Christopher Anderson (014105585)

Dr. Markus Eger

CS 4200 Artificial Intelligence

Lab 1

Lab 1 Report

Part I- Algorithm Implementation

For all:

Created dictionary for back pathing

- Was in the form of dict[node_id] = (came from node_id, edge)

Created path, frontier and expandedNodes lists

- Path was used at the end alongside dictionary
- Taking end node as key, appending the edge it came from to path and applying its cost to length
- Make current key = last key tuple[0] element which was the node where it came from and repeat until starting node is current key.

Used while loop until goal was found- exception to this was A* which also checked the rest of the frontier for better pathing.

For all but A*, node was popped, if not in expanded list then add to expanded, extract neighbors, removed already expanded neighbors, checked if already in the frontier- if not then add to frontier and update dictionary.

A* differed by adding an additional if, if they were already in the frontier then check that versions current cost and heuristic value to see if it should be updated with the new one.

BFS:

Main difference in BFS was creating a for loop for frontier length, this loop allowed all current frontier nodes to be processed in groups creating the level by level tree processing of BFS. This was done using a list/queue with .pop(0)

DFS:

Much like BFS but without the forloop and used a stack applying .pop(-1).

Greedy:

Implemented similar to DFS but used .pop(0) for queue style and pre-sorted frontier using frontier.sort(key = heuristic, reverse = False).

A*:

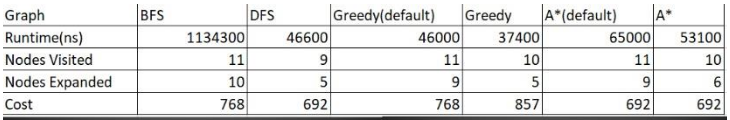
Similar to greedy but created a custom sort function key frontier.sort(key = aStarKey, reverse = False).

A second dictionary was used to store the current cost to get to that node with the node_id as its key. This would be updated when they were added or edited into the frontier.

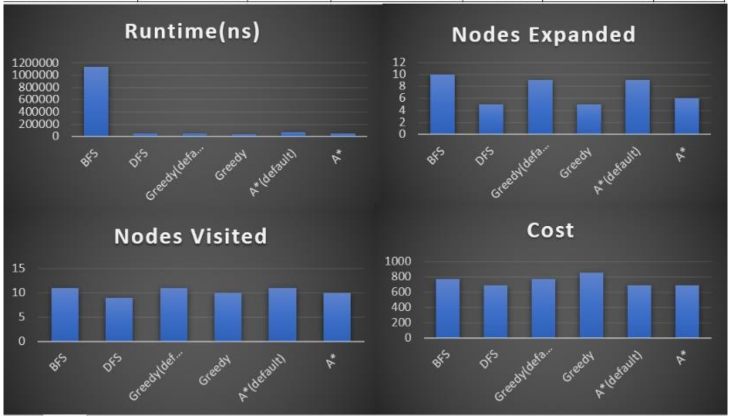
The custom sorting function returned costDict value + heuristic value for sorting.

Part II- Performance:

Austria:



Graph	BFS	DFS	Greedy(default)	Greedy	A*(default)	A*
Runtime(ns)	1134300	46600	46000	37400	65000	53100
Nodes Visited	11	9	11	10	11	10
Nodes Expanded	10	5	9	5	9	6
Cost	768	692	768	857	692	692



Best performances

Runtime: Greedy

Visited: DFS

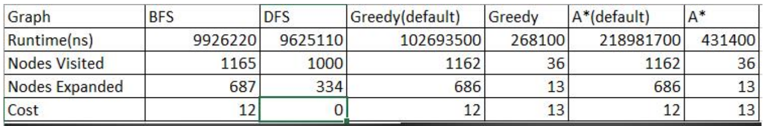
Expanded: DFS, Greedy

Cost: DFS, A* both versions

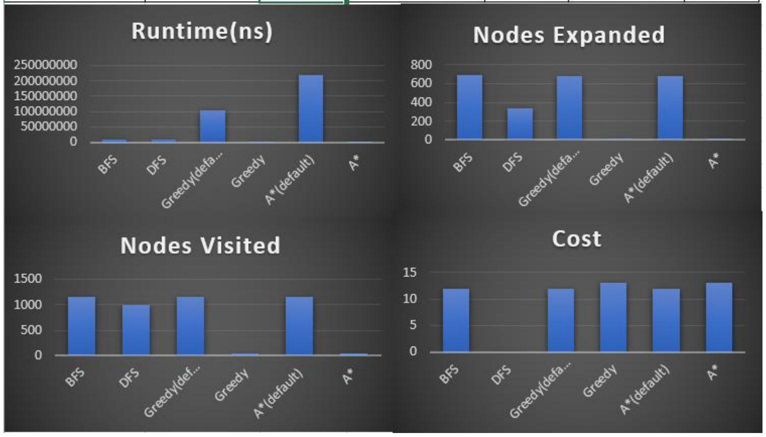
For Austria, most algorithmns performed roughly the same in run time except for BFS which took a very long time in comparison as seen above.

