Sokoban, Solved: Modeling Sokoban Puzzles as Search Problems

Matthew Days and Yangzi Jiang
Davidson College Departments of Mathematics and Computer Science

1.Overview

- Create a program that uses different search algorithms to solve Sokoban
- Search algorithms like IDA* & MCTS have numerous real-world applications
  - AI planning, pathfinding in video games, autonomous vehicles
- Sokoban represents a complex domain in which to evaluate these algorithms
- Solving Sokoban is difficult - it is a NP-hard, PSPACE-complete problem

Figure 1: Initial State of the original level-1 Sokoban Puzzle

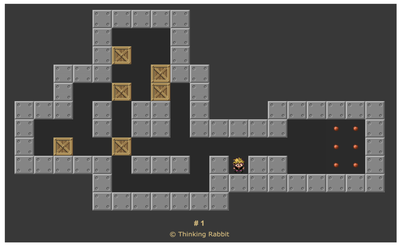
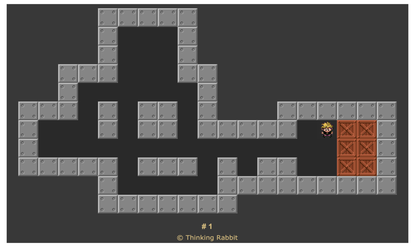


Figure 2: Solved State of the original level-1 Sokoban Puzzle



2. Sokoban Features

- A puzzle game created in 1981 by Hiroyuki Imabayashi published by Thinking Rabbit
- Sokoban is a Japanese word meaning “warehouse keeper”
- Objective: Players push boxes vertically or horizontally into designated goal spaces
- NP-hard, PSPACE-complete
  - At least as difficult as NP-complete problems and requires large memory spaces

Deadlocks: Need to detect a deadlocks as they would render puzzle unsolvable

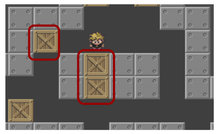


Figure 3: Two examples of deadlocks due to frozen boxes.

Tunnels: Model consecutive pushes along the tunnel as one move to reduce complexity

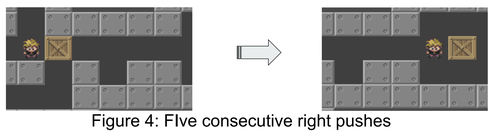
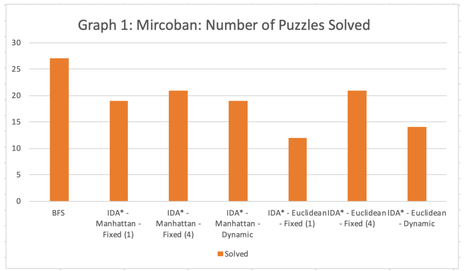


Figure 4: FIve consecutive right pushes

4. Results

Table 1: Comparisons of Performances by BFS and IDA* on 30 *Microban* Puzzles

	Solved in 2 Mins			Nodes Generated		Nodes in Fringe		Nodes Explored		Duration (secs)		Solution Length	
Method - Heuristics - Bound Increment	Solved	Total	Solved %	Total	Avg	Total	Avg	Total	Avg	Total	Avg	Total	Avg
BFS	27	30	90.00%	16413	607.89	3096	114.67	70217	2600.63	347.99	12.89	1304	48.30
IDA* - Manhattan - Fixed (1)	19	30	63.33%	9142	481.16	2163	113.84	73463	3866.47	313.71	16.51	678	35.68
IDA* - Manhattan - Fixed (4)	21	30	70.00%	11117	529.38	1814	86.38	52067	2479.38	254.65	12.13	901	42.90
IDA* - Manhattan - Dynamic	19	30	63.33%	10135	533.42	2241	117.95	65918	3469.37	296.81	15.62	700	36.84
IDA* - Euclidean - Fixed (1)	12	30	40.00%	6030	502.50	1877	156.42	54073	4506.08	220.85	18.40	391	32.58
IDA* - Euclidean - Fixed (4)	21	30	70.00%	11569	550.90	1942	92.48	61388	2923.24	305.47	14.55	866	41.24
IDA* - Euclidean - Dynamic	14	30	46.67%	8037	574.07	2710	193.57	67637	4831.21	288.99	20.64	477	34.07



