

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN MÔN HỌC

CƠ SỞ TRÍ TUỆ NHÂN TẠO

|ĐỀ TÀI|

THUẬT TOÁN TÌM KIẾM TRÊN ĐỒ THỊ

HỌ TÊN : LA NHẬT HY

MSSV : 18120402

HỌC KỲ 1 / 2020 - 2021

Thành Phố Hồ Chí Minh – Năm 2020



A. ĐÁNH GIÁ ĐỒ ÁN.....	3
B. NỘI DUNG ĐỒ ÁN.....	3
I. Lý thuyết cơ bản.....	3
1. Breadth first search	3
2. Depth first search.....	4
4. A star search	5
5. Khác biệt giữa UCS và A*	6
II. Cài đặt thuật toán	6
1. Testcase	6
2. Breadth First Search.....	8
3. Depth First Search.....	9
4. Uniform Cost Search	12
5. A Star search	14
Tài liệu tham khảo	16

A. ĐÁNH GIÁ ĐỒ ÁN

STT	Yêu cầu	Mức độ hoàn thành (%)
1	Lý thuyết cơ bản về các thuật toán	100
2	Cài đặt các thuật toán	100
3	Minh họa các testcase	100
4	Nhận xét các thuật toán	100

B. NỘI DUNG ĐỒ ÁN

I. Lý thuyết cơ bản

1. Breadth first search

a) Ý tưởng

- Xuất phát từ 1 đỉnh và đi tới các đỉnh kề nó, tiếp tục cho đến khi không còn đỉnh nào có thể đi.
- Trong quá trình đi đến đỉnh kề, tiến hành lưu lại đỉnh cha của đỉnh kề để khi đi ngược lại từ đỉnh Kết thúc đến đỉnh Xuất phát, ta có được đường đi ngắn nhất.
- Sở dĩ thuật toán này tìm được đường đi ngắn nhất là nhờ vào cơ chế tô màu và lưu đỉnh cha. Quá trình tô màu khiến 1 đỉnh không thể xét 2 lần trở lên và có thể xem được đường đi từ đỉnh Kết Thúc đến đỉnh Xuất phát dựa vào việc lưu đỉnh cha.

b) Độ phức tạp

- Không gian: Nếu V là tập hợp đỉnh của đồ thị và $|V|$ là số đỉnh thì không gian cần dung của thuật toán là $O(|V|)$.
- Thời gian: Nếu V , và E là tập hợp các đỉnh và cung của đồ thị, thì thời gian thực thi của thuật toán là $O(|E| + |V|)$ vì trong trường hợp xấu nhất, mỗi đỉnh và cung của đồ thị được thăm đúng một lần. Ghi chú: $O(|E| + |V|)$ nằm trong khoảng từ $O(|V|)$ đến $O(|V|^2)$, tùy theo số cung của đồ thị.

c) Tính chất

- Trong đồ thị không có trọng số, thuật toán tìm kiếm theo chiều rộng luôn tìm ra đường đi ngắn nhất có thể.

2. Depth first search

a) Ý tưởng

- Xuất phát từ 1 đỉnh và đi mãi cho đến khi không thể đi tiếp, sau đó đi về lại đỉnh đầu. Trong quá trình quay lại:
 - + Nếu gặp đường đi khác thì đi cho đến khi không đi tiếp được nữa
 - + Nếu không tìm ra đường đi nào khác thì ngừng việc tìm kiếm.

b) Độ phức tạp

- DFS được gọi đúng 1 lần ứng với mỗi đỉnh.
- Mỗi cạnh được xem xét đúng 2 lần, mỗi lần từ một đỉnh kề với nó.
- Với n_s đỉnh và m_s cạnh thuộc thành phần liên thông chứa s , một phép DFS bắt đầu tại s sẽ chạy với thời gian $O(n_s + m_s)$ nếu:
 - + Đồ thị được biểu diễn bằng cấu trúc dữ liệu dạng danh sách kề.
 - + Đặt nhãn cho một đỉnh là "đã thăm" và kiểm tra xem một đỉnh "đã thăm" chưa tốn chi phí $O(\text{degree})$.
 - + Bằng cách đặt nhãn cho các đỉnh là "đã thăm", ta có thể xem xét một cách hệ thống các cạnh kề với đỉnh hiện hành nên ta sẽ không xem xét một cạnh quá 1 lần.

c) Tính chất

- Trong quá trình đi đến đỉnh khác, thuật toán sẽ lưu lại đỉnh cha vừa đi qua để khi đi ngược lại từ đỉnh Kết thúc đến đỉnh Xuất phát, ta có thể xem được đường đi từ đỉnh Kết thúc đến đỉnh Bắt Đầu

3. Uniform cost search

a) Ý tưởng

- Việc tìm kiếm bắt đầu tại nút gốc. Việc tìm kiếm tiếp tục bằng cách duyệt các nút tiếp theo với trọng lượng hay chi phí thấp nhất tính từ nút gốc. Các nút được duyệt tiếp tục cho đến khi đến được nút đích cần đến.

b) Độ phức tạp

- Thời gian:
 - + Gọi C^* là Chi phí của giải pháp tối ưu, và ϵ là mỗi bước để tiến gần hơn đến nút mục tiêu. Khi đó số bước là $C^* / \epsilon + 1$. Ở đây chúng tôi đã lấy $+1$, khi chúng tôi bắt đầu từ trạng thái 0 và kết thúc đến C^* / ϵ .
 - + Do đó, độ phức tạp thời gian trong trường hợp xấu nhất của Tìm kiếm theo chi phí thống nhất là $O(b^{1 + \lceil C^* / \epsilon \rceil})$.
- Không gian: Logic tương tự đối với độ phức tạp không gian.

c) Tính chất

- Tìm kiếm chi phí thống nhất là tối ưu vì ở mọi trạng thái, con đường có chi phí nhỏ nhất được chọn.

4. A star search

a) Ý tưởng

- Xét bài toán tìm đường - bài toán mà A* thường được dùng để giải. A* xây dựng tăng dần tất cả các tuyến đường từ điểm xuất phát cho tới khi nó tìm thấy một đường đi chạm tới đích. Tuy nhiên, cũng như tất cả các thuật toán tìm kiếm có thông tin, nó chỉ xây dựng các tuyến đường "có vẻ" dẫn về phía đích.
- Để biết những tuyến đường nào có khả năng sẽ dẫn tới đích, A* sử dụng một "đánh giá heuristic" về khoảng cách từ điểm bất kỳ cho trước tới đích. Trong trường hợp tìm đường đi, đánh giá này có thể là khoảng cách đường chim bay - một đánh giá xấp xỉ thường dùng cho khoảng cách của đường giao thông.

b) Độ phức tạp

- Độ phức tạp thời gian của A* phụ thuộc vào đánh giá heuristic. Trong trường hợp xấu nhất, số nút được mở rộng theo hàm mũ của độ dài lời giải, nhưng nó sẽ là hàm đa thức khi hàm heuristic h thỏa mãn điều kiện sau: $|h(x) - h^*(x)| \leq O(\log(h^*(x)))$
- Trong đó h^* là heuristic tối ưu, nghĩa là hàm cho kết quả là chi phí chính xác để đi từ x tới đích. Nói cách khác, sai số của h không nên tăng nhanh hơn logarit của h^*
- Vấn đề sử dụng bộ nhớ của A* còn rắc rối hơn độ phức tạp thời gian. Trong trường hợp xấu nhất, A* phải ghi nhớ số lượng nút tăng theo hàm mũ.

