ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

TRÍ TUỆ NHÂN TẠO

Thuật toán DFS/BFS/UCS cho Sokoban

GVHD: Lương Ngọc Hoàng

Lớp: CS106.O21

Tên: Lê Minh Nhựt

MSSV: 22521060

I. Mô hình hóa Sokoban

Sokoban là một trò chơi logic, trong đó, chúng ta điều khiển một nhân vật để đẩy hộp vào các ô mục tiêu trong một mê cung. Sokoban được mô hình hóa như sau:

- **Trạng thái khởi đầu:** Trạng thái khởi đầu của Sokoban được biểu diễn bằng một bản đồ chứa nhân vật và các hộp trong mê cung, cùng với vị trí của chúng.
- **Trạng thái kết thúc:** Trạng thái kết thúc xảy ra khi tất cả các hộp đã được đẩy vào các ô mục tiêu. Tức là, khi mọi ô mục tiêu đều có một hộp đặt lên.
- **Không gian trạng thái**: Không gian trạng thái của Sokoban bao gồm tất cả các trạng thái có thể xảy ra trong quá trình di chuyển hộp và nhân vật. Đây là một không gian với mỗi trạng thái được định nghĩa bởi vị trí của hộp và nhân vật trên bản đồ.
- **Các hành động hợp lệ:** Trong Sokoban, chúng ta có thể thực hiện các hành động như di chuyển nhân vật lên, xuống, trái hoặc phải để đẩy hộp đến các ô trống. Tuy nhiên, các hành động này chỉ được coi là hợp lệ nếu nhân vật có thể di chuyển và không có chướng ngại vật ngăn cản (hộp chặn hộp, tường).
- **Hàm tiến triển (successor function):** Hàm tiến triển xác định các trạng thái tiếp theo có thể xảy ra sau một hành động nhất định. Nó kiểm tra xem việc thực hiện hành động sẽ dẫn đến trạng thái mới nào (Ví dụ: Đi bên trái sẽ dẫn tới trạng thái nào? Có đẩy hộp hay không?) và kiểm tra tính hợp lệ của trạng thái đó (Ví dụ: Trạng thái mới có phải là trạng thái hợp lệ không?).

Mô hình hóa trên sẽ cho phép máy tính giải quyết Sokoban thông qua các thuật toán tìm kiếm hoặc thông qua việc sử dụng trí tuệ nhân tạo để tìm ra các chiến lược tốt nhất để di chuyển hộp và nhân vật để hoàn thành mê cung.

II. DFS, BFS, UCS cho Sokoban

Cả 3 thuật toán đều thuộc nhóm tìm kiếm không có thông tin:

DFS (Depth-First Search): tìm kiếm theo chiều sâu, hoạt động theo cơ chế của stack (last in first out), node (chuỗi trạng thái) nào vào sau sẽ được xử lý trước, DFS sẽ duyệt đến node sâu nhất của một nhánh trước, rồi mới đến nhánh khác.

BFS (Breadth-First Search): tìm kiếm theo chiều rộng, hoạt động theo cơ chế của queue (first in first out), node nào vào trước sẽ được xử lý trước, BFS sẽ duyệt đến các node nông nhất trước (trong cùng 1 tầng), khi đã duyệt hết tầng này thì mới đến tầng thấp hơn.

UCS (Uniform Cost Search): tìm kiếm theo chi phí, hoạt động theo cơ chế của priority queue, node nào có độ ưu tiên cao sẽ được lấy ra trước, UCS sẽ mở rộng các node dựa trên độ ưu tiên (đối với Sokoban node nào có chi phí thấp nhất thì ưu tiên mở node đó ra).

III. So sánh các thuật toán DFS, BFS, UCS cho Sokoban

Bảng thống kê số bước đi các thuật toán DFS, BFS, UCS

Bước đi / Level	DFS	BFS	UCS
1	79	12	12
2	24	9	9
3	403	15	15
4	27	7	7
5		20	20
6	55	19	19
7	707	21	21
8	323	97	97
9	74	8	8
10	37	33	33
11	36	34	34
12	109	23	23
13	185	31	31
14	865	23	23
15	291	105	105
16	86	34	34
17	Không lời giải	Không lời giải	Không lời giải
18			

Nhận xét:

- **DFS:** cho lời giải đường đi không tối ưu cho các màn.
- **BFS:** cho lời giải đường đi tối ưu cho các màn.
- **UCS:** cho lời giải đường đi tối ưu cho các màn.

*Lời giải đường đi của BFS và UCS đều giống nhau và là lời giải ngắn nhất cho các màn.

IV. Đánh giá chung

- Trong 3 thuật toán trên, ta thấy chỉ có BFS và UCS cho lời giải tối ưu cho bài toán Sokoban. Nhưng UCS sẽ tốt hơn cả vì nó sẽ ưu tiên cho đường đi có chi phí thấp nhất khi chưa dịch chuyển hộp, điều này giúp tiết kiệm thời gian khi tránh phải đi vào những đường đi có số bước đi nhiều trước, đối với những màn có không gian trạng thái càng lớn UCS càng hiệu quả về mặt thời gian hơn so với BFS.
- Bản đồ của màn 18 là khó giải nhất bởi vì nó có không gian trạng thái rất phức tạp, có nhiều hộp cần đặt vào ô mục tiêu và cấu trúc các bức tường gây cản trở lớn cho việc đẩy hộp, điều này tạo ra khó khăn trong việc tìm kiếm lời giải cho cả 3 thuật toán. Ngoài ra, màn 5 cũng gây nhiều khó khăn cho các thuật toán bởi vì không gian trạng thái rộng lớn, làm số lượng hành động trở nên rất nhiều, nhất là đối với DFS khi phải duyệt sâu để tìm lời giải.

File code: BT1_22521060.zip