Inteligência Artificial para Jogos

2016-2017

2º Project Report

Group 19:

Sofia Augusto – 78302
Tiago Baltazar – 78583
Pedro Lopes – 78910

Level 3 – Cluster Graph:

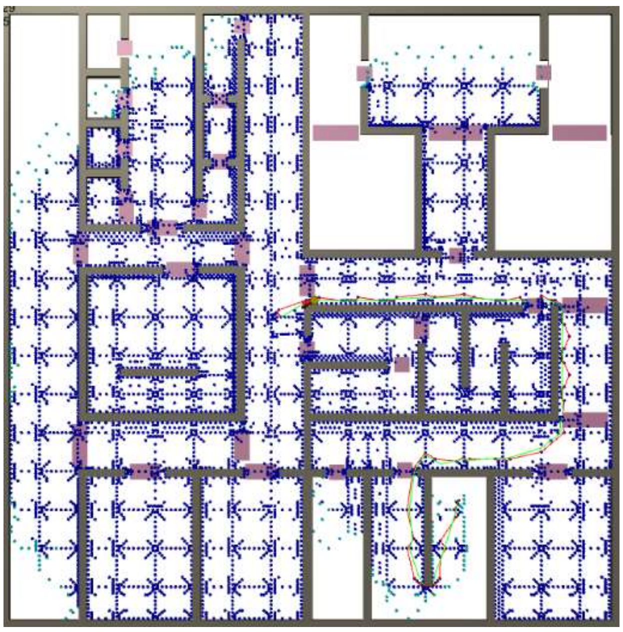
In order to implement the cluster graph creation, we iterate a beginGate(from the startCluster) and do a search with a NodeArrayA*Search algorithm with every combinationGate(from the endCluste). The minimum value found from Gate A to Gate B is then stored with the indexes corresponding to the Gate's id.

This indexation is straight forward and very simple to use. All these values are then stored within Scriptable Objects of the required template, to be serialized by Unity.

Level 4 – Comparing the pathfinding algorithms:

AStarPathfinding:

- Priority heap as open list;
- Dictionary as closed list;
- Using Euclidean Distance;



Total Nodes explored: 8018

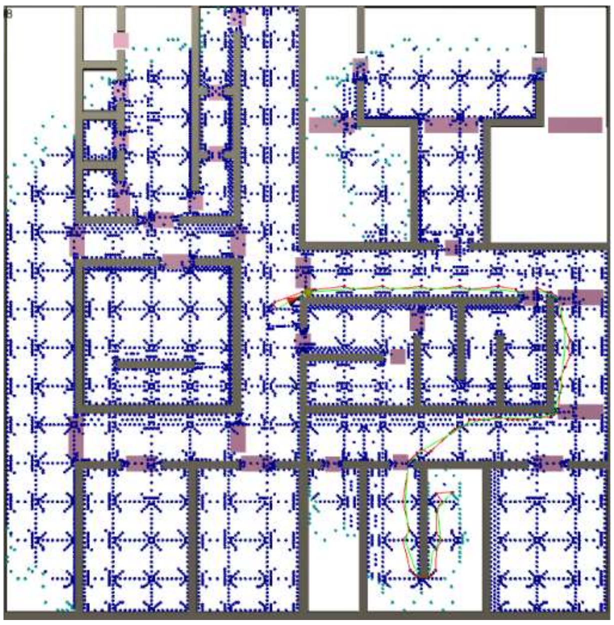
Max Open Nodes: 262

Total Time: 416.0782

Time per Nodes: 0.05402

NodeArrayAStarPathFinding:

- Using Euclidean Distance



Total Nodes explored: 8228

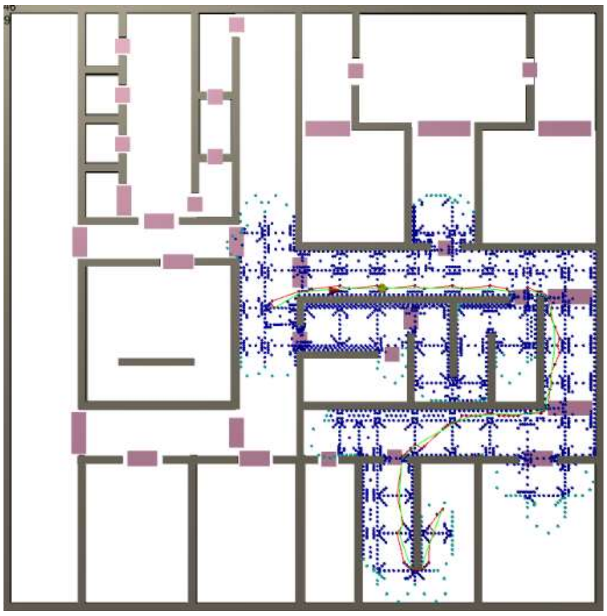
Max Open Nodes: 256

Total Time: 195.6415

Time per Nodes: 0.0240

NodeArrayAStarPathFinding:

