COM1005 Assignment 1 2019 Report

Name: Daniel Whiteman CICS userID: aca18djw

# Testing LatticeState

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | Testing for | Test | Ouput | Pass/Fail |
| SameState | Same states |  | True | Pass |
| SameState | Different states (Different costs) |  | False | Pass |
| SameState | Different states (Different instance variables in the WordH) |  | False | Pass |
| SameState | Different states (Different words) |  | False | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | Testing for | Test | Output | Pass/Fail |
| goalP | Not a goal |  | False | Pass |
| goalP | Is a goal |  | True | Pass |
| getWord | Accesses the WordH |  | True | Pass |
| getCost | Accesses the localCost |  | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | Testing for | Test | Output | Pass/Fail |
| getSuccessors | Correct successors from \*start\* | Running on a sample lattice.  (latt4.txt) |  | Pass – correct successors |
| getSuccessors | Correct successors at the end | Running on a sample lattice.  (latt4.txt) |  | Pass – No successors as we have reached a goal. |
| getSuccessors | Correct successors in the middle of the algorithm. | Running on a sample lattice.  (latt4.txt) |  | Pass – Correct successors at iteration 7 |

# Testing LatticeSearch

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | Testing for | Test | Output | Pass/Fail |
| getLatt | Accesses the latt object |  | True | Pass |
| getBg | Accesses the bg object |  | True | Pass |
| getGoal | Accesses the goal(36 for the word lattice) |  | 36 | Pass |

# Experiments with search strategies

Word lattice: latt1.txt

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TV (20) | off (150) |  | | volume (30) | |
|  | radio (100) | | on (60) |  | |
| three (50) | channel (70) | | five (20) | | playback (120) |
|  | | | seven (200) | |  |

0

10

20

30

40

50

60

70

|  |  |  |  |
| --- | --- | --- | --- |
| Search Method | Solution Path | Global cost | Efficiency |
| Branch and Bound | \*start\*  TV  channel  five  playback | 390 | 0.33333334 |
| Depth first | \*start\*  three  channel  seven  playback | 690 | 1.0 |
| Breadth first | \*start\*  TV  channel  on  volume | 440 | 0.3125 |

For this test I used the TVLM.txt language model and vocabulary. Here, only branch and bound finds a solution path with the lowest cost and does this with an efficiency of 0.33333334.

Word lattice: latt2.txt

|  |  |  |  |
| --- | --- | --- | --- |
| TV (30) |  | | |
|  | channel (20) | | five (20) |
| playback (50) |  | | nine (70) |
|  | off (60) |  | |

30

40

20

10

0

|  |  |  |  |
| --- | --- | --- | --- |
| Search Method | Solution Path | Global cost | Efficiency |
| Branch and Bound | \*start\*  TV  channel  five | 120 | 0.5714286 |
| Depth first | \*start\*  Playback  channel  nine | 240 | 0.8 |
| Breadth first | \*start\*  TV  channel  five | 120 | 0.5 |

For this test I used the TVLM.txt language model and vocabulary. Here, branch and bound finds a solution path with the lowest cost and does this with an efficiency of 0.5714286. Breadth first also finds the least costly path but only has an efficiency of 0.5.

Word lattice: latt3.txt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| playback (30) | channel (50) | |  | |
|  | record (15) | | | eight (40) |
| radio (70) |  | | | nine (70) |
|  | on (30) |  | | |

