Heuristics Analysis

Victor Osório

Introduction

In this project we complete the skeletons of classes that solve deterministic planning problems for Air Cargo transport system using search agents. This is a classical problem of deterministic planning.
In this review it will be analyzed which search algorithm is better to find solutions in this context.

Problems

This project presents three problems that should be solved using Planning Algorithms. All of the three problems has optimum solutions that are described below.

Problem	Definition	Solution	Steps
1	Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(P1, SFO) ∧ At(P2, JFK) ∧ Cargo(C1) ∧ Cargo(C2) ∧ Plane(P1) ∧ Plane(P2) ∧ Airport(JFK) ∧ Airport(SFO)) Goal(At(C1, JFK) ∧ At(C2, SFO))	Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P1, SFO, JFK) Fly(P2, JFK, SFO) Unload(C1, P1, JFK) Unload(C2, P2, SFO)	6
2	Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL) ∧ At(P1, SFO) ∧ At(P2, JFK) ∧ At(P3, ATL) ∧ Cargo(C1) ∧ Cargo(C2) ∧ Cargo(C3) ∧ Plane(P1) ∧ Plane(P2) ∧ Plane(P3) ∧ Airport(JFK) ∧ Airport(SFO) ∧ Airport(ATL)) Goal(At(C1, JFK) ∧ At(C2, SFO) ∧ At(C3, SFO))	Load(C1, P1, SFO) Load(C2, P2, JFK) Load(C3, P3, ATL) Fly(P1, SFO, JFK) Fly(P2, JFK, SFO) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) Unload(C2, P2, SFO) Unload(C1, P1, JFK)	9
3	Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL) ∧ At(C4, ORD) ∧ At(P1, SFO) ∧ At(P2, JFK) ∧ Cargo(C1) ∧ Cargo(C2) ∧ Cargo(C3) ∧ Cargo(C4) ∧ Plane(P1) ∧ Plane(P2) ∧ Airport(JFK) ∧ Airport(SFO) ∧ Airport(ATL) ∧ Airport(ORD)) Goal(At(C1, JFK) ∧ At(C3, JFK) ∧ At(C2, SFO) ∧ At(C4, SFO))	Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P2, ORD, SFO) Unload(C4, P2, SFO) Fly(P1, ATL, JFK) Unload(C3, P1, JFK) Unload(C2, P2, SFO) Unload(C1, P1, JFK)	12

A solution is considered optimal when the number of steps is minimal, regardless of their order.

Algorithm Analysis

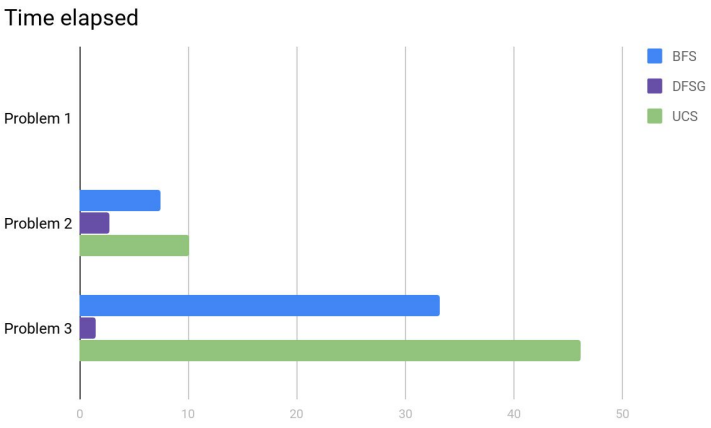
For our first analysis we will run the three problems with Breadth First Search (**BFS**), Depth First Graph Search (**DFGS**) and Uniform Cost Search (**UCS**).

Breadth First Search

According with AIMA, the BFS is complete and optimal. The only problem with BFS is that the memory usage grows in $O\left(b^d\right)$ where b is the branching factor and d is the deep of the tree.

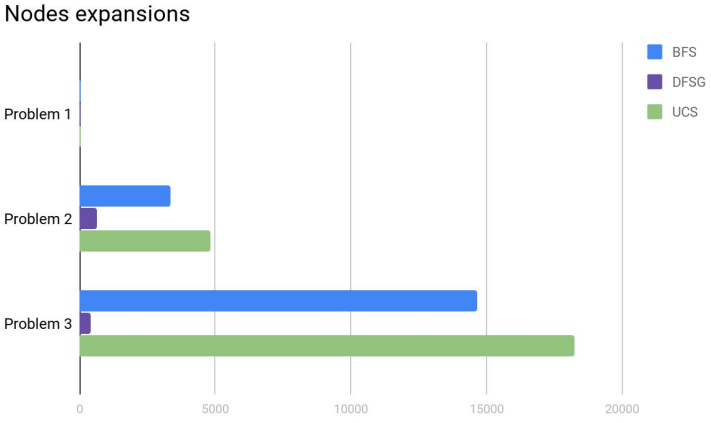
Depth First Graph Search

According with AIMA, DFS is only complete when is applied in Graphs. In our project, we can see that for some problems (P2 and P3) it does not returns any results in three hours. This occurs because the searching in tree can results in repeated states and redundant paths. This can be avoid by using graphs. DFS is not optimal.
The great advance of DFGS is that use less memory than other searches, and it not grows as UCS or BFS.



Uniform Cost Search

UCS can present in some cases good results, but in this cases it will not. UCS has, according with AIMA, complexity in time and space of $O\left(b^{1+\lfloor C*/\varepsilon \rfloor}\right)$ that can be greater than BFS.
The UCS choose the next node to expand with minimal cost of $g(n)$, that in this case is the number of steps from the solution.

