Quiz Summary

Question 1

- Consider the following graph in Fig. 1, with start state S and goal state G. Edge costs are labeled on the edges, and heuristic values are given by the H values next to each state. In the search procedures below, break any ties alphabetically, so that if nodes on your fringe are tied in values, the state that **comes first alphabetically** is **expanded first**.

(1) What is the path returned by greedy graph search with the given heuristic?

(2) What is the path returned by A* graph search with the given heuristic?

(3) List the nodes in the order they are expanded by A* graph search in the above (2) procedure.

(4) Is this heuristic admissible? If not, find a minimal set of nodes that would need to have their values changed to make the heuristic admissible.

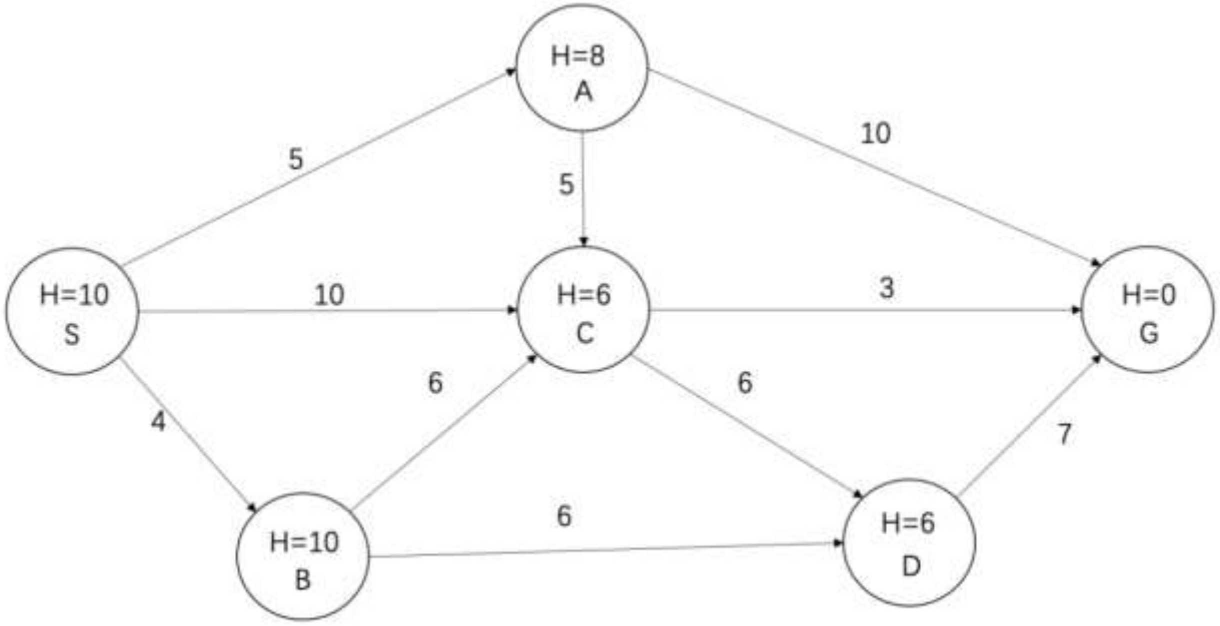


Fig. 1

Review heuristic search

Uninformed search

no

Domain knowledge

yes

Heuristic search

evaluation function

f(x)

diff from BFS

node x with the **lowest**
$f(x)$ is **expanded first**

$f(x) = h(x)$

Greedy Best-first Search(Greedy search)

- $g(x)$: *cost so far to reach x.*
- $\boxed{h(x)\text{: estimated cost to goal from } x.}$
- $f(x)$: *estimated total cost of path through x to goal.*

$A^* \xrightarrow{h(x)=0}$ UCS

$f(x) = g(x) + h(x)$

$f(x) = g(x)$

Heuristic function h is admissible if:
$\forall x \rightarrow h(x) \leq h_*(x)$, $h_*(x)$: the true cost from x to goal.
A* with h is optimal if h is admissible.

For admissible h1 and h2,if h2(x) ≥h1(x),$\forall n$,then h2 dominates h1 and is more efficient for search.

Question 1.(1)

(1) What is the path returned by greedy graph search with the given heuristic?

Greedy Best-first Search(Greedy search)

$f(x) = h(x)$

node x with the **lowest** $h(x)$ is **expanded first**

iter	a pq ordered by h(x)	**The nodes expanded**
0	{**[S,0]**}	[S,0]
1	{**[C, 6]**,[A, 8],[B, 10]}	[S,0], [C, 6]
2	{**[G,0]**,[A, 8],[B, 10]}	[S,0], [C, 6], [G,0]