Overall DFS and A* performed the best in cost, but DFS came out slightly ahead due to getting lucky on pathing.

Inf Graph(simple):



Graph	BFS	DFS	Greedy(default)	Greedy	A*(default)	A*
Runtime(ns)	9926220	9625110	102693500	268100	218981700	431400
Nodes Visited	1165	1000	1162	36	1162	36
Nodes Expanded	687	334	686	13	686	13
Cost	12	0	12	13	12	13



Best performances

Runtime:Greedy

Visited: Greedy, A*

Expanded: Greedy, A*

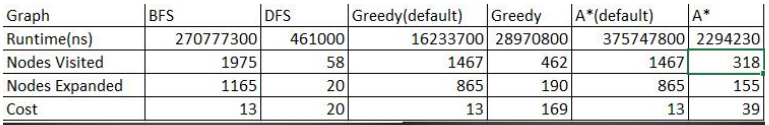
Cost: BFS, Greedy(default), A*(default)

DFS failed this one.

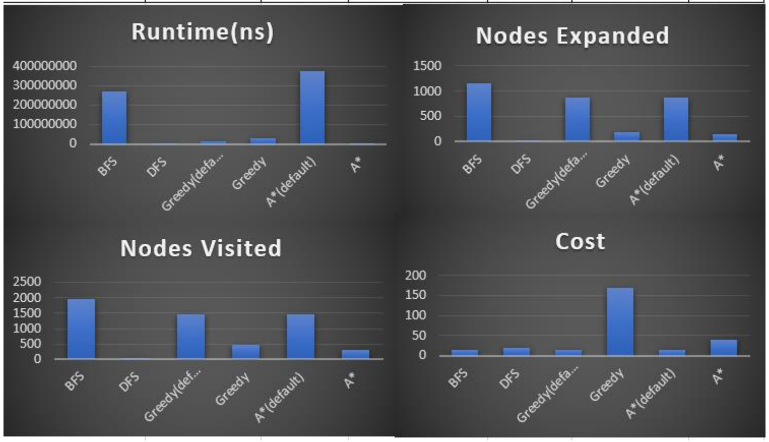
From the data we can see using heuristics greatly reduced runtime, nodes expanded and visited.

All versions were relatively close in cost besides DFS which failed to find a path.

Inf Graph(Multi):



Graph	BFS	DFS	Greedy(default)	Greedy	A*(default)	A*
Runtime(ns)	270777300	461000	16233700	28970800	375747800	2294230
Nodes Visited	1975	58	1467	462	1467	318
Nodes Expanded	1165	20	865	190	865	155
Cost	13	20	13	169	13	39



Best performances

Runtime: DFS

Visited: DFS

Expanded: DFS

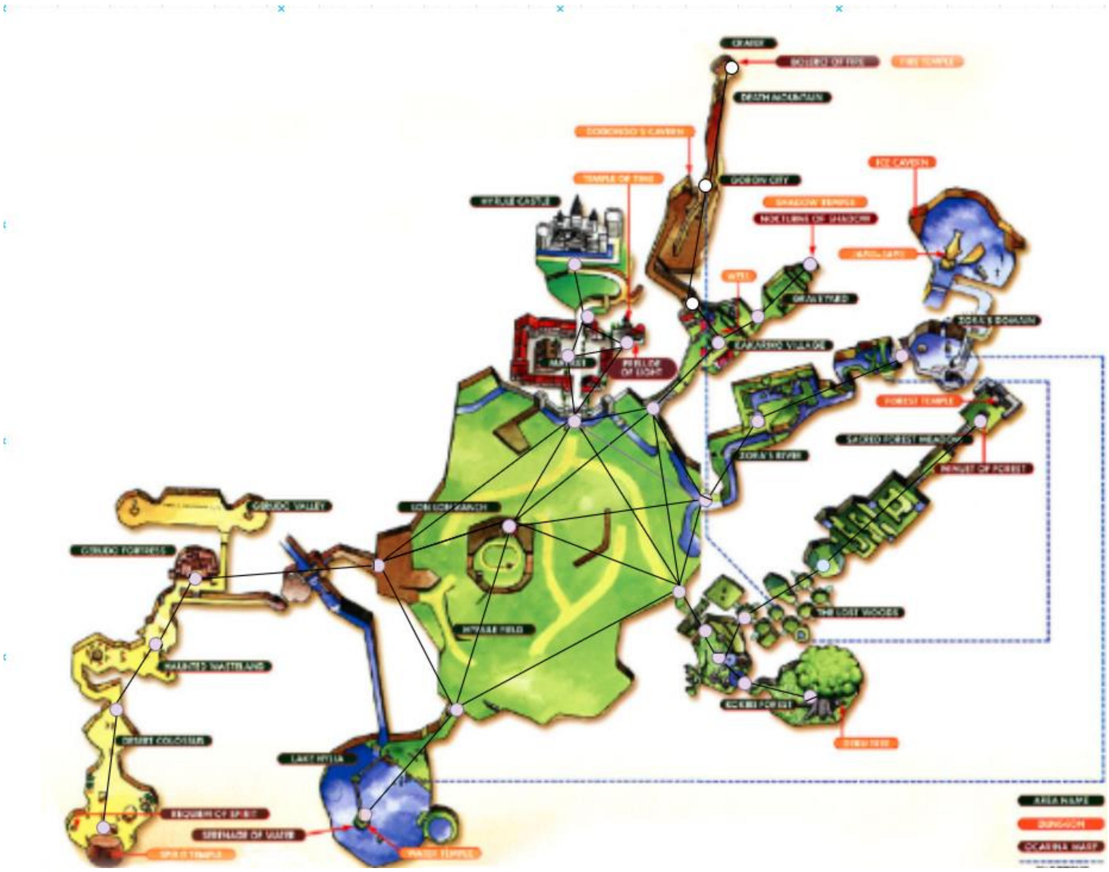
Cost: BFS, Greedy(default), A*(default)

