National University of Singapore
School of Computing
IT5005 Artificial Intelligence

Uninformed and Informed Search

1. Consider the grid problem shown in Fig. 1. The grid extends to infinity in all directions. The agent can only move along the grid lines. Action *East*, *West*, *North*, and *South* takes the agent from state (r, c) to $(r, c+1)$, $(r, c-1)$, $(r+1, c)$, $(r-1, c)$, respectively. Each action costs one unit. The agent is initially assumed to be at state $(0, 0)$, and objective is to reach state $(2, 2)$. All action costs are same and equal to 1.

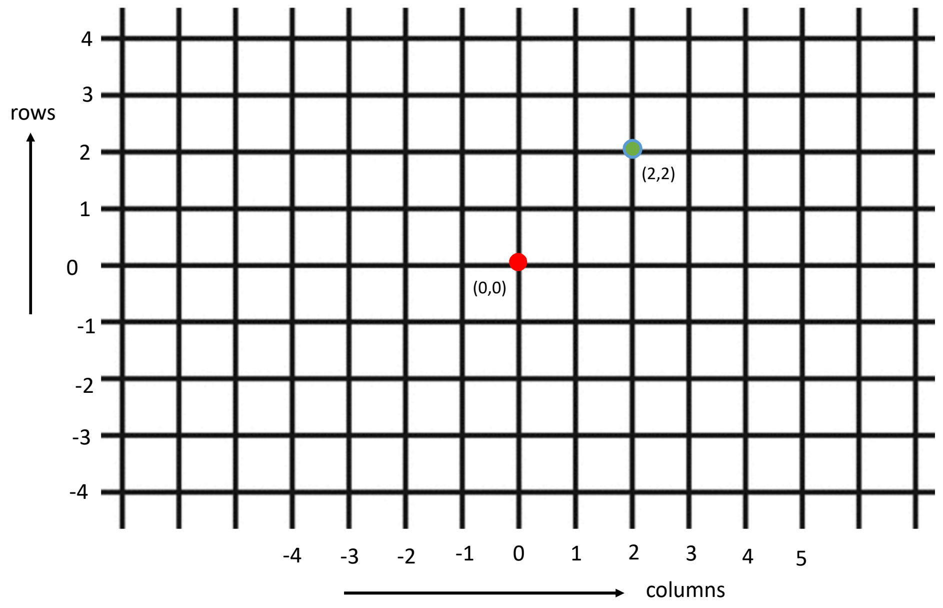

Figure 1: State is represented as (r, c), where r and c represent the row and column, respectively. The agent can move one step along the grid in *East*, *West*, *North*, and *South* directions; for example action *East* takes the agent from initial state $(0, 0)$ to $(0, 1)$; When choosing an action, ties are always broken in the order *East*, *West*, *North*, and *South*. For example, if you need to choose between *East* and *North*, first choose *East* and then *North*.

(a) Which algorithm would you recommend: tree search/graph search? Provide justification.

(b) Identify whether the following algorithms are complete and optimal. Provide the rationale for each answer.

i. Breadth-First Search
ii. Depth-Limited Search
iii. Depth-First Search
iv. Iterative Deepening Search

Solution

(a) Graph search is preferred as the tree search leads to infinite loops.

(b) i. The action-cost is 1; hence breadth-first search is complete and optimal.
ii. Depth-limited search is complete if the depth of the search is set appropriately. The DLS is optimal if the depth-limit is set as 3. However, we cannot generalize this result.
iii. Depth-first search is not guaranteed to be complete as the grid is extended to infinite.
iv. The action-cost is 1; hence iterative deepening search is complete and optimal.

2. An explicit state space graph is shown in Fig. 2. Several heuristics (i.e., estimated costs to reach the goal state from a node) are presented in Table 1. As an example, the column corresponding to h_5 can be interpreted as $h_5(A) = 5$, $h_5(B) = 7$, $h_5(C) = 2$, and $h_5(D) = 0$.