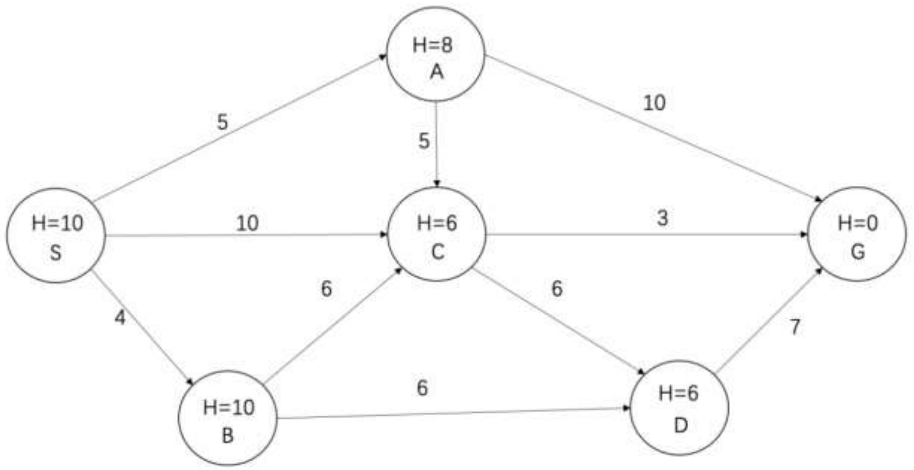


Fig. 1

Answer: $S \rightarrow C \rightarrow G$

Question 1.(2)(3)

(2) What is the path returned by A* graph search with the given heuristic?

(3) List the nodes in the order they are expanded by A* graph search in the above (2) procedure.

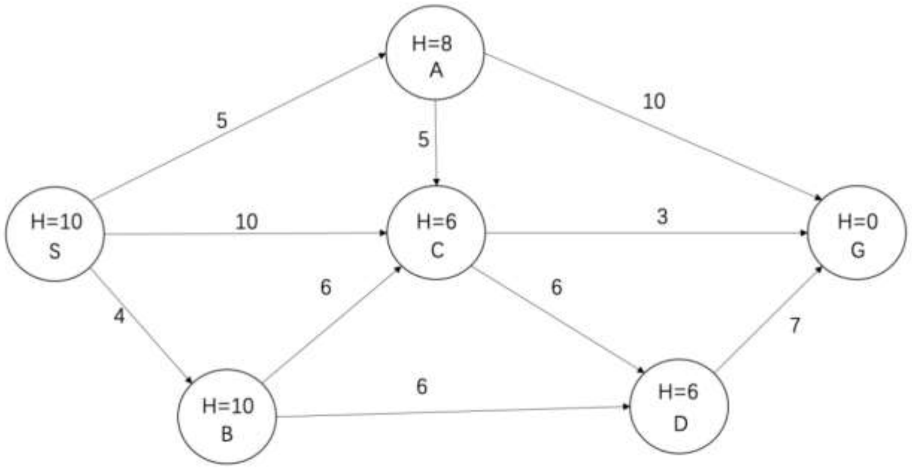


Fig. 1

A*

$f(x) = g(x) + h(x)$

node x with the **lowest** $f(x)$ is **expanded first**

iter	a pq ordered by h(x)	**The nodes expanded**
0	{[S,0+10]}	[S,10]
1	{[A, 5+8],[B, 4+10],[C, 10+6]}	[S,0], [A, 13]
2	{[B, 4+10],[G, 15+0],[C, 10+6]}	[S,0], [A, 13], [B,14]
3	{[G, 15+0],[C, 10+6]}	[S,0], [A, 13], [B,14], [G, 15]

A's father is S, S is the start state. **A→S**

G is the goal state, G's father is A. **G→A**

Answer: (2) $S \to A \to G$

(3) S, A, B, G

Question 1.(4)

(4) Is this heuristic admissible? If not, find a minimal set of nodes that would need to have their values changed to make the heuristic admissible.

Heuristic function h is admissible if:

$\forall x \rightarrow h(x) \leq h^*(x)$, $h^*(x)$: the true cost from x to goal.

A* with h is optimal if h is admissible.