0

10

20

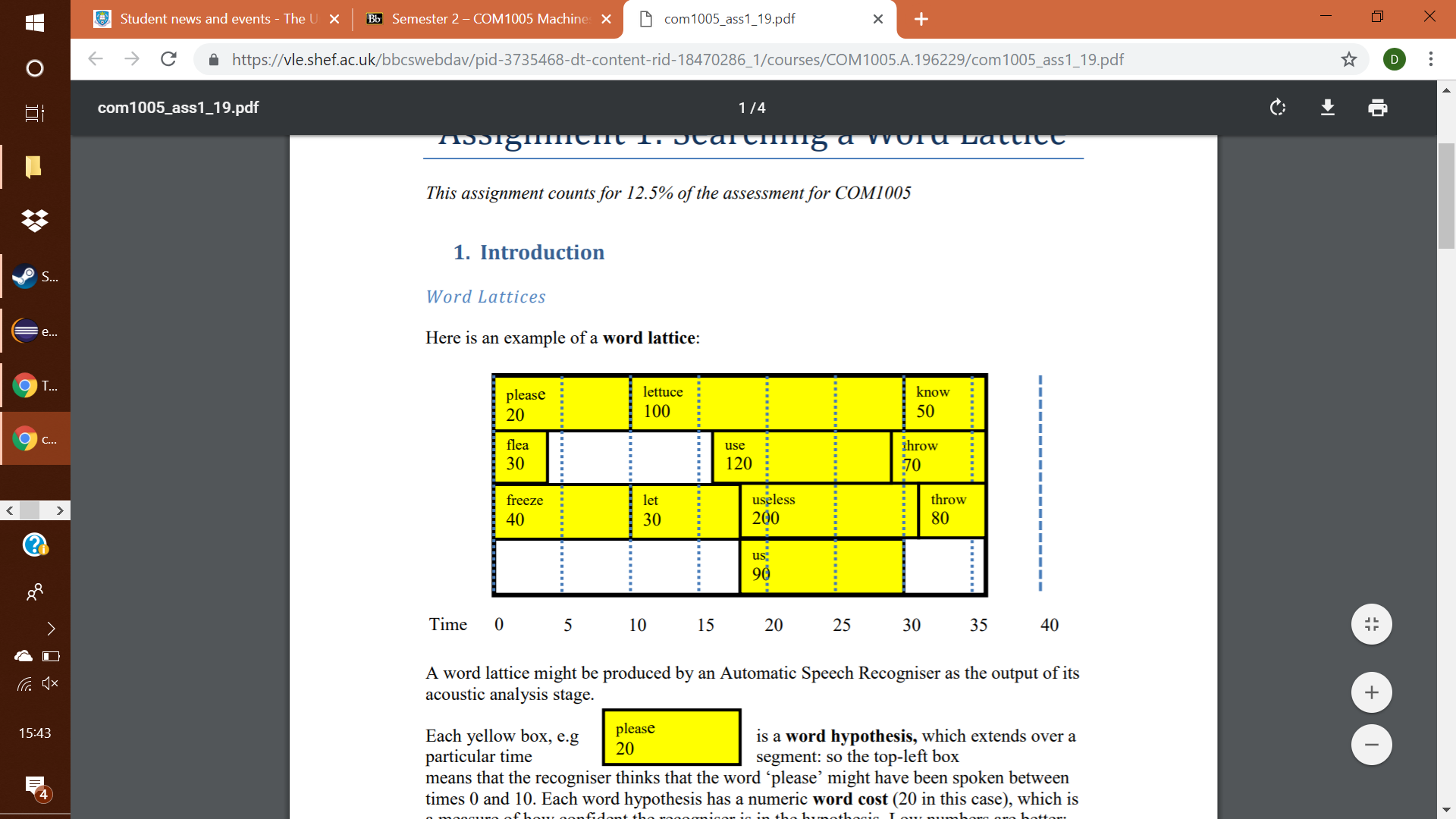
30

40

|  |  |  |  |
| --- | --- | --- | --- |
| Search Method | Solution Path | Global cost | Efficiency |
| Branch and Bound | \*start\*  radio  record  eight | 325 | 0.4 |
| Depth first | \*start\*  radio  record  nine | 345 | 0.8 |
| Breadth first | \*start\*  radio  record  eight | 325 | 0.4 |

For this test I used the TVLM.txt language model and vocabulary. Here, branch and bound finds a solution path with the lowest cost and does this with an efficiency of 0.4. Breadth first also finds the least costly path and only has an efficiency of 0.4.

Word lattice: latt4.txt



|  |  |  |  |
| --- | --- | --- | --- |
| Search Method | Solution Path | Global cost | Efficiency |
| Branch and Bound | \*start\*  please  let  us  know | 290 | 0.5 |
| Depth first | \*start\*  freeze  Let  us  know | 360 | 1.0 |
| Breadth first | \*start\*  freeze  lettuce  know | 370 | 0.44444445 |

For this test I used the original language model and vocabulary found in RunLatticeSearch. Here, only branch and bound finds a solution path with the lowest cost and does this with an efficiency of 0.5.

Word lattice: latt5.txt

|  |  |  |  |
| --- | --- | --- | --- |
| playback (70) |  | | up (20) |
|  | volume(40) | | eight (40) |
| radio (50) |  | | nine (70) |
|  | on (30) | record (40) |  |

