Uninformed Search:
DFS: 效率低 DLS: 可能找不到 IDS:目标很深效率低 BFS

Algorithm	Time Complexity	Space Complexity	Derivative
DFS	O(b^m)	O(bm)	
DLS	O(b^l)	O(bl)	DFS
IDS	O(b^d)	O(bd)	DLS
BFS	O(b^d)	O(b^d)	
UCS	O(b^d)	O(b^d)	BFS
BIDI （Bidirectional）	O($b^{d/2}$)	O($b^{d/2}$)	BFS（两端采用 BFS）

b, branching factor
d, tree depth of the solution
m, tree depth
l, search depth limit
State Space (aka Problem Space) = all possible valid configurations of the environment
Optimality / Admissibility – an admissible algorithm will find a solution with minimum cost
Completeness – a complete algorithm will find a solution (not all)
A goal test is applied to a node's state to determine if it is a goal node
Informed Search:
f(n) = g(n) + h(n)
其中 h(n)就是 heuristic 函数，g(n)是 cost 估算函数。g(n)表示从初始节点到当前节点所需的开销（例如可以用路径的长度/权来表示），而 h(n)表示从当前节点到目标节点还剩的差距（有多种表示方法，比如可以用当前状态与目标状态 Conflicts 的个数来表示），g(n)和 h(n)两者共同决定了 f(n)，即一个状态的质量/估价。有时为了区分 g(n)和 h(n)的权重，也可以在前面加上常系数，即表示为
f(n)=α*g(n) +β*h(n)

```
BFS:（1）将初始节点插入 OPEN List 中
 当 OPEN List 非空时
 （2）从 OPEN List 出堆，放入 CLOSED List 中，如果该节点即为目标节点则算法结束（backtracking 路径）
 （3）扩展该节点的所有后继节点
对每一个后继节点，若后继节点不在 CLOSED 中，则把它插入 OPEN List 中
A*:（1）将初始节点插入 OPEN List 中
 当 OPEN List 非空时
 （2）从 OPEN List 出堆，放入 CLOSED List 中，如果该节点即为目标节点则算法结束（backtracking 路径）
 （3）扩展该节点的所有后继节点
   对每一个后继节点
     if 后继节点在 CLOSED List 中，则抛弃它，continue
     else if 后继节点在 OPEN List 中
        if 后继节点的 g(n)值比出现在 OPEN List 上的那个节点的 g(n)值好
         则用后继节点替换掉原来出现在 OPEN List 上的那个节点
     else
      把后继节点加到 OPEN List 中
```

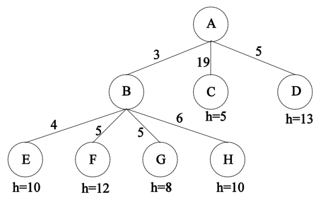
A heuristic function that utilizes relative distance from the goal state.
A heuristic, h, is called consistent (aka monotonic) if, for every node n and every successor n' of n, the estimated cost of reaching the goal from n is no greater than the step cost of getting to n' plus the estimated cost of reaching the goal from n':
c(n, n') ≥ h(n) − h(n')
or, equivalently: h(n) ≤ c(n, n') + h(n')
•Values of f(n) along any path are nondecreasing
•Consistency is a stronger condition than admissibility
All consistent heuristics are admissible, but not all admissible heuristics are consistent
An admissible heuristic h(n) never overestimates the cost from n to the goal.

$$0 \le H(Y \mid X) \le 1$$

Hill-Climbing Algorithm

```
1. Pick initial state s
2. Pick t in neighbors(s) with the largest f(t)
3. if f(t) ≤ f(s) then stop and return s
4. s = t. Goto Step 2.
```

Consider the following *search tree* produced after expanding nodes A and B, where each arc is labeled with the cost of the corresponding operator, and the leaves are labeled with the value of a heuristic function, *h*. For uninformed searches, assume children are expanded left to right. In case of ties, expand in alphabetical order.