c) Tính chất

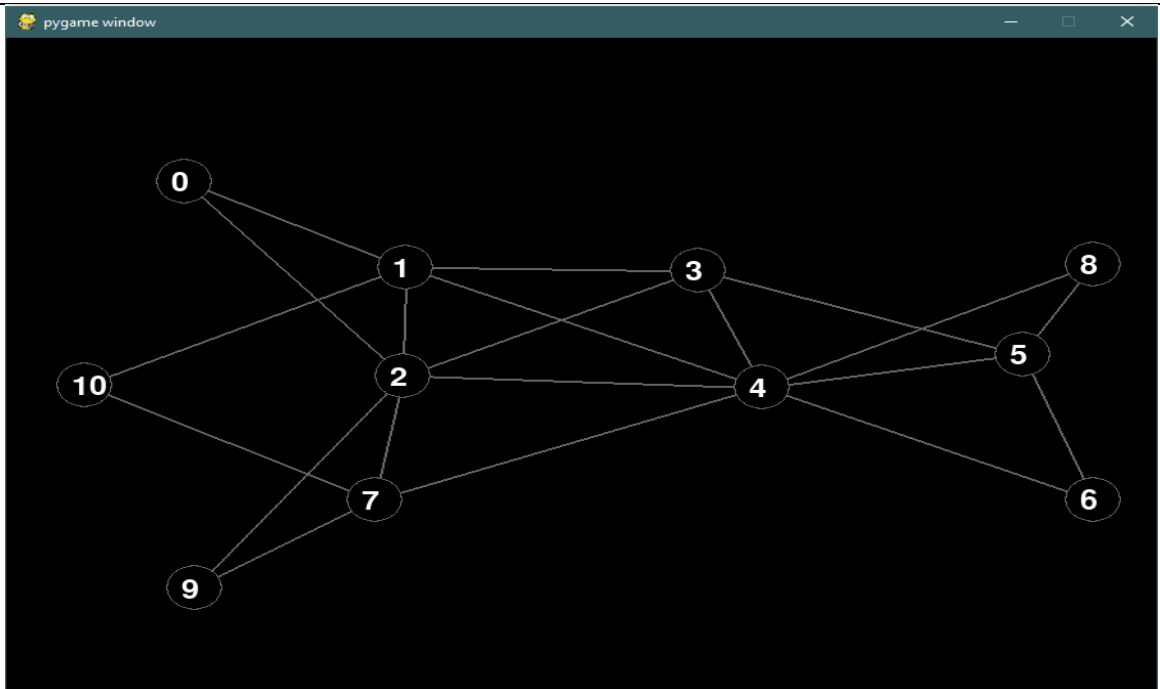
- Cũng như tìm kiếm theo chiều rộng (*breadth-first search*), A* là thuật toán *đầy đủ* (*complete*) theo nghĩa rằng nó sẽ luôn luôn tìm thấy một lời giải nếu bài toán có lời giải

5. Khác biệt giữa UCS và A*

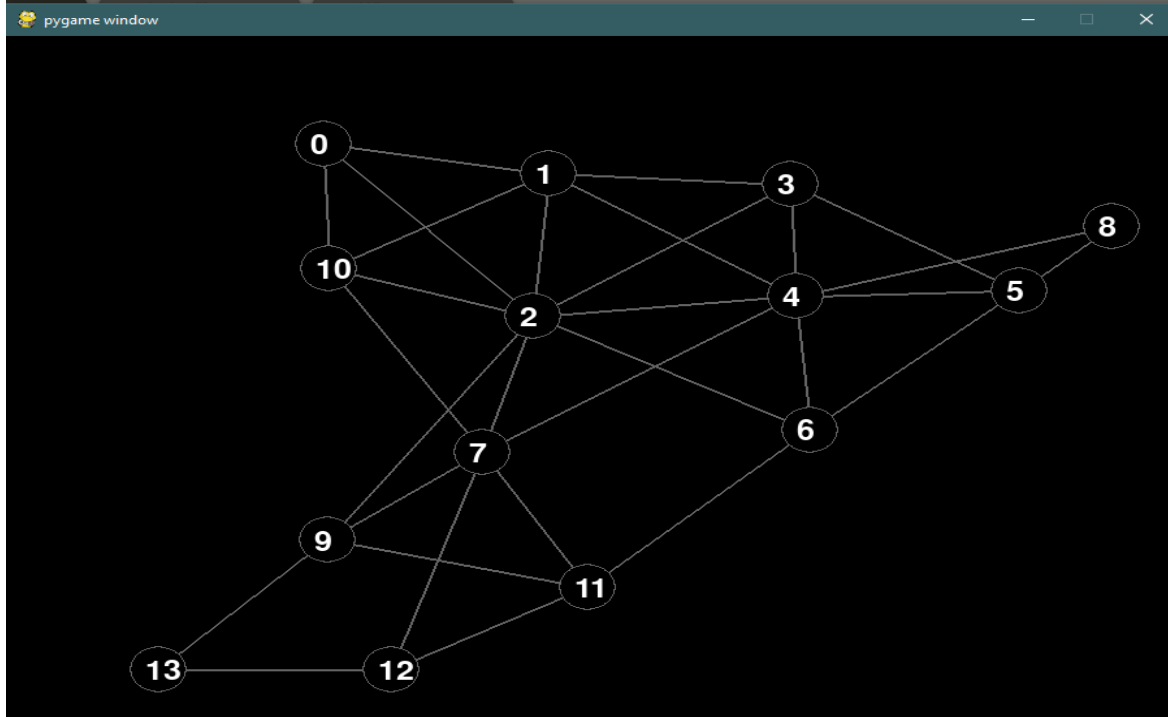
- UCS:
 - Không quan tâm đến số bước liên quan đến việc tìm kiếm và chỉ quan tâm đến chi phí đường dẫn. Do đó thuật toán này có thể bị mắc kẹt trong một vòng lặp vô hạn.
 - Khi đồ thị có chi phí ở mỗi bước là như nhau thì thuật toán trở thành phương pháp tìm kiếm theo chiều rộng.
 - Tối ưu
- A star:
 - Sử dụng một hàm đánh giá dựa trên kinh nghiệm để tìm ra đường dẫn.
 - Quá trình tìm kiếm có thể đi xa khỏi lời giải.
 - Một hàm đánh giá tốt có thể giảm thời gian và không gian nhớ một cách đáng kể.
 - Tối ưu khi hàm heuristic có tính chất thu nạp

