National University of Singapore
School of Computing
CS3243 Introduction to AI

Tutorial 2: Uninformed Search

Issued: January 20, 2021 Discussion in: Week 4

Important Instructions:

- **Assignment 2** consists of **Question 4** from this tutorial.
- Your solutions for this tutorial must be TYPE-WRITTEN.
- You are to submit your solutions on LumiNUS by **Week 3, Saturday, 2359 hours**.
- Refer to LumiNUS for submission guidelines

Note: you may discuss the content of the questions with your classmates (outside your group). But each group should work out and write up ALL the solutions individually. If caught plagiarising, you may be awarded an F Grade for the module.

---

1. Sudoku is a popular number puzzle that works as follows: we are given a 9×9 square grid; some squares have numbers, while some are blank. Our objective is to fill in the blanks with numbers from $1 - 9$ such that each row, column and the highlighted 3×3 squares contain no duplicate entries (see Figure 1). Solving Sudoku puzzles is often modeled as a CSP (which we will cover later in the course). Consider, however, the problem of *generating* Sudoku puzzles. Consider the following procedure: start with a completely full number grid (see Figure 2), and iteratively make some squares blank. We continue blanking out squares as long as the resulting puzzle can be completed in only one way.

   Complete the following:

   - Give the representation of a state in this problem;
   - Using the state representation defined above, specify the initial state and goal state.
   - Define its actions; and
   - Using the state representation and actions defined above, specify the transition function T (i.e., when each of the actions defined above is applied to a current state, what is the resulting next state?).

2	5			3		9		1
	1				4			
4		7				2		8
		5	2					
				9	8	1		
	4				3			
			3	6			7	2
	7							3
9		3				6		4

Figure 1: A simple Sudoku Puzzle

2	5	8	7	3	6	9	4	1
6	1	9	8	2	4	3	5	7
4	3	7	9	1	5	2	6	8
3	9	5	2	7	1	4	8	6
7	6	2	4	9	8	1	3	5
8	4	1	6	5	3	7	2	9
1	8	4	3	6	9	5	7	2
5	7	6	1	4	2	8	9	3
9	2	3	5	8	7	6	1	4

Figure 2: Solution to the Sudoku puzzle in Figure 1.

2. Consider the graph shown in Figure 3. Let S be the initial state and G be the goal state. The cost of each action is as indicated.

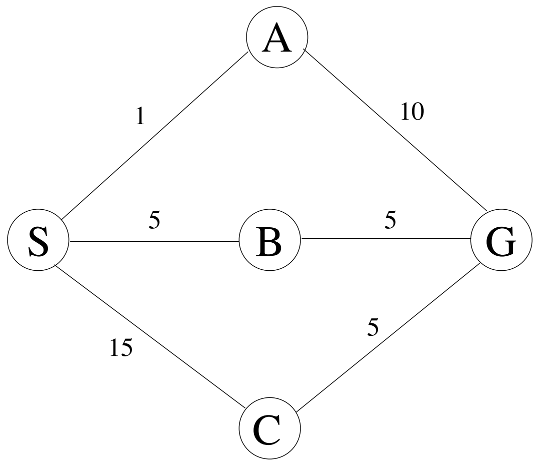

Figure 3: Graph of routes between S and G.

   (a) Trace the application of the uniform-cost search algorithm on this graph.
   (b) When A generates G, which is the goal with a path cost of 11, why doesn't the algorithm halt and return the search result since the goal has been found?

3. Prove that the Uniform-cost Search (UCS) algorithm is optimal when all step costs are greater than some small positive constant ϵ.

4. We have seen various search strategies in class, and analyzed their worst-case running time. Prove that *any deterministic search algorithm* will, in the worst case, search the entire state space. More formally, prove the following theorem

   Theorem 1. *Let $\mathcal{A}$ be some complete, deterministic search algorithm. Then for any search problem defined by a finite connected graph $G = \langle V, E\rangle$ (where V is the set of possible states and E are the transition edges between them), there exists a choice of start node s_0 and goal node g so that $\mathcal{A}$ searches through the entire graph G.*