Informe Proyecto 2

Autor: Adelina Figueira
Carnet: 15-10484

0.1. El problema

Implementar árboles de juego para una variante del juego Othello con 6x6 casillas en lugar de 8x8. Se desea implementar los algoritmos Negamax sin poda alpha-beta, Negamax con poda alpha-beta, Scout y Negascout. Los algoritmos se implementaron basándose en el pseudo código dado en clase, además de modificar el archivo othello_cut.h para verificar las diagonales y los movimientos en estas, se creó una función que retorna los hijos del nodo, llamada moves que da como resultado los movimientos posibles para un jugador dado el color del mismo.
Se realizó la corrida de los algoritmos con un límite de tiempo de 5 min o 300 segundos, debido a que a partir de ciertos valores de la variación principal los algoritmos toman una gran cantidad de tiempo.

0.2. Resultados obtenidos

0.2.1. Negamax sin poda alpha-beta

PV	Expanded	Generated	Seconds	Generated/Seconds	Time spent
34	0	1	0.00000500027	199989	0.00000500027
33	1	2	0.0000109999	181821	0.0000109999
32	3	5	0.0000140001	357140	0.0000140001
31	4	6	0.0000109999	545462	0.0000109999
30	9	13	0.000025	520001	0.000025
29	10	14	0.0000169999	823534	0.0000169999
28	64	91	0.000132	689394	0.000132
27	125	177	0.000286	618881	0.000286
26	744	1049	0.00114	920176	0.00114
25	3168	4498	0.003921	1147160	0.003921
24	8597	11978	0.010018	1195650	0.010018
23	55127	76826	0.032988	2328910	0.032988
22	308479	428402	0.133661	3205140	0.133661
21	2525249	3478735	0.925154	3760170	0.925154
20	9459570	13078933	3.66476	3568830	3.66476
19	65121519	90647895	24.588	3686670	24.588
18	625084814	876269598	279.707	3132820	279.707

El algoritmo Negamax sin poda alpha-beta es el único algoritmo que llega hasta una variación principal de 18, el resto de los algoritmos logran iterar sobre más variaciones en el límite de tiempo establecido.

0.2.2. Negamax con poda alpha-beta

PV	Expanded	Generated	Seconds	Generated/Seconds	Time spent
34	0	1	0	inf	0
33	1	2	0	inf	0
32	3	5	0	inf	0
31	4	6	0	inf	0
30	9	13	0	inf	0
29	10	14	0	inf	0
28	21	27	0	inf	0
27	62	82	0	inf	0
26	186	238	0	inf	0
25	769	1003	0	inf	0
24	1152	1502	0.003793	395993	0.003793
23	3168	4068	0.003652	1.11391e+06	0.003652
22	7031	9130	0.007657	1.19237e+06	0.007657
21	76021	98755	0.057439	1.7193e+06	0.057439
20	98129	127644	0.045296	2.818e+06	0.045296
19	205017	267604	0.096616	2.76977e+06	0.096616
18	960343	1259430	0.467375	2.69469e+06	0.467375
17	1549785	2031924	0.993281	2.04567e+06	0.993281
16	22325108	29501798	13.9136	2.12036e+06	13.9136
15	32949019	43574643	24.1705	1.80281e+06	24.1705
14	82016158	107642871	44.4215	2.42321e+06	44.4215
13	315074162	415909956	193.695	2.14724e+06	193.695

0.2.3. Scout

PV	Expanded	Generated	Seconds	Generated/Seconds	Time spent
34	0	1	3.99956e-06	250027	3.99956e-06
33	1	2	1.00001e-05	199998	1.00001e-05
32	2	5	1.29999e-05	384619	1.29999e-05
31	3	6	1.20001e-05	499996	1.20001e-05
30	11	20	8.70004e-05	229884	8.70004e-05
29	12	21	3.60003e-05	583329	3.60003e-05
28	18	34	4.90001e-05	693876	4.90001e-05
27	35	84	0.000205	409756	0.000205
26	164	401	0.000708001	566384	0.000708001
25	663	1748	0.002082	839578	0.002082
24	982	2622	0.002788	940459	0.002788
23	1540	4141	0.005642	733960	0.005642
22	4674	13285	0.007564	1.75635e+06	0.007564
21	18848	54461	0.034858	1.56237e+06	0.034858
20	29822	89091	0.035468	2.51187e+06	0.035468
19	69511	204351	0.099753	2.04857e+06	0.099753
18	209486	649026	0.233371	2.78109e+06	0.233371
17	391669	1189883	0.440771	2.69955e+06	0.440771
16	2494074	7989969	2.87368	2.7804e+06	2.87368
15	9853100	30953399	14.437	2.14404e+06	14.437
14	23453007	74581266	52.8986	1.40989e+06	52.8986
13	86804999	276935284	126.722	2.18538e+06	126.722

Podemos observar que el algoritmo de Negamax con poda alpha-beta y Scout logran alcanzar ambos una variación principal de 13, Negamax logra generar y expandir una mayor cantidad de estados para una variación de 13, pero Scout logra alcanzar esta variación en una menor cantidad de tiempo.