II. Cài đặt thuật toán

1. Testcase

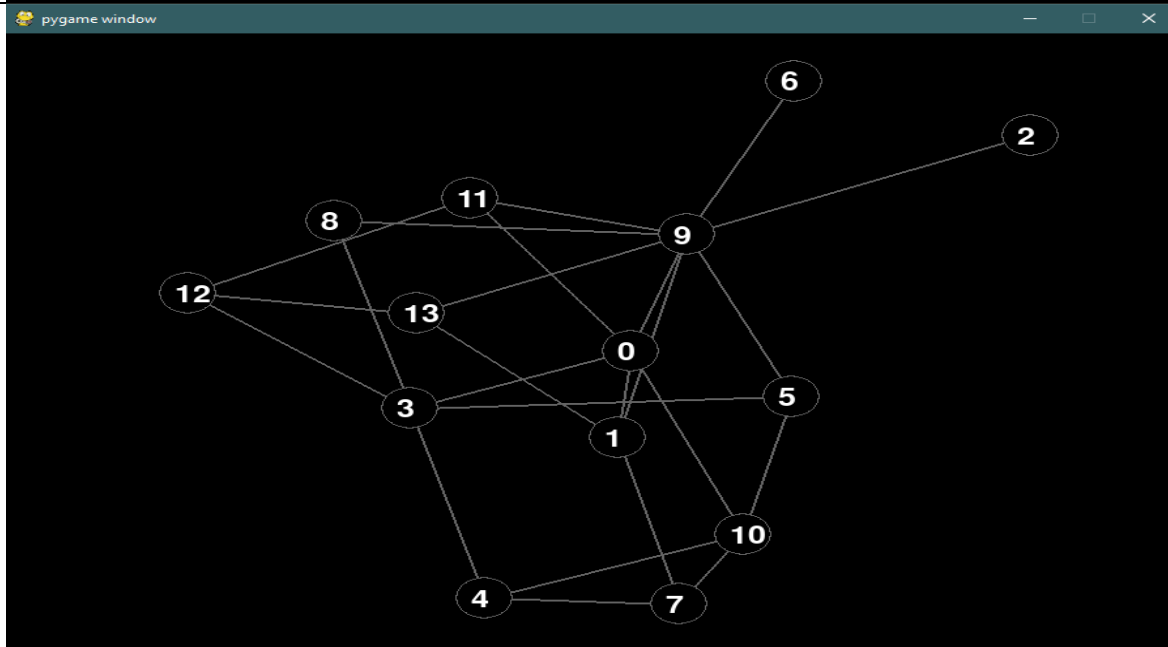
STT	Testcase	Ghi chú
1		Đồ thị 10 liên thông 10 đỉnh

2



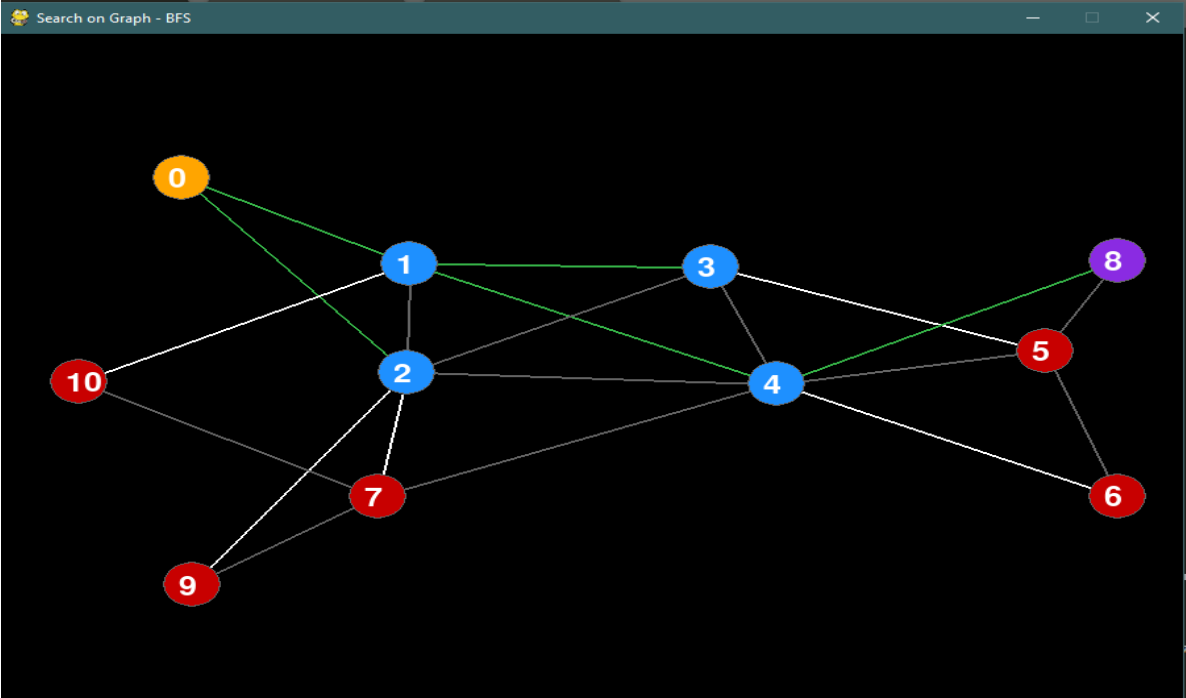
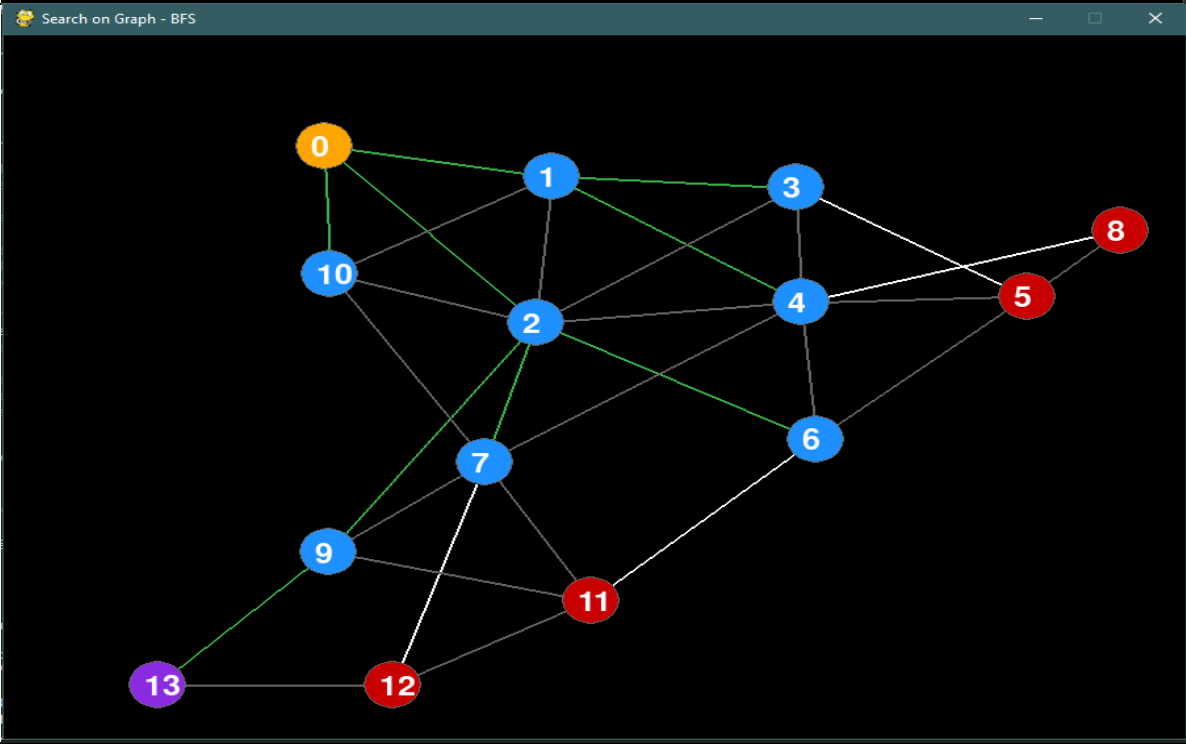
**Đồ thị
liên
thông
13
đỉnh**

