Tarea 1 Inteligencia Artificial
Fecha de entrega: Domingo 5 de Mayo de 2019

1 - (10 pts) De un ejemplo de un espacio de búsqueda (considerando factores como factor de ramificación, profundidad, ubicación del objetivo en profundidad) en el cual el método de búsqueda de profundidad iterativa tenga un peor desempeño que el de búsqueda en profundidad (básico). Puede usar un ejemplo artificial, es decir, el espacio de estados puede no corresponder a ningún problema real.

2.- Considere un problema en el cual tiene una mesa y cuatro bloques, del mismo tamaño. Cada bloque es identificado por una letra, A, B, C o D. Cada bloque puede estar en la mesa, o sobre otro bloque. Al comienzo, los bloques A, B y D están sobre la mesa, y el bloque C está sobre el bloque D. Un bloque puede moverse a la vez, sólo si no tiene otro bloque encima. Cada bloque puede moverse ya sea a la mesa o sobre otro bloque (si es que no hay nada sobre ese bloque). El objetivo es crear una torre con los bloques A, B y C, donde A es el bloque de más arriba y C el de más abajo (sobre la mesa).

2.1 (10 pts) Describa la formulación del problema (cuáles son los estados, estado inicial, test objetivo y acciones)

2.2 (5 pts) Proponga una heurística no trivial (no h(n)=constante). Es la heurística propuesta admisible?

3.- (15 pts) Los componentes g y h juegan un rol distintivo en A*. Cuáles son esos roles? Qué ocurre cuando se enfatiza o desenfatiza uno de ellos usando diferentes factores en f(n) (por ejemplo, f(n) = (1-w)g(n) + wh(n)) ?

4.- (15 pts) Suponga que tiene dos heurísticas admisibles, h_1 y h_2. Se propone usar h_1 y h_2 para crear las siguientes heurísticas :

5.1. h_4(n) = max(h_1(n), 1.1 * h_2(n))
5.2. h_5(n) = max(h_1(n), h_2(n)/2)
5.3. h_6(n) = min(h_1(n), h_2(n)/2)

Para cada una de las nuevas heurísticas, especifique si es admisible o no, justificando su respuesta. Usaría una de estas heurísticas en lugar de h_1 o h_2? por qué?

5.- Programación. Debe adjuntar el código fuente documentado e instrucciones para su ejecución (se descontará puntaje de no contar con una documentación adecuada y/o instrucciones)

5.1.- (25 pts) Implemente un programa (en C, C++, Java o Python) que, utilizando una estructura de búsqueda tipo árbol, encuentre una solución para el problema de encontrar una ruta entre A y H, diagramado en la Figura 1. Debe utilizar los siguientes métodos:

- Búsqueda en profundidad (escogiendo un sucesor al azar)
- Búsqueda por costo uniforme
- Búsqueda greedy
- A*

Para cada tipo de búsqueda, su código debe retornar:

- El camino encontrado y su costo (y si es la solución óptima, que ud. ya conoce)
- Cantidad de nodos expandidos (veces por nodo y en total)

5.2 .- (10 pts) Responda:

- Qué puede decir de la comparación entre los métodos implementados? En concreto, los que encontraron la solución óptima, a qué se debió? Y los que no, por qué no la encontraron?
- De los métodos vistos en clase, hay alguno que NO retorne una solución (es decir, que se mantenga realizando búsqueda ad infinitum) en este problema? Si es así, cuál? Y si no, por qué no?

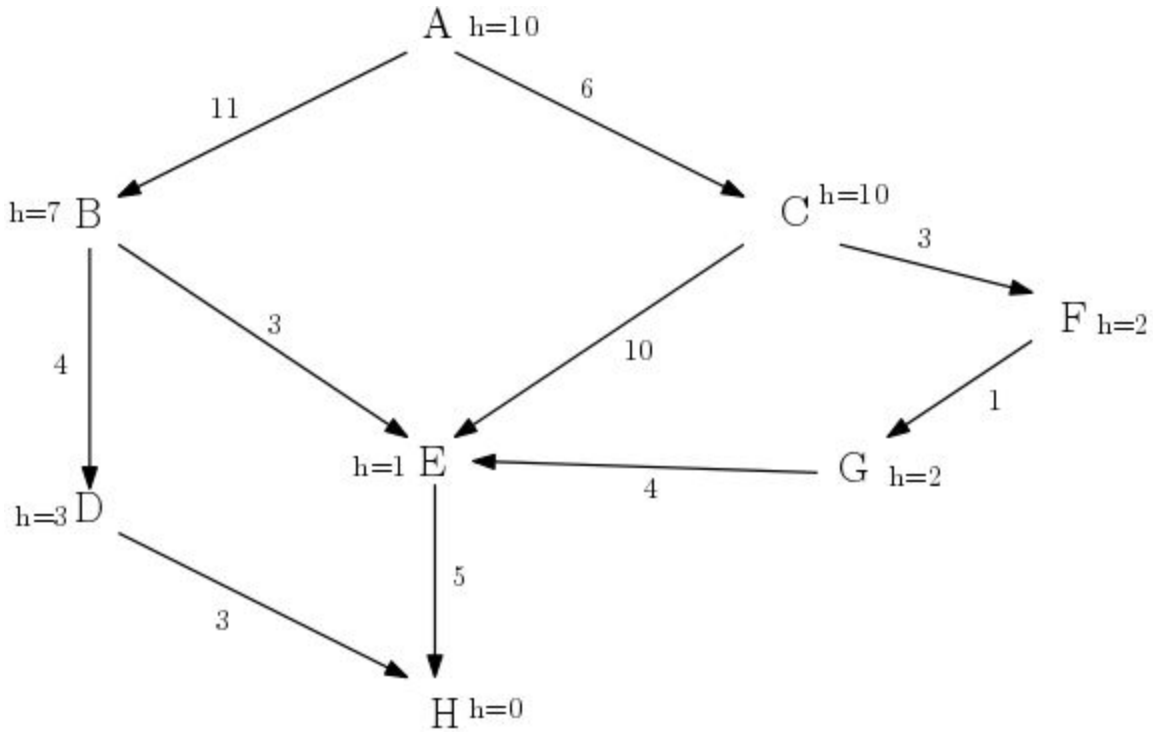

Figura 1