Sokoban: Search in a complex domain

Yann Chazallon, Nicolas Dossou-Gbété, Tony Chan Ki Hong and Michal Staniaszek

October 19, 2013

Table of Contents

Problem: Sokoban

- A puzzle game which written in 1981 by Hiroyuki Imabayashi
- Control an agent to push boxes onto goal locations
- PSPACE-complete single agent search problem

Why is it interesting

- It can be applied in real-life situation
- High Branching factor: 4N where N = number of box
- Depth of the search tree can be infinity

Table of Contents

Map Representation

- Wall: # Player: @ Player in goal: +
- Goal: . Box: $ Box in goal: *
- Recieve map as a txt file with symbols representing different objects
- Static Objects in Static Board
- Dynamic Objects + Static Objects in Board

Deadlock Detection

- Player cannot pull a box, may push box into lose state
- Static Lock : push the box to the goal even without existance of other boxes
- Dynamic Lock : some boxes blocking in the middle and impossible to remove the blocking box

Search Method

- A*
- Best First
- BFS

Heuristic Method

- Mahattan Distance
- Static Cost Evaluation
- Minimum Matching

Table of Contents

Object-Oriented Programming

- Board class - Stores the state of the board and methods to change Board State
- Search class - Allow bi-directional search
- SearchNode class

Table of Contents

Search Method	Time limit		
	5 sec	11 sec	15 sec
A*	12	15	16
Best First	56	60	64
Bi-directional Best First	76	81	82
Bi-directional A*	39	41	43

Table : The number of test cases solved out of 100 available in the offline test set, running various search implementations with the manhattan distance heuristic. Note the lack of large differences in the number of solved maps with different time limits.

Search method	Avg expanded	Avg opened	Avg rejected
A*	9149	33382	17929
Best First	10741	40473	19499
Bi-directional Best First	3801	12727	4136
Bi-directional A*	4591	16462	4340

Table : Some statistics for the search nodes, and the average solution time in the tests in Table 1. Expanded nodes are those nodes whose successors have been generated. Opened nodes is the total number of nodes placed onto the open list. Rejected nodes are successors not added to the open list because they would generate locked states.

Table of Contents

Things that we can do better

- Members may not fully understand the code
- Didn't meet too often
- Only have a vague idea

Initial Idea

- Use motion of the player as the state expansion
- A Search Implementation

Current Method

- More efficent equality check
- improved heuristic - minimum matching heuristic

What will we do differently if we do again

- Start with simple implementation that worked
- A better heuristic can be used