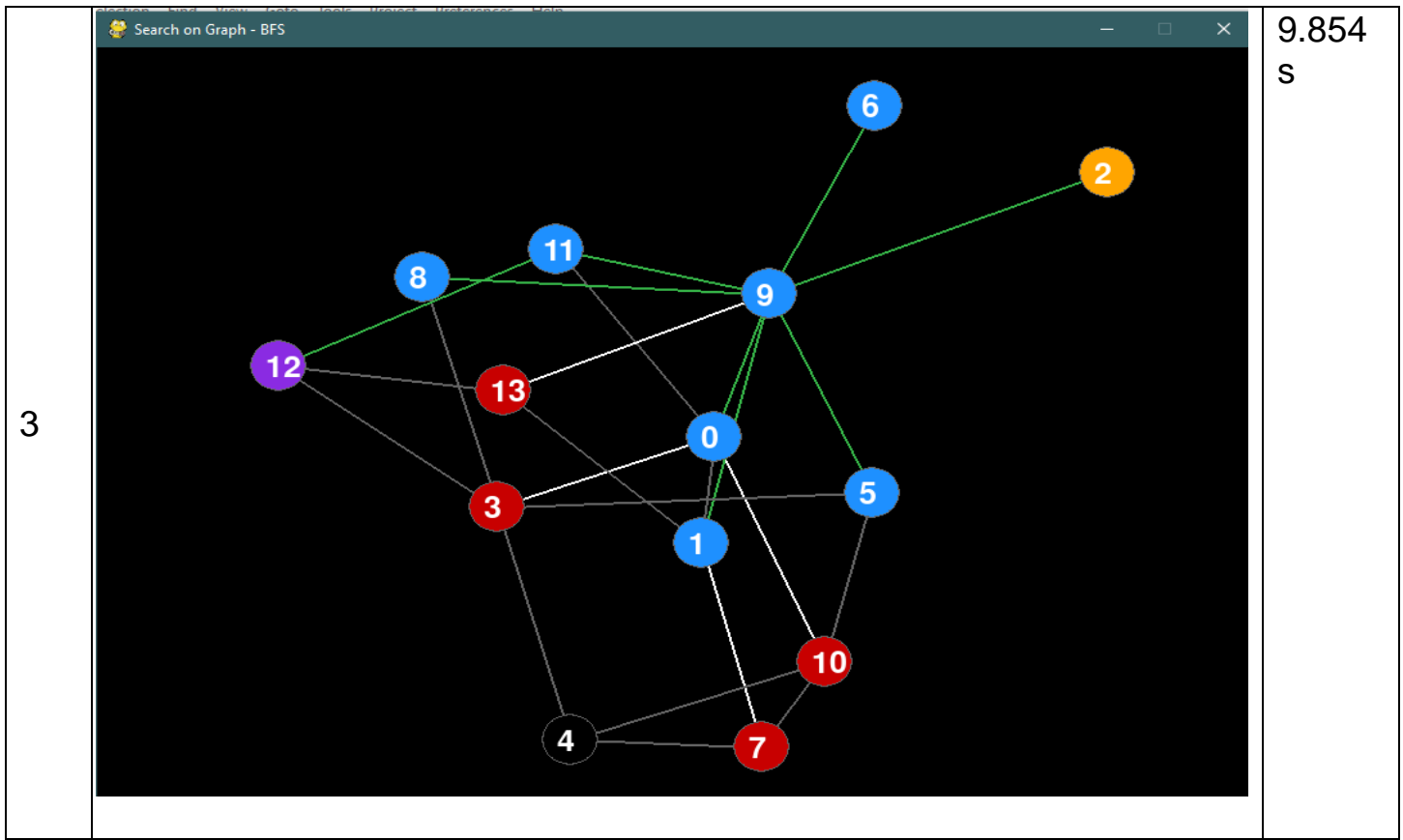
3



**Đồ thị
13
đỉnh
và có
các
đỉnh
treo**

2. Breadth First Search

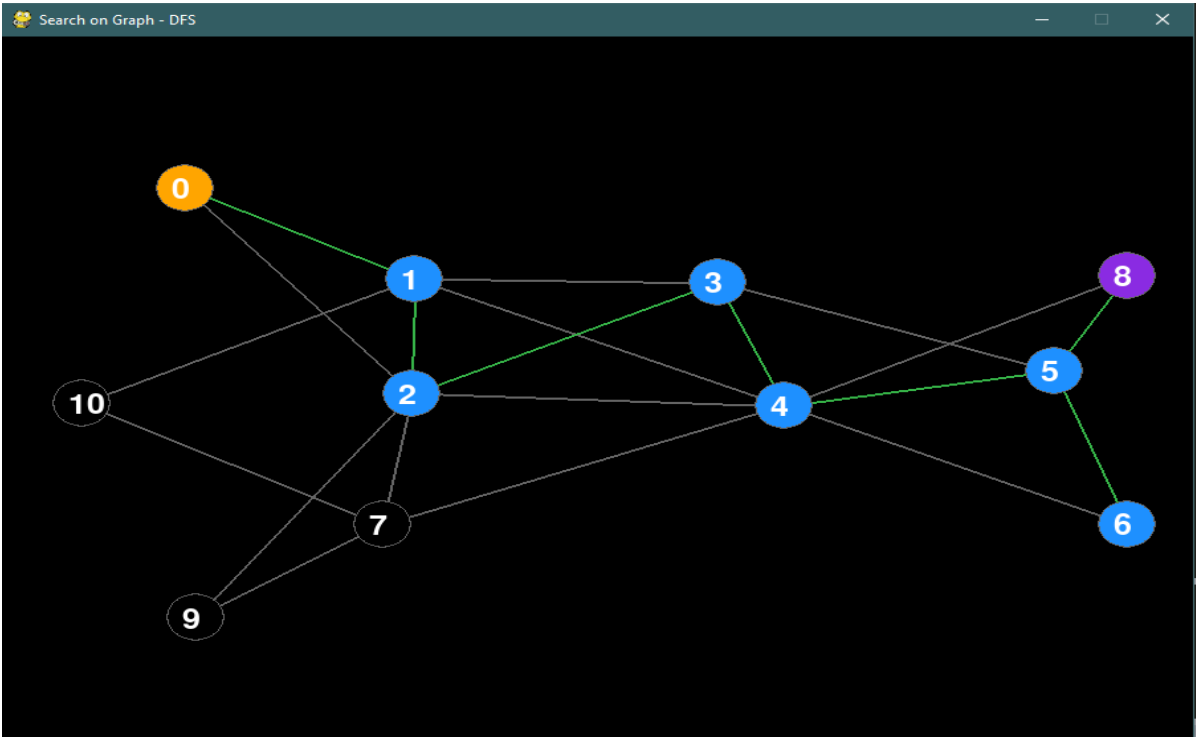
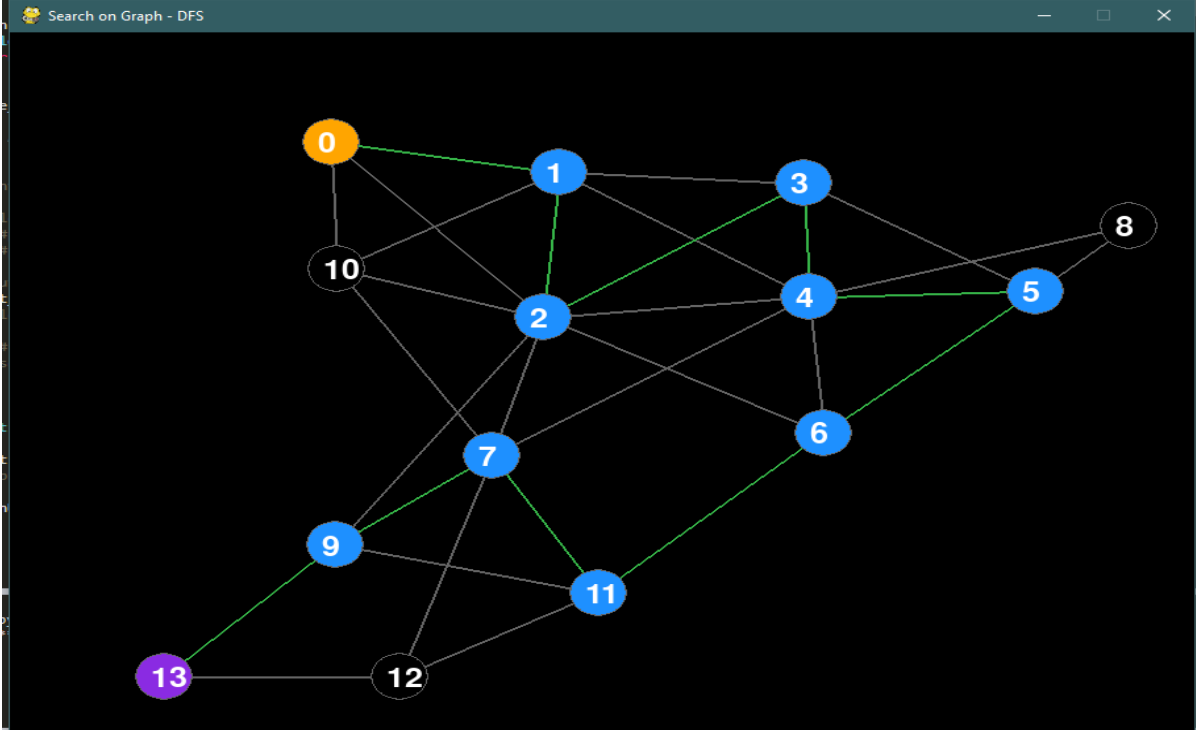
ST T	Kết quả	Thời gian
1		7.24s
2		10.86 s