For runtime both BFS and A* using default heuristics took exceptionally long compared to the others.

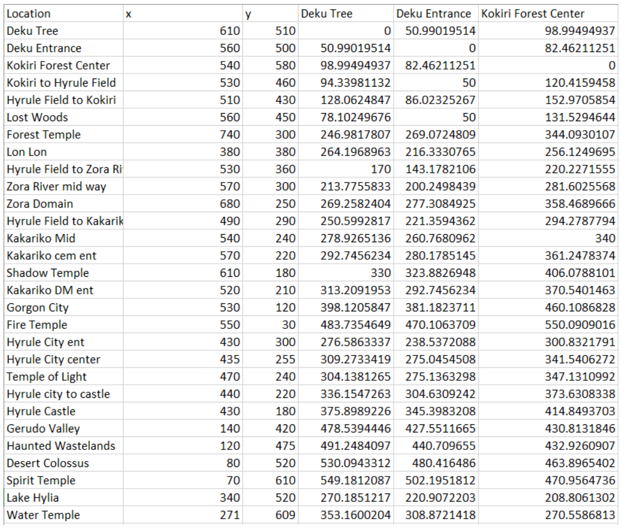
DFS, appears to be the best overall for this test, but it could of also just got lucky on pathing as it did before- needs further testing on different graphs.

The biggest surprise in this test was how the graphs using non-default heuristics did, while they did a better job going through fewer nodes, the cost they output wasn't nearly as good as default.

