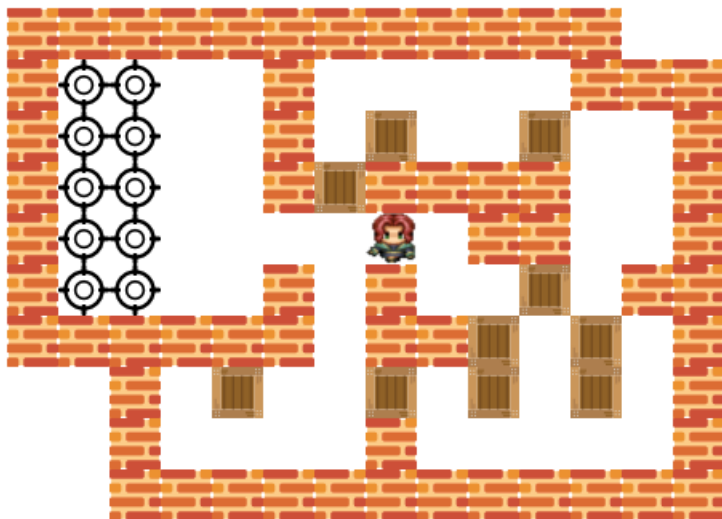


# BÁO CÁO MÔN CS112

BT1 – BFS/DFS/UCS for Sokoban



*Giảng viên hướng dẫn:*

*Lương Ngọc Hoàng*

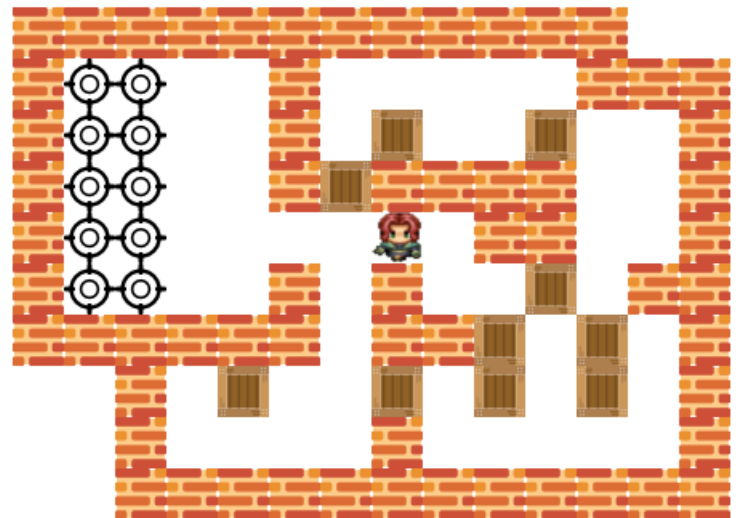
*Sinh viên thực hiện:*

*Nguyễn Gia Bảo - 22520109*

## I. Mô hình hóa game Sokoban và các quy định của trò chơi (trong khuôn khổ bài tập này)

- Trong trò chơi Sokoban, các màn chơi được thiết kế và lưu dưới dạng file .txt. Các ký tự trong file sẽ được chuyển thành các vật thể trong trò chơi. Sự phân bố và số lượng các vật thể sẽ tạo nên độ khó cho các màn chơi. Các ký tự sẽ được chuyển hóa với quy tắc như sau:
  - # → Tường (wall)
  - B → Thùng (box)
  - & → Người chơi (player)
  - . → Đích
- Ví dụ:

```
1 #####
2 #.. #   ###
3 #.. # B B #
4 #.. #B### #
5 #.. & ## #
6 #.. # # B ##
7 ##### ##B B #
8 # B B B B #
9 #   #   #
10 #####
```



- Trong các màn chơi, sẽ có các khái niệm:
  - Không gian trạng thái (state space): coi mỗi vị trí của người chơi và các thùng là 1 trạng thái (state), thì không gian trạng thái là tất cả những vị trí có thể của người chơi và các thùng trên bản đồ.

- Trạng thái khởi đầu (start state): là vị trí ban đầu của player và các thùng
  - Trạng thái kết thúc (end state): là trạng thái khi tất cả các thùng đều đã ở vị trí đích
  - Các hành động hợp lệ (legal actions): là các bước đi hợp lệ có thể có của người chơi, để dẫn từ trạng thái này sang trạng thái khác. Mỗi bước đi được cho là hợp lệ khi vị trí mới của player không trùng với thùng hoặc tường, vị trí mới của thùng không trùng với tường hoặc thùng khác.
  - Hàm tiền triển (successor function): Là 1 hàm để tìm ra 1 chuỗi các bước đi từ start state thành end state.
- Dưới góc nhìn của các thuật toán, các màn chơi của sokoban được mô hình hóa thành 1 cây gồm các node là các trạng thái của trò chơi, mỗi node có thể mở ra các node con thông qua các hành động hợp lệ của nó. Các thuật toán tìm kiếm sẽ duyệt qua các node từ node khởi đầu, sau đó mở rộng ra các node con lân cận có thể mở thông qua các bước đi hợp lệ, và dừng lại khi tìm thấy được node đích (vị trí các thùng trùng với các vị trí đích)