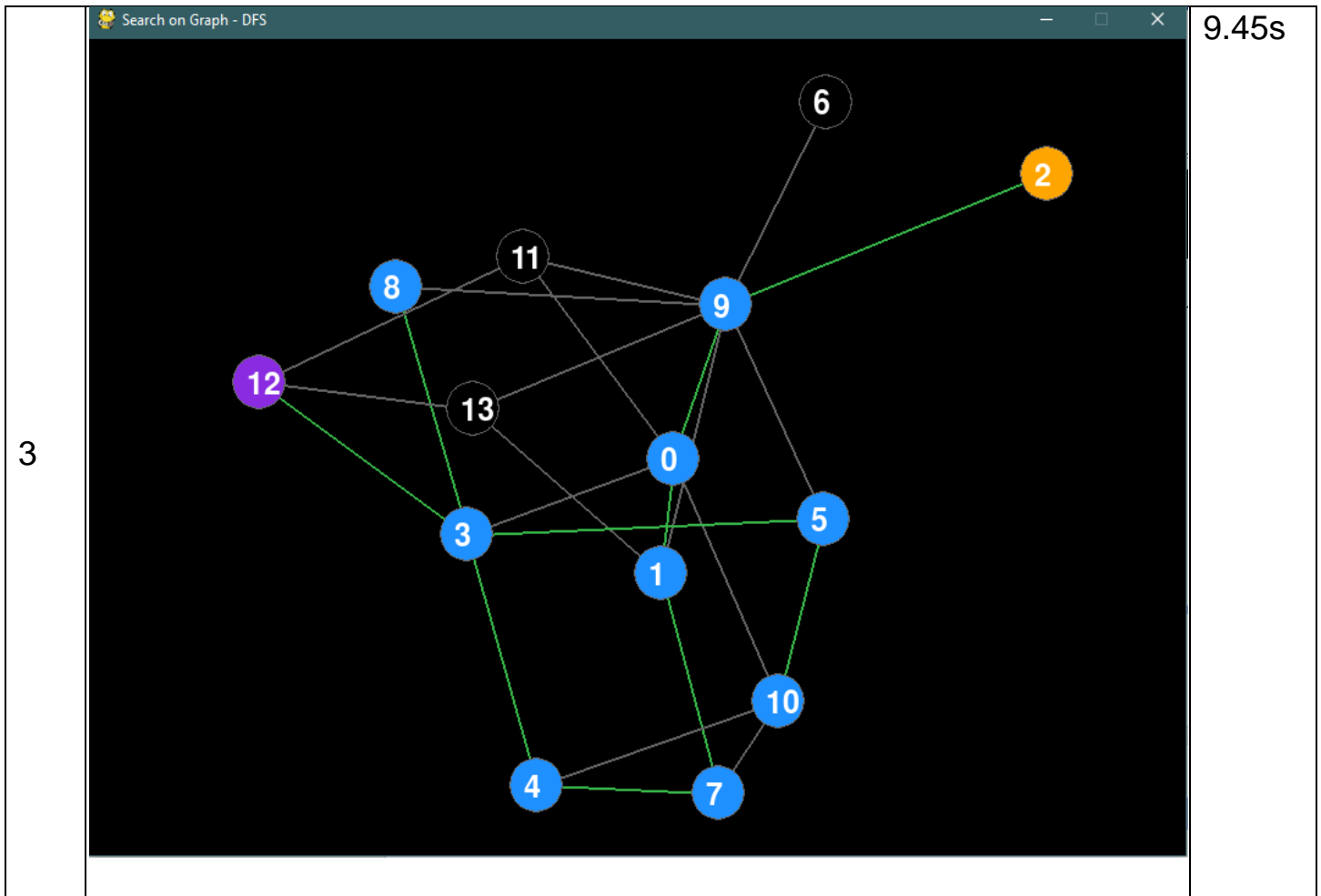


Nhận xét:

- Thời gian tìm lời giải trung bình: 9.318s (max)
- Tại mỗi đỉnh, thuật toán luôn xét đến tất cả các đỉnh chưa xét mà kề với đỉnh đang xét. Do đó, độ phức tạp thời gian lớn dẫn đến việc tìm lời giải có vẻ chậm hơn so với các thuật toán khác
- Có thể sử dụng thuật toán tìm kiếm theo chiều rộng cho hai mục đích: tìm kiếm đường đi từ một đỉnh gốc cho trước tới một đỉnh đích, và tìm kiếm đường đi từ đỉnh gốc tới tất cả các đỉnh khác.

