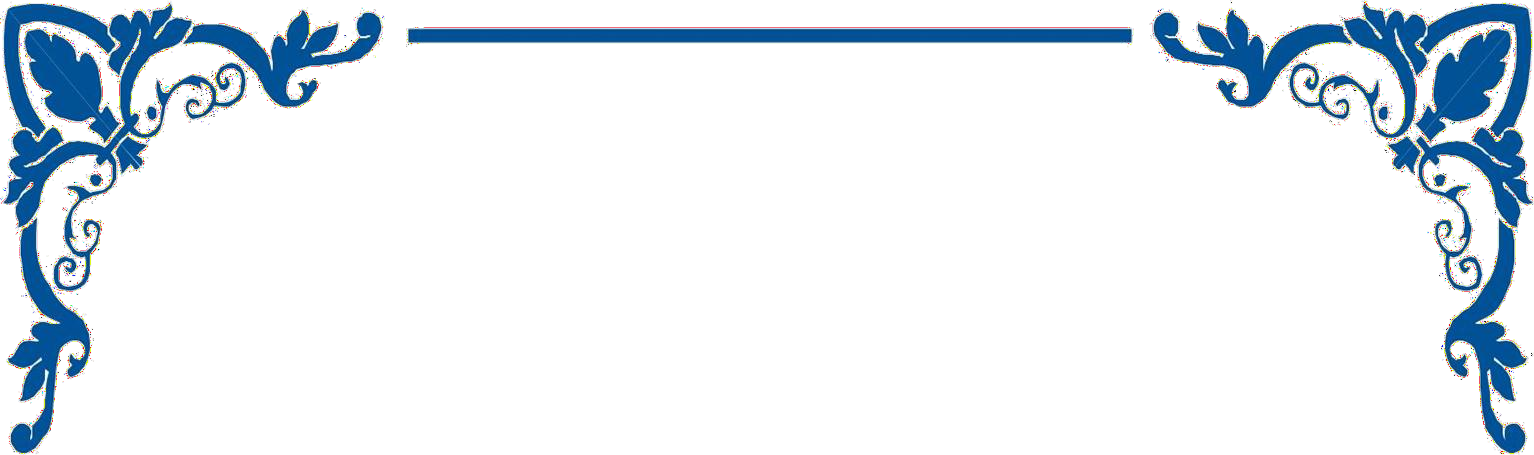
**TRƯỜNG ĐẠI HỌC SƯ PHẠM TP HCM**



**KHOA CÔNG NGHỆ THÔNG TIN**

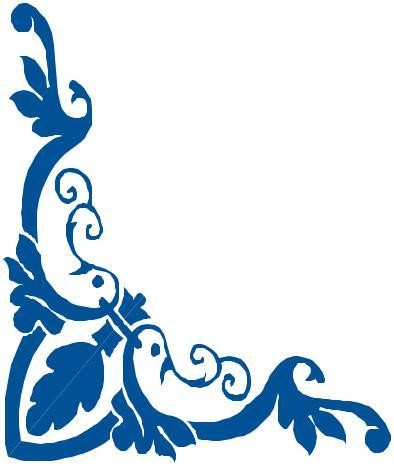
**🙡🕮🙣**



**TIỂU LUẬN**

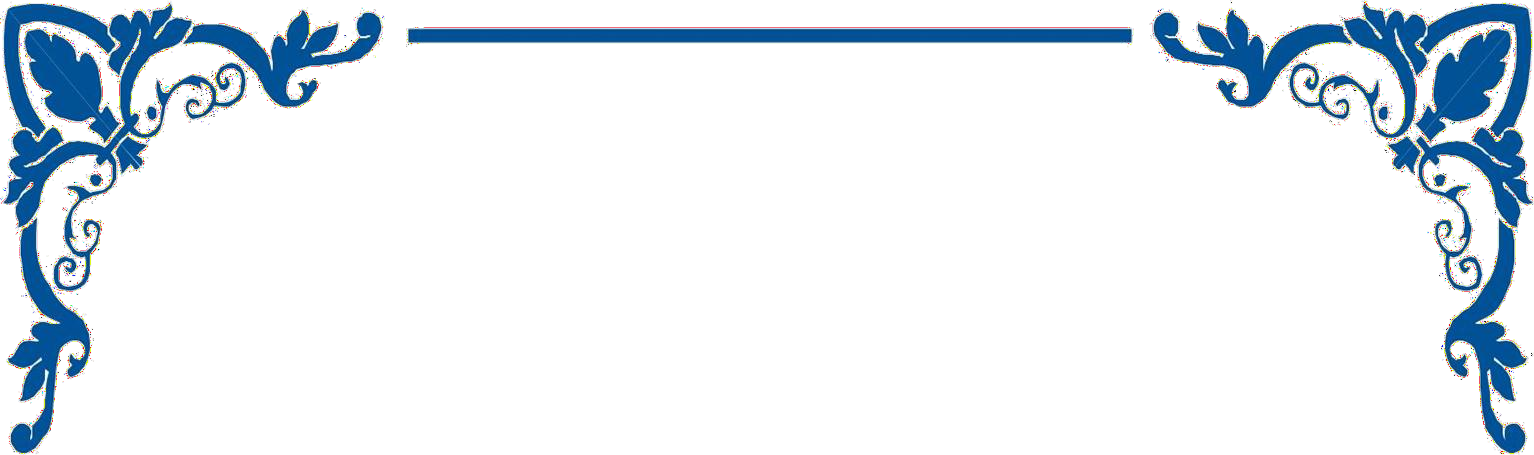
**Đề tài: Minh họa trực quan bằng giao diện đồ họa các thuật toán lý thuyết đồ thị Kruskal và Floyd**

