Test Cases Summary

By: Kevin Offemaria

Initial State:

Country	Population	MetallicElements	Timber	MetallicAlloys	Electronics	Housing	HousingWaste	ElectronicsWaste	MetallicAlloysWaste
NewCaliforniaRepublic	80	50	50	40	130	15	20	20	20
NewReno	60	40	40	20	90	15	20	20	20
Arroyo	30	150	150	150	4	4	0	0	0
SanFrancisco	100	150	120	120	160	40	40	40	40
Gecko	40	20	10	20	10	20	80	80	80

Initial Resource Weights:

Resource	Weight
Population	0.5
MetallicElements	0.5
Timber	0.5
MetallicAlloys	0.5
Electronics	1
Housing	0.5
HousingWaste	-0.2
MetallicAlloysWaste	-0.2
ElectronicsWaste	-0.6

State Quality:

In terms of the State Quality definition, a country values its happiness according to how much `Electronics` it currently owns. Therefore, its State Quality measure is the count of the amount of available Electronics. Since Electronics is a produced resource, there is value in having the necessary raw materials to produce them though it would still need resources and time to produce them. This continuity of production is counted towards the SQ score but with a 50% penalty. All of these are reflected in the resource weights. There is a cap on Electronics against the country's population assumed at a 2:1 ratio of Electronics per person. If a country has more Electronics against its own population, we penalize Electronics, MetallicElements, MetallicAlloys. This is necessary to shift focus on other resources that may potentially increase population satisfaction such as `Housing`, which has an increased role in EU contribution. We assume the optimal ratio of persons to a house is 4:1, or 4 persons per single house. A bonus to this resource is applied to incentivize building houses to accommodate a country's population. However, just like Electronics, once the limit goes above the ratio a penalty will be applied to discourage building unnecessary houses for a satisfied population.

Search Function:

There are 3 search functions that can be used in the program: Greedy Best First Search, Heuristic Depth First Search, Breadth First Search. GBFS utilizes the *PriorityQueue* class from the Python library, *queue*, as the storage frontier. While the others use the built-in Python List object. The algorithm of all three functions is generally the same whereby they start with a root node inserted into the frontier and generate children nodes based on a root or parent node. It then attempts to generate Action Templates for TRANSFORMs and TRANSFERs. The resource values for the current node's country are modified by the Action. The new child nodes are randomly shuffled and appended to the frontier. Next, the Expected Utility score on the current node is calculated and evaluated against the best scoring EU node. If so, replace the best EU node with the current node, and write all the Action Templates for all parent nodes of the current node. The program ends when the maximum depth, or frontier size has been exceeded, or the frontier is empty. (See architecture)

Expected Utility constants:

These amounts resulted in moderate EU scores ranging from the 20s to high 90s based on the max depth bound and frontier sizes. Keeping the logistic growth rate (K) to a minimum helped curb the amount to reaching the hundreds or thousands.

GAMMA = 0.95
COST_OF_FAILURE = -10
L = 1
K = 0.001
X_0 = 5

Test Case Overview:

The goal of each test case is to match each available search strategy (3) playing as two (2) different countries in each test case iteration for a total of 6 test cases. Both countries to be tested offers unique characteristics. Common parameters throughout the tests are max number of schedules are set at 20 to maximize the best schedule we can find given the search strategy.

1. Play as *New California Republic*
    a. Moderate raw resources
    b. High starting Electronics/Housing resources to population ratio
2. Play as *Arroyo*
    a. High raw resources
    b. Low starting Electronics/Housing resources to population ratio

Test Case 1:

We use the following parameters in line with GBFS as I found these are the optimal parameters to get reach the highest scores without having a larger frontier or depth bound. Out of all the searches, this takes the most advantage of using the defined heuristic to find the best possible scores. Compared to the other searches, this was also the quickest search type to finish. The country's moderate resources and high Electronics to start plays a huge role in acquiring more resources in order to build more electronics.