3. Depth First Search

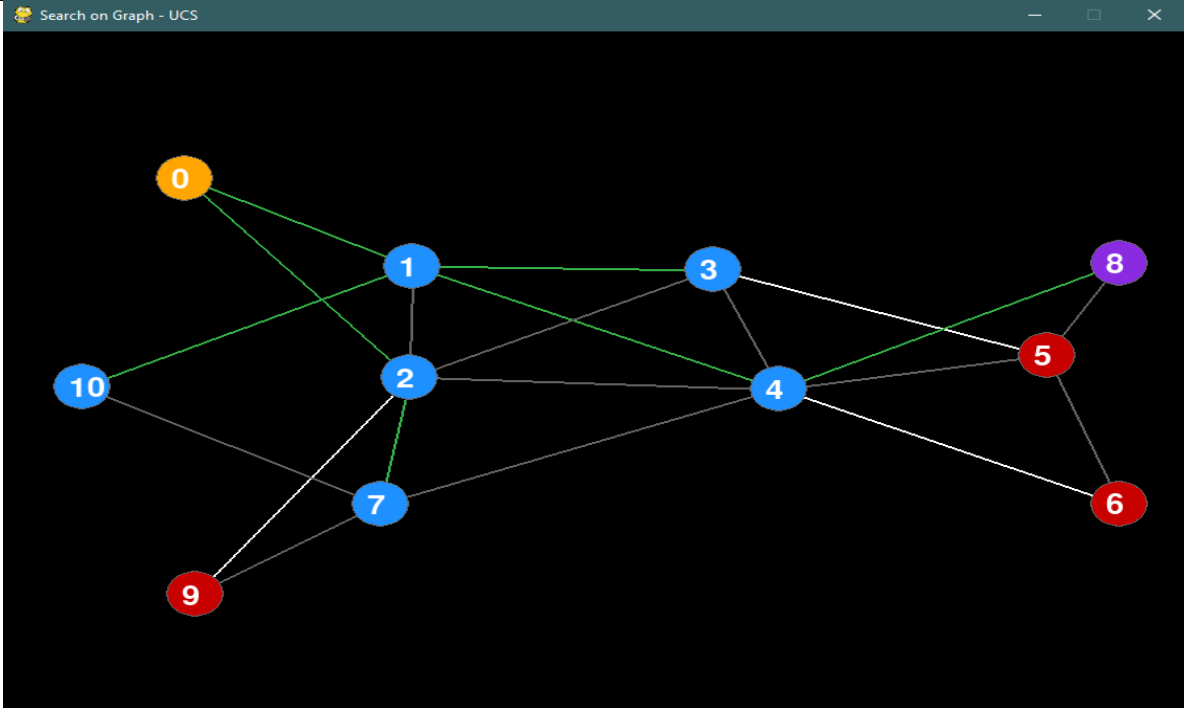
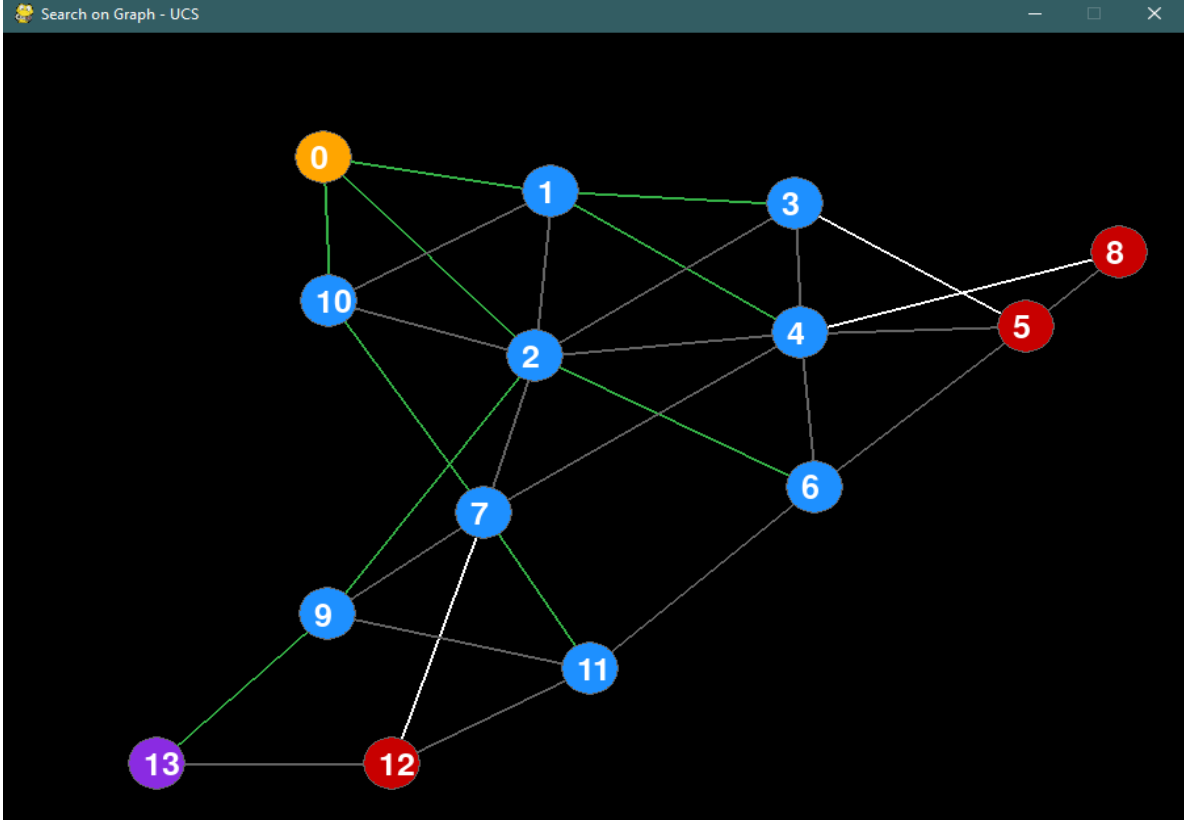
ST T	Kết quả	Thời gian
1		6.236 s
2		8.237 s

