Universidad Simón Bolívar
CI5437 - Inteligencia Artificial I
Kevin Mena 13-10869
Leonardo López 14-10576
Pedro Samuel Fagundez 15-10460

Proyecto 1

15 Puzzle

Heurística: Para este problema empleamos dos tipos de heurísticas, la primera la distancia Manhattan la cual pre calculamos para cada una de las casillas para así ganar tiempo a medida que más instancias se intentan resolver. La segunda una pdb aditiva la cual solo tomaba en cuenta 5 casillas a la vez para al final tener 5-5-5, la heurística termina siendo la suma de las tres. Se intentó calcular la apdb más eficiente que es una 7-8 pero no se tuvieron los recursos necesarios para lograr obtener los resultados.

Algoritmos: A continuación se presenta una tabla con los resultados obtenidos de correr 50 instancias distintas del problema y resolverlas con los distintos algoritmos y heurísticas. Un problema que se encontró es que debido a la representación escogida (con el espacio en blanco abajo a la derecha) muchos de las instancias de prueba no se pueden resolver por lo tanto hubo que pasarlos por un proceso para que fueran resolubles.

Algoritmo	Tiempo Promedio (ms)	Nodos Promedios	Distancia Promedio
IDA* Manhattan	15526012.81	256322728.25	52.81
IDA* APDB	18750038.9	78608786.99	52.81

Para A* no se contaron con los recursos suficientes como para correr las instancias con distancia al goal mayor a 50, y dado que las instancias eran de esa distancia no se logró obtener información debido a que la memoria se acabó.

Por otro lado, podemos notar con IDA* que a pesar de que el tiempo de resolver el problema con Manhattan en promedio es menor, la cantidad de nodos expandidos es bastante menor con APDB. Esto debido a que la cantidad de nodos por bound es menor cuando se subdivide el problema.

Árbol de búsqueda:

Distancia máxima calculada	Número de estados	Factor de ramificación
25	36142146	1.822937875

24 Puzzle

Heurística: Para este problema se tomó una pdb aditiva que consiste en solo tomar en cuenta una sección del puzzle a la vez, la solución óptima es una división 6-6-6-6 pero no se consiguió obtener los resultados por lo que se tuvo que reducir a una 5-5-5-5-4 para poder intentar solucionar el problema.

Algoritmo: Se intentó resolver 50 instancias del problema, pero no se logró con los recursos que se tenían. Todas las instancias pasaron el tiempo que se puso como cota y no se lograron obtener resultados. Se piensa que deben ser resultados parecidos al de 15 puzzle.

Árbol de búsqueda:

Distancia máxima calculada	Número de estados	Factor de ramificación
20	14524718	2.255801522

Top Spin 12

Heurística: Para este problema se tomaron en cuenta 3 pdbs distintas, la primera en la que las últimas cuatro fichas se mapean al primero para reducir el número efectivo de estados del problema, la segunda se quitan las últimas cinco fichas y el último pdb se mapean todas para solo dejar las fichas pares. Finalmente se escoge el valor máximo entre estas tres.

Algoritmos: A continuación se presenta una tabla con los resultados de 25 instancias utilizando tanto A* cómo IDA*. Como se puede observar IDA* es sustancialmente mejor tanto en tiempo de resolución como nodos expandidos como distancia promedio recorrida. Estos son resultados

esperados, por lo menos la parte de la cantidad de memoria usada ya que IDA* no utiliza tanta como A*.

Algoritmo	Tiempo Promedio (ms)	Nodos Promedios	Distancia Promedio
A*	226847.2	121034.38	40.58
IDA*	120.36	35.22	7.38

Árbol de búsqueda:

Distancia máxima calculada	Número de estados	Factor de ramificación
7	17849952	5.745601465

Top Spin 14

Heurística: Para este problema se tomaron en cuenta 3 pdbs distintas, la primera en la que las últimas siete fichas se mapean al primero para reducir el número efectivo de estados del problema, la segunda se quitan las primeras siete fichas y el último pdb se mapean siete fichas del medio del problema. Finalmente se escoge el valor máximo entre estas tres.

Algoritmos: A continuación se presenta una tabla con los resultados de 25 instancias utilizando tanto A* cómo IDA*. Como se puede observar IDA* es sustancialmente mejor tanto en tiempo de resolución como nodos expandidos como distancia promedio recorrida, en este problema IDA* es un poco más lento pero igualmente es muy eficiente. Estos son resultados esperados, por lo menos la parte de la cantidad de memoria usada ya que IDA* no utiliza tanta como A*.