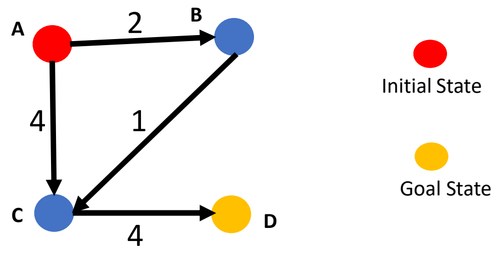

Figure 2: Explicit state space graph

Table 1: Heuristics for the state space graph shown in Fig. 2

Node	h_1	h_2	h_3	h_4	h_5
A	0	7	7	5	5
B	0	5	4	3	7
C	0	4	1	2	2
D	0	0	0	0	0

(a) Comment on admissibility and consistency of the heuristic functions $h_i, i = \{1, \ldots, 5\}$.

(b) Find the path from initial state to goal state using uniform-cost search, A* search, and greedy best first search algorithms. Use the heuristics in Table 1 when needed.

Solution

(a) **Checking for Admissibility:** A heuristic is admissible if estimated cost at each node to reach the goal state is less than true cost to reach the goal state.

Let us denote the actual cost of reaching the goal by h^*. They can be evaluated as:

$$\begin{aligned} h^*(A) &= 7 \\ h^*(B) &= 5 \\ h^*(C) &= 4 \\ h^*(D) &= 0 \end{aligned}$$

It can be verified from Table 1 that h_1, h_2, h_3, and h_4 are admissible. h_5 is not admissible, because $h_5(B) > h^*(B)$.

Checking for Consistency: A heuristic is consistent if at each node n, the following triangle inequality is satisfied for all neighbors $n' \in \mathcal{N}(n)$ of node n,

$$\begin{aligned} & h(n) \leq h(n') + c(n, a, n'), \forall n, n' \in \mathcal{N}(n) \\ \implies \quad & h(n) - h(n') \leq c(n, a, n'), \forall n, n' \in \mathcal{N}(n) \end{aligned}$$

where $\mathcal{N}(n)$ indicate the set of neighbors of node n, $c(n, a, n')$ indicates the cost of reaching node n' from node n with action a. For the given problem, $\mathcal{N}(A) = \{B, C\}$, $\mathcal{N}(B) = \{C\}, \mathcal{N}(C) = \{D\}$. Therefore, the conditions for consistency of heuristic h_1 can be written as:

$$\begin{aligned} h_1(A) - h_1(B) = 0 \leq C(A, B) \\ h_1(A) - h_1(C) = 0 \leq C(A, C) \\ h_1(B) - h_1(C) = 0 \leq C(B, C) \\ h_1(C) - h_1(D) = 0 \leq C(C, D) \end{aligned}$$

where $C(A, B), C(A, C), C(B, C), C(C, D)$ are the action costs. For example, $C(A, B)$ indicates the action cost to move from vertex A to vertex B. Therefore, h_1 is consistent. Similarly, it can be verified that h_2 and h_4 are consistent.

On the other hand, the heuristics h_3 and h_5 are not consistent. They violate the triangle inequality as shown below:

$$h_3(A) - h_3(B) = 3 > C(A, B)$$
$$h_5(B) - h_5(C) = 5 > C(B, C)$$

Note: The triangle inequality must be satisfied at each state in the graph. If it is violated at any of the states, the heuristic is inconsistent.

(b) **Uniform-Cost Search** The trace of the algorithm and search tree for UCS are shown in Table 2 and Fig. 3, respectively. From Fig. 3, path identified by the uniform-cost search is $A - B - C - D$ with a path-cost of 7.

Table 2: Uniform-Cost Search Algorithm

#	Node	Is-Goal(Node)	s	Frontier	Reached
0	A_0			$[A_0]$	$\{A_0\}$
1	A_0	No	B_2	$[B_2]$	$\{A_0, B_2\}$
2			C_4	$[B_2, C_4]$	$\{A_0, B_2, C_4\}$
3	B_2	No	C_3	$[C_4, C_3]$	$\{A_0, B_2, \mathbf{C_3}\}$
4	C_3	No	D_7	$[C_4, D_7]$	$\{A_0, B_2, C_3, D_7\}$
5	C_4	No	D_8	$[D_7]$	$\{A_0, B_2, C_3, D_7\}$
6	D_7	Yes			