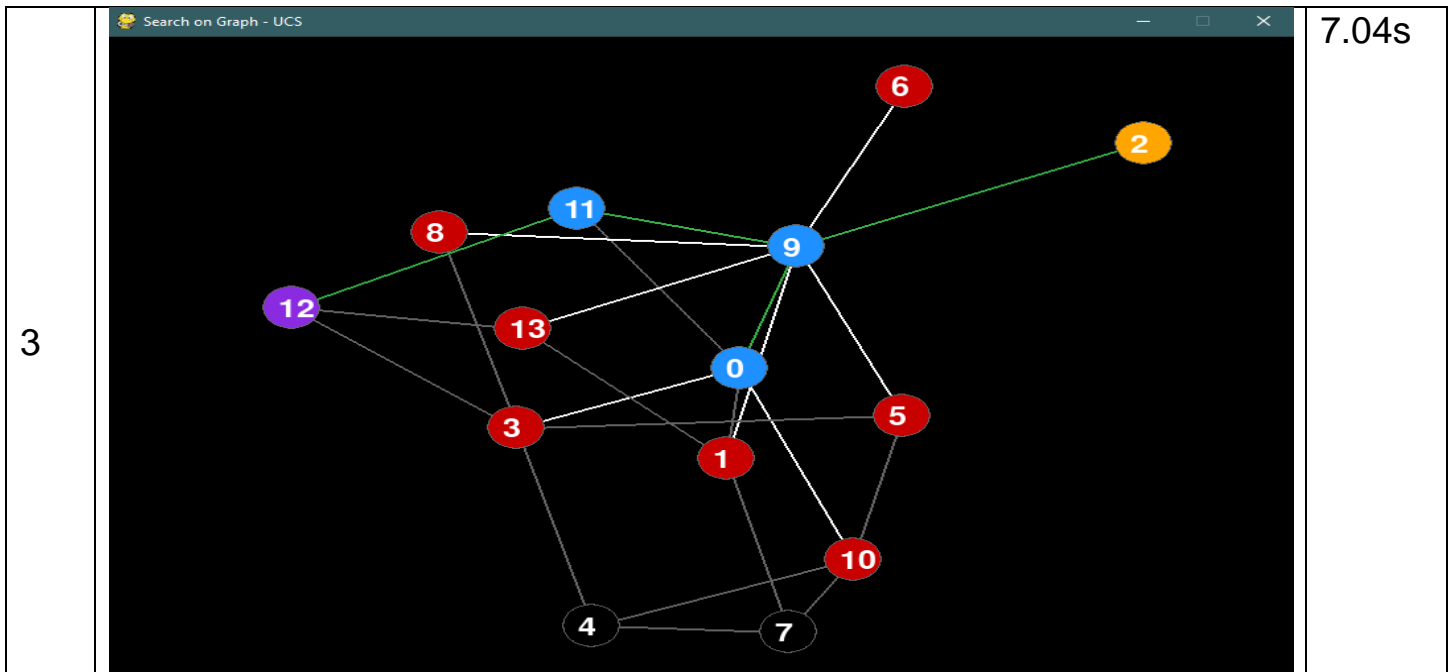


Nhận xét:

- Thời gian tìm lời giải trung bình: 7.97s
- Tuy độ phức tạp không gian của DFS thấp hơn so với BFS. Độ phức tạp về thời gian của hai thuật toán là như nhau. Trong mỗi bước, DFS sẽ phát triển xa nhất theo mỗi nhánh, do đó khả năng dẫn đến trường hợp xấu có thể cao
- Mất ít thời gian hơn để đến được nút mục tiêu so với thuật toán BFS (nếu nó đi đúng đường)
- Nhiều giải thuật sử dụng tìm kiếm theo chiều sâu:
 - Xác định các thành phần liên thông của đồ thị
 - Sắp xếp tô-pô cho đồ thị
 - Xác định các thành phần liên thông mạnh của đồ thị có hướng
 - Kiểm tra một đồ thị có phải là đồ thị phẳng hay không

4. Uniform Cost Search

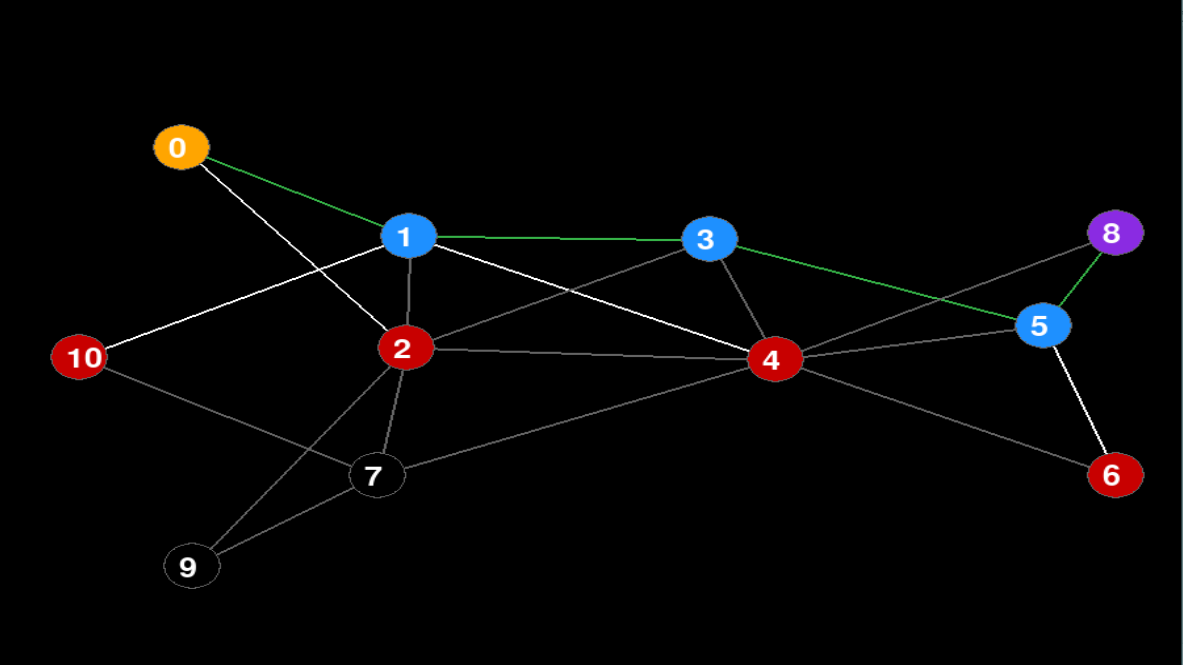
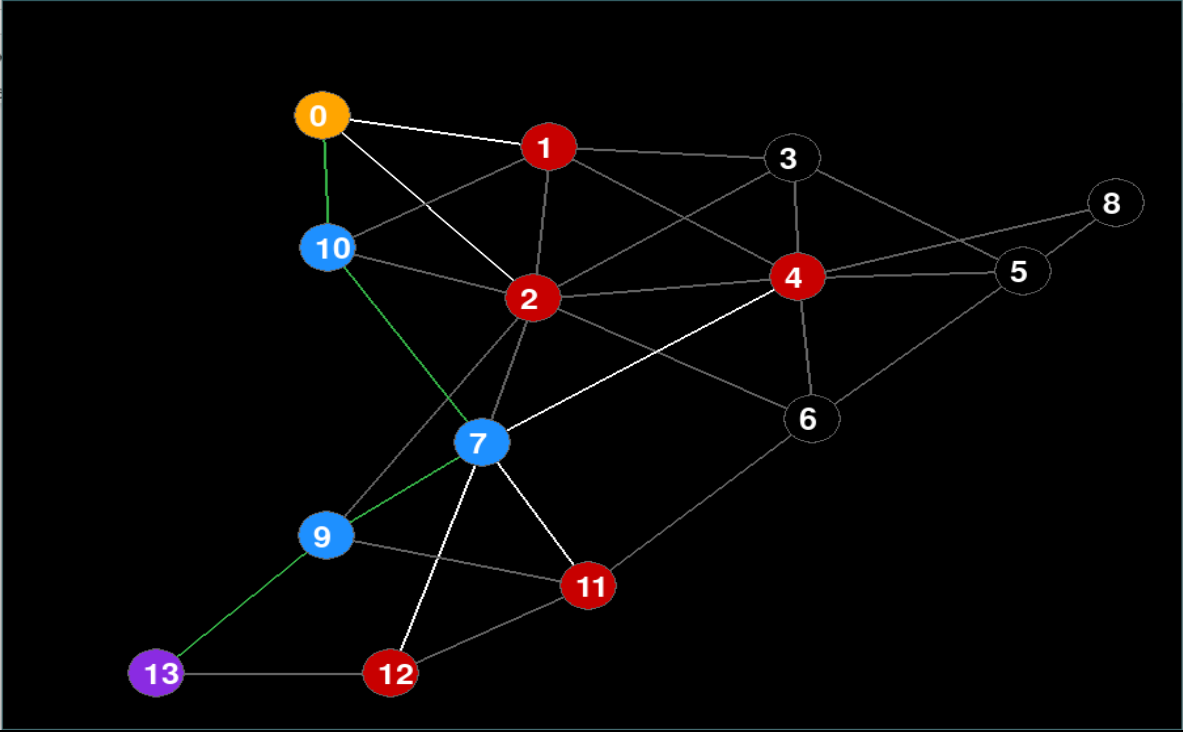
ST T	Kết quả	Thời gian
1		8.45s
2		11.46 s