Algoritmo	Tiempo Promedio (ms)	Nodos Promedios	Distancia Promedio
A*	22112830.12	6415390.02	54.36
IDA*	33540.08	12805.2	9.64

Árbol de búsqueda:

Distancia máxima calculada	Número de estados	Factor de ramificación
6	6541598	7.219446247

Top Spin 17

Heurística: Para este problema se tomaron en cuenta 3 pdbs distintas, la primera en la que las últimas once fichas se mapean al primero para reducir el número efectivo de estados del problema, la segunda se quitan las primeras once fichas y el último pdb se mapean todas las fichas a simplemente tres para reducir el problema. Finalmente se escoge el valor máximo entre estas tres.

Algoritmos: A continuación se presenta una tabla con los resultados de 25 instancias utilizando tanto A* cómo IDA*. Como se puede observar IDA* es sustancialmente mejor tanto en tiempo de resolución como nodos expandidos como distancia promedio recorrida, en este problema IDA* es un poco más lento pero igualmente es muy eficiente. Estos son resultados esperados, por lo menos la parte de la cantidad de memoria usada ya que IDA* no utiliza tanta como A*. Una cosa es acotar es que para A* no se logró obtener resultados para todas las instancias debido a acabarse la memoria muy rápido, pero aun así se puede observar la diferencia entre los algoritmos.

Algoritmo	Tiempo Promedio (ms)	Nodos Promedios	Distancia Promedio
A*	18927853.4	5385709.7	42.37
IDA*	281106.28	119955.3	10.04

Árbol de búsqueda:

Distancia máxima calculada	Número de estados	Factor de ramificación
6	16266076	7.890649096

Cubo de Rubik

Heurística: Para este problema se tomaron en cuenta 3 pdbs, 1 corner y dos edges. La pdb corner se refiere a solo tomar en cuenta las esquinas del cubo para reducir el problema a un cubo 2 x 2 x 2 y también se mapean colores para reducir el espacio de estados. Las pdbs edges es tomar en cuenta todo menos las esquinas, la diferencia aquí fue que colores mapear para reducir el espacio. Finalmente se tomó el máximo de los tres.

Algoritmos: A continuación se presentan los resultados con 35 instancias del problema. Dado que no se logró terminar todas las instancias con A* por problemas de memoria el conjunto de resultados es menor, pero aun así se puede notar como IDA* en este problema obtiene mejores resultados en promedio.

Algoritmo	Tiempo Promedio (ms)	Nodos Promedios	Distancia Promedio
A*	8608219.65	784302.2	25.70
IDA*	227094.3	28753.4	8.97

Árbol de búsqueda:

Distancia máxima calculada	Número de estados	Factor de ramificación
5	574908	13.296052175

Torres de Hanoi 12 discos 4 postes

Heurística: Para el pdb de este problema se optó por dos heurísticas distintas, ambas consisten en dividir el número de variables (ya sea mitad u otra proporción) hacer la suma de esas dos mitades para poder obtener una heurística admisible y finalmente se escoge la que tenga mayor valor entre la dos.

Algoritmos: Se intentó resolver 35 instancias del problema, pero no se logró con los recursos que se tenían. Ni siquiera se pudo compilar los solvers con los distintos algoritmos para intentar resolver.

Torres de Hanoi 14 discos 4 postes

Heurística: Para el pdb de este problema se optó por dos heurísticas distintas, ambas consisten en dividir el número de variables (ya sea mitad u otra proporción) hacer la suma de esas dos mitades para poder obtener una heurística admisible y finalmente se escoge la que tenga mayor valor entre la dos.

Algoritmos: Se intentó resolver 35 instancias del problema, pero no se logró con los recursos que se tenían. Ni siquiera se pudo compilar los solvers con los distintos algoritmos para intentar resolver.

Torres de Hanoi 18 discos 4 postes

Heurística: Para el pdb de este problema se optó por dos heurísticas distintas, ambas consisten en dividir el número de variables (ya sea mitad u otra proporción) hacer la suma de esas dos mitades para poder obtener una heurística admisible y finalmente se escoge la que tenga mayor valor entre la dos.

Algoritmos: Se intentó resolver 35 instancias del problema, pero no se logró con los recursos que se tenían. Ni siquiera se pudo compilar los solvers con los distintos algoritmos para intentar resolver.