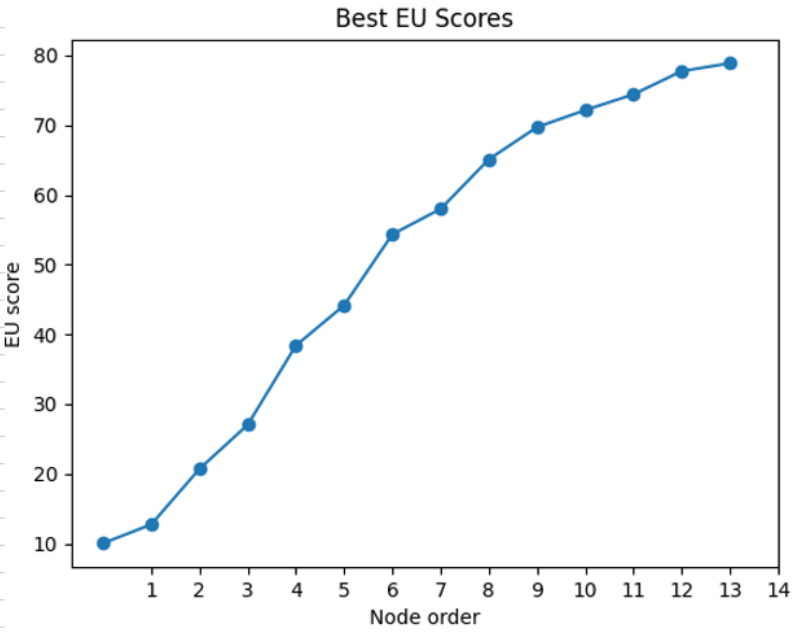
- Search strategy: Greedy Best First Search
- Play as *New California Republic*
- *country_schedule()* parameters:
    - country = "NewCaliforniaRepublic"
    - search_type = "gbfs"
    - num_schedules = 20
    - depth_bound = 20
        - I wanted to go deep enough to potentially find high scoring nodes as GBFS utilizes high EU scores as a guide in generating the best schedules.
    - frontier_size = 10000
        - I found this is the optimal number of nodes to find the best nodes.

Observations:

- Generally, more TRANSFERs than TRANSFORMs
- Higher schedule count, between 15-18
- Avg EU score between 70 to 80

Result summary:

```
====================
SCHEDULE SUMMARY
Country: NewCaliforniaRepublic
Best EU score: 78.86291324705465
Node count: 10007
Schedule count: 15
TRANSFORM count: 45
TRANSFER count: 111
Population: 80
Electronics: 132
MetallicElements: 115
MetallicAlloys: 72
Housing: 17
Timber: 33
====================
```



Test Case 2:

We use the same exact parameters and search type as in Test Case 1, except using a country with much higher resources to in its initial state. Due to its high amount of raw resources and the randomness of a transfer percentage (10-20%), a relatively large transfer ultimately occurs which results in a sudden spike in EU score in the latter parts of the schedule, notice between Nodes 11-12. This also has higher variance in average EU scores.

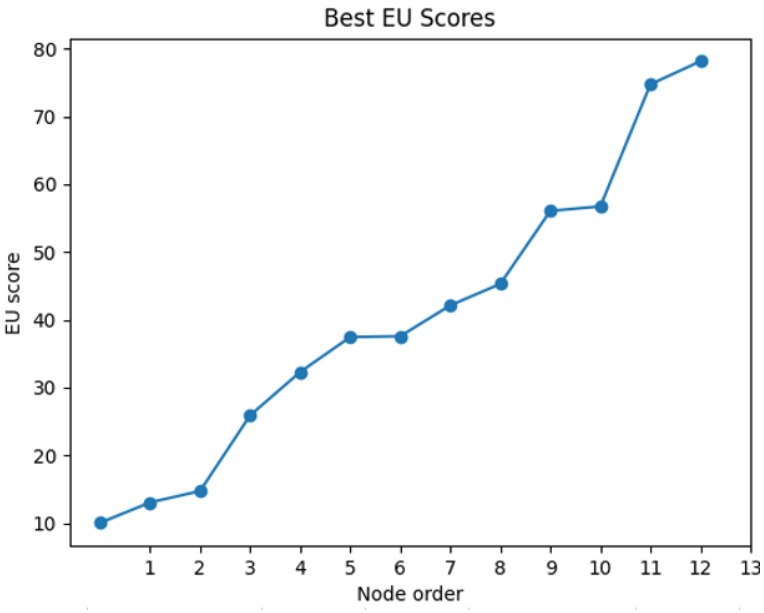
- Search strategy: Greedy Best First Search
- Play as *Arroyo*
- *country_schedule()* parameters:
    - country = "Arroyo"
    - search_type = "gbfs"
    - num_schedules = 20
    - depth_bound = 20
        - I wanted to go deep enough to potentially find high scoring nodes as GBFS utilizes high EU scores as a guide in generating the best schedules.
    - frontier_size = 10000
        - I found this is the optimal number of nodes to find the best nodes.