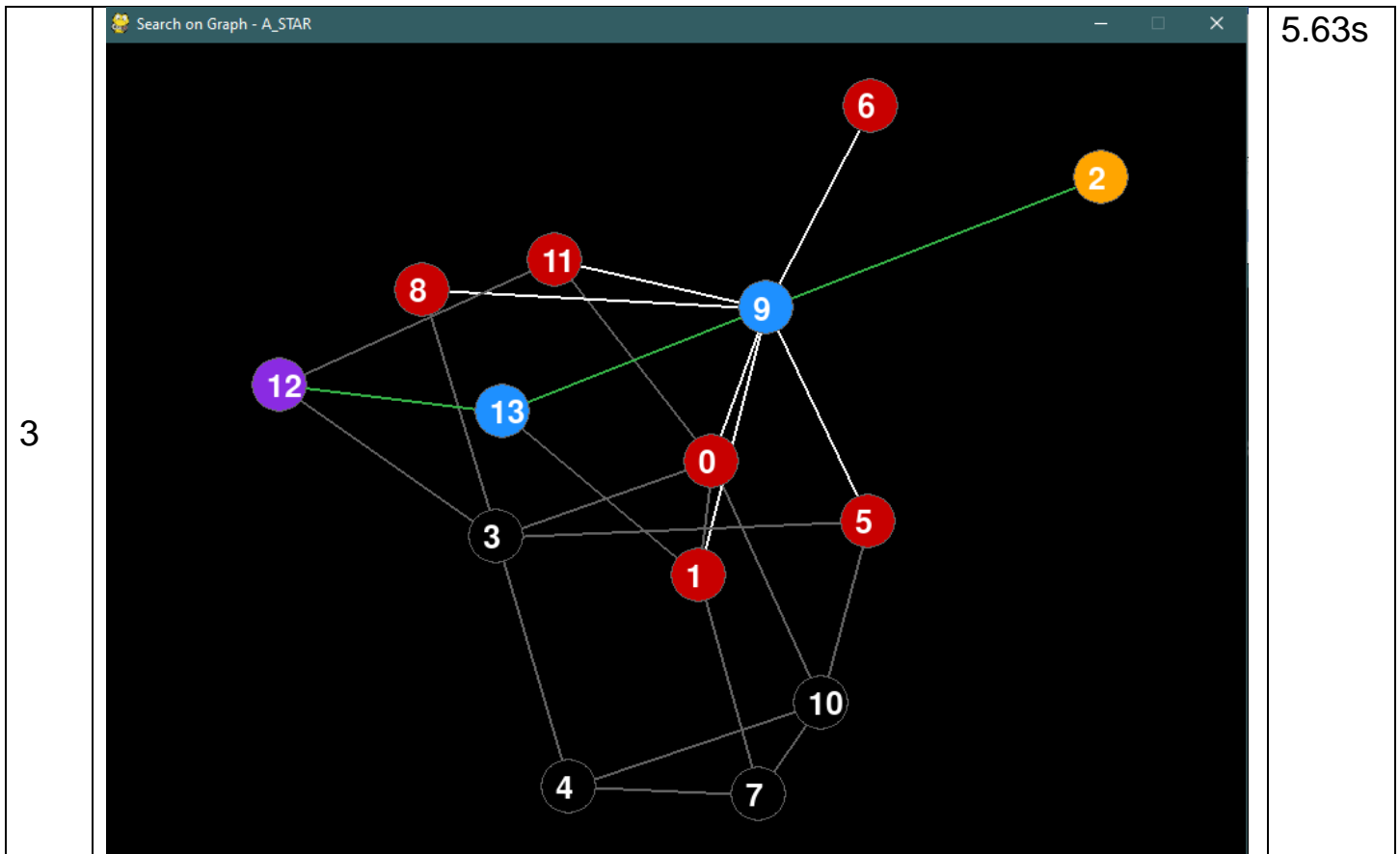


Nhận xét:

- Thời gian tìm lời giải trung bình: 8.98s
- Thuật toán UCS luôn tìm được đường đi với chi phí thấp nhất.
- Qua mỗi bước duyệt, các node còn lại trong danh sách đều được so sánh để sắp xếp lại và chọn ra node "được cho là" có chi phí thấp nhất. Do chỉ đánh giá tự trên bản thân nó mà không sử dụng hàm đánh giá khách quan, nên dễ đi sai đường. Do đó thời gian tìm được lời giải chậm hơn thuật toán A*.

5. A Star search

STT	Kết quả	Thời gian
1		5.82s
2		6.23s



Nhận xét:

- Thời gian tìm lời giải trung bình: 5.89s (min)
- Đánh giá heuristic của thuật toán A* là ước lượng chi phí từ đỉnh đang xét đến đích. Hàm heuristic trả về giá trị là độ dài thẳng (đường chim bay) từ vị trí xét đến đích.
- Ở mỗi bước duyệt đỉnh, đánh giá sẽ đưa ra một đỉnh có vẻ gần đích nhất trong các tập cùng xét và tiến về đỉnh đó.
- Cài đặt thuật toán sử dụng heuristic này vì tính đơn giản, trực quan và dễ dàng cài đặt.

TÀI LIỆU THAM KHẢO

Tài liệu tham khảo

- [1] C. L. Hoang. [Trực tuyến]. Available: <https://lhchuong.wordpress.com/2013/11/24/thua%CC%A3t-toan-dfs-tim-kiem-theo-chieu-sau/>.
- [2] T. Trần, “Viblo,” [Trực tuyến]. Available: <https://viblo.asia/p/cac-thuat-toan-co-ban-trong-ai-phan-biet-best-first-search-va-uniform-cost-search-ucs-Eb85omLWZ2G>.
- [3] “Wikipedia,” [Trực tuyến]. Available: https://vi.wikipedia.org/wiki/T%C3%ACm_ki%E1%BA%BFm_theo_chi%E1%BB%81u_s%C3%A2u.
- [4] “Wikipedia,” [Trực tuyến]. Available: [https://vi.wikipedia.org/wiki/Gi%E1%BA%A3i_thu%E1%BA%ADt_t%C3%ACm_ki%E1%BA%BFm_A*](https://vi.wikipedia.org/wiki/Gi%E1%BA%A3i_thu%E1%BA%ADt_t%C3%ACm_ki%E1%BA%BFm_A%2A).
- [5] “Wikipedia,” [Trực tuyến]. Available: https://vi.wikipedia.org/wiki/T%C3%ACm_ki%E1%BA%BFm_theo_chi%E1%BB%81u_r%E1%BB%99ng.
- [6] “Wikipedia,” [Trực tuyến]. Available: https://vi.wikipedia.org/wiki/T%C3%ACm_ki%E1%BA%BFm_chi_ph%C3%AD_%C4%91%E1%BB%81u.
- [7] “JavaTPoint,” [Trực tuyến]. Available: <https://www.javatpoint.com/ai-uninformed-search-algorithms#:~:text=The%20same%20logic%20is%20for,with%20the%20lowest%20path%20cost..>