**HỌC PHẦN:** COMP170102 – Lý thuyết đồ thị và ứng dụng



**-------- Thành phố Hồ Chí Minh, ngày 30 tháng 11 năm 2021--------**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM TP HCM**



**KHOA CÔNG NGHỆ THÔNG TIN**

**🙡🕮🙣**



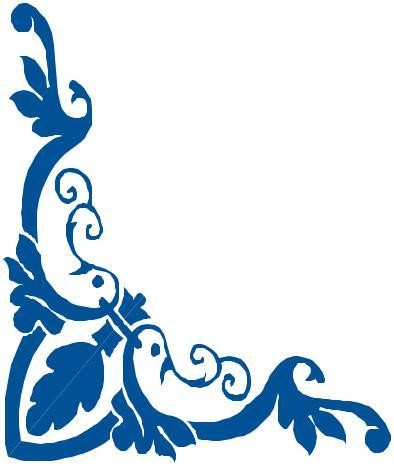
**TIỂU LUẬN**

**Đề tài: Minh họa trực quan bằng giao diện đồ họa các thuật toán lý thuyết đồ thị Kruskal và Floyd**

**Nhóm:** I\_WinX\_5

**Mã lớp học phần:** COMP170102

**Giảng viên hướng dẫn:** Thầy Nguyễn Viết Hưng  
 & Thầy Lương Trần Ngọc Khiết



**-------- Thành phố Hồ Chí Minh, ngày 30 tháng 11 năm 2021--------**

**MỤC LỤC**

LỜI CẢM ƠN 4

BẢNG PHÂN CÔNG 5

A.THUẬT TOÁN KRUSKAL 6

1.Khái niệm 6

2.Tư tưởng thuật toán 6

3.Mô tả thuật toán 6

4.Thực thi thuật toán Kruskal 7

B.THUẬT TOÁN FLOYD 7

1.Khái niệm 8

2.Mô tả thuật toán 8

3.Ứng dụng của thuật toán Floyd 9

LINK THÔNG TIN CHƯƠNG TRÌNH 12

TÀI LIỆU THAM KHẢO 13

***LỜI CẢM ƠN!***

*Lời đầu tiên, chúng em xin gửi lời cảm ơn chân thành nhất đến Thầy Nguyễn Viết Hưng và Thầy Lương Trần Ngọc Khiết. Trong quá trình học tập và tìm hiểu bộ môn Lý thuyết đồ thị và ứng dụng, chúng em đã nhận được sự quan tâm giúp đỡ, hướng dẫn tận tình, tâm huyết của thầy. Thầy đã giúp chúng em tích lũy thêm nhiều kiến thức để có cái nhìn sâu sắc và hoàn thiện hơn trong quá trình học tập. Từ những kiến thức mà thầy truyền tải, chúng em đã dần thành thục hơn về việc thực hành trong bộ môn này. Thông qua bài tiểu luận này, chúng em xin trình bày lại những gì mà chúng em đã tìm hiểu về vấn đề “*Minh họa trực quan bằng giao diện đồ họa các thuật toán lý thuyết đồ thị Kruskal và Floyd*”, chúng em xin gửi đến Thầy.*

*Có lẽ kiến thức là vô hạn mà sự tiếp nhận kiến thức của bản thân chúng em luôn tồn tại những hạn chế nhất định. Do đó, trong quá trình hoàn thành bài tiểu luận, chắc chắn không tránh khỏi những thiếu sót. Bản thân chúng em rất mong nhận được những đóng góp ý kiến đến từ thầy để bài tiểu luận của chúng em được hoàn thiện hơn.*

*Kính chúc thầy sức khỏe, hạnh phúc thành công con trên con đường sự nghiệp giảng dạy!*