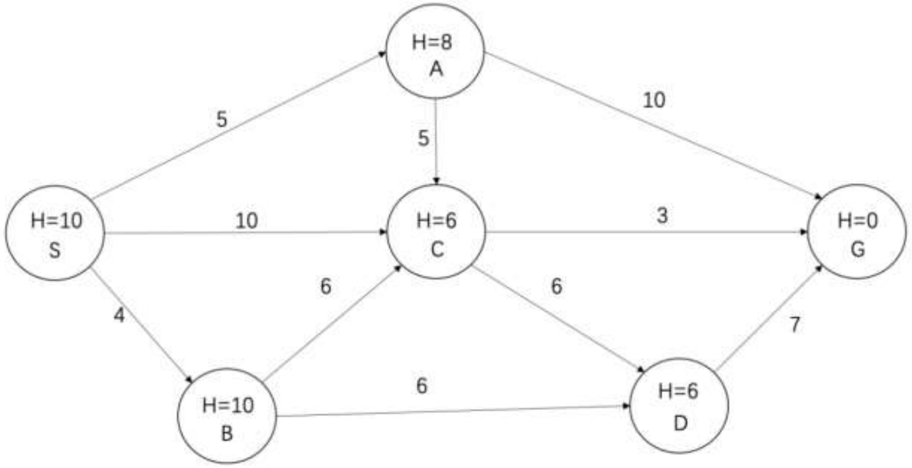


Fig. 1

$h^*(S) = 13$
$h^*(A) = 8$
$h^*(B) = 9$
$h^*(C) = 3$
$h^*(D) = 7$
$h^*(G) = 0$

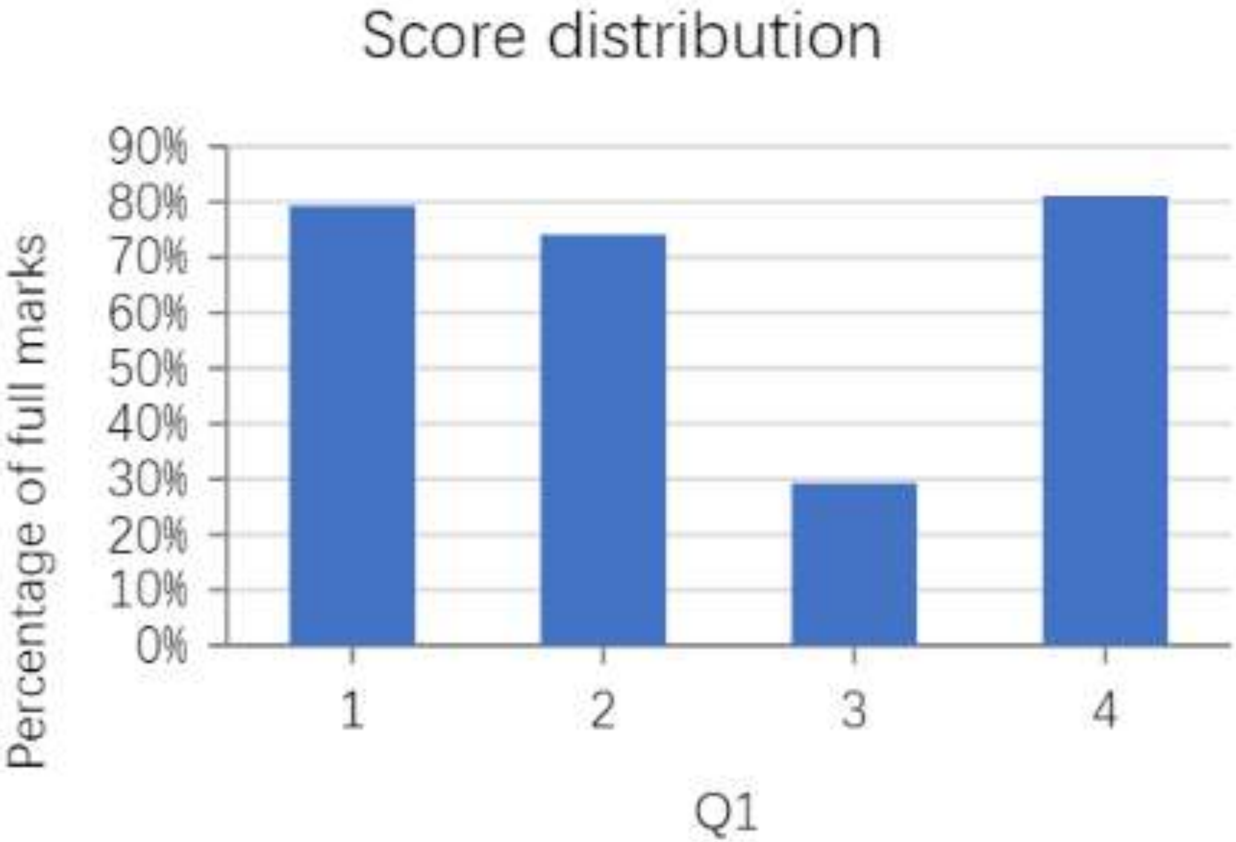
Notice that $h(C) = 6 > h^*(C)$ and $h(B) = 10 > h^*(B)$, so this heuristic is not admissible.

Answer:

$\boldsymbol{No. The\ set\ of\ nodes: \{C, B\}.}$

$\boldsymbol{Need\ to\ make\ h(C) \leq\ 3\ and\ h(B) \leq\ 9.}$

Question 1 Completion statistics



Question 1 Common Errors

(3) List the nodes in the order they are expanded by A* graph search in the above (2) procedure.

Error case : S, A, B, C, G

(4) Is this heuristic admissible? If not, find a minimal set of nodes that would need to have their values changed to make the heuristic admissible.

Error case : only find the node C

Question 2

- You now control two long-lost insect friends in a rectangular maze-like environment of size $M \times N$, as shown in Fig. 2. At each time step, an insect can move north, east, south, or west (but not diagonally) to an adjacent square if that square is free, or remain in place. Squares may be blocked by walls (represented by black squares). The map is provided as input. The insects' starting positions are also input, though initially unknown to you. Your goal is to help the bugs reunite by defining a search problem that yields a sequence of actions ensuring they meet on the same square, regardless of where they start. The specific meeting square doesn't matter; the only goal is for the insects to reunite. Both insects follow the actions blindly, without knowing if their moves succeed; if an action leads them toward a blocked square, they stay in place. They cannot jump over walls, and both insects move simultaneously at each time step, with each time step incurring a cost of one. The following questions refer to the general problem, not just the example maps shown.

