Sokoban Puzzle

Sokoban is a single agent grid based puzzle in which the agent's goal is to push the boxes scattered around the grid to the designated storage locations.

Objectives and rules

1. The environment is a rectangular grid of walls and floor squares. The floor squares may be empty or contain boxes or targets.
2. The objective is the player has to push each box onto goal square so that every target is covered and no box is remaining.
3. The player can move one square at a time in any of the four directions if the square is empty. Player must not pass through walls or boxes
4. If the player is in the adjacent square of the box, it can be pushed. Once the box is pushed it cannot be reversed.
5. Pushing the box diagonally, pushing more than one box or pulling boxes are not allowed.
6. If the box cannot be retreived from it's position and remains stuck, it is a deadlock condition and the puzzle becomes unsolvable.
7. The puzzle is considered solved when each box has been moved to a distinct goal position.

Symbols used in Sokoban

1) Wall (#)
2) Floor ( )
3) Box ($)
4) Player (@)
5) Goal (.)
6) Box-on-goal (*)
7) Player-on goal (+).

Date
Page

Problem Formulation

(1) State representation - It contains the current position of the player as coordinates (x, y) and the positions of all the boxes as a distinct coordinate. Here x represents row number and y is column number.
Syntax: $s = (P, \{b_1, b_2\})$ for two boxes.

(2) Actions - The allowed actions from any given state is to shift the position to any adjacent empty square. If the adjacent cell in direction d contains the box and the next cell beyond in d is empty it will be pushed. Else it is an invalid state. The four actions are MoveUp, Move Down, Move Left and Move Right.
Syntax: ACTION(s)

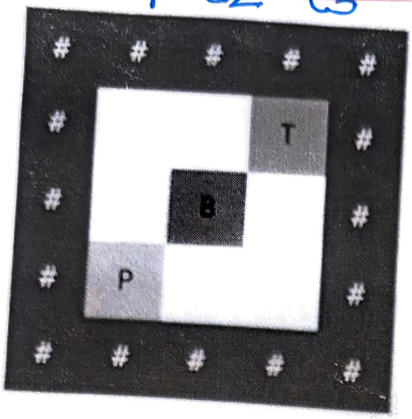
(3) Transition Model - It defines the result of applying each action to a given state.
Syntax: RESULT $((P, b), d) = (P', b')$
If the player moves without pushing, $P' = p + d$, $b' = b$ otherwise $p' = b$ and $b' = b + d$. In the first case only player position is changed and in second case both player and box position gets updated. An action is invalid if an agent pushes the box to the wall

(4) Goal Test - If the current coordinates of all boxes is B_pos and the fixed coordinates of all target squares is T_pos, the goal test is true if and only if these two sets are identical.

(5) Action Cost - This is a metric for evaluating solution. Each move usually has unit cost and push has cost = 2
For any action a taken in state s, results in s'
ACTION COST $(s, a, s') = 1$ or 2

C1 C2 C3

R1
R2
R3



① Formulate and solve the sokoban problem in a 3X3 grid with the player, box and target coordinates as shown,

Number of states $= \left[\frac{n!}{b!\,(n-b)!}\right](n-b)$

where n = number of non obstacle cells and b is no of cells

For 3X3 grid above, No. of states $= \frac{9!}{1!\,(9-1)!}(9-1) = \frac{(362820)\times 8}{40320}$

$= 72$

Step 1: State is represented in the form of [player_pos, box_pos] = [(3,1), (2,2)]

Step 2: Actions: The following actions can be done on the player: Move Right, Move Left, Move Up and Move Right

Step 3: Transition Model: Format: RESULT(s, a) = s'

RESULT ((3,1), (2,2), Move Right) = ((3,2), (2,2))

RESULT((3,2), (2,2), Move Up) = ((2,2), (1,2))

RESULT ((2,2), (1,2), Move Left) = ((2,1), (1,2))

RESULT ((2,1), (1,2), Move Up) = ((1,1), (1,2))

RESULT ((1,1), (1,2), Move Right) = ((1,2), (1,3))

Step 4: Action Cost

		Category
i)	Action Cost ((3,1), (2,2)), Move Right ((3,2), (2,2)) = 1	Simple Move
ii)	Action Cost ((3,2), (2,2)), Move Up ((2,2), (1,2)) = 2	Push
iii)	Action Cost ((2,2) (1,2)), Move Left ((2,1) (1,2)) = 1	Simple Move
iv)	Action Cost ((2,1) (1,2), Move Up ((1,1), (1,2)) = 1	Simple Move
v)	Action Cost ((1,1) (1,2), Move Right ((1,2), (1,3)) = 2	Push

The cumulitive cost of solving the problem = 07

Step 5: Goal Test

B_Pos = (1,3) and T_Pos = (1,3). The two sets are identical hence the goal test is PASS.

```
1 #  #  #  #  #  #
      1  2  3  4
#  1T1  -  B      #
#2  _  _  _  _    #
#3  P  B  _  T2   #
#4  _  _  _  _    #
#  #  #  #  #  #
```

(2) Example 2

Solve ~~Consider a~~ Sokoban problem in a 4X4 grid with player position, positions of boxes and targets are shown.

Step 1: State is represented as (player_pos, box1_pos, box2_pos)

Initial state ((3,1), (1,3), (3,2))

Step 2: Actions:- The four possible actions are MoveUp, Move Down, Move Left, Move Right.

