# CS4750/7750 HW#2

## Uniform Cost Tree and Graph Search

We implemented Uniform Cost Search in a single function that behaves differently depending on the treeSearch argument that you specify when you run it.
The algorithm heavily relies on the cost it takes to proceed to the next node and always visits the next cheapest node. The key difference between the Tree and the Graph implementations is that the Graph one remembers the visited states. It obviously takes up more memory to store visited states but significantly decreases execution time.
(Full report is on the next page and in the .log documents from our submission)

# Uniform Cost Graph Search Report

```
Graph 1. Initial agent location: (2,2). Dirty squares: (1,2), (2,4), (3,5).
```

```
=====================================================

|(1, 1)   |  |(2, 1)   |  |(3, 1)   |  |(4, 1)   |

|(1, 2) d|  |(2, 2)* |  |(3, 2)   |  |(4, 2)   |

|(1, 3)   |  |(2, 3)   |  |(3, 3)   |  |(4, 3)   |

|(1, 4)   |  |(2, 4) d|  |(3, 4)   |  |(4, 4)   |

|(1, 5)   |  |(2, 5)   |  |(3, 5) d|  |(4, 5)   |

=====================================================


First five nodes visited: [2, 2], [2, 3], [2, 1], [3, 2], [1, 2]

=====================================================

Nodes created: 1156

Nodes expanded: 366

Time elapsed: 0.21797966957092285 seconds

Solution cost: 6.699999999999999

Final path: [3, 5](sucking) <- [3, 5] <- [2, 5] <- [2, 4](sucking) <- [2, 4] <- [2, 3]
<- [1, 3] <- [1, 2](sucking) <- [1, 2] <- [2, 2]

Actions list: Action.SUCK <- Action.RIGHT <- Action.DOWN <- Action.SUCK <- Action.DOWN
<- Action.RIGHT <- Action.DOWN <- Action.SUCK <- Action.LEFT <- Action.NONE

Number of moves: 10
```

```
Graph 2. Initial agent location: (3,2). Dirty squares: (1,2), (2,1), (2,4), (3,3).

=====================================================

|(1, 1)  |  |(2, 1) d|  |(3, 1)  |  |(4, 1)  |

|(1, 2) d|  |(2, 2)  |  |(3, 2)* |  |(4, 2)  |

|(1, 3)  |  |(2, 3)  |  |(3, 3) d|  |(4, 3)  |

|(1, 4)  |  |(2, 4) d|  |(3, 4)  |  |(4, 4)  |

|(1, 5)  |  |(2, 5)  |  |(3, 5)  |  |(4, 5)  |

=====================================================

First five nodes visited: [3, 2], [3, 3], [3, 1], [4, 2], [2, 2]

=====================================================

Nodes created: 6921

Nodes expanded: 2152

Time elapsed: 5.763530015945435 seconds

Solution cost: 8.899999999999999

Final path: [1, 2](sucking) <- [1, 2] <- [1, 1] <- [2, 1](sucking) <- [2, 1] <- [2, 2]
<- [2, 3] <- [2, 4](sucking) <- [2, 4] <- [3, 4] <- [3, 3](sucking) <- [3, 3] <- [3,
2]

Actions list: Action.SUCK <- Action.DOWN <- Action.LEFT <- Action.SUCK <- Action.UP <-
Action.UP <- Action.UP <- Action.SUCK <- Action.LEFT <- Action.DOWN <- Action.SUCK <-
Action.DOWN <- Action.NONE

Number of moves: 13
```