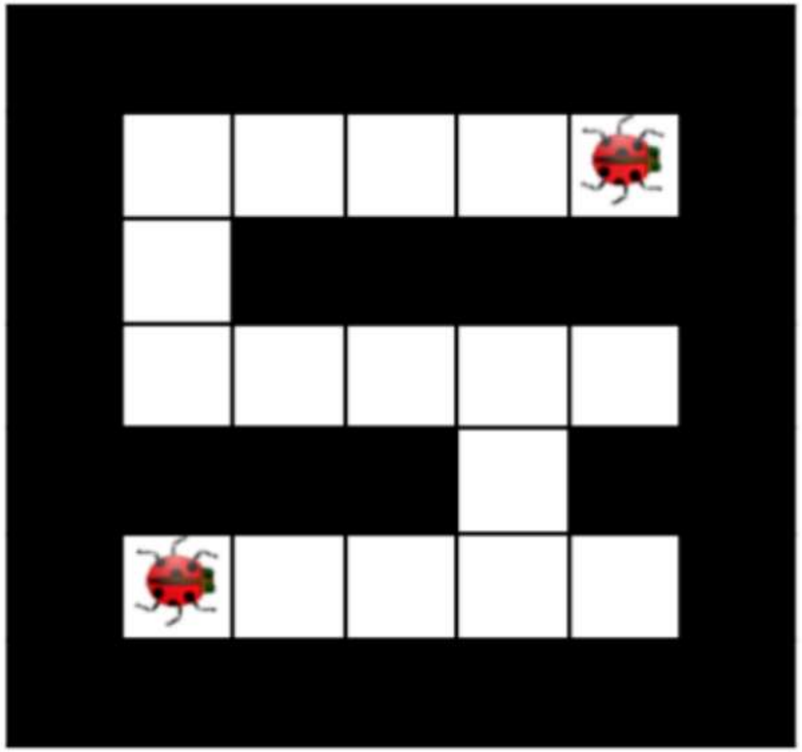


Fig. 2

(1) Give a minimal state representation for the above search problem.
(2) Give the size of the state space for this search problem.
(3) Give a nontrivial admissible heuristic for this search problem.

Question 2.(1)(2)

(1) Give a minimal state representation for the above search problem.
(2) Give the size of the state space for this search problem.

Some possible representations:

- **Representation1**: Using a pair of integers to represent a bug's location a state contains 2 bugs' locations:

  $[x_1, y_1, x_2, y_2]\ or [(x_1, y_1), (x_2, y_2)]\ \forall x_i \in [1, m], y_i \in [1, n], i \in [1,2]$

  The size of the state space using this representation: $(MN)^2$

- **Representation2**: Using a binary sequence to represent if a bug in this square or not

  $[p_1, p_2, \dots, p_l]\ \ l = the\ number\ of\ white\ blocks$

  The size of the state space using this representation: $C_l^2 + C_l^1$

$C_l^2 + C_l^1 < (MN)^2$

Answer: Representation2

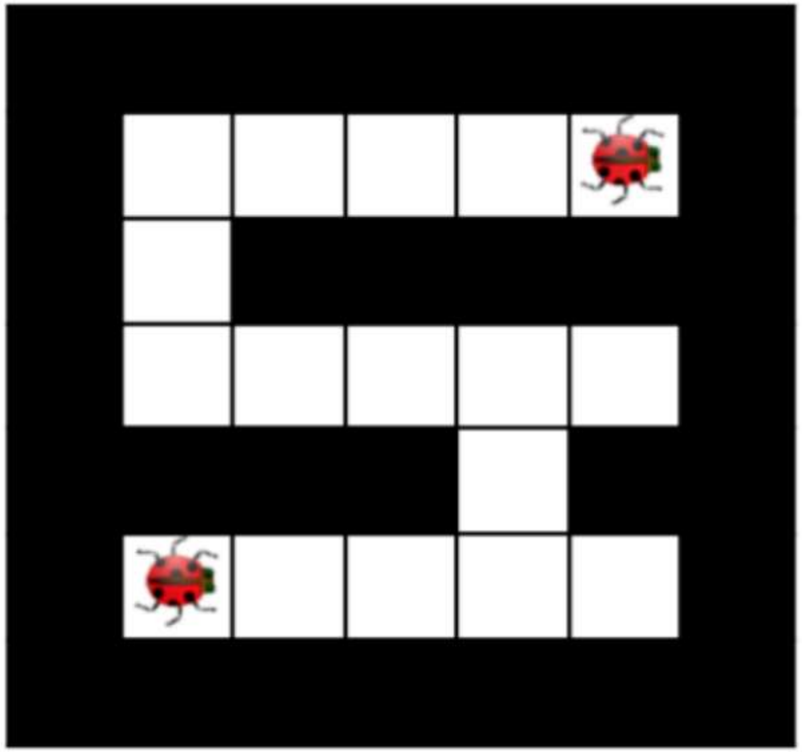


Fig. 2

Question 2.(3)

(3) Give a nontrivial admissible heuristic for this search problem.

Question description about how an insect moves:
At each time step, an insect can move North, East, South, or West (but not diagonally) into an adjacent square if that square is currently free, or the insect may stay in its current location.