Which one node will be expanded *next* by each of the following search methods?
Which one node will be expanded *next* by each of the following search methods?

(a) [3] *Depth-First search*

```
E
```

(b) [3] *Greedy Best-First search*

```
C (smallest h value)
```

(c) [3] *Uniform-Cost search*

```
D (smallest g value)
```

(d) [3] *A* search*

```
G (smallest g+h value)
```

Question 2. [13] Search Short Answer

(a) [2] True or False: *Greedy Best-First search* using an admissible heuristic is guaranteed to find an optimal solution.

```
False
```

(b) [2] True or False: If a heuristic is consistent (aka monotonic), it is also admissible.

```
True
```

(c) [2] True or False: If h_1 and h_2 are both admissible heuristics, it is guaranteed to be better to use the heuristic $h_3(n) = \max(h_1(n), h_2(n))$ rather than the heuristic $h_4(n) = \min(h_1(n), h_2(n))$.

```
True
```

(d) [2] True or False: If h_1 and h_2 are both admissible heuristics, $2h_1 - h_2$ is guaranteed to be an admissible heuristic.

```
False
```

(e) [2] True or False: *Beam search* with a beam width *W*=3 and an admissible heuristic is *not* guaranteed to find a solution (optimal or not).

```
True
```

(f) [3] Say we have a state space where all arc costs are 1 but we don't have a heuristic function. We want to use a space-efficient search algorithm (in terms of the maximum number of nodes stored at any point during the search) but also want to guarantee that we find an optimal solution. What search method would be best to use in this situation?

```
Depth-First Iterative Deepening because it is space efficient
like DFS but also guarantees finding an optimal solution because
all arc costs are 1.
```

Question 3. [9] Local Search

(a) [3] The 8-Queens problem can be represented as an 8-digit vector where the i^{th} digit indicates the row number of the queen in the i^{th} column. Given a state defined by such a vector, a successor function could be defined by randomly selecting a new row for a single queen. Which *one* of the following actions in a *Genetic Algorithm* is *most similar* to the operation defined by this successor function?
(i) Selection
(ii) Fitness
(iii) Crossover
(iv) Mutation

```
(iv) Mutation
```

(b) [3] How would *Simulated Annealing* perform if the temperature, *T*, is always equal or very close to 0? (Select *one* of the following.)
(i) A random walk
(ii) Hill-Climbing
(iii) It would halt immediately
(iv) It would never halt

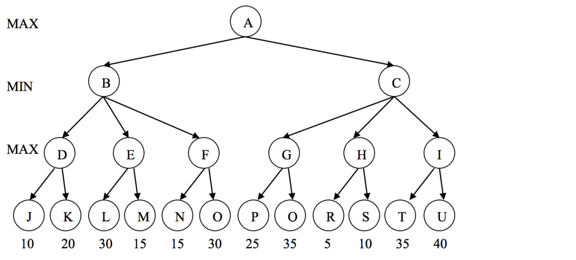
```
(ii) Hill-Climbing
```

(c) [3] In *Simulated Annealing* if the current state's evaluation function value is 15, the selected successor state's value is 11, and the current temperature is *T* = 10, what is the probability of moving to the successor state? (Assume we are maximizing the evaluation function.) Give your answer as either an expression or a number.

$p = e^{\Delta E/T}$ where ΔE f(successor) − f(current) = 11 − 15 = −4
So, $p = e^{-.4} = 0.67$

Question 4. [11] Game Playing

Consider the following game tree. The root is a maximizing node, and children are visited left to right.



(a) [6] *Ignoring the values given above at the leaf nodes*, could there *possibly* exist some set of values at the leaf nodes in the above tree so that *Alpha-Beta pruning* would do the following. Answer *Yes* or *No*.