# Uniform Cost Tree Search Report

```
Graph 1. Initial agent location: (2,2). Dirty squares: (1,2), (2,4), (3,5).
=====================================================

|(1, 1)   |  |(2, 1)   |  |(3, 1)   |  |(4, 1)   |
|(1, 2) d|  |(2, 2)* |  |(3, 2)   |  |(4, 2)   |
|(1, 3)   |  |(2, 3)   |  |(3, 3)   |  |(4, 3)   |
|(1, 4)   |  |(2, 4) d|  |(3, 4)   |  |(4, 4)   |
|(1, 5)   |  |(2, 5)   |  |(3, 5) d|  |(4, 5)   |
=====================================================
First five nodes visited:
[2, 2], [2, 3], [2, 1], [3, 2], [1, 2]
=====================================================
Nodes created: 96793
Nodes expanded: 27874
Time elapsed: 239.94411873817444 seconds
Solution cost: 6.699999999999999
Final path: [3, 5](sucking) <- [3, 5] <- [2, 5] <- [2, 4](sucking) <- [2, 4] <- [2, 3]
<- [1, 3] <- [1, 2](sucking) <- [1, 2] <- [2, 2]
Actions list: Action.SUCK <- Action.RIGHT <- Action.DOWN <- Action.SUCK <- Action.DOWN
<- Action.RIGHT <- Action.DOWN <- Action.SUCK <- Action.LEFT <- Action.NONE
Number of moves: 10

Graph 2. Initial agent location: (3,2). Dirty squares: (1,2), (2,1), (2,4), (3,3).
=====================================================

|(1, 1)   |  |(2, 1) d|  |(3, 1)   |  |(4, 1)   |
|(1, 2) d|  |(2, 2)   |  |(3, 2)* |  |(4, 2)   |
|(1, 3)   |  |(2, 3)   |  |(3, 3) d|  |(4, 3)   |
|(1, 4)   |  |(2, 4) d|  |(3, 4)   |  |(4, 4)   |
|(1, 5)   |  |(2, 5)   |  |(3, 5)   |  |(4, 5)   |
=====================================================
First five nodes visited + graph representation:
[3, 2], [3, 3], [3, 1], [4, 2], [2, 2]
=====================================================
Nodes created: 365987
Nodes expanded: 103061
Time elapsed: 3600.219456911087 seconds
The solution was not found within an hour
```

# Iterative Deepening Search

The biggest difference between this algorithm and Uniform Cost is that it doesn't necessarily propagate to the next cheapest node. The key principle is that it will go down the search tree until it hits a depth limit.
In our experience, this algorithm was efficient in terms of execution time while providing a solution with the scores almost as low as the uniform cost.

## IDS Report

```
Graph 1
=====================================================


|(1, 1)   |   |(2, 1)   |   |(3, 1)   |   |(4, 1)   |
|(1, 2) d|   |(2, 2)* |   |(3, 2)   |   |(4, 2)   |
|(1, 3)   |   |(2, 3)   |   |(3, 3)   |   |(4, 3)   |
|(1, 4)   |   |(2, 4) d|   |(3, 4)   |   |(4, 4)   |
|(1, 5)   |   |(2, 5)   |   |(3, 5) d|   |(4, 5)   |

=====================================================
First five nodes visited: [[2, 2], [2, 2], [2, 1], [1, 2], [1, 2]]
=====================================================
Nodes created: 48
Nodes Expanded: 59
Time Elapsed: 0.0003349539999999984 seconds
=====================================================
Solution Sequence: ['LEFT', 'SUCK', 'DOWN', 'DOWN', 'DOWN', 'RIGHT', 'UP', 'SUCK',
'DOWN', 'RIGHT', 'SUCK']
Number of moves: 11
Solution Cost: 8.200000000000001
=====================================================


Graph 2
=====================================================


|(1, 1)   |   |(2, 1) d|   |(3, 1)   |   |(4, 1)   |
|(1, 2) d|   |(2, 2)   |   |(3, 2)* |   |(4, 2)   |
|(1, 3)   |   |(2, 3)   |   |(3, 3) d|   |(4, 3)   |
|(1, 4)   |   |(2, 4) d|   |(3, 4)   |   |(4, 4)   |
|(1, 5)   |   |(2, 5)   |   |(3, 5)   |   |(4, 5)   |

=====================================================
First five nodes visited: [[3, 2], [3, 2], [3, 1], [2, 2], [3, 3]]
=====================================================
Nodes created: 44
Nodes Expanded: 44
Time Elapsed: 0.00015419299999999692 seconds
=====================================================
Solution Sequence: ['DOWN', 'SUCK', 'DOWN', 'LEFT', 'SUCK', 'UP', 'UP', 'UP', 'SUCK',
'LEFT', 'DOWN', 'SUCK']
Number of moves: 12
Solution Cost: 8.899999999999999
=====================================================
```