0.2.4. Negascout

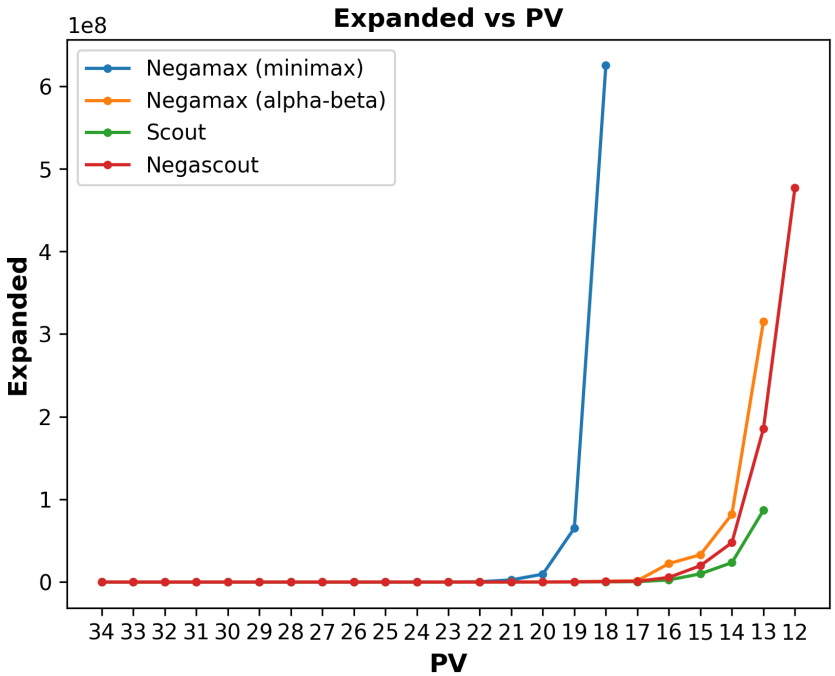
PV	Expanded	Generated	Seconds	Generated/Seconds	Time spent
34	0	0	1.00001e-06	0	1.00001e-06
33	1	1	2.00002e-06	499996	2.00002e-06
32	3	4	3.00002e-06	1.33332e+06	3.00002e-06
31	4	5	2.00002e-06	2.49998e+06	2.00002e-06
30	14	17	6.00005e-06	2.83331e+06	6.00005e-06
29	15	18	6.00005e-06	2.99998e+06	6.00005e-06
28	26	31	1.99999e-05	1.55001e+06	1.99999e-05
27	64	81	4.60001e-05	1.76087e+06	4.60001e-05
26	312	389	0.000156	2.49359e+06	0.000156
25	1275	1631	0.000637	2.56044e+06	0.000637
24	1894	2421	0.000911	2.65752e+06	0.000911
23	3051	3843	0.001341	2.86577e+06	0.001341
22	9329	11979	0.004107	2.91673e+06	0.004107
21	37988	48477	0.023732	2.04268e+06	0.023732
20	63570	81631	0.031525	2.58941e+06	0.031525
19	142595	184144	0.065639	2.80541e+06	0.065639
18	466161	605947	0.222081	2.7285e+06	0.222081
17	870050	1134288	0.403584	2.81054e+06	0.403584
16	5518091	7221811	2.65526	2.71981e+06	2.65526
15	19705373	25831374	10.3405	2.49809e+06	10.3405
14	47600678	62051335	26.0077	2.38588e+06	26.0077
13	185297022	242584981	86.7592	2.79607e+06	86.7592
12	477003110	623011942	246.92	2.52313e+06	246.92

El algoritmo Negascout es el único capaz de alcanzar una variación principal de 12 y Scout logra generar una mayor cantidad de estados para la variación 13 que Negascout.