Part III- Custom Graph – Hyrule from Legend of Zelda: Ocarina of Time

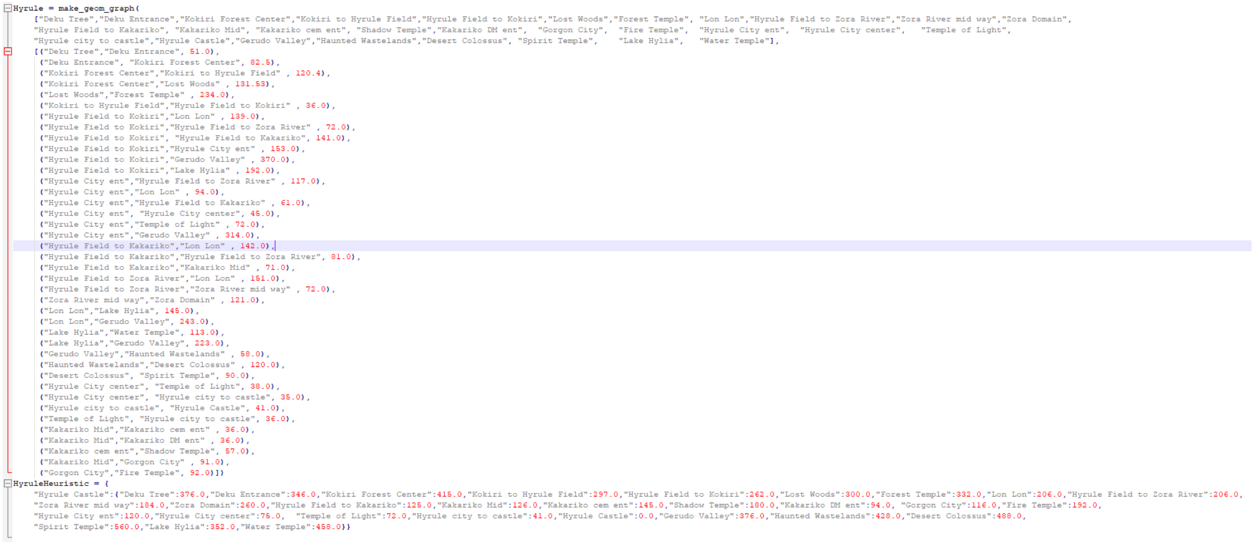


For this graph, draw.io was used and I drew over a clipped picture I inserted. Attempting to move circles gave x, y coordinates which I input into excel.



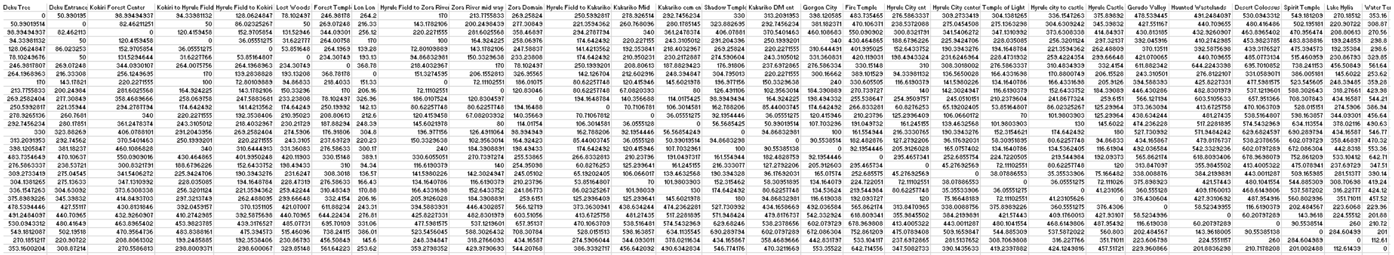
Location	x	y	Deku Tree	Deku Entrance	Kokiri Forest Center
Deku Tree	610	510	0	50.99019514	98.99494937
Deku Entrance	560	500	50.99019514	0	82.46211251
Kokiri Forest Center	540	580	98.99494937	82.46211251	0
Kokiri to Hyrule Field	530	460	94.33981132	50	120.4159458
Hyrule Field to Kokiri	510	430	128.0624847	86.02325267	152.9705854
Lost Woods	560	450	78.10249676	50	131.5294644
Forest Temple	740	300	246.9817807	269.0724809	344.0930107
Lon Lon	380	380	264.1968963	216.3330765	256.1249695
Hyrule Field to Zora Ri	530	360	170	143.1782106	220.2271555
Zora River mid way	570	300	213.7755833	200.2498439	281.6025568
Zora Domain	680	250	269.2582404	277.3084925	358.4689666
Hyrule Field to Kakarik	490	290	250.5992817	221.3594362	294.2787794
Kakariko Mid	540	240	278.9265136	260.7680962	340
Kakariko cem ent	570	220	292.7456234	280.1785145	361.2478374
Shadow Temple	610	180	330	323.8826948	406.0788101
Kakariko DM ent	520	210	313.2091953	292.7456234	370.5401463
Gorgon City	530	120	398.1205847	381.1823711	460.1086828
Fire Temple	550	30	483.7354649	470.1063709	550.0909016
Hyrule City ent	430	300	276.5863337	238.5372088	300.8321791
Hyrule City center	435	255	309.2733419	275.0454508	341.5406272
Temple of Light	470	240	304.1381265	275.1363298	347.1310992
Hyrule city to castle	440	220	336.1547263	304.6309242	373.6308338
Hyrule Castle	430	180	375.8989226	345.3983208	414.8493703
Gerudo Valley	140	420	478.5394446	427.5511665	430.8131846
Haunted Wastelands	120	475	491.2484097	440.709655	432.9260907
Desert Colossus	80	520	530.0943312	480.416486	463.8965402
Spirit Temple	70	610	549.1812087	502.1951812	470.9564736
Lake Hylia	340	520	270.1851217	220.9072203	208.8061302
Water Temple	271	609	353.1600204	308.8721418	270.5586813

Here is in example of the excel spread sheet, using the distance formula to find the differences between nodes to generate edge length and heuristic straight line distance.