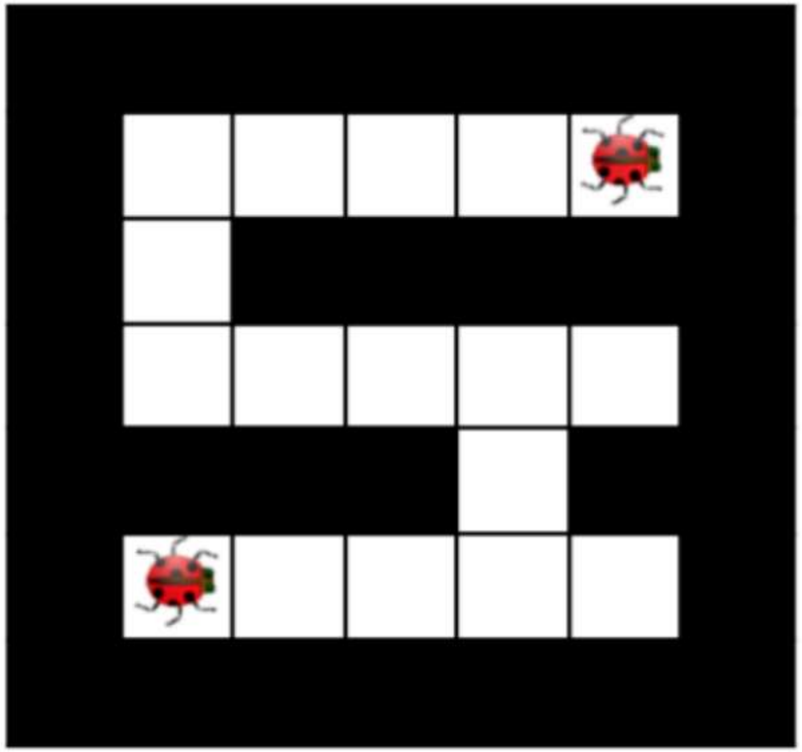


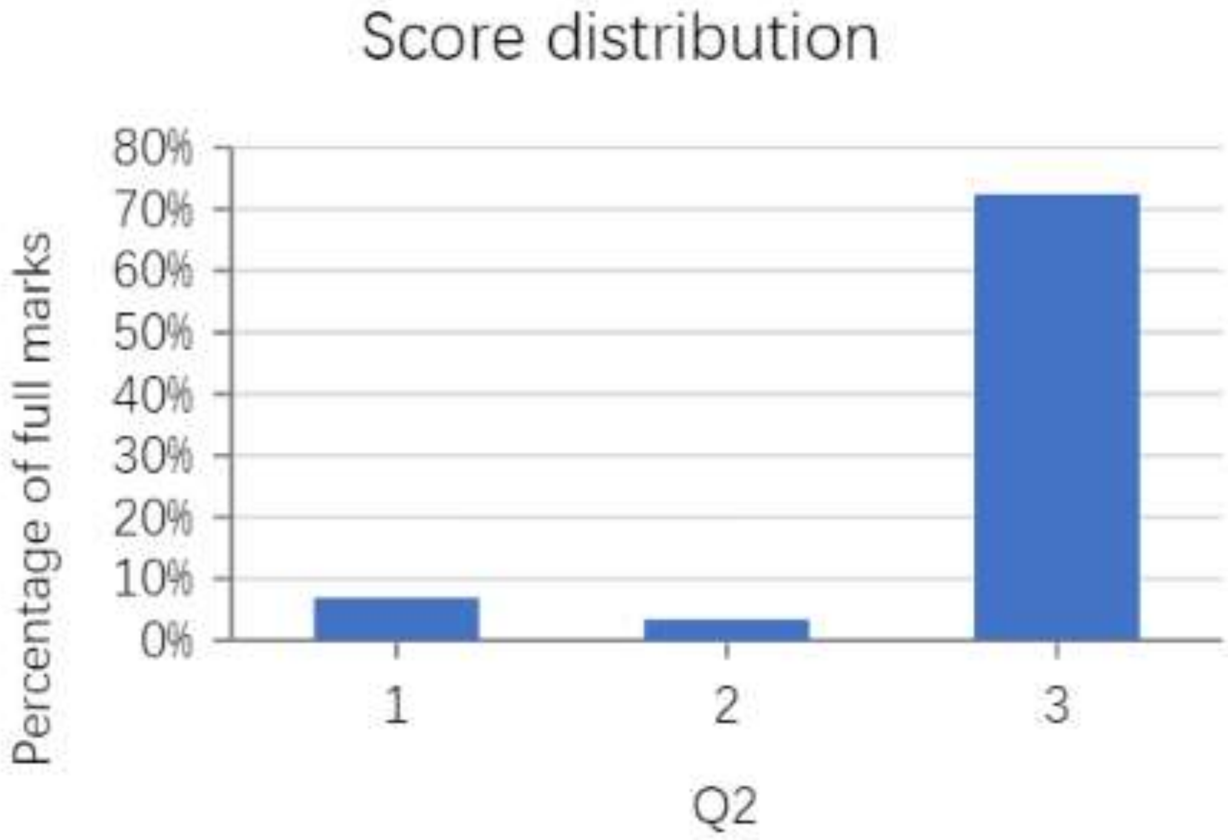
Fig. 2

Answer:

$h(n) =$ *the Manhattan distance from the current location to the target location.*
or
$h(n) =$ *the straight line distance from the current location to the target location.*

In this case, the Manhattan distance is better than the straight-line distance.