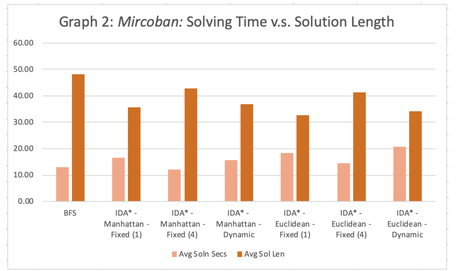
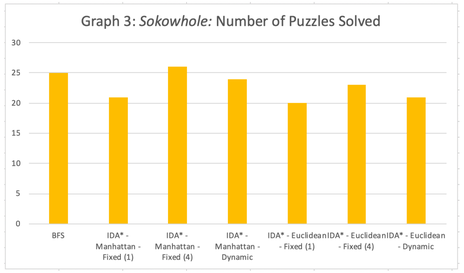
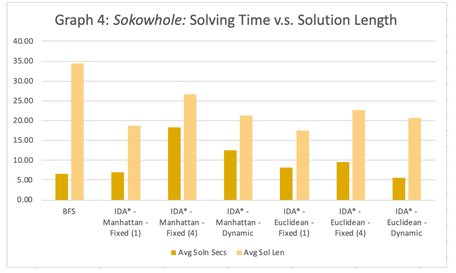


Table 2: Comparisons of Performances by BFS and IDA* on 30 *Sokowhole* Puzzles

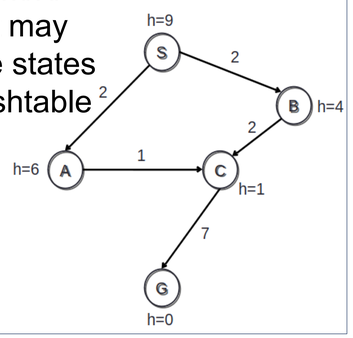
	Solved in 2 Mins			Nodes Generated		Nodes in Fringe		Nodes Explored		Duration (secs)		Solution Length	
Method - Heuristics - Bound Increment	Solved	Total	Solved %	Total	Avg	Total	Avg	Total	Avg	Total	Avg	Total	Avg
BFS	25	30	83.33%	63162	2526.48	335	13.40	18717	748.68	163.44	6.54	859	34.36
IDA* - Manhattan - Fixed (1)	21	30	70.00%	6263	298.24	1719	68.76	34715	1388.60	173.44	6.94	470	18.80
IDA* - Manhattan - Fixed (4)	26	30	86.67%	28666	1102.54	5564	222.56	69785	2791.40	459.18	18.37	666	26.64
IDA* - Manhattan - Dynamic	24	30	80.00%	13765	573.54	3600	144.00	55687	2227.48	314.18	12.57	532	21.28
IDA* - Euclidean - Fixed (1)	20	30	66.67%	3789	189.45	2621	104.84	43976	1759.04	206.20	8.25	436	17.44
IDA* - Euclidean - Fixed (4)	23	30	76.67%	11278	490.35	2393	95.72	40416	1616.64	237.53	9.50	569	22.76
IDA* - Euclidean - Dynamic	21	30	70.00%	7602	362.00	1471	58.84	24548	981.92	139.05	5.56	519	20.76





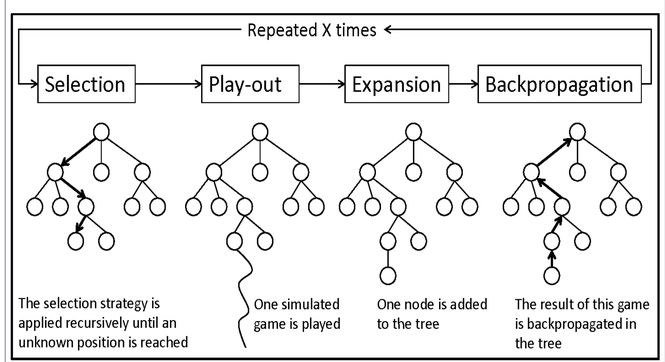
3a. Iterative Deepening A* (IDA*)

- Graph search algorithm that uses heuristic function as guide
- Trade off between time and space:
  - IDA* Saves storage space with DFS & pruning
    - Addresses the high space-complexity issue with with Sokoban.
  - Vanilla IDA* is more time-costly as it may revisit the same states
    - Transition hashtable to record visited states



3b. Monte Carlo Tree Search (MCTS)

- MCTS does not require domain-specific knowledge, only legal moves and goal conditions (this lack of knowledge causes a performance hit)



5. Conclusion

- Modeled Sokoban as a search problem to test various search algorithms
- BFS runtime outperformed IDA* runtime
  - Result of heuristic calculation, which allows for less nodes generated
- IDA*: move-optimal; BFS not
- As puzzle difficulty increases, performance declines dramatically
- MCTS too slow to solve Sokoban without optimizations & domain dependent info
- Further exploration:
  - More domain-specific info: improved deadlocks, PI-corrals, goal & tunnel macros
  - Improved heuristics: Minmatching (minimum cost perfect matching in bipartite graph of boxes and goals)

6. Acknowledgements & Sources

- We would like to thank Dr. Raghu Ramanujah for his guidance on this project
- Game figures: Original puzzle game written by Hiroyuki Imabayashi; © 1982 by THINKING RABBIT Inc. JAPAN; retrieved from sokoban.info