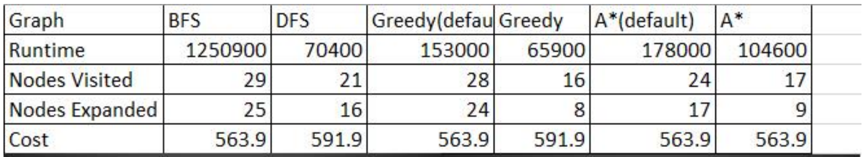
```
Hyrule = make_geom_graph(
    ["Deku Tree","Deku Entrance","Kokiri Forest Center","Kokiri to Hyrule Field","Hyrule Field to Kokiri","Lost Woods","Forest Temple", "Lon Lon","Hyrule Field to Zora River","Zora River mid way","Zora Domain",
    "Hyrule Field to Kakariko", "Kakariko Mid", "Kakariko cem ent", "Shadow Temple","Kakariko DM ent",  "Gorgon City",  "Fire Temple",  "Hyrule City ent",  "Hyrule City center",   "Temple of Light",
    "Hyrule city to castle","Hyrule Castle","Gerudo Valley","Haunted Wastelands","Desert Colossus", "Spirit Temple",    "Lake Hylia",   "Water Temple"],
    [("Deku Tree","Deku Entrance", 51.0),
     ("Deku Entrance", "Kokiri Forest Center", 82.5),
     ("Kokiri Forest Center","Kokiri to Hyrule Field" , 120.4),
     ("Kokiri Forest Center","Lost Woods" , 131.53),
     ("Lost Woods","Forest Temple" , 234.0),
     ("Kokiri to Hyrule Field","Hyrule Field to Kokiri" , 36.0),
     ("Hyrule Field to Kokiri","Lon Lon" , 139.0),
     ("Hyrule Field to Kokiri","Hyrule Field to Zora River" , 72.0),
     ("Hyrule Field to Kokiri", "Hyrule Field to Kakariko", 141.0),
     ("Hyrule Field to Kokiri","Hyrule City ent" , 153.0),
     ("Hyrule Field to Kokiri","Gerudo Valley" , 370.0),
     ("Hyrule Field to Kokiri","Lake Hylia" , 192.0),
     ("Hyrule City ent","Hyrule Field to Zora River" , 117.0),
     ("Hyrule City ent","Lon Lon" , 94.0),
     ("Hyrule City ent","Hyrule Field to Kakariko" , 61.0),
     ("Hyrule City ent", "Hyrule City center", 45.0),
     ("Hyrule City ent","Temple of Light" , 72.0),
     ("Hyrule City ent","Gerudo Valley" , 314.0),
     ("Hyrule Field to Kakariko","Lon Lon" , 142.0),
     ("Hyrule Field to Kakariko","Hyrule Field to Zora River", 81.0),
     ("Hyrule Field to Kakariko","Kakariko Mid" , 71.0),
     ("Hyrule Field to Zora River","Lon Lon" , 151.0),
     ("Hyrule Field to Zora River","Zora River mid way" , 72.0),
     ("Zora River mid way","Zora Domain" , 121.0),
     ("Lon Lon","Lake Hylia", 145.0),
     ("Lon Lon","Gerudo Valley", 243.0),
     ("Lake Hylia","Water Temple", 113.0),
     ("Lake Hylia","Gerudo Valley", 223.0),
     ("Gerudo Valley","Haunted Wastelands" , 58.0),
     ("Haunted Wastelands","Desert Colossus" , 120.0),
     ("Desert Colossus", "Spirit Temple", 90.0),
     ("Hyrule City center", "Temple of Light", 38.0),
     ("Hyrule City center", "Hyrule city to castle", 35.0),
     ("Hyrule city to castle", "Hyrule Castle", 41.0),
     ("Temple of Light", "Hyrule city to castle", 36.0),
     ("Kakariko Mid","Kakariko cem ent" , 36.0),
     ("Kakariko Mid","Kakariko DM ent" , 36.0),
     ("Kakariko cem ent","Shadow Temple", 57.0),
     ("Kakariko Mid","Gorgon City" , 91.0),
     ("Gorgon City","Fire Temple", 92.0)])
HyruleHeuristic = {
    "Hyrule Castle":{"Deku Tree":376.0,"Deku Entrance":346.0,"Kokiri Forest Center":415.0,"Kokiri to Hyrule Field":297.0,"Hyrule Field to Kokiri":262.0,"Lost Woods":300.0,"Forest Temple":332.0,"Lon Lon":206.0,"Hyrule Field to Zora River":206.0,
    "Zora River mid way":184.0,"Zora Domain":260.0,"Hyrule Field to Kakariko":125.0,"Kakariko Mid":126.0,"Kakariko cem ent":145.0,"Shadow Temple":180.0,"Kakariko DM ent":94.0, "Gorgon City":116.0,"Fire Temple":192.0,
    "Hyrule City ent":120.0,"Hyrule City center":75.0,  "Temple of Light":72.0,"Hyrule city to castle":41.0,"Hyrule Castle":0.0,"Gerudo Valley":376.0,"Haunted Wastelands":428.0,"Desert Colossus":488.0,
    "Spirit Temple":560.0,"Lake Hylia":352.0,"Water Temple":458.0}}
```

I then built the graph using the same style used for Austria, for the Heuristic I only included one for Hyrule castle since that is what I was testing on- and the excel sheet was massive to include all 28 x 28 values.