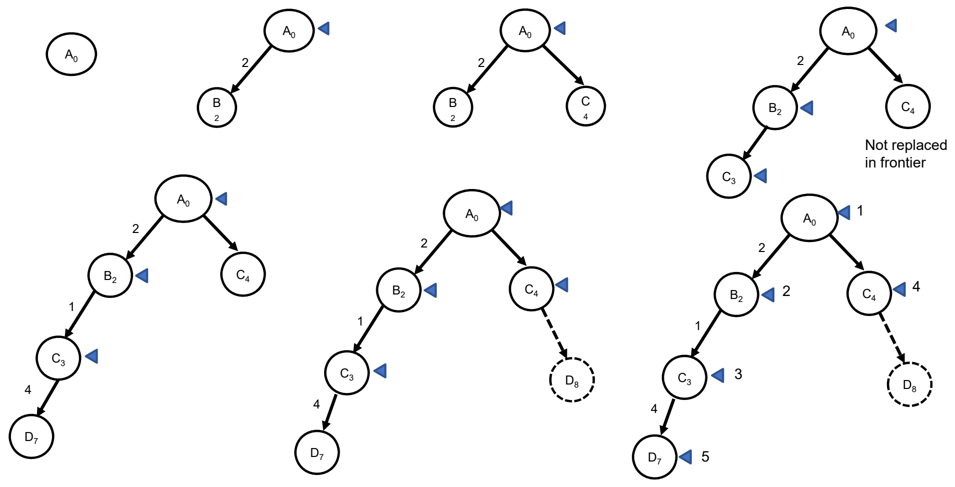

Figure 3: Search Tree for Uniform-Cost Search; the node with state D_8 is generated, but it will not be added to the frontier

A* Search with heuristic h_1: This heuristic is both admissible and consistent. The trace of the algorithm and search tree for A* search with heuristic h_1 are shown in

Table 3 and Fig. 4, respectively. The search tree of A* algorithm with heuristic h_1 is the same as that of UCS. It is because heuristic cost is zero for all nodes. The output of is $A - B - C - D$ with a path-cost of 7.

Table 3: A* Search for heuristic h_1; $h_1(A) = h_1(B) = h_1(C) = h_1(D) = 0$

#	Node	Is-Goal(Node)	s	Frontier	Reached
0	A_{0+0}			$[A_{0+0}]$	$\{A_{0+0}\}$
1	A_{0+0}	No	B_{2+0}	$[B_{2+0}]$	$\{A_{0+0}, B_{2+0}\}$
2			C_{4+0}	$[B_{2+0}, C_{4+0}]$	$\{A_{0+0}, B_{2+0}, C_{4+0}\}$
3	B_{2+0}	No	C_{3+0}	$[C_{4+0}, C_{3+0}]$	$\{A_{0+0}, B_{2+0}, \mathbf{C_{3+0}}\}$
4	C_{3+0}	No	D_{7+0}	$[C_{4+0}, D_{7+0}]$	$\{A_{0+0}, B_{2+0}, C_{3+0}, D_{7+0}\}$
5	C_{4+0}	No	D_{8+0}	$[D_{7+0}]$	$\{A_{0+0}, B_{2+0}, C_{3+0}, D_{7+0}\}$
6	D_{7+0}	Yes		$[]$	$\{A_{0+0}, B_{2+0}, C_{3+0}, D_{7+0}\}$