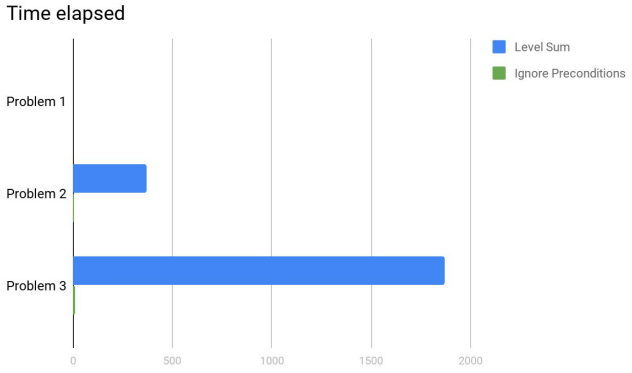


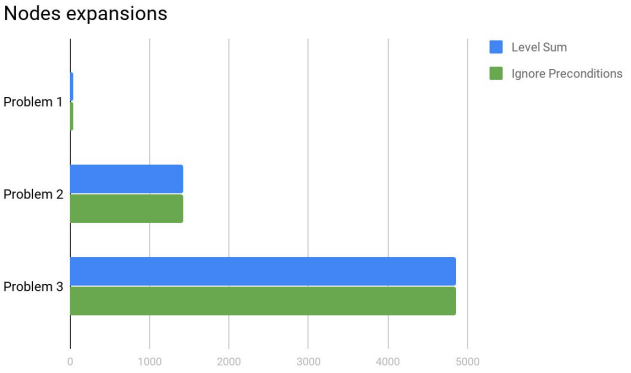
Results Analysis

As we can see in results, the BFS and UCS grows exponential with the problem size while the DFSG not. This occurs because using a graph search many states are avoided. But DFSG does not returns a optimal solution in any case.

Heuristic Analysis

Now, it will be analyzed the search using A* with differents heuristics. According with AIMA, A* will be complete and it will be optimal only if the used heuristic is admissible.
The A* algorithm is similar to UCS, but it does not only use the cost, the next node to expand is choosing using the function $f(n) = g(n) + h(n)$ where $h(n)$ is the heuristic for the current state.
The biggest problem for A* is that for each analyzed noode it is executed $h(n)$, so the heuristic should be quickly calculated. A greater heuristic that is slow to calculate does not present good effects in time.





Ignore Preconditions

This heuristic is simple, it only checks which goal state condition is not present in the actual state and estimate that for each state it is need one action to reached it.
This heuristic is admissible, only if there is no action that changes more than one condition.

Level Sum

This heuristic is a little more complex, it take the sum of the levels where any literal ofthe goal first appears. This is a not admissible heuristic.

Results Analysis

As we can see, using both heuristics the A* algorithm expand the same number of nodes, but it does not grows in time equally. The Level Sum take more time to execute, so this make it much more slow than Ignore precondition.
According with AIMA a better heuristic has it branching factor (b^*) near 1. A heuristic will better than other if it allow find the solution with less nodes analysed.
As we can see both heuristic has the same b^*, because they produce the same number of nodes for the same problem. If one heuristic take more time to calculate, this time should be returned by reducing the b^*. That is not happening in both heuristic, they analysis the same number of nodes. So we can conclude that Ignore Preconditions is better than Level Sum.

Results

Here it is presented all results for all tests. For these test execution it was estimated a timeout of two hours, so when it is show any value with ∞ that means that the execution does not present any results in two hours.

	Algorithm	Heuristic	Expansions	Goal Tests	New Nodes	Execution Time	Optimum
Problem 1	Breadth First Search	N/A	43	56	180	0.02440	Yes
	Breadth First Tree Search	N/A	1458	1459	5960	0.75331	Yes
	Depth First Graph Search	N/A	21	22	84	0.01170	No
	Depth Limited Search	N/A	101	271	414	0.07170	No
	Uniform Cost Search	N/A	55	57	224	0.03131	Yes
	Recursive Best First Search	1	4229	4230	17023	2.30165	Yes
	Greedy Best First Graph Search	1	7	9	28	0.00596	Yes
	A* Search	1	55	57	224	0.05885	Yes
	A* Search	Ignore Preconditions	41	43	170	0.02766	Yes
	A* Search	PG Levelsum	41	43	170	1.35887	Yes

Problem 2	Breadth First Search	N/A	3343	4609	30509	7.48874	Yes
	Breadth First Tree Search	N/A	∞	∞	∞	∞	-
	Depth First Graph Search	N/A	624	625	5602	2.76486	No
	Depth Limited Search	N/A	222719	2053741	2054119	772.27607	No
	Uniform Cost Search	N/A	4823	4825	43774	10.10210	Yes
	Recursive Best First Search	1	∞	∞	∞	∞	-
	Greedy Best First Graph Search	1	385	387	3512	0.77715	No
	A* Search	1	4823	4825	43774	10.40354	Yes
	A* Search	Ignore Preconditions	1421	1423	13022	3.08169	Yes
	A* Search	PG Levelsum	1421	1423	13022	368.14584	Yes
Problem 3	Breadth First Search	N/A	14663	18098	129631	33.13345	Yes
	Breadth First Tree Search	N/A	∞	∞	∞	∞	-
	Depth First Graph Search	N/A	408	409	3364	1.50615	No
	Depth Limited Search	N/A	∞	∞	∞	∞	-
	Uniform Cost Search	N/A	18235	18237	159716	46.15212	Yes
	Recursive Best First Search	1	∞	∞	∞	∞	-
	Greedy Best First Graph Search	1	4198	4200	37094	9.53958	No
	A* Search	1	18235	18237	159716	41.67847	Yes
	A* Search	Ignore Preconditions	4859	4861	43129	12.85926	Yes
	A* Search	PG Levelsum	4859	4861	43129	1868.06382	Yes