Question 2 Completion statistics



Question 2 Common Error

(1) Give a minimal state representation for the above search problem.
(2) Give the size of the state space for this search problem.

Error case 1: Using Representation1, but it is not the minimal state representation.

Some possible representations:

- **Representation1**: Using a pair of integers to represent a bug's location

  a state contains 2 bugs' locations:

  $[x_1, y_1, x_2, y_2]\ or [(x_1, y_1), (x_2, y_2)]\ \forall x_i \in [1, m], y_i \in [1, n], i \in [1,2]$

  The size of the state space using this representation: $(MN)^2$

- **Representation2**: Using a binary sequence to represent if a bug in this square or not

  $[p_1, p_2, \dots, p_l]\ \ l = the\ number\ of\ white\ blocks$

  The size of the state space using this representation: $C_l^2 + C_l^1$

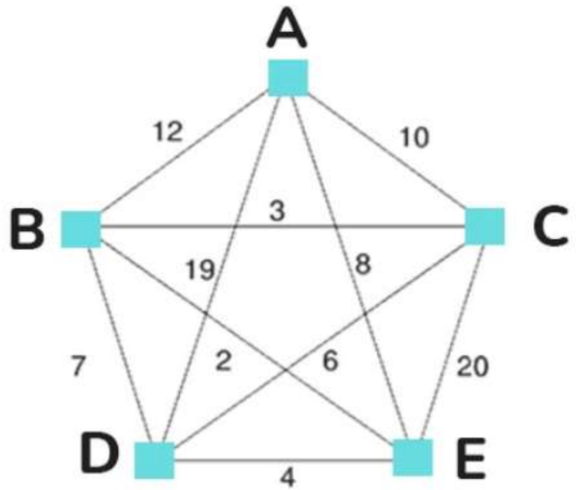
Error case 2: confused by state and action

Error case 3: Only specific examples are taken into account, without abstraction into general expressions or descriptions.

Question 3

- Travelling Salesman Problem (TSP): Given a set of cities and distances between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns to the starting city.
- An example is shown in Fig. 3. Use the Evolutionary Algorithm to solve the TSP problem:
  - How can an individual be represented?
  - Define an appropriate fitness function.
  - How to generate a new individual?
  - Use a flowchart or pseudocode to describe the algorithmic process of solving the TSP. (general problem, not just the example figure shown.)

Input:



Output:

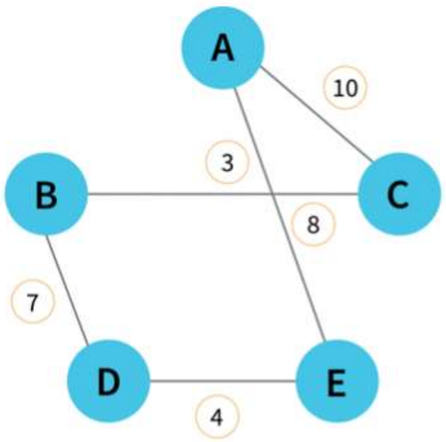


Fig. 3

Question 3

(1) How can an individual be represented?

n cities can be represented by n different numbers, for example, $1 \sim n$.

Using an integer sequence to represent an individual:

$$\boldsymbol{x = [c_1, c_2, \dots, c_n], \forall c_i \in [1, n]\ and\ c_i \neq c_j\ while\ i \neq j}$$

The solution space: all permutations of n numbers.

The solution size: $n!$

(2) The TSP problem representation

$$minimaze\ f(x) = \sum_{i=1}^{n-1} dis\left(c_i, c_{(i+1)}\right) \quad + dis(c_n, c_1)$$ or $$f(x) = \sum_{i=1}^{n} dis\left(c_i, c_{(i+1)\%n}\right)$$

$$subject\ to: \forall c_i \in [1, n], i = 1,2, \dots, n$$

$$c_i \neq c_j\ while\ i \neq j$$

(4) Use the Evolutionary Algorithm to solve the TSP problem.

Generate P_0 *with N initial solutions*

In TSP, can generate N solutions $\boldsymbol{by\ suffle\ the\ sequence}$ $[\mathbf{1}, \mathbf{2}, \dots, \boldsymbol{n}]$

$i = 0$

Generate P_{i+1} *base on* P_i:

1. Get fitness value for each individual in P_i
2. Generate M offsprings, each offspring generated by the following workflow:

   1.1 select 2 parents from P_i

   In TSP, can using roulette-wheel selection or tournament selection.

   1.2 obtain 1 or 2 offsprings from crossover the 2 parents.

   In TSP, Order crossover(OX for short) or Position-based crossover (PBX for short) can be used. Or you can describe your own crossover process in detail.

   1.3 apply mutation to the offspring with a small probability.

   In TSP, swap mutation, exchange mutation or inversion mutation can be used. Or you can describe your own mutation process in detail.

   1.4 calculate the fitness values of the offsprings.

   In TSP, fitness value: $\boldsymbol{f(x) = \sum_{i=1}^{n-1} dis\left(c_i, c_{(i+1)}\right) + dis(c_n, c_1)}$

3. Select **N individuals(**$\boldsymbol{P_{i+1}}$**)** based on P_i *and* M offsprings using some selection(replacement) strategy.