(i) [2] Pruning occurs at the arc from D to K.

```
No
```

(ii) [2] Pruning occurs at the arc from E to M.

```
Yes
```

(iii) [2] Pruning occurs at the arc from C to G.

```
No
```

(b) [3] In the tree above assume that the root node is a MAX node, nodes B and C are MIN nodes, and the nodes D, …, I are ***not*** MAX nodes but instead are positions where a fair coin is flipped (so going to each child has probability 0.5). What is the *Expectimax* value at node A?

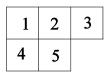
```
15
```

(c) [2] True or False: When using Expectimax to compute the best move at the root, rescaling the values at *all* the leaf nodes by multiplying them *all* by 10 may result in a different move at the root.

```
False
```

Question 5. [8] Constraint Satisfaction

Consider the problem of assigning colors to the five squares on board below such that horizontally adjacent and vertically adjacent squares do not have the same color. Assume there are possible two colors, red (*R*) and black (*B*). Formulated as a constraint satisfaction problem, there are five variables (the squares, named 1, …, 5) and two possible values (*R*, *B*) for each variable.



(a) [4] If initially every variable has both possible values and we then assign variable 1 to have value *R*, what is the result of the *Forward Checking algorithm* (no search, just forward checking deletions)?

```
1 = { R },  2 = { B },  3 = { R, B },  4 = { B },  5 = { R, B }

Forward Checking propagates constraints from the current assigned
variable to all adjacent, unassigned variables, which in this
case are variables 2 and 4.
```

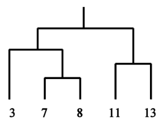
(b) [4] If initially every variable has both possible values except variable 5 has only value *B*, what is the result of the *Arc Consistency algorithm* (AC-3) (no search, just AC-3 deletions)?

```
1 = { B },  2 = { R },  3 = { B },  4 = { R },  5 = { B }
```

Question 6. [6] Hierarchical Clustering

(a) [4] Show the results of *Hierarchical Agglomerative Clustering* (all iterations until it halts) using *complete-linkage* with the one-dimensional dataset containing five points: {3, 7, 8, 11, 13}.

```
Iteratively merge closest two clusters, where cluster distance is
measured by the farthest apart two points.  So, first, merge 7
and 8 (distance 1), then merge 11 and 13 (distance 2), then merge
3 and {7,8} (distance 5), and finally merge {3,7,8} and {11,13}
(distance 10).
```



(b) [2] What clusters are created if you want *three* clusters?

```
Cut off after first two steps, creating clusters {3}, {7, 8} and
{11, 13}
```

Question 7. [8] K-Nearest Neighbor (*k*-NN) Classifier

(a) [4] Consider a set of five training examples given as ((x_i, y_i), c_i) values, where x_i and y_i are the input attribute values and c_i is the output class: {((1, 1), −1), ((1, 7), +1), ((3, 3), +1), ((5, 4), −1), ((2, 5), −1)}. Classify a test example at coordinates (3, 6) using a *k-NN classifier* with *k* = 3 and the City-Block distance defined by $d((u, v), (p, q)) = |u - p| + |v - q|$.

```
City-block distances from (3, 6) to each training example are:
(1, 1) is distance 2+5=7; (1, 7) is distance 2+1=3; (3, 3) is
distance 0+3=3; (5, 4) is distance 2+2=4; and (2, 5) is distance
1+1=2.  So, points (2, 5, -1), (1, 7, +1) and (3, 3, +1) are the
3 nearest neighbors and their majority class is +1, so classify
(3, 6) as class +1.
```

(b) [2] True or False: The decision boundary associated with a *1-NN classifier* defined by the above five examples consists of a sequence of line segments that are each parallel to exactly one of the two axes associated with the two features, *x* and *y*.

```
False
```

(c) [2] True or False: The results of a general *k-NN classifier* that uses Euclidean distance *may* change if we multiply each example's attribute value by 0.1.

```
False
```

Question 8. [9] K-Means Clustering

(a) [2] True or False: *K-Means Clustering* is guaranteed to converge (i.e., terminate).

```
True
```

(b) [7] Show the results of performing *K-Means Clustering* on the one-dimensional dataset containing four data points: {5, 7, 10, 12} assuming *k* = 2 and the initial cluster centers are given as c_1 = 3.0 and c_2 = 13.0.