Observations:

- Generally more TRANSFORMs than TRANSFERs
- Lower schedule count, between 10 to 14
- Avg EU score between 70 to 90, higher variance than test case 1

Result summary:

```
====================
SCHEDULE SUMMARY
Country: Arroyo
Best EU score: 78.1375655368197
Node count: 10005
Schedule count: 14
TRANSFORM count: 50
TRANSFER count: 47
Population: 30
Electronics: 14
MetallicElements: 116
MetallicAlloys: 57
Housing: 5
Timber: 265
====================
```



Test Case 3:

For HDFS, we greatly reduce the depth bound parameter to prevent the agent from going deeper into a low scoring node. Conversely, this also limits the agent's ability to find the better scores, and this also takes slightly longer than GBFS to finish. The country's moderate resources and high Electronics to start plays a huge role in acquiring more resources in order to build more electronics. Overall, the nature of HDFS will almost always have limited EU scores because the algorithm goes down the leftmost node and may never recover from a low EU score.

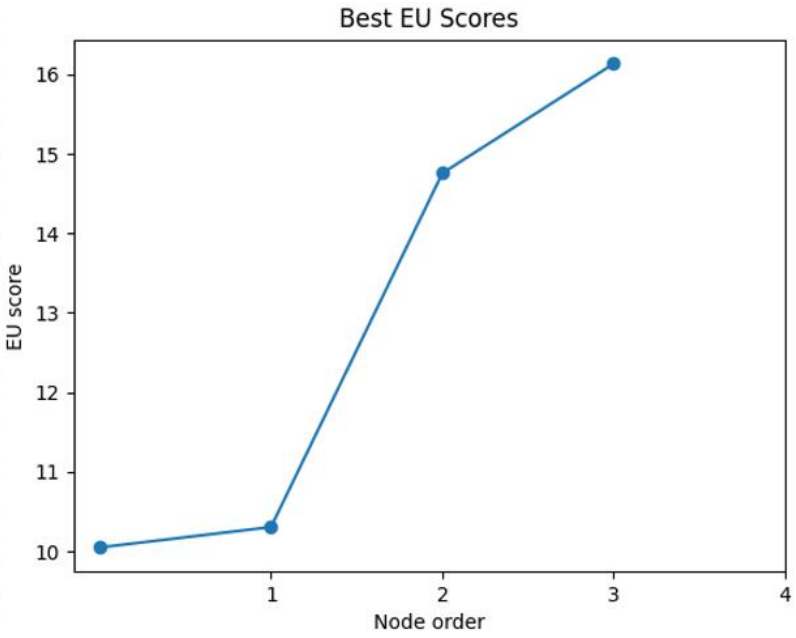
- Search strategy: Heuristic Depth First Search
- Play as *New California Republic*
- *country_schedule()* parameters:
    - country = "NewCaliforniaRepublic"
    - search_type = "hdfs"
    - num_schedules = 20
    - depth_bound = 5
        - Keep the depth bound to a minimum
    - frontier_size = 20000
        - Double the frontier size in an attempt to find better schedules.

Observations:

- Slightly more TRANSFERs than TRANSFORMs, almost even
- Very low schedule count, between 3-5
- Very low EU scores, as low as in the teens, lowest among all search algorithms

Result summary:

```
====================
SCHEDULE SUMMARY
Country: NewCaliforniaRepublic
Best EU score: 16.1342202568635
Node count: 7526
Schedule count: 5
TRANSFORM count: 6
TRANSFER count: 6
Population: 80
Electronics: 130
MetallicElements: 62
MetallicAlloys: 46
Housing: 15
Timber: 25
====================
```



Test Case 4:

Similar to the previous test case 3, the search resulted in limited schedules despite a higher frontier size. The only difference having higher resources and the randomness of Transfer percentages and other injected random features (random encounters/recycling) increased the EU scores while returning slightly higher node schedules.