## II. Thực hiện các thuật toán

Lần lượt chạy các thuật toán DFS, BFS, UCS để giải các màn chơi có sẵn trong Sokoban, ta ghi nhận được bảng thống kê sau:

Level	DFS		BFS		UCS		
	Time (second)	Steps	Time (second)	Steps	Time (second)	Steps	Cost
1	0.05	79	0.07	12	0.06	12	6
2	0.00	24	0.01	9	0.00	9	5
3	0.21	403	0.15	15	0.11	15	9
4	0..00	27	0.00	7	0.00	7	4
5	-	-	175.1	20	136.84	20	10
6	0.01	55	0.01	19	0.01	19	14
7	0.49	707	0.75	21	0.56	21	10
8	0.07	323	0.18	97	0.21	97	65
9	0.26	74	0.01	8	0.01	8	5
10	0.01	37	0.01	33	0.01	33	25
11	0.01	36	0.01	34	0.02	34	27
12	0.13	109	0.08	23	0.09	23	16
13	0.16	185	0.13	31	0.18	31	24
14	3.49	865	2.51	23	3.05	23	16
15	0.14	291	0.24	105	0.28	105	62
16	-	-	21.12	34	19.80	34	22
17	22.89	0	27.53	0	24.63	0	0
18	-	-	-	-	-	-	-

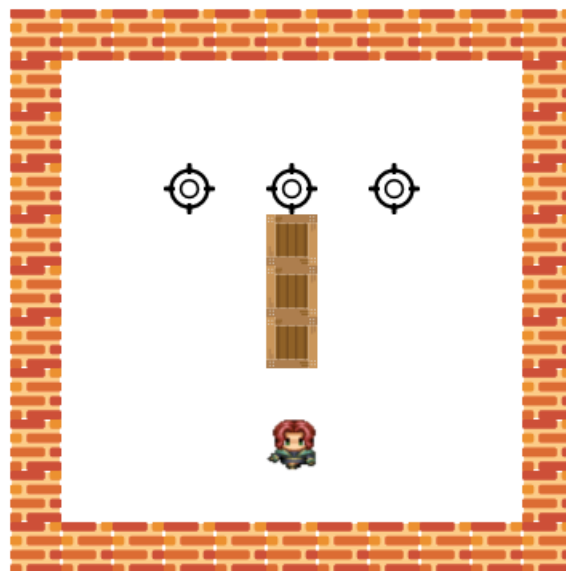
*Chú thích: “-” là thuật toán không trả về kết quả do giới hạn của phần cứng. Các màn chơi có số bước bằng 0 nghĩa là màn chơi không có lời giải.*

### III. Kết luận

Từ bảng thống kê trên, ta có thể rút ra được các kết luận sau:

- Trong cả 3 thuật toán, DFS có số bước đi lớn hơn rõ rệt so với 2 thuật toán còn lại. Về mặt thời gian, DFS có thể nhanh hơn BFS và UCS trong đa số các màn chơi.

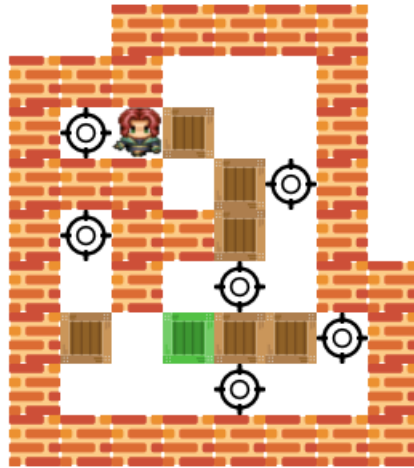
- Còn về BFS và UCS, số bước đi tìm được bởi 2 thuật toán này giống nhau, còn thời gian thực thi của UCS nhanh hơn BFS ở đa số các màn. Điều này là do BFS coi tất cả các bước đi đều có chi phí là 1, trong khi UCS cần thêm thời gian để tính toán chi phí cho từng bước đi. Nhưng xét trên tổng thể thì UCS vẫn vượt trội hơn BFS, và xếp cuối là DFS.
- Trong màn chơi level 5, DFS không thể tìm được đáp án, nhưng UCS và BFS lại làm được, với UCS thậm chí thời gian cần thiết cũng ngắn hơn BFS rất nhiều. Điều này là do ở level 5, bản đồ chỉ có 4 bức tường bao quanh, bên trong có rất nhiều ô trắng, nghĩa là số lượng bước đi hợp lệ được sinh ra là rất lớn, khiến thuật toán tìm kiếm theo chiều sâu như DFS không đủ bộ nhớ để lưu trữ hết các trạng thái đã đi qua. Ngược lại, ở UCS và BFS các trạng thái được khám phá theo chiều ngang (BFS) và chiều ngang có hàng đợi ưu tiên (UCS). Điều này giúp giảm gánh nặng lưu trữ khi có thể sớm tìm được các trạng thái đích ở tầng cao của cây trạng thái.