*Chúng em xin chân thành cảm ơn Thầy!*

*Tp.HCM, ngày 30 tháng 11 năm 2021.*

**BẢNG PHÂN CÔNG:**

|  |  |  |
| --- | --- | --- |
| **MSSV** | **Họ tên** | **Nhiệm vụ** |
| 46.01.104.089 | Nguyễn Tuấn Kiệt | Tạo và chỉnh winform, viết thuật toán Floyd C++, làm Word, kiếm ma trận và đồ thị,làm ppt\* |
| 46.01.104.047 | Trần Ngọc Bảo Hân | Tạo và chỉnh winform, chuyển C++ sang C#, demo cách giải tay Floyd, giới thiệu Word, demo chương trình, thuyết trình |
| 46.01.104.085 | Trần Anh Khoa | Tạo và chỉnh winform\*, viết thuật toán Kruskal C++, kiếm ma trận và đồ thị,ppt |
| 46.01.104.135 | Huỳnh Ngọc Nhung | Trang trí winform, tìm hiểu 2 thuật toán và làm Word, kiếm ma trận và đồ thị,ppt |
| 46.01.104.199 | Nguyễn Thị Tú Trinh | Trang trí winform, demo cách giải tay Kruskal, tìm hiểu 2 thuật toán và làm Word |

**A/ Thuật toán Kruskal:**

1. **Khái niệm:**

- Thuật toán Kruskal (Kruskal Algorithm) được biết đến với tên là cây khung nhỏ nhất.

- Thuật toán Kruskal là một thuật toán trong lý thuyết đồ thị để tìm cây bao trùm nhỏ nhất của một đồ thị liên thông có trọng số. Nói cách khác, nó tìm một tập hợp các cạnh tạo thành một cây chứa tất cả các đỉnh của đồ thị và có tổng trọng số các cạnh là nhỏ nhất.

- Thuật toán này xuất bản lần đầu tiên năm 1956, bởi Joseph Kruskal.

- Một vài thuật toán khác cho bài toán này bao gồm thuật toán Prim, thuật toán xóa ngược và thuật toán Boruvka.

1. **Tư tưởng thuật toán:**

Thuật toán Kruskal dựa trên mô hình xây dựng cây khung nhỏ nhất bằng thuật toán hợp nhất.

- Thuật toán không xét các cạnh với thứ tự tùy ý.

- Thuật toán xết các cạnh theo thứ tự đã sắp xếp theo trọng số.

- Để xây dựng tập n-1 cạnh của cây khung nhỏ nhất - tạm gọi là tập K, Kruskal đề nghị cách kết nạp lần lượt các cạnh vào tập đó theo nguyên tắc như sau:

* Ưu tiên các cạnh có trọng số nhỏ hơn.
* Kết nạp cạnh khi nó không tạo chu trình với tập cạnh đã kết nạp trước đó.

Đó là một nguyên tắc chính xác và đúng đắn, đảm bảo tập K nếu thu đủ n-1 cạnh sẽ là cây khung nhỏ nhất.

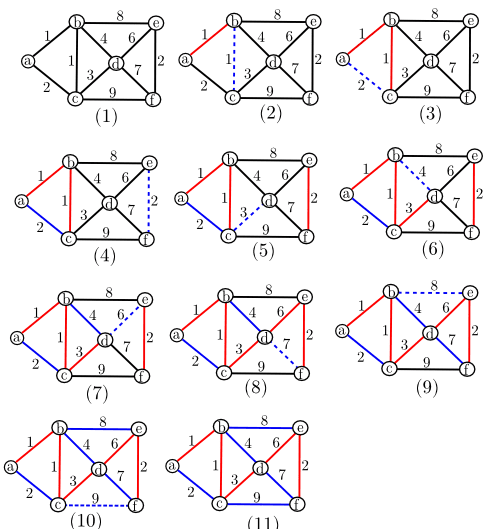
1. **Mô tả thuật toán:**

Giả sử ta cần tìm cây bao trùm nhỏ nhất của đồ thị G. Thuật toán bao gồm các bước sau:

- Khởi tạo rừng F (tập hợp các [cây](https://vi.wikipedia.org/wiki/C%C3%A2y_(l%C3%BD_thuy%E1%BA%BFt_%C4%91%E1%BB%93_th%E1%BB%8B))), trong đó mỗi đỉnh của G tạo thành một cây riêng biệt.

- Khởi tạo tập S chứa tất cả các cạnh của G. Bao giờ S còn [khác rỗng](https://vi.wikipedia.org/w/index.php?title=Kh%C3%A1c_r%E1%BB%97ng&action=edit&redlink=1) và F gồm hơn một cây, thì xóa cạnh nhỏ nhất trong S.

- Nếu cạnh đó nối hai cây khác nhau trong F, thì thêm nó vào F và hợp hai cây kề với nó làm một. Nếu không thì loại bỏ cạnh đó.