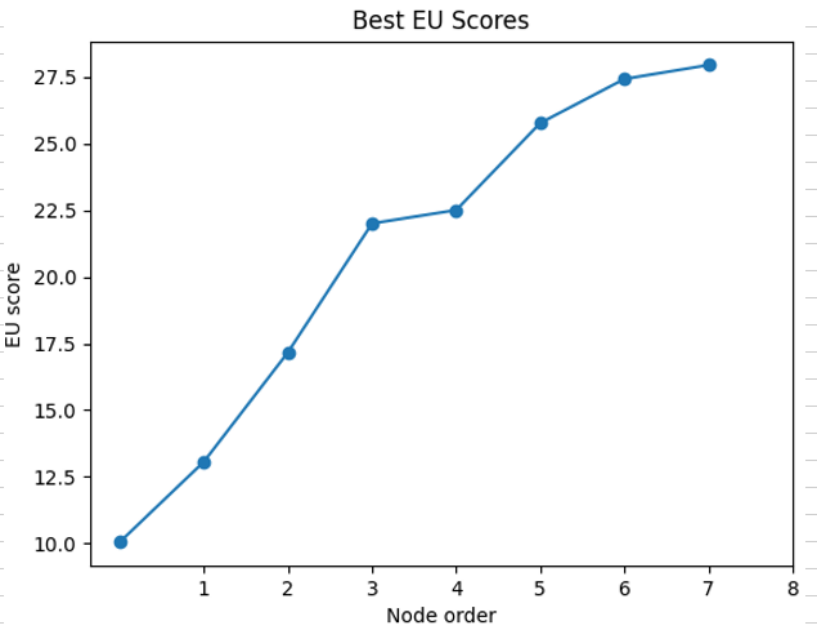
- Search strategy: Heuristic Depth First Search
- Play as *Arroyo*
- *country_schedule()* parameters:
    - country = "Arroyo"
    - search_type = "hdfs"
    - num_schedules = 20
    - depth_bound = 5
        - Keep the depth bound to a minimum
    - frontier_size = 20000
        - Double the frontier size in an attempt to find better schedules.

Observations:

- Slightly more TRANSFORMs than TRANSFERs, almost even
- Low schedule count, between 7 to 9
- Low EU scores, between 22 to 27

Result Summary:

```
====================
SCHEDULE SUMMARY
Country: Arroyo
Best EU score: 27.9488882364542
Node count: 9275
Schedule count: 9
TRANSFORM count: 15
TRANSFER count: 12
Population: 30
Electronics: 6
MetallicElements: 145
MetallicAlloys: 115
Housing: 4
Timber: 180
====================
```



Test Case 5:

We introduce Breath First Search with a middle-sized depth bound compared to other search types explored. However, we use an extremely large frontier size to account for the nature of the algorithm to search wide *before* going down the search space. This returns relatively low EU scores as it does not go deep enough to search for better nodes. The country's moderate resources return a lower variance of scores and schedule counts. Overall, this BFS is not an efficient algorithm to be used on an infinite search space.

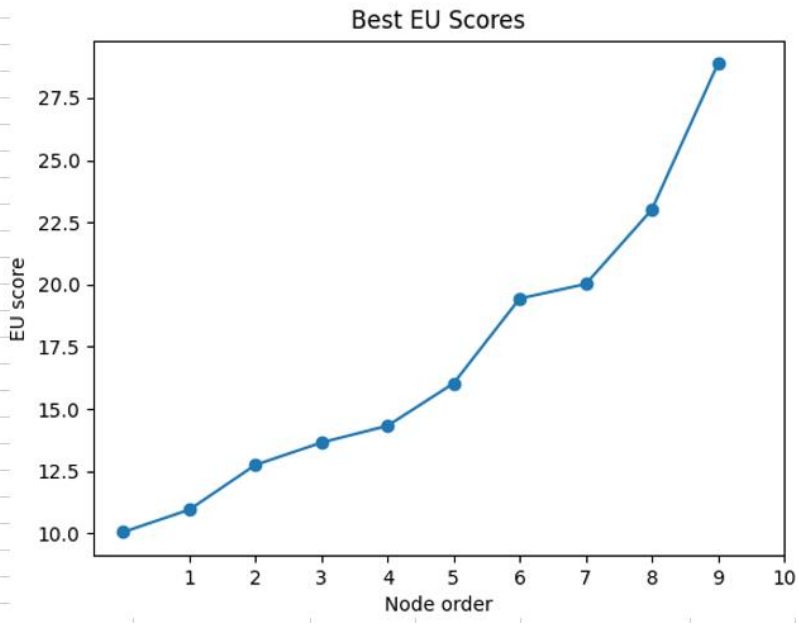
- Search strategy: Breadth First Search
- Play as *New California Republic*
- *country_schedule()* parameters:
    - country = "New California Republic"
    - search_type = "bfs"
    - num_schedules = 20
    - depth_bound = 10
        - Middle sized depth bound.
    - frontier_size = 50000
        - Extremely large frontier size.