*Bản đồ của level 5*

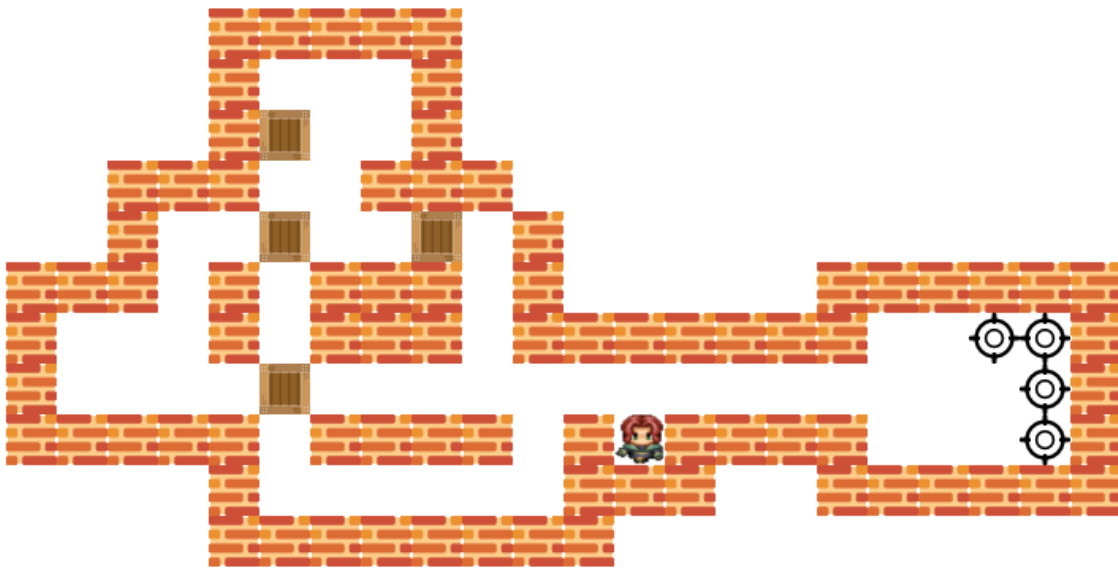
- Điều tương tự lại xảy ra khi tới level 16. Lúc này số lượng ô trắng đã không còn nhiều như level 5, nhưng thay vào đó số lượng thùng

và đích lại gấp đôi level 5. Một lần nữa BFS và UCS lại thể hiện tốt hơn DFS.



*Bản đồ level 16*

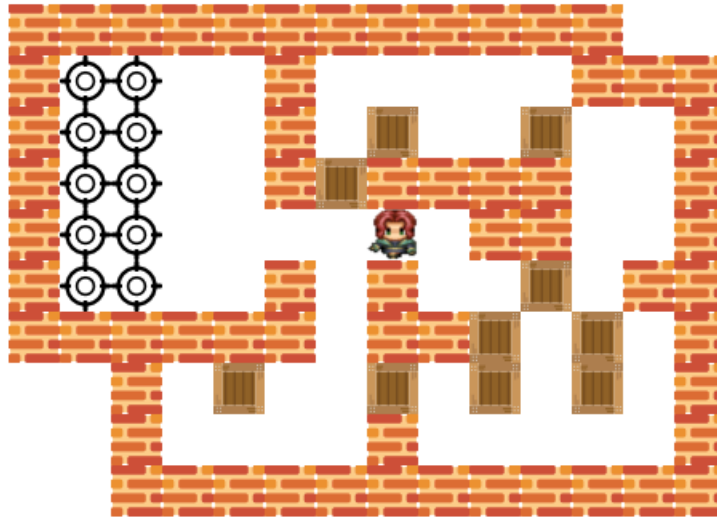
- Ở level 17, là một màn chơi mà không có lời giải, cả 3 thuật toán đều trả về với số steps = 0



*Bản đồ level 17*

- Cuối cùng, level 18 chính là level khó nhất, với 10 thùng cần phải đẩy vào 10 vị trí đích. Cả 3 thuật toán DFS, BFS, UCS mặc dù có

sự hiệu quả khác nhau nhưng ở level này thì cả 3 đều không thể trả về đáp án. Lí giải cho độ khó này của level 18 là vì số lượng thùng lớn nhất (10 thùng) và vị các thùng gần nhau, khiến trò dễ rơi vào trạng thái chết (một hoặc nhiều thùng bị kẹt vào góc tường).



*Bản đồ level 18*

➔ **Tổng kết:** Trong 3 thuật toán: DFS, BFS và UCS, thuật toán tối ưu nhất là UCS. Trong 18 level, level khó nhất là level 18.

\_\_\_HẾT\_\_\_