3 Parte 3

3.1 Respuesta 1

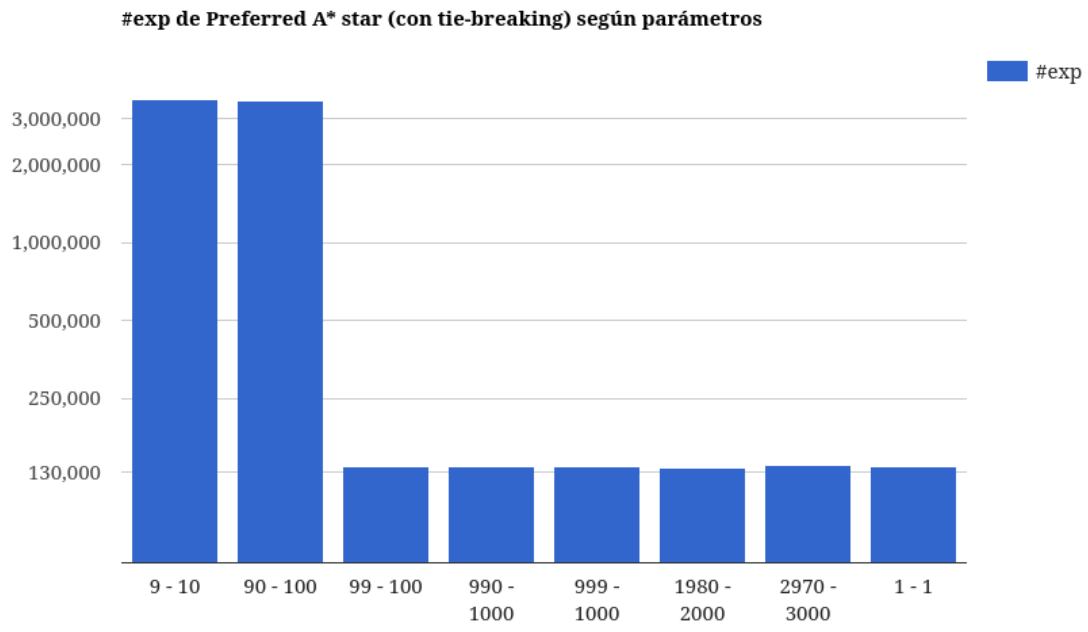

Figure 1: Gráfico de barras mostrando #Exp vs. Parámetros

El gráfico anterior muestra las expansiones según los parámetros `num_pref` y `out_of` que se usen al probar el algoritmo Preferred A* Star en los tests. Los valores de cada barra corresponden a la suma de las expansiones de todos los tests. Notar que la escala es logarítimica.

Se puede notar que la eficiencia del algoritmo aumenta significativamente cuando la razón entre los parámetros se acerca a 1, pasando desde 3 millones cuando la razón es de 0.9, a aproximadamente 135,000 cuando la razón es aún más cercana a 1. Razones menores a 0.9 resultaron en tiempos de ejecución demasiado largos (y por lo tanto, más expansiones), por lo que no fueron incluídos en el diagrama. Cuando `num_pref` = `out_of`, se obtienen 136069 expansiones, sin embargo, esto no corresponde al óptimo para estos tests, logré encontrar mejores resultados para otras combinaciones, con 133880 expansiones para el caso $1980 - 2000$, y 132343 para el caso $2000 - 2025$ (no está en el diagrama). Cabe destacar que las mejoras encontradas no fueron extremadamente distintas al caso con razón 1, a diferencia de este último con los casos con razón ≤ 0.9. Si bien no se puede concluír definitivamente que los valores $2000 - 2025$ son los mejores para el problema, si se puede decir con certeza que mientras más cercana a 1 sea la razón (más se confíe en la NN), obtendremos mejores resultados.

3.2 Respuesta 3

3.3 Respuesta 5

Al calcular m (el valor mínimo para $g(s) + h(s), s \in \text{OPEN} \cup \text{PREFERRED}$), el m encontrado corresponderá al valor de un estado que se encuentra en el camino óptimo, y tendría un valor de suboptimalidad total igual a 1. Dado que al hacer la comparación $g(t) + h(t) > wm$ multiplicamos

a m por w (la suboptimalidad máxima que nuestro algoritmo va a tolerar), nos aseguramos de que t no pertenezca a un camino que sobrepase esta esta cota. Además, dado que al reinsertar t en OPEN (en caso de que la condición sea verdadera), la OPEN se ordena automáticamente según el valor f, nos aseguramos de que la próxima vez que se extraiga un nodo de la OPEN, se prefiera sacar un nodo con menor valor f (y que haga la condición falsa), manteniendo así el camino de la posible solución dentro de la cota de suboptimalidad establecida.

3.4 Respuesta 6

w	1		1.1		1.2		1.4		2	
# puzzle	orig	NN	orig	NN	orig	NN	orig	NN	orig	NN
1	32470	31757	30277	23411	9187	3086	8431	45	4000	45
2	48443	47565	32296	82389	7253	25676	14663	42	7656	42
3	66296	64189	66738	24603	53827	16927	35271	42	5760	42
4	142928	142674	142527	74891	39631	45752	19412	974	13131	41
5	154019	149756	116004	62886	30315	27242	4946	46	3615	46
6	179269	175750	162662	28798	59747	42644	19050	47	2255	47
7	191088	186553	185407	72354	108485	17880	16152	44	27910	44
8	273541	250460	147901	52365	91182	53	16337	53	11660	53
9	330838	315560	330781	138058	281753	32176	33492	4167	67487	46
10	486106	472625	489087	331803	728325	97439	120281	46	23708	46
Total	1904998	1836889	1703680	891558	1409705	308875	288035	5506	167182	452

Como se puede notar en la tabla, si bien para una cota superior de suboptimalidad o peso igual a 1, ambas implementaciones tienen un rendimiento similar en cuanto a cantidad de expansiones. Sin embargo, a medida que se va aumentando el w, si bien para ambos empiezan a disminuir las expansiones, esto es mucho más pronunciado para la implementación con redes neuronales. Valores superiores de w que los mostrados en la tabla no mejoran las expansiones en el caso de la red neuronal, y mejoran lentamente el algoritmo original, pero nunca alcanzando la eficiencia de la NN.