Step 3: Transition model

i) RESULT((3,1),(1,3),(3,2), MoveRight) = ((3,2),(1,3),(3,3))

ii) RESULT((3,2),(1,3),(3,3), Move Right) = ((3,3),(1,3),(3,4))

iii) RESULT((3,3),(1,3),(3,4), Move Up) = ((2,3),(1,3),(3,4))

iv) RESULT((2,3),(1,3),(3,4), MoveRight) = ((2,4),(1,3),(3,4))

v) RESULT((1,4),(1,3),(3,4), Move Up) = ((1,4),(1,3),(3,4))

vi) RESULT((1,4),(1,3),(3,4), Move Left) = ((1,3),(1,2),(3,4))

vii) RESULT((1,3),(1,2),(3,4), Move Left) = ((1,2),(1,1),(3,4))

Step 4: Action Cost

If the player moves to an empty square, cost = 1

If the player pushes a box, cost = 2

i) ACTION COST((3,1),(2,3)), Move Right, ((3,2),(1,3),(3,3)) = 2

ii) ACTION COST((3,2),(1,3),(3,3)), Move Right, ((3,3),(1,3),(3,4)) = 2

iii) ACTION COST((3,3),(1,3),(3,4), Move Up, (2,3),(1,3),(3,4)) = 1

iv) ACTION COST = 1

v) ACTION COST = 1

vi) ACTION COST = 2

vii) ACTION COST = 2

The cumulative cost of solving the problem = 11

Step 5: Goal Test: (1) (1,3) = (1,1), Box1 position is same as target 1 position (2) (3,2) = (3,4), Box2 position is same as target 2 position, Hence Goal test is PASS.

(3)

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	#	.	.	.	.	.	·2·	#
2	#	.	.	~~2~~	.	.	.	#
3	#	.	.	.	.	.	2	#
4	#	.	1	.	.	.	.	#
5	#	.	.	.	P	.	.	#
6	#	.	·1·	.	.	.	.	#
7	#	#	#	#	#	#	#	#

Formulate and Solve the Sokoban problem in 6X 6 grid shown in the figure. P denotes player position, 1 and 2 are the box positions, ·1· and ·2· are the goal positions of the respective boxes.

Step 1 - The initial state is given by (Player-Pos, Box-Pos)

Initial state = [(5,4), { (4,2), (3,6) }]

Step 2 - Actions - The agent can perform four actions : Move Up, Move Down, Move Left, Move Right

Step 3 and 4 - Transition model and Action Cost

Command	COST
RESULT ((5,4), {(4,2),(3,6)}, Move Right = (5,5), (4,2), (3,6))	1
RESULT ((5,5), (4,2), (3,6)}, Move Right = (5,6), (4,2), (3,6)	1
RESULT ((5,6), (4,2), (3,6)}, Move ~~Right~~ UP = (4,6), (4,2), (3,6)	1
RESULT ((5,6), (4,2), (3,6)}, Move UP = (3,6), (4,2), (3,6)	2
RESULT ((3,6) (4,2), (3,6)}, Move UP = (2,6), (4,2), (1,6)	2
RESULT ((2,6), (4,2), (1,6)}, Move Down = (3,6), (4,2), (1,6)	1
RESULT ((3,6), (4,2), (1,6)}, Move Left = (3,5), (4,2), (1,6)	1
RESULT ((3,5), (4,2) (1,6)} Move Left = (3,4), (4,2), (1,6)	1
RESULT ((3,4), (4,2), 1,6)} Move Left = (3,3), (4,2), 1,6)	1
RESULT ((3,3), (4,2), (1,6)} Move Left = (3,2), (4,2), (1,6)	1
RESULT ((3,2), (4,2), (1,6)} Move Down = (4,2), (5,2), 1,6)	2
RESULT ((4,2), (5,2), (1,6)} Move Down = ((5,2), (6,2), (1,6)	2

Cumulative Cost of solving the problem = 16

Step 5 - Goal Test

Box_Pos = ((6,2), (1,6)), Target_Pos = ((6,2), (1,6))

The two sets are identical and hence the goal test is PASS.

Deadlock conditions in Sokoban Problem:

(1) Corner Deadlock:

	0	1	2
0	#	#	#
1	#	$	.
2	#	P	.

A box pushed into a corner where two adjacent walls meet becomes trapped. Here player can only push the box at (1,1) and cannot pull, ~~the~~ But it is blocked by walls at (0,1) and (1,0). None of the move can extract it from there, hence it is ~~corn~~ a corner deadlock condition.

(2) Freeze Deadlock

	0	1	2
0	#	#	#
1		$	$
2	P	#	.

Two boxes pushed side by side against a wall become mutually frozen. Box at (1,1) cannot move right (blocked by box 2), up (blocked by wall. Neither of boxes can be repositioned as it will move to a corner or wall making deadlock.

(3) Wall Deadlock

	0	1	2
0	#	#	#
1	#	•	•
2	#	$	P

A box positioned ~~on a~~ adjacent to wall and if a further push might lead to corner or against another wall, box cannot reach the goal. At position (2,1), box can only slide horizontally along the wall row 2. If the target does not exist along this segments it results in deadlock.

(4) Internal Blockage Deadlock:

	0	1	2	3	4
0	#	#	#	#	#
1	#	$	·2·	.	#
2	#	#	$	#	#
3	#	·1·	.	P	#
4	#	#	#	#	#

It occurs when one or more boxes block access to an area containing other boxes, preventing from reaching the target. Here box at (1,1) in the upper chamber needs to reach (3,1) at lower chamber. The only connection between chambers at (2,2) is blocked by another box at (2,2).