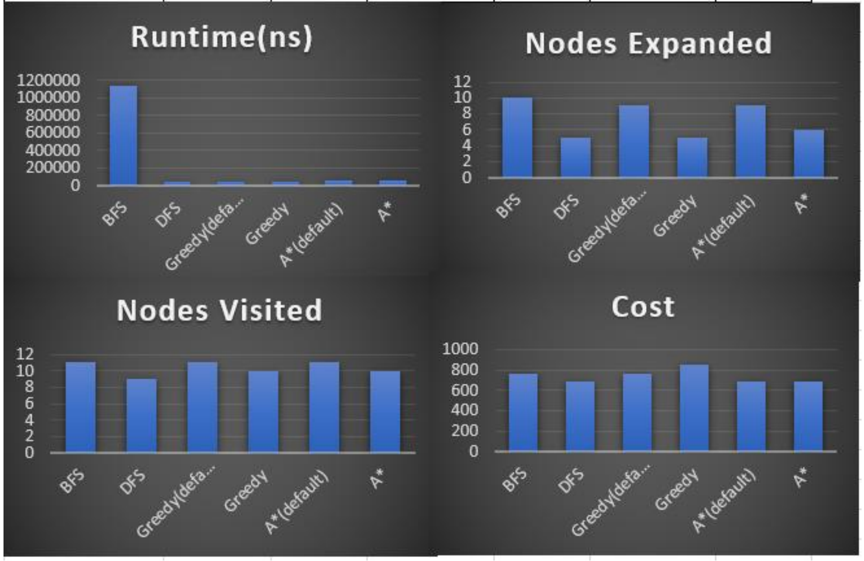


Deku Tree	Deku Entrance	Kokiri Forest Center	Kokiri to Hyrule Field	Hyrule Field to Kokiri	Lost Woods	Forest Templ	Lon Lon	Hyrule Field to Zora Rive	Zora River mid way	Zora Domain	Hyrule Field to Kakariko	Kakariko Mid	Kakariko cem en	Shadow Templ	Kakariko DM ent	Gorgon City	Fire Temple	Hyrule City ent	Hyrule City center	Temple of Light	Hyrule city to castle	Hyrule Castle	Gerudo Valley	Haunted Wastelands	Desert Colossus	Spirit Temple	Lake Hylia	Water Te
0	50.990195	98.99494937	94.33981132	128.0624847	78.102497	246.98178	264.2	170	213.7755833	269.25824	250.5992817	278.926514	292.7456234	330	313.2091953	398.120585	483.735465	276.5863337	309.2733419	304.1381265	336.1547263	375.89892	478.539445	491.2484097	530.0943312	549.181209	270.18512	353.16
50.99019514	0	82.46211251	50	86.02325267	50	269.07248	216.33	143.1782106	200.2498439	277.30849	221.3594362	260.768096	280.1785145	323.882695	292.7456234	381.182371	470.106371	238.5372088	275.0454508	275.1363298	304.6309242	345.39832	427.551167	440.709655	480.416486	502.195181	220.90722	308.87
98.99494937	82.462113	0	120.4159458	152.9705854	131.52946	344.09301	256.12	220.2271555	281.6025568	358.46897	294.2787794	340	361.2478374	406.07881	370.5401463	460.108683	550.090902	300.8321791	341.5406272	347.1310992	373.6308338	414.84937	430.813185	432.9260907	463.8965402	470.956474	208.80613	270.56
94.33981132	50	120.4159458	0	36.05551275	31.622777	264.00758	170	100	164.924225	258.06976	174.642492	220.227155	243.3105012	291.204396	250.1999201	340	430.464865	188.6796226	225.9424706	228.035085	256.3201124	297.32137	392.045916	410.2742985	453.9823785	483.838816	199.24859	298.8
128.0624847	86.023253	152.9705854	36.05551275	0	53.851648	264.1969	139.28	72.80109889	143.1782106	247.58837	141.4213562	192.353841	218.4032967	269.25824	220.2271555	310.644491	401.995025	152.6433752	190.3943276	194.1648784	221.3594362	262.48809	370.13511	392.5875698	439.3176527	475.394573	192.35384	298.6
78.10249676	50	131.5294644	31.6227766	53.85164807	0	234.30749	193.13	94.86832981	150.3329638	233.23808	174.642492	210.950231	230.2172887	274.590604	243.3105012	331.360831	420.119031	198.4943324	231.6246964	228.4731932	259.4224354	299.66648	421.070065	440.709655	485.0773134	515.460959	230.86793	329.85
246.9817807	269.07248	344.0930107	264.0075756	264.1968963	234.30749	0	368.78	218.4032967	170	78.102497	250.1999201	208.80613	187.8829423	176.91806	237.6972865	276.586334	330.15148	310	308.3018002	276.5863337	310.4834939	332.4154	611.882342	644.2243398	695.7010852	738.241153	456.50849	561.64
264.1968963	216.33308	256.1249695	170	139.2838828	193.13208	368.78178	0	151.3274595	206.1552813	326.95565	142.126704	212.602916	248.394847	304.795013	220.2271555	300.16662	389.101529	94.33981132	136.5650028	166.4331698	170.8800749	206.15528	243.310501	276.8122107	331.0589071	386.005181	145.6022	253.62
170	143.17821	220.2271555	100	72.80109889	94.86833	218.4033	151.33	0	72.11102551	186.01075	80.62257748	120.415946	145.6021978	196.977156	150.3329638	240	330.605505	116.6190379	141.5980226	134.1640786	166.4331698	205.9126	394.588393	425.8227331	477.5981575	523.545605	248.39485	359.28
