**Đường đi Euler và chu trình Euler được ra đời từ bài toán nào?**

Trả lời:

Bài toán Konigsberg

**Nếu đưa ra một vấn đề về đồ thị, làm thế nào để chúng ta biết liệu chúng ta cần sử dụng thuật toán bfs hay dfs ??? hoặc khi nào chúng ta sử dụng thuật toán dfs hoặc thuật toán bfs . Sự khác biệt và lợi thế của cái này so với cái khác là gì?**

Trả lời:

BFS sẽ sử dụng nhiều bộ nhớ hơn tùy thuộc vào yếu tố phân nhánh ... tuy nhiên, BFS là một thuật toán hoàn chỉnh ... có nghĩa là nếu bạn đang sử dụng nó để tìm kiếm thứ gì đó ở độ sâu thấp nhất có thể, BFS sẽ cung cấp cho bạn giải pháp tối ưu . Độ phức tạp của không gian BFS là O(b^d)... hệ số phân nhánh được nâng lên độ sâu (có thể là RẤT NHIỀU bộ nhớ).

Mặt khác, DFS tốt hơn nhiều về không gian tuy nhiên nó có thể tìm ra giải pháp tối ưu. Có nghĩa là, nếu bạn chỉ tìm kiếm một đường dẫn từ đỉnh này sang đỉnh khác, bạn có thể tìm thấy giải pháp tối ưu (và dừng ở đó) trước khi bạn tìm thấy đường dẫn ngắn nhất thực sự. Độ phức tạp của không gian DFS là O(|V|)... có nghĩa là bộ nhớ nhiều nhất nó có thể chiếm là con đường dài nhất có thể.

Chúng có cùng độ phức tạp thời gian.

Về mặt triển khai, BFS thường được triển khai với Queue, trong khi DFS sử dụng Stack.

**Sự khác biệt giữa DFS và BFS, nhưng tôi muốn biết khi nào thực tế hơn để sử dụng cái này hơn cái kia?**

Điều đó phụ thuộc nhiều vào cấu trúc của cây tìm kiếm và số lượng và vị trí của các giải pháp (còn gọi là các mục tìm kiếm).

* Nếu bạn biết một giải pháp không xa gốc của cây, thì lần tìm kiếm đầu tiên (BFS) có thể tốt hơn.
* Nếu cây rất sâu và các giải pháp rất hiếm, thì việc tìm kiếm đầu tiên (DFS) có thể mất một thời gian rất dài, nhưng BFS có thể nhanh hơn.
* Nếu cây rất rộng, BFS có thể cần quá nhiều bộ nhớ, vì vậy nó có thể hoàn toàn không thực tế.
* Nếu các giải pháp là thường xuyên nhưng nằm sâu trong cây, BFS có thể không thực tế.
* Nếu cây tìm kiếm rất sâu, bạn sẽ cần phải hạn chế độ sâu tìm kiếm cho độ sâu tìm kiếm đầu tiên (DFS), dù sao (ví dụ với độ sâu lặp lại).