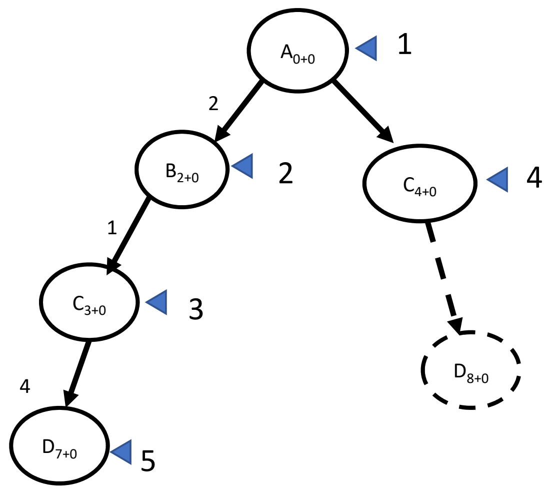

Figure 4: A* search tree with heuristic h_1

A* Search with heuristic h_2: This heuristic is both admissible and consistent. Moreover, the heuristic cost is also same as the actual cost of reaching the goal state. The trace of the algorithm and search tree for A* search with heuristic h_2 are shown in Table 4 and Fig. 5, respectively. Note that the search tree of A* search with the heuristic h_2 is different compared to that of UCS. The A* search algorithm does not explore the node C_{4+4}. The path identified by the A* search tree is the same as that of UCS, i.e.,

$A - B - C - D$ with a path-cost of 7. The A* search algorithm does not explore nodes outside the optimal path as the perfect estimate of actual cost is known at each state.

Table 4: A* Search for heuristic h_2; $h_2(A) = 7, h_2(B) = 5, h_2(C) = 4, h_2(D) = 0$

#	Node	Is-Goal(Node)	s	Frontier	Reached
0	A_{0+7}			$[A_{0+7}]$	$\{A_{0+7}\}$
1	A_{0+7}	No	B_{2+5}	$[B_{2+5}]$	$\{A_{0+7}, B_{2+5}\}$
2			C_{4+4}	$[B_{2+5}, C_{4+4}]$	$\{A_{0+7}, B_{2+5}, C_{4+4}\}$
3	B_{2+5}	No	C_{3+4}	$[C_{4+4}, C_{3+4}]$	$\{A_{0+7}, B_{2+5}, \mathbf{C_{3+4}}\}$
4	C_{3+4}	No	D_{7+0}	$[C_{4+4}, D_{7+0}]$	$\{A_{0+7}, B_{2+5}, C_{3+4}, D_{7+0}\}$
5	D_{7+0}	Yes		-	-

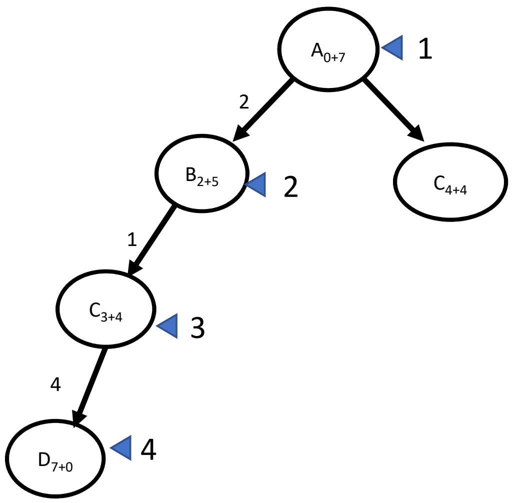

Figure 5: A* search tree with heuristic h_2

A* Search with heuristic h_3: This heuristic is admissible but not consistent. The trace of the algorithm and the search tree for the A* search with heuristic h_3 are shown in Table 5 and Fig. 6, respectively. The path identified by the A* search algorithm is optimal (due to admissibility of the heuristic) and it is the same as that of UCS, i.e., $A - B - C - D$ with a path-cost of 7.

Now, let us investigate the impact of inconsistent heuristic. First, the state C (node C_{4+1}) is reached through a suboptimal path, that is, $A - C$. Consequently, the child nodes of this node are added to the search tree. The same state C is also reached later via the optimal path (node C_{3+1}), i.e., $A - B - C$. The child nodes of the state C_{3+1} on the optimal path are promising, as their path costs would be lower than those generated through suboptimal path, and they would also be added to the search tree. That means we are adding same states to search tree multiple times, leading to additional time and memory complexity.

However, if the heuristic is consistent, the state along the optimal path would be expanded first. Subsequently, even if the same state is reached via a suboptimal path, the child nodes would not be added to the search tree as their path costs would be greater than the path costs of the nodes generated via the node on the optimal path. Therefore, if the heuristic is consistent, the search is efficient.

