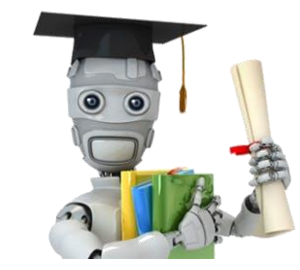
AI Agent to Play Sokoban Video Games

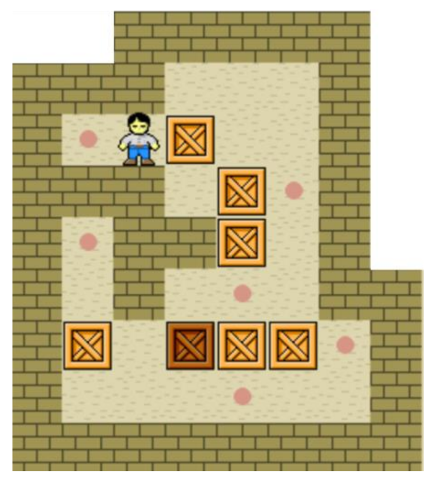
Tzu-Ling Liu (tzuling@stanford.edu) and Zhaoxuan Wang (wangzx@stanford.edu)



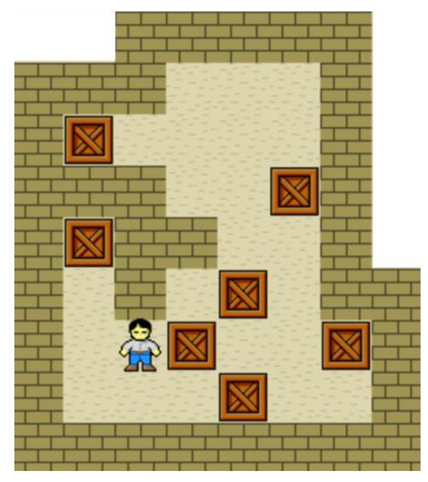
I. Motivation

Among the various applications of Artificial Intelligence (AI), AI games where AI is used to generate non-player characters similar to human-like intelligence has expanded greatly. Hence, in this report, we are trying to develop an AI agent to solve a Japanese video games, Sokoban, whose goal is to move all the boxes to the storage locations.

Initial state



goal state



II. State definition

(1) action: the action that was used to generate this state
(2) cost: the cost (an integer) of getting to this state.
(3) parent: the parent state from which this state was generated by applying the action
(4) map information: the current location of agent and boxes. The definition of the objects in the map are as follows:

- '#': walls and obstacles
- '&': agent
- '*': storage location
- '@': box that is not at storage location
- '$'': box that is at storage location

```
# # # # # #
# # #     #
#   @ *   #
#   & $   #
#       # #
# # # # # #
```

III. Problem definition

- s_{start} (starting state): action = "START", cost = 0, parent = None, map information (e.g. agent, box, storage, obstacle locations...etc)
- Actions(s): possible actions (UP, DOWN, LEFT, RIGHT)
- Cost(s, a): action cost (add 1 for every move)
- Succ(s, a): successor (the state after valid move)
- IsEnd(s): reached end state? (end state: all the boxes are moved to the storage)

IV. Challenges & Approaches

Challenges

- Huge state space
- Definition of heuristic function
- Initiation of a game board with a valid solution
- Definition of the successor state
- Prune method for time efficiency

Approaches

- Depth-first search (DFS)
- Breadth-first search (BFS)
- Uniform cost search (UCS)
- A*, heuristic functions:
  - h_1: shortest Euclidean distances from any box to any storage
  - h_2: shortest Manhattan distances from any box to any storage
  - h_3: average of the Euclidean distances from all boxes to all storages
  - h_4: average of the Manhattan distances from all boxes to all storages
  - h_5: a numeric value that estimates the obstacle condition of boxes that not at storage locations.

V. Results & Discussion

i. Comparison of 4 approaches

# boxes		2 boxes	3 boxes	4 boxes
Map - initial state		# # # # # # # # # # # @ * # # @ * # # & # # # # # # # #	# # # # # # # # * # & # # # @ # # * @ # # # @ # # * # # # # # # # # #	# # # # # # # # # * * * # # # # * @ # # @@@ # # # # # & # # # # # # # # #
DFS	Time	10.4 sec	322.6 sec	> 2 hr
	Cost	5	75	NA
	# states	3.7×10^5	7.9×10^6	NA
BFS	Time	< 0.1 sec	0.6 sec	133.8 sec
	Cost	5	41	55
	# states	105	4.0×10^4	8.0×10^6
UCS	Time	< 0.1 sec	1.5 sec	189.6 sec
	Cost	5	41	55
	# states	81	4.0×10^4	8.0×10^6
A* (h_2)	Time	< 0.1 sec	0.6 sec	96.5 sec
	Cost	5	41	55
	# states	26	3.4×10^4	2.8×10^6

- DFS explores more states than other approaches, resulting in longer resolve time
- BFS and UCS show similar performance, because every action (move) gets the same cost
- A* is the best approach with consistent heuristic function.
- Generally, heuristic functions using Manhattan distance are better than using Euclidean distance because the action/move is based on Manhattan distance.
- For simple problems, h_1 and h_2 show the best results because calculation for h_5 is time expensive, while h_5 has best performance for complicated problems with consideration of obstacle circumstance.

ii. Comparison of different heuristic functions

Map - initial state		# # # # # # # # # * * * # # # # * @ # # @@@ # # # # # & # # # # # # # # #	# # # # # # # # # # # # # # @ # # * * * $ @& # # # @ # # # # # # # # # # # # # #	# # # # # # # # * # # # * # # @&@ # # # # @ @ # # * # # # * # # # # # # # #
h_1	Time	130.2 sec	184.3 sec	0.8 sec
	Cost	55	46	20
	# states	3.7×10^6	4.0×10^6	2.9×10^4
h_2	Time	96.5 sec	130.4 sec	0.5 sec
	Cost	55	46	20
	# states	2.8×10^6	3.5×10^6	1.9×10^4
h_3	Time	239.2 sec	278.4 sec	5.3 sec
	Cost	55	46	20
	# states	6.7×10^6	7.5×10^6	9.6×10^4
h_4	Time	204.5 sec	225.9 sec	1.7 sec
	Cost	55	46	20
	# states	6.3×10^6	7.0×10^6	7.8×10^4
h_5	Time	3.0 sec	4.6 sec	1.3 sec
	Cost	55	46	20
	# states	1.6×10^5	2.4×10^5	6.4×10^4

VI.Reference

1. Suleman, M., et al. Int. J. Adv. Comput. Sci. Appl. (2017)
2. Anthony, T., et al. IEEE Trans. Comput. Intell. AI Games (2014)
3. Luzhnica, G., et al. in Adjunct Proceedings of the 2016 IEEE International Symposium on Mixed and Augmented Reality, ISMAR-Adjunct 2016 (2017).