In TSP, Elite selection can be used or you can just say select top N. Or you can describe your own selection process in detail.

(3)How to generate a new individual?

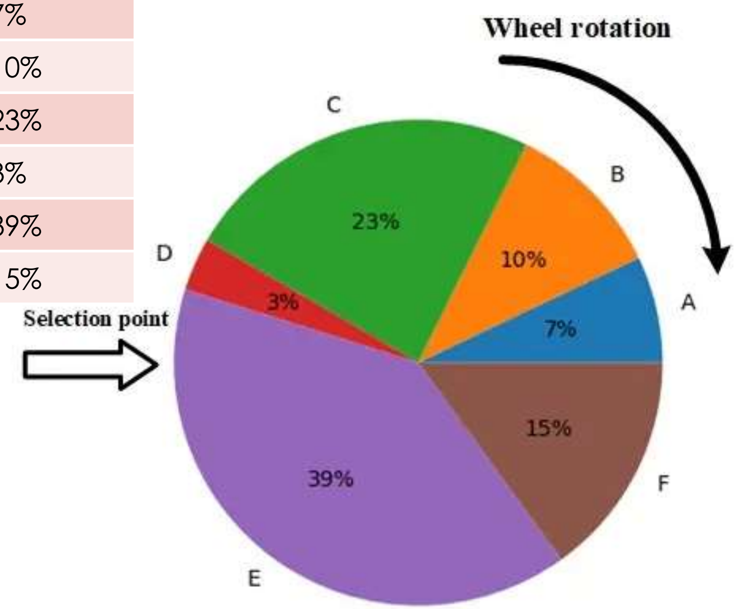
$i = i + 1$

Stop?