Table 5: A* Search for heuristic h_3; $h_3(A) = 7, h_3(B) = 4, h_3(C) = 1, h_3(D) = 0$

#	Node	Is-Goal(Node)	s	Frontier	Reached
0	A_{0+7}			$[A_{0+7}]$	$\{A_{0+7}\}$
1	A_{0+7}	No	B_{2+4}	$[B_{2+4}]$	$\{A_{0+7}, B_{2+4}\}$
2			C_{4+1}	$[B_{2+4}, C_{4+1}]$	$\{A_{0+7}, B_{2+4}, C_{4+1}\}$
3	C_{4+1}	No	D_{8+0}	$[B_{2+4}, D_{8+0}]$	$\{A_{0+7}, B_{2+4}, C_{4+1}, D_{8+0}\}$
4	B_{2+4}	No	C_{3+1}	$[D_{8+0}, C_{3+1}]$	$\{A_{0+7}, B_{2+4}, \mathbf{C_{3+1}}, D_{8+0}\}$
5	C_{3+1}	No	D_{7+0}	$[D_{8+0}, D_{7+0}]$	$\{A_{0+7}, B_{2+4}, C_{3+1}, \mathbf{D_{7+0}}\}$
6	D_{7+0}	Yes		-	-

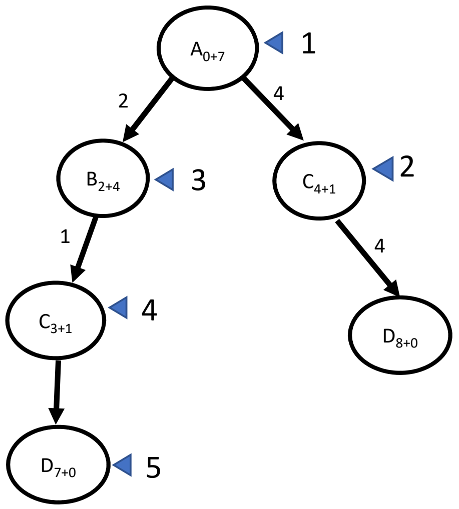

Figure 6: A* search tree with heuristic h_3; note that node with state D_{8+0} is added to the frontier and it is part of the search tree. If the heuristic is consistent, this node wouldn't in the search tree

A* Search with heuristic h_4: This heuristic is admissible and consistent. The trace of the algorithm and the search tree for the A* search with heuristic h_4 are shown in Table 6 and Fig. 7, respectively. It can be seen that the state C (node C_{3+2}) is first reached by the optimal path, that is, $A - B - C$. The same state C (node C_{4+2}) is reached via the suboptimal path (node C_{3+1}). The child nodes of this node are not added to the frontier as the same child node with lower path-cost is visited via optimal path. It shows that if the heuristic is consistent, the search is efficient. The path identified by the A* search tree is the same as that of UCS, i.e., $A - B - C - D$ with a path cost of 7.

Table 6: A* Search for heuristic h_4; $h_4(A) = 5, h_4(B) = 3, h_4(C) = 2, h_4(D) = 0$

#	Node	Is-Goal(Node)	s	Frontier	Reached
0	A_{0+5}			$[A_{0+5}]$	$\{A_{0+5}\}$
1	A_{0+5}	No	B_{2+3}	$[B_{2+3}]$	$\{A_{0+5}, B_{2+3}\}$
2			C_{4+2}	$[B_{2+3}, C_{4+2}]$	$\{A_{0+5}, B_{2+3}, C_{4+2}\}$
3	B_{2+3}	No	C_{3+2}	$[C_{4+2}, C_{3+2}]$	$\{A_{0+5}, B_{2+3}, \mathbf{C_{3+2}}\}$
4	C_{3+2}	No	D_{7+0}	$[C_{4+2}, D_{7+0}]$	$\{A_{0+5}, B_{2+3}, C_{3+2}, D_{7+0}\}$
5	C_{4+2}	No	D_{8+0}	$[D_{7+0}]$	$\{A_{0+5}, B_{2+3}, C_{3+2}, D_{7+0}\}$
6	D_{7+0}	Yes		-	-