0.3. Resultados Gráficos

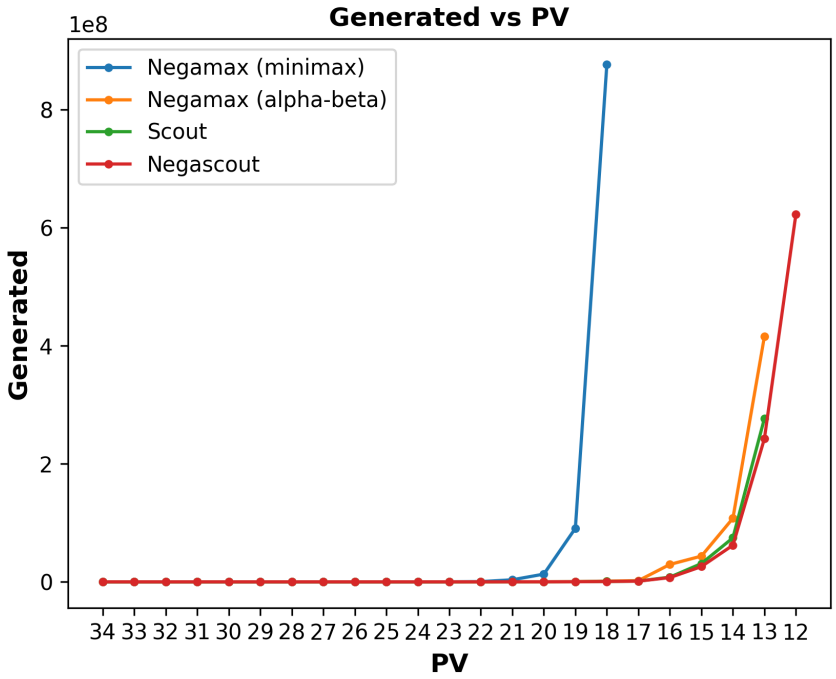
Con las tablas anteriores se crearon los siguientes gráficos.

0.3.1. Gráfico Expanded vs PV



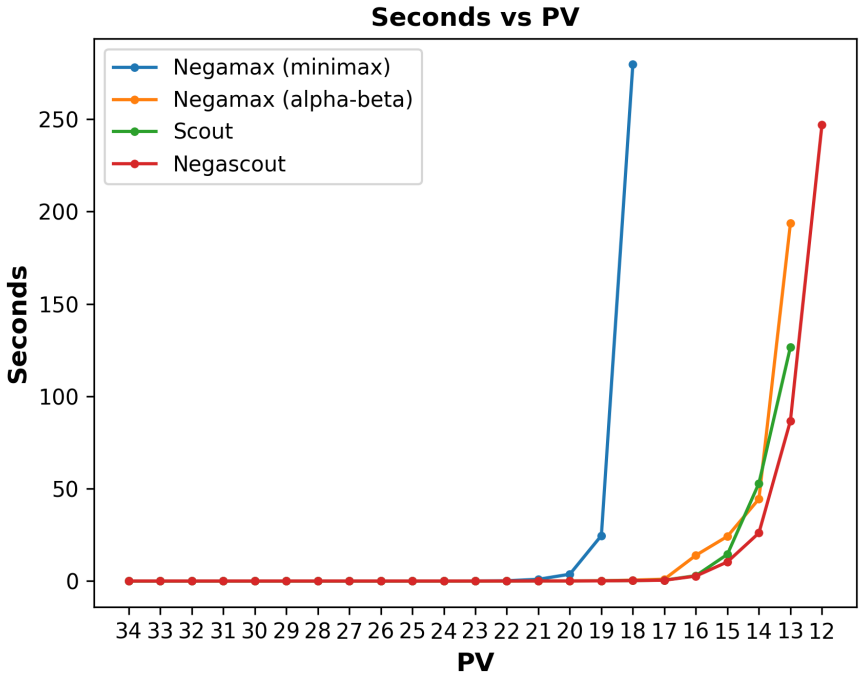
Por la gráfica anterior podemos ver que el algoritmo que expande una mayor cantidad de estados es el algoritmo Negascout seguido por el algoritmo Scout, mientras que el algoritmo con el peor rendimiento para la expansión de los estados es Negamax sin poda alpha-beta.

0.3.2. Gráfico Generated vs PV



Por la gráfica anterior, podemos ver que el algoritmo que logra generar una mayor cantidad de estados es el algoritmo Negascout y al igual que para la expansión, está seguido por el algoritmo Scout y el peor rendimiento lo tiene el algoritmo Negamax sin poda alpha-beta al igual que para la expansión.