- Using Gateway



Total Nodes explored: 4289

Max Open Nodes: 209

Total Time: 1344.5990

Time per Nodes: 0.3134994

Performance Analysis:

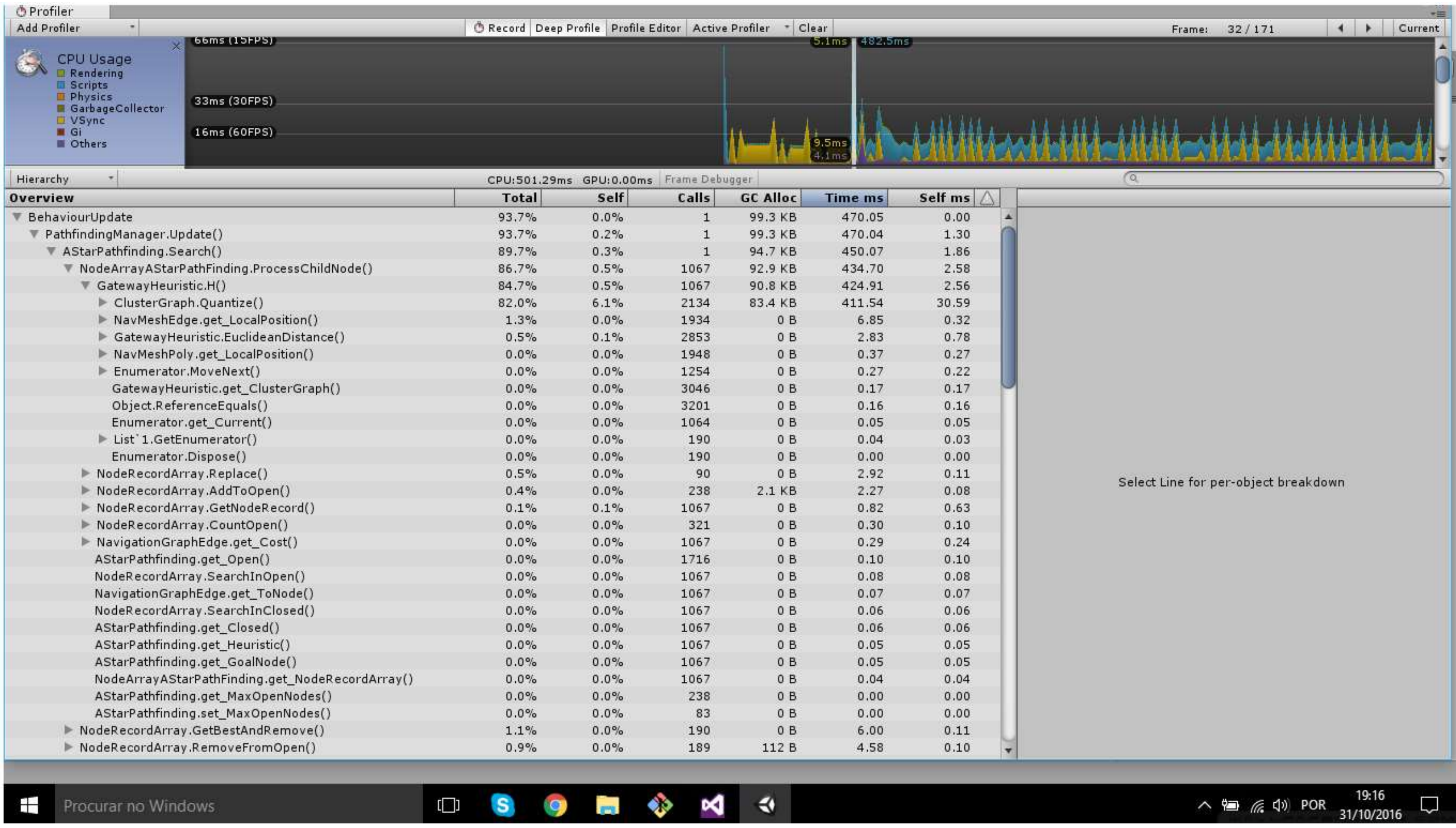
In terms of total processing time the NodeArrayAStarPathFinding search with Euclidean Distance Heuristic is better comparing to the other two implementations. However, in terms of total nodes explored, the variation with Gateway Heuristic has half of the explored nodes and the lesser fill of the map.