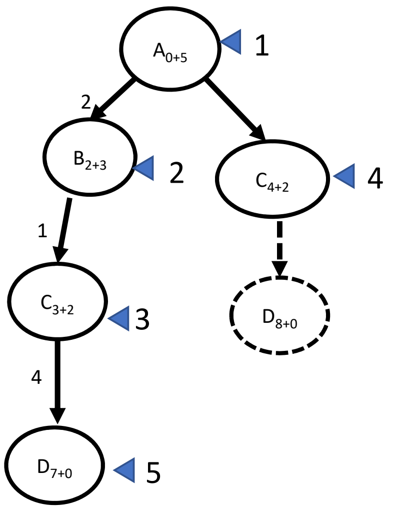

Figure 7: A* search tree with heuristic h_4

A* Search with heuristic h_5: This heuristic is not admissible and consistent. The trace of the algorithm and search tree for A* search with heuristic h_5 are shown in Table 7 and Fig. 8, respectively. The solution is $A - C - D$ with a path-cost of 8 and it is suboptimal.

Table 7: A* Search for heuristic h_5; $h_5(A) = 5, h_5(B) = 7, h_5(C) = 2, h_5(D) = 0$

#	Node	Is-Goal(Node)	s	Frontier	Reached
0	A_{0+5}			$[A_{0+5}]$	$\{A_{0+5}\}$
1	A_{0+5}	No	B_{2+7}	$[B_{2+7}]$	$\{A_{0+5}, B_{2+7}\}$
2			C_{4+2}	$[B_{2+7}, C_{4+2}]$	$\{A_{0+5}, B_{2+7}, C_{4+2}\}$
3	C_{4+2}	No	D_{8+0}	$[B_{2+7}, D_{8+0}]$	$\{A_{0+5}, B_{2+7}, C_{4+2}, D_{8+0}\}$
4	D_{8+0}	Yes	-	-	-

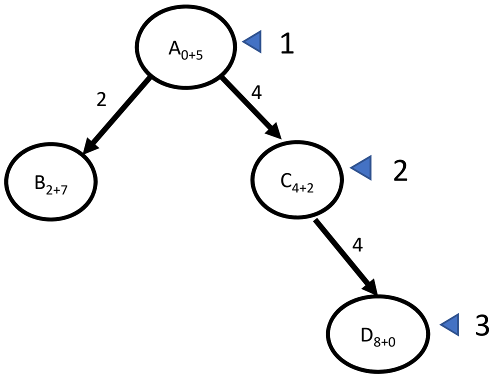

Figure 8: A* search tree with heuristic h_5

Greedy Best First Search:

The trace of greedy best first search (GBFS) algorithm with heuristic h_5 is shown in Table 8. The trace of the algorithm for other heuristics is omitted. The search trees of GBFS with all heuristics are shown in Fig. 9. It can be seen that GBFS yields suboptimal solution for all heuristics. For the heuristic h_1, the GBFS algorithm can provide an optimal solution, but is not guaranteed.

Table 8: Greedy Best First Search for heuristic h_5

#	Node	Is-Goal(Node)	s	Frontier	Reached
0	A_5			$[A_5]$	$\{A_5\}$
1	A_5	No	B_7	$[B_7]$	$\{A_5, B_7\}$
2			C_2	$[B_7, C_2]$	$\{A_5, B_7, C_2\}$
3	C_2	No	D_0	$[B_7, C_2, D_0]$	$\{A_5, B_7, C_2, D_0\}$
4	D_0	Yes	-	-	-

