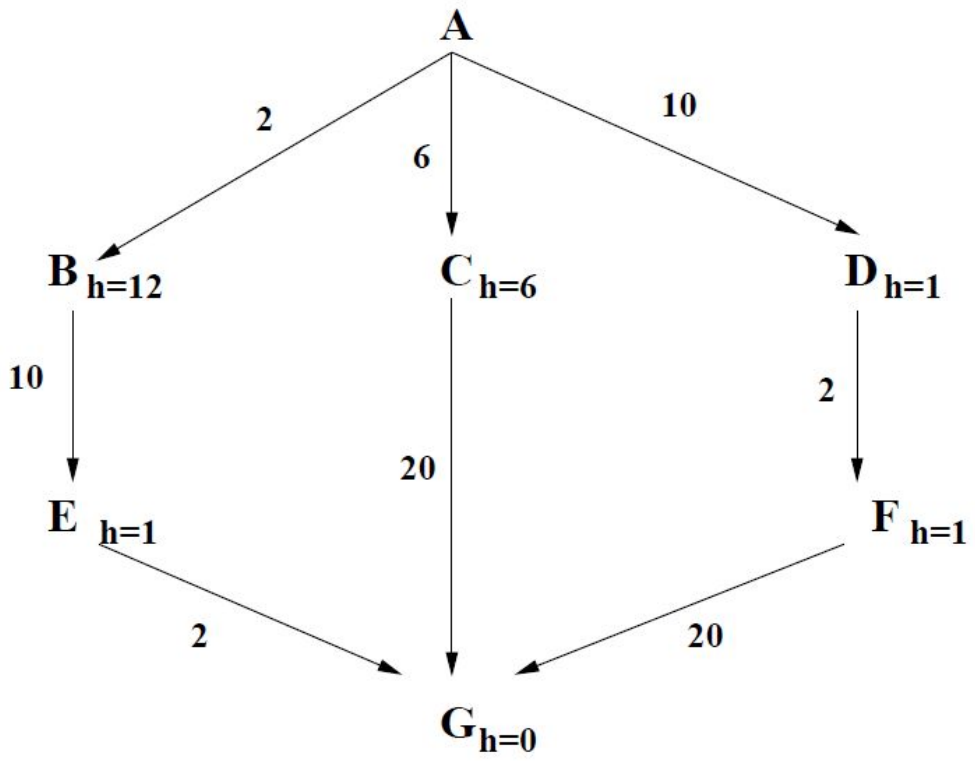
Tarea 1 - Inteligencia Artificial

Entrega: Domingo 7 de Junio

Nombre:

1.- (10 pts) Considere el siguiente grafo, donde cada nodo tiene un identificador (letra), y un valor de heurística. El número asociado a cada arista indica su costo.



1.1.- Muestre en qué orden A* expande los nodos desde A hasta G. Para cada nodo expandido, indique los valores de f y g, además del nodo del que fue generado.
1.2.- ¿Cuál es la solución que entrega A*?
1.3.- ¿Es la función h admisible? ¿Es consistente? Justifique.

2.- (10 pts) Suponga que ud. Desea utilizar un método de búsqueda informada con la función de evaluación f(n) = (1- w) g(n) + wh(n)
2.1.- Para esta pregunta suponga también que h(n) es admisible. ¿Cuáles son los valores de w que garantizan que el algoritmo encontrará una solución óptima? Justifique
2.2.- ¿Hay algún rango de valores de w que garantice que la heurística del algoritmo, al utilizar f, sea admisible? De ser así, ¿cuál es el rango? Justifique.

3.- (10 pts) Suponga que tiene dos heurísticas admisibles, h1 y h2. Ud. decide crear nuevas heurísticas de la siguiente manera:
h3(n) = max (h1(n), h2(n))

h4(n) = max (h1(n), 1.1*h2(n))
h5(n) = min(h1(n), 3*h2(n))
h6(n) = (h1(n) + h2(n))/2

Para cada una de estas nuevas heurísticas, especifique si son, o no, admisibles. Justifique su respuesta. ¿Usaría ud. alguna de estas nuevas heurísticas en lugar de h1 o h2?

4.- (5 pts) ¿Es el algoritmo de Hill Climbing apropiado para el problema de misioneros y caníbales? Justifique

5.- (5 pts) Es posible que el método de búsqueda en profundidad iterativa tenga un peor desempeño que búsqueda en profundidad? Justifique.

6.- (35 pts) Se adjunta código “tarea.cpp”. Este código contiene un esqueleto de un programa que, originalmente, resolvía mediante varios métodos de búsqueda el problema resuelto en el cuestionario evaluado 2. Para esta tarea, se le pide que complete el esqueleto con los códigos de los métodos de:

-A*
- greedy
- búsqueda en profundidad (DFS)
- Además, implemente búsqueda en anchura

Todo esto para resolver el problema del puzzle-8 en tres versiones: puzzle-3 (en un tablero de 2x2 con fichas del 1 al 3), puzzle-8 (en un tablero de 3x3, con fichas del 1 al 8) y puzzle-15 (en un tablero de 4x4, con fichas del 1 al 15), desde cualquier configuración inicial, usando como acción el movimiento de una ficha a la celda vacía, y como heurística la suma de las distancias de cada ficha en el estado actual, con respecto a su posición en el estado objetivo (sólo usando movimientos laterales y verticales, no diagonales).

Ej:
Puzzle-3, estado objetivo:

1	2
3	

Puzzle-8, estado objetivo:

1	2	3
4	5	6
7	8	

Puzzle-16, estado objetivo:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

Se debe entregar el código final y las instrucciones para su ejecución, y se evaluará en un conjunto de casos de prueba de configuraciones iniciales para los tres tipos de puzzles. El tiempo máximo a utilizar para cada búsqueda es de 1 hora (es decir, si pasa 1 hora y no entrega resultado, se considera que el método es muy ineficiente para el problema, asumiendo que está bien programado).

***BONUS (20 pts): Crear una copia de tarea.cpp y modificar para implementar alguno de los métodos de búsqueda local vistos en el material, para resolver el problema de N-reinas (donde N es un parámetro de entrada al programa). En caso de que el método devuelva distintos resultados, utilizar los 3 mejores resultados devueltos y explicarlos. Al igual que en el ejercicio 6, para tener acceso a los puntos de bonus se deberá entregar el código junto con instrucciones para su ejecución, además del reporte de los resultados.