213.7755833	200.24984	281.6025568	164.924225	143.1782106	150.33296	170	206.16	72.11102551	0	120.83046	80.62257748	67.0820393	80	126.491106	102.9563014	184.390889	270.739727	140	142.3024947	116.6190379	152.6433752	184.39089	446.430286	482.8301979	537.1219601	588.302643	318.27661	429.98
269.2582404	277.30849	358.4689666	258.069758	247.5883681	233.23808	78.102497	326.96	186.0107524	120.8304597	0	194.1648784	140.356688	114.0175425	98.9949494	164.924225	198.494332	255.538647	254.9509757	245.0510151	210.2379604	241.8677324	259.6151	566.127194	603.5105633	657.951366	708.307843	434.16587	544.21
250.5992817	221.35944	294.2787794	174.642492	141.4213562	174.64249	250.19992	142.13	80.62257748	80.62257748	194.16488	0	70.7106781	106.3014581	162.788206	85.44003745	174.642492	266.833281	60.8276253	65.19202405	53.85164807	86.02325267	125.29964	373.363094	413.6725758	470.1063709	528.015151	274.5906	386.94
278.9265136	260.7681	340	220.2271555	192.3538406	210.95023	208.80613	212.6	120.4159458	67.08203932	140.35669	70.71067812	0	36.05551275	92.1954446	36.05551275	120.415946	210.23796	125.2996409	106.0660172	70	101.9803903	125.29964	438.634244	481.27435	538.5164807	598.163857	344.09301	456.64
292.7456234	280.17851	361.2478374	243.3105012	218.4032967	230.21729	187.88294	248.39	145.6021978	80	114.01754	106.3014581	36.0555128	0	56.5685425	50.99019514	107.703296	191.049732	161.245155	139.4632568	101.9803903	130	145.6022	474.236228	517.2281895	574.5432969	634.113554	378.02116	490.63
330	323.88269	406.0788101	291.2043956	269.2582404	274.5906	176.91806	304.8	196.977156	126.4911064	98.994949	162.788206	92.1954446	56.56854249	0	94.86832981	100	161.554944	216.3330765	190.3943276	152.3154621	174.642492	180	527.730992	571.9484242	629.6824597	690.289794	434.16587	546.77
313.2091953	292.74562	370.5401463	250.1999201	220.2271555	243.3105	237.69729	220.23	150.3329638	102.9563014	164.92423	85.44003745	36.0555128	50.99019514	94.8683298	0	90.5538514	182.482876	127.2792206	96.17692031	58.30951895	80.62257748	94.86833	434.165867	479.8176737	538.2378656	602.079729	358.46897	470.32
398.1205847	381.18237	460.1086828	340	310.6444913	331.36083	276.58633	300.17	240	184.3908891	198.49433	174.642492	120.415946	107.7032961	100	90.55385138	0	92.1954446	205.9126028	165.0757402	134.1640786	134.5362405	116.61904	492.036584	542.3329236	602.0797289	672.086304	442.8318	553.36
483.7354649	470.10637	550.0909016	430.464865	401.9950248	420.11903	330.15148	389.1	330.6055051	270.7397274	255.53865	266.8332813	210.23796	191.0497317	161.554944	182.4828759	92.1954446	0	295.4657341	252.6855754	224.7220505	219.544984	192.09373	565.862174	618.8093406	678.9698079	752.861209	533.10412	642.71
276.5863337	238.53721	300.8321791	188.6796226	152.6433752	198.49433	310	94.34	116.6190379	140	254.95098	60.8276253	125.299641	161.245155	216.333077	127.2792206	205.912603	295.465734	0	45.27692569	72.11102551	80.62257748	120	313.847097	355.9845502	413.4005322	475.078941	237.69729	347.51
309.2733419	275.04545	341.5406272	225.9424706	190.3943276	231.6247	308.3018	136.57	141.5980226	142.3024947	245.05102	65.19202405	106.066017	139.4632568	190.394328	96.17692031	165.07574	252.685575	45.27692569	0	38.07886553	35.35533906	75.166482	338.008876	384.2199891	443.0011287	509.165985	281.51377	390.14
304.1381265	275.13633	347.1310992	228.035085	194.1648784	228.47319	276.58633	166.43	134.1640786	116.6190379	210.23796	53.85164807	70	101.9803903	152.315462	58.30951895	134.164079	224.722051	72.11102551	38.07886553	0	36.0551275	72.111026	375.898923	421.57443	480.1041554	544.885309	308.70698	419.24