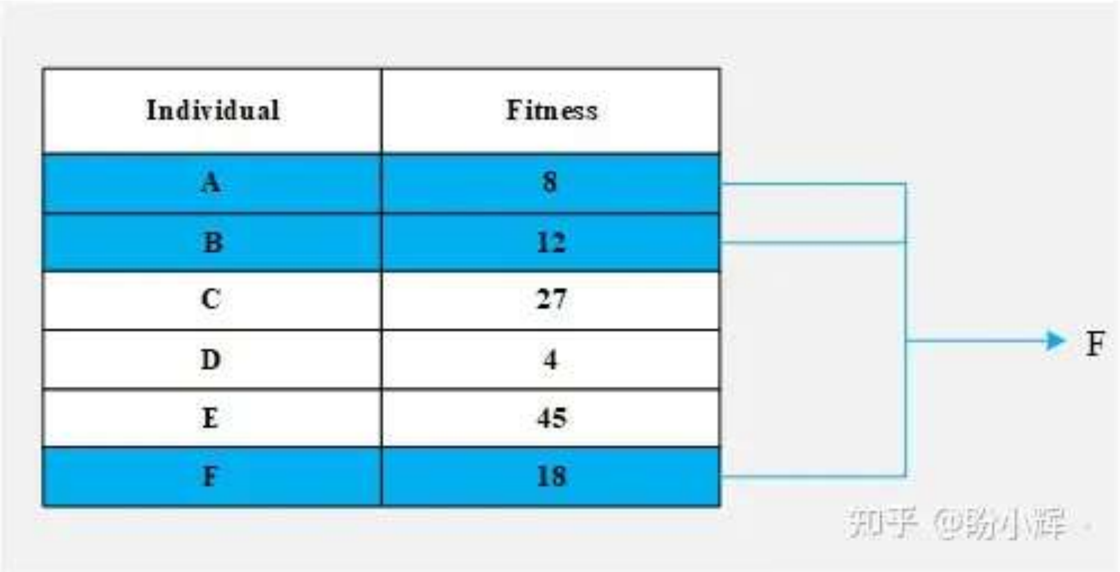
Return the best

Review Parent selection

Solution	Fitness	proportio n
A	8	7%
B	12	10%
C	27	23%
D	4	3%
E	45	39%
F	17	15%



roulette wheel selection



Individual	Fitness
A	8
B	12
C	27
D	4
E	45
F	18

Tournament selection

Review crossover

单点交叉（Single-point crossover）

parent1	1	2	3	4	5	6	7	8
parent2	a	b	c	d	e	f	g	h

两点交叉（Two-points crossover）

parent1	1	2	3	4	5	6	7	8
parent2	a	b	c	d	e	f	g	h

多点交叉（Multi-point crossover）

parent1	1	2	3	4	5	6	7	8
parent2	a	b	c	d	e	f	g	h

均匀交叉（Uniform crossover）

parent1	1	2	3	4	5	6	7	8
parent2	a	b	c	d	e	f	g	h

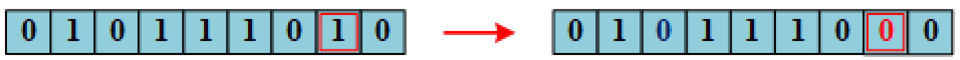
顺序交叉（Order crossover, OX）

parent1	1	2	3	4	5	6	7	8
parent2	3	5	8	1	7	4	2	6

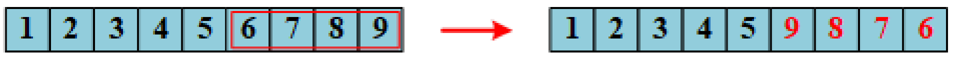
位置交叉（Position-based crossover , PBX）

parent1	1	2	3	4	5	6	7	8
parent2	3	5	8	1	7	4	2	6

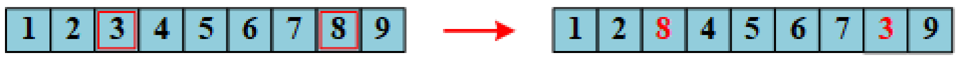
Review Mutation



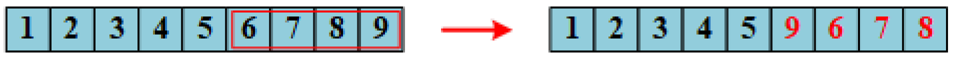
Flip bit mutation



Inversion mutation



Swap\Exchange mutation

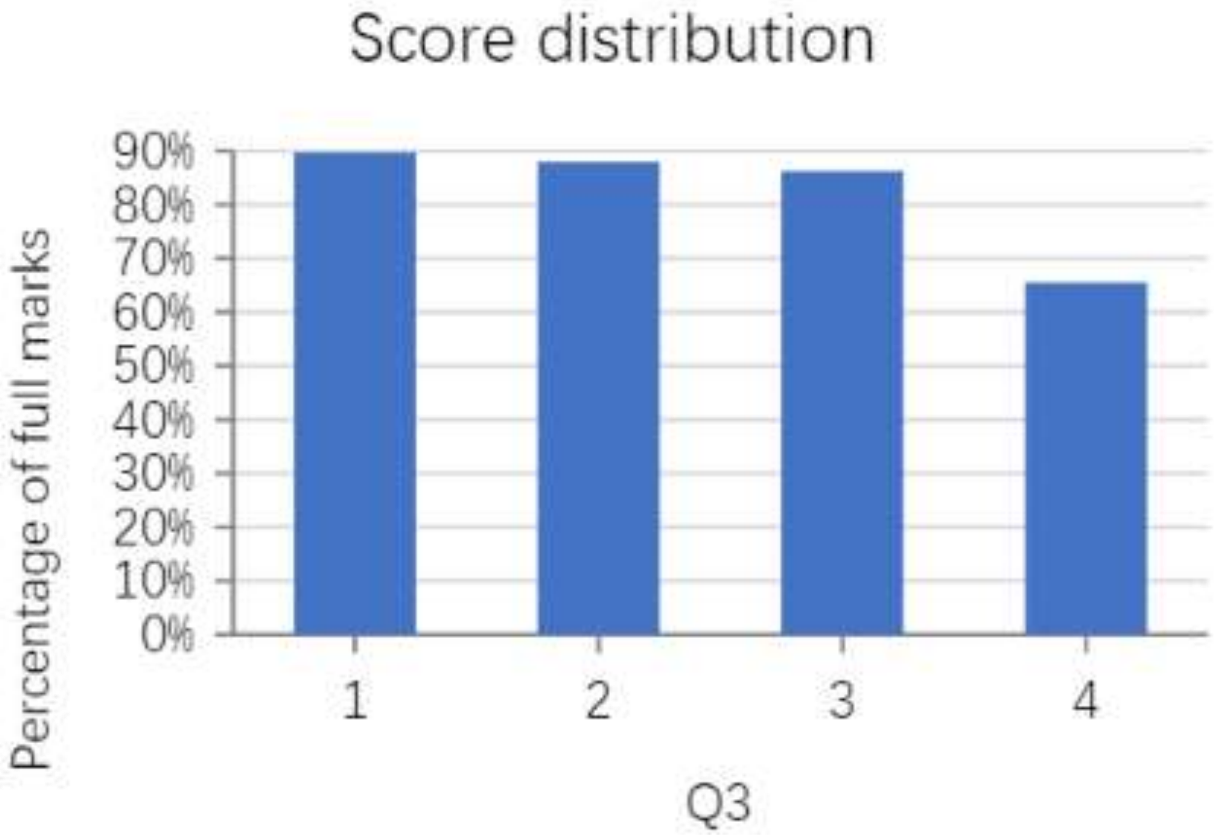


Scramble mutation

Review Selection strategy

Elite selection	The larger the fitness, the better
Non-Elite selection	Not entirely dependent on the fitness

Question 3 Completion statistics



Question 3 Common Error

(1) How can an individual be represented?

Error case 1: mistakenly thought that simulated annealing was an EA algorithm

Error case 2: Only the example provided in the question is considered, without being abstracted into general expressions or descriptions, such as mathematical formulas.

(2) How to generate a new individual?

Error case 1: *Not taking constraints into account*

Error case 2: *just says how mutates, not how crosses*

Summary

- Problem Representation
  - Solution representation
  - State VS. Action
  - state/solution space
- Classic Search
  - Heuristic search vs. uninformed search
  - A* and admirable function
- EA
  - EA workflow
  - How to select parent, crossover, mutate, replace the old population