(i) [4] Give the initial cluster assignments. (That is, which examples are in cluster c_1 and which examples are in cluster c_2?)

Distance	5	7	10	12
c_1 = 3	2	4	7	9
c_2 = 13	8	6	3	1

```
So, the initial clusters are {5, 7} and {10, 12}
```

(ii) [3] Give the new cluster centers after making the assignments in (i).

c_1 = (5 + 7)/2 = 6 and c_2 = (10 + 12)/2 = 11

Question 9. [15] Decision Trees

(a) [4] What is the *conditional entropy*, *H*(*C* | *A*), for the following set of 8 training examples. Give either an expression containing logs or else a single value using log 0.1 = -3.3, log 0.2 = -2.3, log 0.25 = -2.0, log 0.3 = -1.74, log 0.33 = -1.58, log 0.4 = -1.3, log 0.45 = -1.15, log 0.5 = -1.0, log 0.55 = -0.85, log 0.6 = -0.7, log 0.67 = -0.58, log 0.7 = -0.5, log 0.75 = -0.4, log 0.8 = -0.3, log 0.9 = -0.15, log 1 = 0, where all logs are base 2.

A	*B*	*C*
T	F	T
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F

```
H(C|A) = 5/8 H(5/5, 0/5) + 3/8 H(1/3, 2/3)
       = (5/8)(0) + (3/8)((-.33)log(.33) + (-.67)log(.67))
       = .375((-.33)(-1.58) + (-.67)(-.58))
       = 0.34
```

(b) [3] In a problem where each example has *n* binary attributes, to select the best attribute for a decision tree node at depth *k*, where the root is at depth 0, how many *conditional entropies* must calculated?

```
n - k
```

(c) [2] True or False: The Information Gain at the nodes along any path from the root of a Decision Tree to a leaf is always monotonically non-increasing.

```
False
```

(d) [3] Given a problem where each example is defined by a single, three-valued (values 1, 2 or 3) attribute, what is the *entropy* of these 8 training examples: 1, 3, 2, 3, 1, 3, 3, 2. Use the log values given in (a).

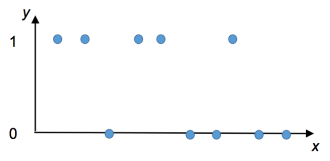
```
H(2/8, 2/8, 4/8) = (-.25)log .25 + (-.25)log .25 + (-.5)log .5
  = 0.5+0.5 + 0.5 = 1.5
```

(e) [3] Which of the following is the (one) *main* reason, in general, for *pruning* a *decision tree*:
(i) To save computing time during testing
(ii) To save space for storing the tree
(iii) To avoid overfitting
(iv) To make the training set error smaller

```
(iii) To avoid overfitting
```

Question 10. [9] Cross-Validation

Suppose you are using a Majority Classifier on the following training set containing 10 examples where each example has a real-valued input, *x*, and a class label, y, with value 0 or 1. Define your Majority Classifier to output the class label that is in the *majority in the training set*, regardless of the input value. In case of ties, always output class 1.



(a) [3] What is the *training set accuracy*?

```
5/10 = 50%
```

(b) [3] What is the *Leave-1-Out Cross-Validation accuracy*?

```
Each of the 10 test examples is classified incorrectly because
the majority class of the other 9 is in the opposite class, so
the average accuracy is 0%
```

(c) [3] What is the *2-fold Cross-Validation accuracy*? Assume the leftmost 5 points (i.e., the 5 points with smallest *x* values) are in one fold and the rightmost 5 points are in the second fold.

```
For each fold, only 1 of the 5 test examples is classified
correctly, so the average accuracy on the two folds is 20%
```