- Khi thuật toán kết thúc, rừng chỉ gồm đúng một cây và đó là một cây bao trùm nhỏ nhất của đồ thị G.

1. **Thực thi thuật toán Kruskal:**

**Bước 1:** Sắp xếp các cạnh của đồ thị theo thứ tự trọng số tăng dần.

**Bước 2:** Khởi tạo T:= Ø

**Bước 3:** Lần lượt lấy từng cạnh thuộc danh sách đã sắp xếp. Nếu T+{e} không chứa chu trình thì gán T:=T+{e}.

**Bước 4:** Nếu T đủ n-1 phần tử thì dừng, ngược lại làm tiếp bước 3.

**B/ Thuật toán Floyd:**

1. **Khái niệm:**

- Thuật toán Floyd (Floyd-Warshall) được biết đến 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 này được tìm ra vào năm 1962, bởi Robert Floyd.

1. **Mô tả thuật toán:**

- Nếu đỉnh k nằm trên đường đi ngắn nhất từ đỉnh i tới đỉnh j thì đoạn đường từ i tới k và từ k tới j phải là đường đi ngắn nhất từ i tới k và từ k tới j tương ứng. Do đó ta sử dụng ma trận A để lưu độ dài đường đi ngắn nhất giữa mọi cặp đỉnh.

- Ban đầu ta đặt A[i,j] = C[i,j], tức là ban đầu A chứa độ dài đường đi trực tiếp (không đi qua đỉnh nào cả).

- Sau đó thực hiện n lần lặp, sau lần lặp thứ k, ma trận A sẽ chứa độ dài đường đi ngắn nhất giữa mọi cặp đỉnh chỉ đi qua các đỉnh thuộc tập {1,2,…,k}. Như vậy, sau n lần lặp ta nhận được ma trận A chứa độ dài các đường đi ngắn nhất giữa  mọi cặp đỉnh của đồ thị.

- Ký hiệu Ak là ma trận A sau lần lặp thứ k, tức là Ak[i,j] là độ dài đường đi ngắn nhất từ i đến j chỉ đi qua các đỉnh thuộc {1,2,…,k}. Ak[i,j] được tính theo công thức như sau:

Ak[i,j] = min { Ak – 1[i,j], Ak – 1[i,k] + Ak – 1[k,j] }

-Trong quá trình lặp ta phải lưu lại vết đường đi, tức là đường đi ngắn nhất đi qua các đỉnh nào. Khi đó ta sử dụng mảng phụ P[nxn], trong đó P[i,j] lưu đỉnh k nếu đường đi ngắn nhất từ i đến j đi qua đỉnh k. Ban đầu P[i,j] = 0 với mọi i,j vì lúc đó đường đi ngắn nhất là đường đi trực tiếp không đi qua đỉnh nào cả.

Text

Description automatically generated

1. **Ứng dụng của thuật toán Floyd:**

Được sử dụng để tìm đường đi ngắn nhất trong đồ thị có hướng

Để tìm ra ma trận Transitive Closure của đồ thị có hướng.

Để tìm ra nghịch đảo của ma trận thực.

Để kiểm tra xem một biểu đồ vô hướng có phải là đồ thị hai phía hay không.

* **So sánh giữa 2 thuật toán Dijkstra và Floyd-Warshall:**

Ở đây ta sẽ sơ lược đơn giản về sự khác biệt giữa 2 thuật toán Dijkstra và thuật toán Floyd.

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

|  |  |  |
| --- | --- | --- |
|  | **Dijsktra** | **Floyd-Warshall** |
| **Ưu điểm** | Chi phí thấp hơn Thuật toán Floyd-Warshall | Không cần chạy lại thuật toán (có nghĩa là có tính kế thừa từ đường đi lẫn nhau)  Có thể chạy được với trọng số âm |
| **Nhược điểm** | Không chạy được với trọng số âm | Chi phí cao hơn cho mỗi cặp đỉnh |

* **Dựa trên những yêu cầu từ đề tài cùng những thông tin đi kèm được thầy cung cấp, thì sản phẩm của nhóm em thu được sau những tìm hiểu thì đây là kết thu được:**
* Hàm giải quyết thuật toán Kruskal:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

* Hàm giải quyết thuật toán Floyd:

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

**LINK THÔNG TIN CHƯƠNG TRÌNH:**

* Giải tay Kruskal + Floyd + Video demo:

<https://bitly.com.vn/2kqwbr>

* Mã nguồn chương trình:

<https://bitly.com.vn/2kqwbr>

**TÀI LIỆU THAM KHẢO:**

[1] <https://bitly.com.vn/ya3243>

[2] <http://upcoder.hcmue.edu.vn/index.php/problems/mysubmit/pJHmd9221>

[3] <http://upcoder.hcmue.edu.vn/index.php/problems/mysubmit/oz3GD3660>