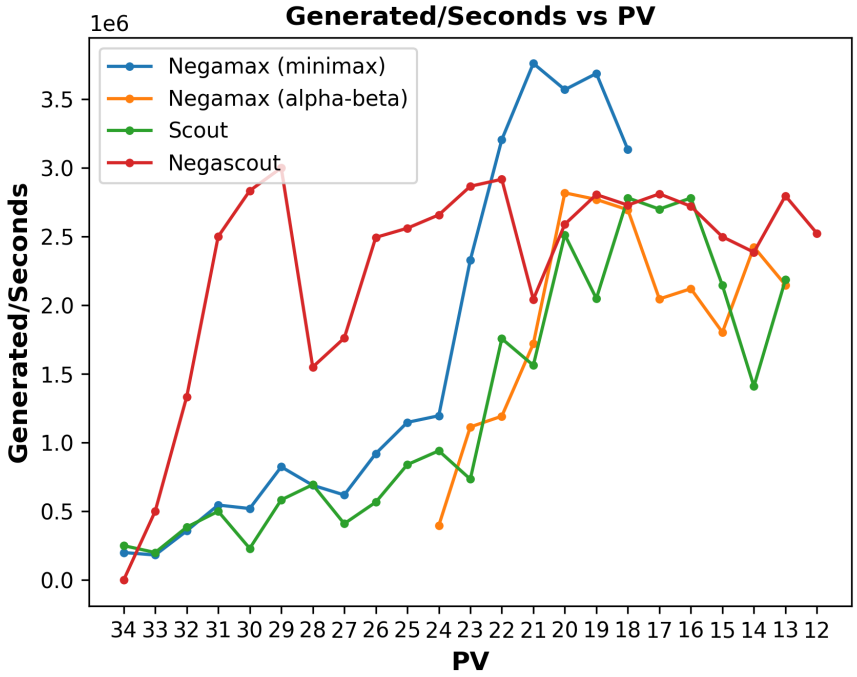
0.3.3. Gráfico Seconds vs PV



Por la gráfica Seconds vs PV podemos observar que el algoritmo que toma una mayor cantidad de tiempo es el de Negamax sin poda alpha-beta, ya que para tan solo una variación principal de 19 toma un tiempo de

24.58 segundos, en comparación a Negascout que tiene el menor tiempo de todos los algoritmos que para una variación principal de 19 toma un tiempo de 0.065639 segundos. El segundo algoritmo más rápido es el Scout y es seguido por Negamax con poda alpha-beta.

0.3.4. Gráfico Generated/Seconds vs PV



El gráfico Generated/Seconds vs PV nos indica la proporción de estados que son generados para la cantidad de segundos que pasan entre cada una de las variaciones principales. Podemos ver que el algoritmo Negamax sin poda alpha-beta tiene un aumento significativo de los estados generados al aumentar también la cantidad de tiempo, por otro lado, el algoritmo Negamax con poda alpha-beta tiene un aumento en la cantidad de generados pero luego este valor desciende.
El algoritmo Scout va en aumento en la cantidad de estados generados desde su inicio, pero tiene una menor proporción para ciertas variaciones principales. Mientras que el algoritmo Negascout es uno de los que comienza generando una gran cantidad de estados y se mantiene generando la misma proporción de estados entre 1.5e6 y 3e6 durante toda la corrida.

0.4. Conclusiones

El algoritmo Negamax sin poda alpha-beta es el algoritmo con peor rendimiento, debido a que toma una gran cantidad de tiempo en comparación al resto de los algoritmos y a pesar de generar una gran cantidad de estados, para generarlos toma mucho tiempo.
Al aplicar la poda alpha-beta en el algoritmo Negamax se mejora su rendimiento pudiendo alcanzar una variación principal de 13 al igual que Scout y Negascout, pero aún así tiene un rendimiento inferior tanto para la cantidad de estados generados, la duración y la expansión de los estados en comparación a los otros dos algoritmos.
Negamax con poda alpha-beta al igual que Scout logra alcanzar la variación principal de 13, es decir, conseguir 21 movimientos de los jugadores, pero el algoritmo Scout tiene un mejor rendimiento que Negamax con poda alpha-beta, ya que, aunque no logra expandir la misma cantidad de estados que este, generar una cantidad de estados similares y Scout puede hacerlo en un tiempo mucho menor a Negamax con poda alpha-beta.
Finalmente, el algoritmo Negascout es el algoritmo que tiene el mejor rendimiento de los algoritmos que se han implementado. Debido a que puede alcanzar una variación principal de 12, es decir, genera 22 movimientos para los jugadores con un tiempo límite de 5 minutos. Además puede generar una cantidad considerable de estados para cada variación y toma un menor tiempo que los algoritmos mencionados anteriormente, lo que le ofrece una gran ventaja, ya que el cálculo de los 33 movimientos tiene un tiempo que se asemeja a la exponencial.
Por esto, es posible afirmar que el mejor algoritmo para el cálculo de los movimientos de los jugadores en un juego de Othello 6x6 con valor -4 es el algoritmo Negascout, debido a su rapidez en comparación al resto de los algoritmos y su capacidad de generar una gran cantidad de estados de manera consecutiva en cada una de las iteraciones.