This is due to the fact that the Gateway Heuristic computes several Euclidean Distances for each H value of a node, plus two quantize methods.

Level 6 – Optimizations:

Pre Optimization:

The cluster graph was implemented exactly as described in level 3, in which we identified the Quantize method as the most expensive, due to its number of calls (dependent on the chosen path, in this case around 2134 calls as shown in the image below) and time spent searching for the corresponding cluster of a given node.



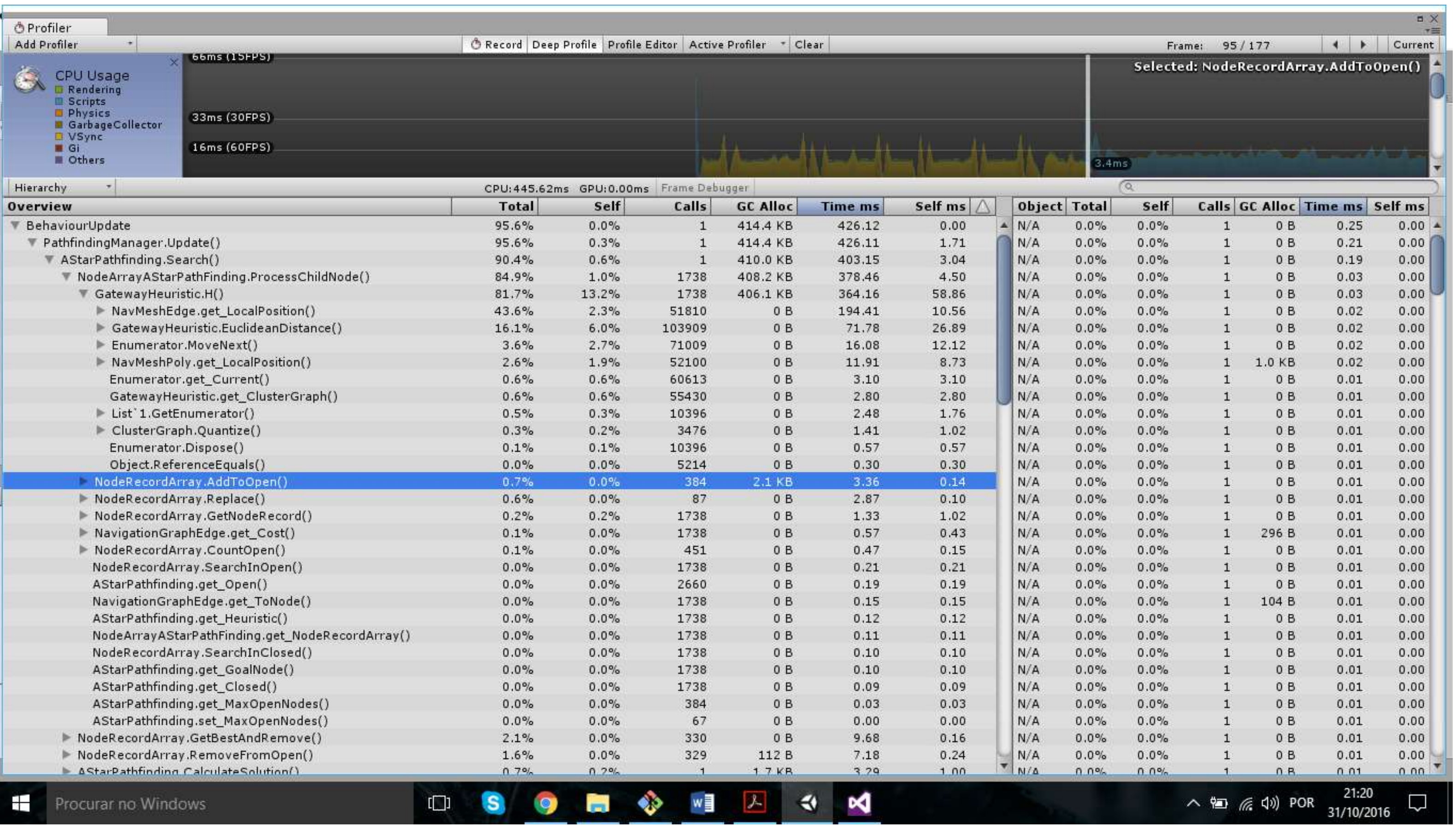
CPU:501.29ms GPU:0.00ms

Overview	Total	Self	Calls	GC Alloc	Time ms	Self ms
▼ BehaviourUpdate	93.7%	0.0%	1	99.3 KB	470.05	0.00
▼ PathfindingManager.Update()	93.7%	0.2%	1	99.3 KB	470.04	1.30
▼ AStarPathfinding.Search()	89.7%	0.3%	1	94.7 KB	450.07	1.86
▼ NodeArrayAStarPathFinding.ProcessChildNode()	86.7%	0.5%	1067	92.9 KB	434.70	2.58
▼ GatewayHeuristic.H()	84.7%	0.5%	1067	90.8 KB	424.91	2.56
▶ ClusterGraph.Quantize()	82.0%	6.1%	2134	83.4 KB	411.54	30.59
▶ NavMeshEdge.get_LocalPosition()	1.3%	0.0%	1934	0 B	6.85	0.32
▶ GatewayHeuristic.EuclideanDistance()	0.5%	0.1%	2853	0 B	2.83	0.78
▶ NavMeshPoly.get_LocalPosition()	0.0%	0.0%	1948	0 B	0.37	0.27
▶ Enumerator.MoveNext()	0.0%	0.0%	1254	0 B	0.27	0.22
GatewayHeuristic.get_ClusterGraph()	0.0%	0.0%	3046	0 B	0.17	0.17
Object.ReferenceEquals()	0.0%	0.0%	3201	0 B	0.16	0.16
Enumerator.get_Current()	0.0%	0.0%	1064	0 B	0.05	0.05
▶ List`1.GetEnumerator()	0.0%	0.0%	190	0 B	0.04	0.03
Enumerator.Dispose()	0.0%	0.0%	190	0 B	0.00	0.00
▶ NodeRecordArray.Replace()	0.5%	0.0%	90	0 B	2.92	0.11
▶ NodeRecordArray.AddToOpen()	0.4%	0.0%	238	2.1 KB	2.27	0.08
