uninformed planning search metrics

For problem 1:

Search method	Expansions	Goal tests	New Nodes	Time elapse	Plan length
breadth_first_search	43	56	180	0.052	6
depth_first_graph_search	21	22	84	0.023	20
uniform_cost_search	55	57	224	0.087	6

The solution using breadth_first_search is:

```
Solving Air Cargo Problem 1 using breadth_first_search...
find solution

Expansions   Goal Tests   New Nodes
    43          56         180

Plan length: 6  Time elapsed in seconds: 0.05232491202203399
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
```

For problem 2:

Search method	Expansions	Goal tests	New Nodes	Time elapse	Plan length
breadth_first_search	3343	4609	30509	27.27	9
depth_first_graph_search	624	625	5602	8.067	619
uniform_cost_search	4852	4854	44030	83.83	9

The solution using uniform_cost_search is:

```
Solving Air Cargo Problem 2 using uniform_cost_search...

Expansions   Goal Tests   New Nodes
   4852        4854       44030

Plan length: 9  Time elapsed in seconds: 83.83094709184068
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(C1, P1, JFK)
Unload(C2, P2, SFO)
```

For problem 3:

Search method	Expansions	Goal tests	New Nodes	Time elapse	Plan length
breadth_first_search	14663	18098	129631	333.44	12
depth_first_graph_search	408	409	3364	3.8947	392
uniform_cost_search	18235	18237	159716	954.54	12

The solution using uniform_cost_search is:

```
Solving Air Cargo Problem 3 using uniform_cost_search...

Expansions   Goal Tests   New Nodes
  18235       18237       159716

Plan length: 12  Time elapsed in seconds: 954.5401690926242
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)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
```

For uninformed searches, only depth_first_search failed to find the optimal plan though it has the least node expansions, goal tests and new nodes, and thus takes the least time. The root cause is during the depth first search, it always tries to expand the longest path first for a solution, so the solution is not guaranteed to be optimal (The reason is explained in section “search comparision” in the course “Search”).

Breadth_first_search and uniform_cost_search got optimal plans (The reason is explained in section “search comparision” in the course “Search”), but they take more expansions, goals tests and time. Especial when the problem is complex, the time and space costs become high rapidly. But the plan they come up with are very good plans with least plan length needed. As uniform_cost_search need to calculate the cost during search, the time cost is the highest among the three search methods. So among the 3 search methods, I think breadth_first_search is better.

metrics of A* searches

For problem 1:

Heuristic	Expansions	Goal tests	New Nodes	Time elapse	Plan length
h_ignore_preconditions	41	43	170	0.057	6
h_pg_levelsum	11	13	50	3.19	6

For problem 2:

Heuristic	Expansions	Goal tests	New Nodes	Time elapse	Plan length
h_ignore_preconditions	1506	1508	13820	28.56	9
h_pg_levelsum	86	88	841	567.46	9

For problem 3:

Heuristic	Expansions	Goal tests	New Nodes	Time elapse	Plan length
h_ignore_preconditions	5118	5120	45650	177.57	12
h_pg_levelsum	408	410	3758	2256.9	12

Of the 2 heuristics, both of them found the optimal plan, while h_ignore_preconditions used less time. The reason is the implementation of h_ignore_preconditions is very simple and is light computation. Its disadvantage is it is not an accurate estimate of the cost so it will cost more expansions and tests. On the contrast, h_pg_levelsum can provide very accurate estimate of the cost but is very heavy computation, so it will use less space but takes more time.

What's the best choice?

Of all the methods,

- To get the balance of space and time, A star search will h_ignore_preconditions is the best choice.
- To use the least space and not care about the time, A star search with h_pg_levelsum is the best choice.
- To care about only time, depth_first_graph_search is the best choice.

So A star with heuristics is not always the best in all aspects. The main reason is heuristic functions needs computation, so will have some time cost.