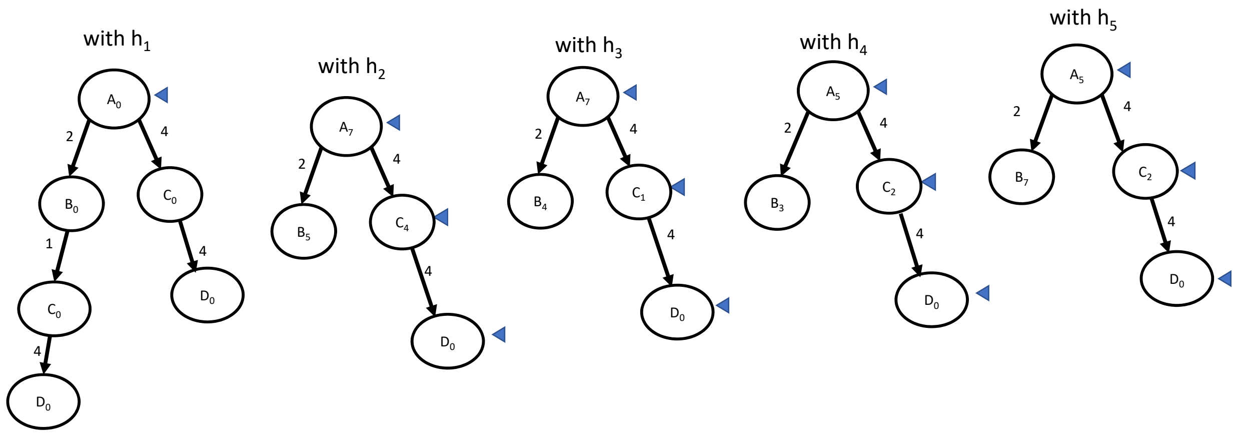

Figure 9: Search Tree for Greedy Best First Search

3. In informed search, we relax problem settings to enable the estimation of path-cost from a given state to a goal state. In an n-puzzle, assume that square B is blank; if square A is horizontally or vertically adjacent to square B, then a tile can move from square A to square B horizontally or vertically, respectively. Consider the following heuristic costs:

   - Heuristic h_1: number of misplaced tiles compared to goal state
   - Heuristic h_2: Manhattan distance of each tile to its position in goal state. It is defined as

$$h_2 = \sum_{i=1}^{n} (|x_{i,a} - x_{i,t}| + |y_{i,a} - y_{i,t}|) \tag{1}$$

     where $(x_{i,a}, y_{i,a})$ is the actual position of a tile and $(x_{i,t}, y_{i,t})$ is its target position. The top left position is (1,1) and the bottom right position is (3,3).

   (a) Show that both heuristics are admissible. Which heuristic can be a good estimate of the actual path-cost?

   (b) Calculate the heuristic costs for the initial state of 8-puzzle shown in the Fig. 10.

Solution:

(a) We can show that they're both admissible by considering two "relaxed problems", which are simplified versions of the game. The resulting optimal heuristic for such games will necessarily be admissible, since the relaxed problems allow for shortcuts.

Actual game:

A tile can move from A to B if A is adjacent to B and B is blank.

Relaxed game for heuristic h_1:

A misplaced tile can be placed directly into its correct position. In this case the optimal heuristic is h_1, since, if we have n misplaced tiles, we place these n tiles into their respective positions.

Relaxed game for heuristic h_2:

A misplaced tile can be moved to any horizontally or vertically adjacent tile regardless of whether it is empty.

The resulting optimal heuristic is h_2 Since both h_1 and h_2 are optimal for the relaxed problem, they are admissible for the actual problem.

As an aside, this is the main way to derive admissible heuristics.

Note that $h_2 \geq h_1$, i.e. h_2 dominates h_1 since version 2 does not allow for a direct placement. Hence the heuristic h_2 is better than h_1.

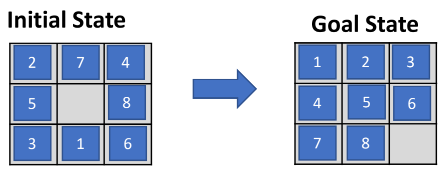

Figure 10: An 8-puzzle problem

(b) **Misplaced Tiles:**

$$h_1 = 8$$

Manhattan Distance:

Table 9: Calculation of Manhattan Distance for Initial State

Tile (i)	Target Position $(x_{i,t}, y_{i,t})$	Actual Position $(x_{i,a}, y_{i,a})$
1	(1,1)	(3,2)
2	(1,2)	(1,1)
3	(1,3)	(3,1)
4	(2,1)	(1,3)
5	(2,2)	(2,1)
6	(2,3)	(3,3)
7	(3,1)	(1,2)
8	(3,2)	(2,3)

$$\begin{aligned} h_2 &= \sum_{i=1}^{8} (|x_{i,a} - x_{i,t}| + |y_{i,a} - y_{i,t}|) \\ &= 3 + 1 + 4 + 3 + 1 + 1 + 3 + 2 = 18 \end{aligned}$$