▶ NodeRecordArray.GetNodeRecord()	0.1%	0.1%	1067	0 B	0.82	0.63
▶ NodeRecordArray.CountOpen()	0.0%	0.0%	321	0 B	0.30	0.10
▶ NavigationGraphEdge.get_Cost()	0.0%	0.0%	1067	0 B	0.29	0.24
AStarPathfinding.get_Open()	0.0%	0.0%	1716	0 B	0.10	0.10
NodeRecordArray.SearchInOpen()	0.0%	0.0%	1067	0 B	0.08	0.08
NavigationGraphEdge.get_ToNode()	0.0%	0.0%	1067	0 B	0.07	0.07
NodeRecordArray.SearchInClosed()	0.0%	0.0%	1067	0 B	0.06	0.06
AStarPathfinding.get_Closed()	0.0%	0.0%	1067	0 B	0.06	0.06
AStarPathfinding.get_Heuristic()	0.0%	0.0%	1067	0 B	0.05	0.05
AStarPathfinding.get_GoalNode()	0.0%	0.0%	1067	0 B	0.05	0.05
NodeArrayAStarPathFinding.get_NodeRecordArray()	0.0%	0.0%	1067	0 B	0.04	0.04
AStarPathfinding.get_MaxOpenNodes()	0.0%	0.0%	238	0 B	0.00	0.00
AStarPathfinding.set_MaxOpenNodes()	0.0%	0.0%	83	0 B	0.00	0.00
▶ NodeRecordArray.GetBestAndRemove()	1.1%	0.0%	190	0 B	6.00	0.11
▶ NodeRecordArray.RemoveFromOpen()	0.9%	0.0%	189	112 B	4.58	0.10

Optimization:

For this optimization we pre-calculate all the clusters for the nodes. We used the GetNodesHack method in order to obtain the nodes from our path finding algorithm. These nodes are indexed, so we just store the corresponding cluster on the corresponding node index. Knowing this will fill the array with repeated clusters (one for each of the nearly 16000 nodes), the access time might be slower than a dictionary would be, but since Unity serialization doesn't allow an easy way to do it, we just went with this method.

As shown in the image below, the time spent on the Quantize method diminished significantly and we can assume it was a decent optimization.



