Tổng quan

Trong phát triển game thuật toán tìm đường được ứng dụng rất phổ biến từ các game AAA cho đến cả những game puzzle trên mobile.

Như cái tên của nó. Thuật toán tìm đường giúp tìm ra con đường từ điểm A tới điểm B tránh khỏi các chướng ngại vật với một lộ trình tối ưu nhất.

Ví dụ trong game moba. Thuật toán tìm đường giúp nhân vật tự động di chuyển đến điểm người chơi đã chọn, giúp creep chạy về phía nhà chính đối phương và né khỏi các chướng ngại vật trên đường đi như trụ, nhà lính, tường. Trong pikachu, thuật toán tìm đường được sử dụng để để kiểu tra 2 con pikachu có match được với nhau không.

Các thu?t toán tìm ???ng th??ng g?p.

	Dijkstra	**A***	**BFS (Breadth-first search)**
Định nghĩa	Tìm đường đi **với chi phí thấp nhất** bằng cách duyệt qua tất cả các node trên bản đồ và tính toán khoảng cách ngắn nhất từ gốc đến mỗi node.	Tìm đường đi **tối ưu nhất** (khoảng cách và chi phí) dựa trên hằng số heuristic để dự đoán khoảng cách từ node đến đích.	Thuật toán **tìm kiếm theo chiều rộng** . Tìm kiếm theo từng lớp bắt đầu từ node gốc sau đó chuyển sang các lớp kế tiếp.
Ưu điểm	Đảm bảo tìm ra đường đi ngắn nhất.	- Tìm đường nhanh, chính xác - Hiệu quả với bản đồ lớn.	- Đơn giản, dễ triển khai. - Hiệu quả với bản đồ không có trọng số
Nhược điểm	Tốn kém (bộ nhớ và CPU) với bản đồ lớn.	Phụ thuộc vào hàm heuristic (nếu hàm không tốt có thể gây tốn tài nguyên).	- Không tối ưu cho tìm đường đi ngắn nhất - Tốc độ chậm với các bản đồ lớn.

Ghi chú:

- Hàm heuristic là một hàm ước lượng khoảng cách từ một node dến đích.
- Bản đồ ở đây là một tập hợp các node và các cạnh nối giữa chúng.

Các yếu tố ảnh hưởng đến lựa chọn thuật toán

Lo?i game

Thể loại game ảnh hưởng lớn đến việc lựa chọn thuật toán. Ví dụ trong một game nhập vai thế giới mở với môi trường rộng lớn và phức tạp thì thuật toán A* hoặc Jump Point Search có thể là lựa chọn tốt hơn Dijkstra. Trong khi với một game puzzle thì BFS có thể hiệu quả hơn.

Môi tr??ng trong game

Cấu trúc của môi trường trong game cũng là một yếu tố quan trọng. Nếu môi trường game là một lưới đơn giản (grid-based) thì Jump Point Search có thể là lựa chọn tối ưu. Nếu môi trường phức tạp bao gồm cả địa hình thì NavMesh có thể là lựa chọn.

Hi?u n?ng

Thuật toán tìm đường sẽ ảnh hưởng đến hiệu năng của game. Đặc biệt với những game có nhiều đối tượng cần tìm đường cùng lúc. Cần cân nhắc giữa độ chính xác của thuật toán và tốc độ tính toán để đảm bảo game có thể chạy chính xác và mượt mà.

S? l??ng nhân v?t

Như đã nói ở trên thì hiệu năng bị ảnh hưởng khi có nhiều nhân vật cùng tìm đường. Thế nên cần cân nhắc lựa chọn thuật toán phù hợp với số lượng nhân vật trong game.

Các y?u t? khác

Ngoài ra còn cần xem xét đến nhiều yếu tố khác như khả năng di chuyển của nhân vật, loại chướng ngại vật, yêu cầu về tính linh hoạt của đường đi...

Ví d?

Trong game chiến thuật thời gian thực (RTS) với hàng trăm đơn vị quân di chuyển trên bản đồ, việc sử dụng thuật toán Dijkstra có thể gây ra hiện tượng giật lag do yêu cầu tính toán lớn. Trong trường hợp này A* hoặc Jump Point Search là lựa chọn phù hợp hơn.

Revision #2
Created 2025-06-22 10:20:11 UTC by ThanhDV
Updated 2025-06-22 10:28:20 UTC by ThanhDV