336.1547263	304.63092	373.6308338	256.3201124	221.3594362	259.42244	310.48349	170.88	166.4331698	152.6433752	241.86773	86.02325267	101.98039	130	174.642492	80.62257748	134.53624	219.544984	80.62257748	35.35533906	36.0551275	0	41.231056	360.555128	409.1760013	468.6149806	537.587202	316.22777	424.12
375.8989226	345.39832	414.8493703	297.3213749	262.488095	299.66648	332.4154	206.16	205.9126028	184.3908891	259.6151	125.2996409	125.299641	145.6021978	180	94.86832981	116.619038	192.093727	120	75.16648189	72.11102551	41.23105626	0	376.430604	427.9310692	487.954916	560.802996	351.71011	457.52
478.5394446	427.55117	430.8131846	392.0459157	370.1351105	421.07007	611.88234	243.31	394.5883931	446.4302857	566.12719	373.3630941	438.634244	474.2362281	527.730992	434.1658669	492.036584	565.862174	313.8470965	338.0088756	375.8989226	360.5551275	376.4306	0	58.52349955	116.6190379	202.484567	223.6068	229.96
491.2484097	440.70965	432.9260907	410.2742985	392.5875698	440.70965	644.22434	276.81	425.8227331	482.8301979	603.51056	413.6725758	481.27435	517.2281895	571.948424	479.8176737	542.332924	618.809341	355.9845502	384.2199891	421.57443	409.1760013	427.93107	58.5234996	0	60.20797289	143.9618	224.55512	201.88
530.0943312	480.41649	463.8965402	453.9823785	439.3176527	485.07731	695.70109	331.06	477.5981575	537.1219601	657.95137	470.1063709	538.516481	574.5432969	629.68246	538.2378656	602.079729	678.969808	413.4005322	443.0011287	480.1041554	468.6149806	487.95492	116.619038	60.20797289	0	90.5538514	260	210.72
549.1812087	502.19518	470.9564736	483.8388161	475.394573	515.46096	738.24115	386.01	523.5456045	588.3026432	708.30784	528.0151513	598.163857	634.1135545	690.289794	602.0797289	672.086304	752.861209	475.0789408	509.1659847	544.885309	537.5872022	560.803	202.484567	143.9618005	90.55385138	0	284.60499	201
270.1851217	220.90722	208.8061302	199.2485885	192.3538406	230.86793	456.50849	145.6	248.394847	318.2766093	434.16587	274.5906044	344.093011	378.0211634	434.165867	358.4689666	442.831797	533.104117	237.6972865	281.5137652	308.7069808	316.227766	351.71011	223.606798	224.5551157	260	284.604989	0	112.61
353.1600204	308.87214	270.5586813	298.8009371	298.600067	329.85148	561.64223	253.62	359.2798352	429.9790693	544.20768	386.9392717	456.642092	490.6342834	546.774176	470.3211669	553.35522	642.714556	347.5082733	390.1435633	419.2397882	424.1249816	457.51721	229.960866	201.8836298	210.7178208	201.002488	112.61439	0

Hyrule performance:



Graph	BFS	DFS	Greedy(defau	Greedy	A*(default)	A*
Runtime	1250900	70400	153000	65900	178000	104600
Nodes Visited	29	21	28	16	24	17
Nodes Expanded	25	16	24	8	17	9
Cost	563.9	591.9	563.9	591.9	563.9	563.9



Best performances

Runtime: Greedy

Visited: A*

Expanded: Greedy

Cost: BFS, Greedy(default), A* both versions

My algorithms performed very similar to the Austria graph. This time DFS didn't get as lucky but BFS ended up tying in total cost with the heuristics but at the cost of a much greater runtime. All algorithmns actually performed decent with roughly 30 units difference in their cost.

The main thing I was testing in this graph compared to the Austria graph was the many dead ends available.