CPU:445.62ms GPU:0.00ms

Overview	Total	Self	Calls	GC Alloc	Time ms	Self ms
BehaviourUpdate	95.6%	0.0%	1	414.4 KB	426.12	0.00
PathfindingManager.Update()	95.6%	0.3%	1	414.4 KB	426.11	1.71
AStarPathfinding.Search()	90.4%	0.6%	1	410.0 KB	403.15	3.04
NodeArrayAStarPathFinding.ProcessChildNode()	84.9%	1.0%	1738	408.2 KB	378.46	4.50
GatewayHeuristic.H()	81.7%	13.2%	1738	406.1 KB	364.16	58.86
NavMeshEdge.get_LocalPosition()	43.6%	2.3%	51810	0 B	194.41	10.56
GatewayHeuristic.EuclideanDistance()	16.1%	6.0%	103909	0 B	71.78	26.89
Enumerator.MoveNext()	3.6%	2.7%	71009	0 B	16.08	12.12
NavMeshPoly.get_LocalPosition()	2.6%	1.9%	52100	0 B	11.91	8.73
Enumerator.get_Current()	0.6%	0.6%	60613	0 B	3.10	3.10
GatewayHeuristic.get_ClusterGraph()	0.6%	0.6%	55430	0 B	2.80	2.80
List`1.GetEnumerator()	0.5%	0.3%	10396	0 B	2.48	1.76
ClusterGraph.Quantize()	0.3%	0.2%	3476	0 B	1.41	1.02
Enumerator.Dispose()	0.1%	0.1%	10396	0 B	0.57	0.57
Object.ReferenceEquals()	0.0%	0.0%	5214	0 B	0.30	0.30
NodeRecordArray.AddToOpen()	0.7%	0.0%	384	2.1 KB	3.36	0.14
NodeRecordArray.Replace()	0.6%	0.0%	87	0 B	2.87	0.10
NodeRecordArray.GetNodeRecord()	0.2%	0.2%	1738	0 B	1.33	1.02
NavigationGraphEdge.get_Cost()	0.1%	0.0%	1738	0 B	0.57	0.43
NodeRecordArray.CountOpen()	0.1%	0.0%	451	0 B	0.47	0.15
NodeRecordArray.SearchInOpen()	0.0%	0.0%	1738	0 B	0.21	0.21
AStarPathfinding.get_Open()	0.0%	0.0%	2660	0 B	0.19	0.19
NavigationGraphEdge.get_ToNode()	0.0%	0.0%	1738	0 B	0.15	0.15
AStarPathfinding.get_Heuristic()	0.0%	0.0%	1738	0 B	0.12	0.12
NodeArrayAStarPathFinding.get_NodeRecordArray()	0.0%	0.0%	1738	0 B	0.11	0.11
NodeRecordArray.SearchInClosed()	0.0%	0.0%	1738	0 B	0.10	0.10
AStarPathfinding.get_GoalNode()	0.0%	0.0%	1738	0 B	0.10	0.10
AStarPathfinding.get_Closed()	0.0%	0.0%	1738	0 B	0.09	0.09
AStarPathfinding.get_MaxOpenNodes()	0.0%	0.0%	384	0 B	0.03	0.03
AStarPathfinding.set_MaxOpenNodes()	0.0%	0.0%	67	0 B	0.00	0.00
NodeRecordArray.GetBestAndRemove()	2.1%	0.0%	330	0 B	9.68	0.16
NodeRecordArray.RemoveFromOpen()	1.6%	0.0%	329	112 B	7.18	0.24
AStarPathfinding.CalculateSolution()	0.7%	0.2%	1	1.7 KB	3.29	1.00

Object	Total	Self	Calls	GC Alloc	Time ms	Self ms
N/A	0.0%	0.0%	1	0 B	0.25	0.00
N/A	0.0%	0.0%	1	0 B	0.21	0.00
N/A	0.0%	0.0%	1	0 B	0.19	0.00
N/A	0.0%	0.0%	1	0 B	0.03	0.00
N/A	0.0%	0.0%	1	0 B	0.03	0.00
N/A	0.0%	0.0%	1	0 B	0.02	0.00
N/A	0.0%	0.0%	1	0 B	0.02	0.00
N/A	0.0%	0.0%	1	0 B	0.02	0.00
N/A	0.0%	0.0%	1	1.0 KB	0.02	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	296 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	104 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00
N/A	0.0%	0.0%	1	0 B	0.01	0.00