Observations:

- Generally more TRANSFERs than TRANSFORMs
- Moderate schedule count, between 8 to 11
- Lower EU score, between 25 to 28
- Never goes deep enough to explore potentially more optional schedules.

Result Summary:

```
====================
SCHEDULE SUMMARY
Country: NewCaliforniaRepublic
Best EU score: 28.870586003586894
Node count: 50007
Schedule count: 11
TRANSFORM count: 9
TRANSFER count: 20
Population: 80
Electronics: 132
MetallicElements: 57
MetallicAlloys: 48
Housing: 15
Timber: 43
====================
```



Test Case 6:

Using the same parameters for Breadth First Search, this last test case produced the EU nodes with at least half the scores as GBFS. Like previous iterations with this country, its high raw resources combined with random transfer percentages result in higher barter amounts accepted.

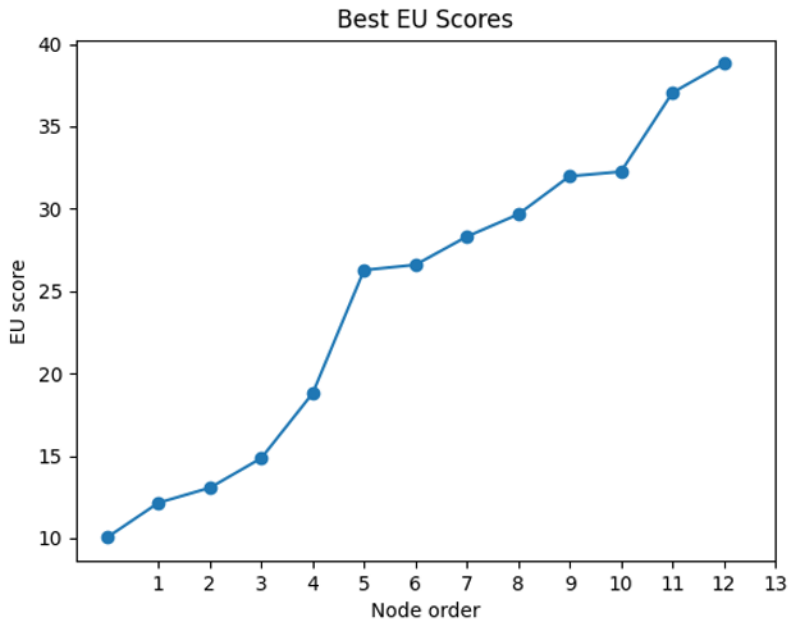
- Search strategy: Breadth First Search
- Play as *Arroyo*
- *country_schedule()* parameters:
    - country = “Arroyo”
    - search_type = “bfs”
    - num_schedules = 20
    - depth_bound = 10
        - Middle sized depth bound.
    - frontier_size = 50000
        - Extremely large frontier size.

Observations:

- Generally more TRANSFERs than TRANSFORMs
- Moderate schedule count, between 8 to 13
- Moderate EU score, between 30 to 40
- Slightly higher EU due to resource weights on raw materials.

Result Summary:

```
====================
SCHEDULE SUMMARY
Country: Arroyo
Best EU score: 38.81528299019058
Node count: 50006
Schedule count: 14
TRANSFORM count: 15
TRANSFER count: 23
Population: 30
Electronics: 4
MetallicElements: 126
MetallicAlloys: 75
Housing: 4
Timber: 222
====================
```



Test Cases Summary:

- Randomness of applied uncertainty on the respective action templates generates higher variance of EU scores due to potential positive/negative effects.
    - Random encounters
    - Chance to recycle resources
- The redefined state quality of giving bonuses to raw resources as a means of production to Electronics meant an excessive amount of them actually provided more value than having Electronics itself.
- The country with higher resources scored a higher potential due to its given value on the means to production.
- Resource decay affected the algorithm towards the latter parts of the search as the country is attempting to replenish its lost resources.
- Since Housing is given some priority, there are more transformations to reflect this.

Search Strategy Observations:

- Of the 3 search types, GBFS is the most efficient at finding the best schedule.
- GBFS relies on accuracy of heuristics (EU) to guide the search.
- HDFS and BFS are very limited when used with a large or infinite search space.
- If HDFS goes on a low scoring node, it may never recover and will not find high EU scores.
- It is almost impossible to find the best schedule for BFS on an infinite search space.
- Since we have no explicit "goal" state, we are not able to accurately estimate the distance to the goal.
- Fine line when introducing alpha-beta pruning because it is technically injecting human intuition to the agent, which is why avoided this risk and removed it altogether.