0

10

20

30

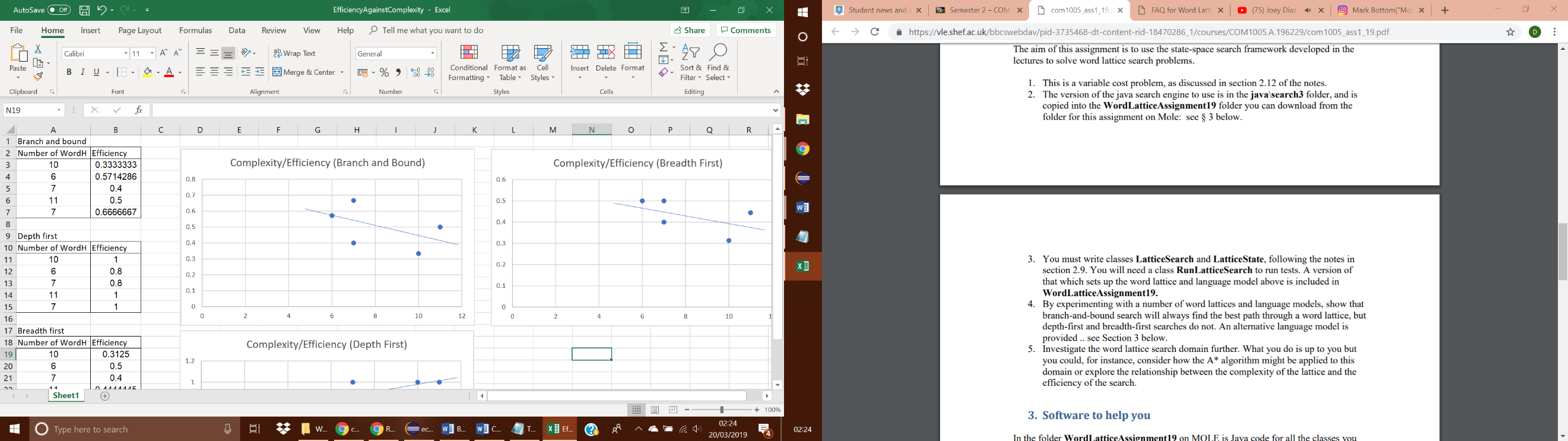
40

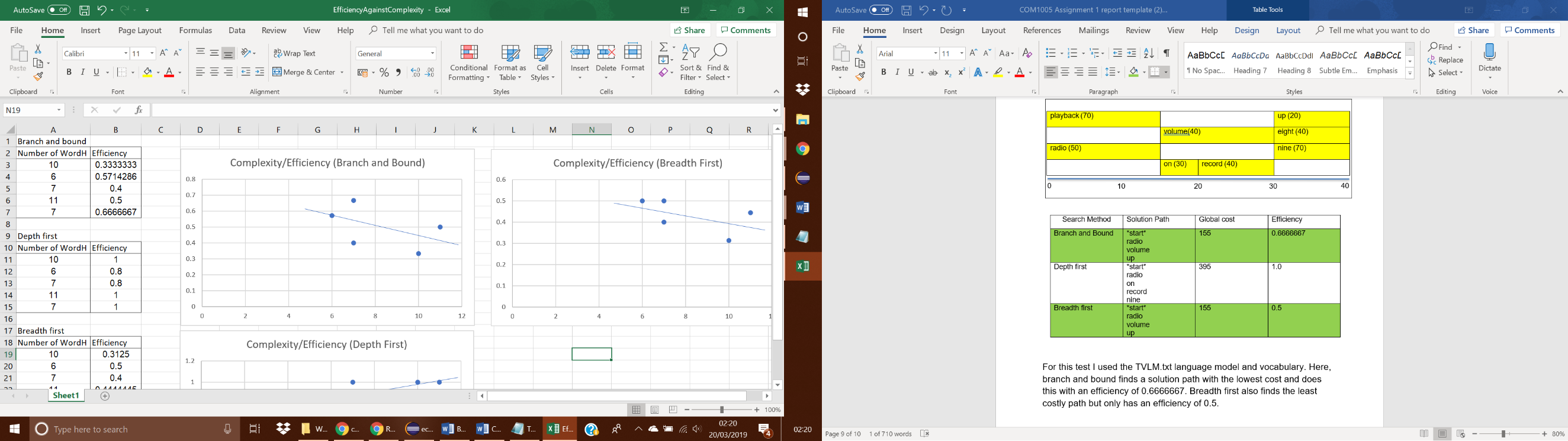
|  |  |  |  |
| --- | --- | --- | --- |
| Search Method | Solution Path | Global cost | Efficiency |
| Branch and Bound | \*start\*  radio  volume  up | 155 | 0.6666667 |
| Depth first | \*start\*  radio  on  record  nine | 395 | 1.0 |
| Breadth first | \*start\*  radio  volume  up | 155 | 0.5 |

For this test I used the TVLM.txt language model and vocabulary. Here, branch and bound finds a solution path with the lowest cost and does this with an efficiency of 0.6666667. Breadth first also finds the least costly path but only has an efficiency of 0.5.

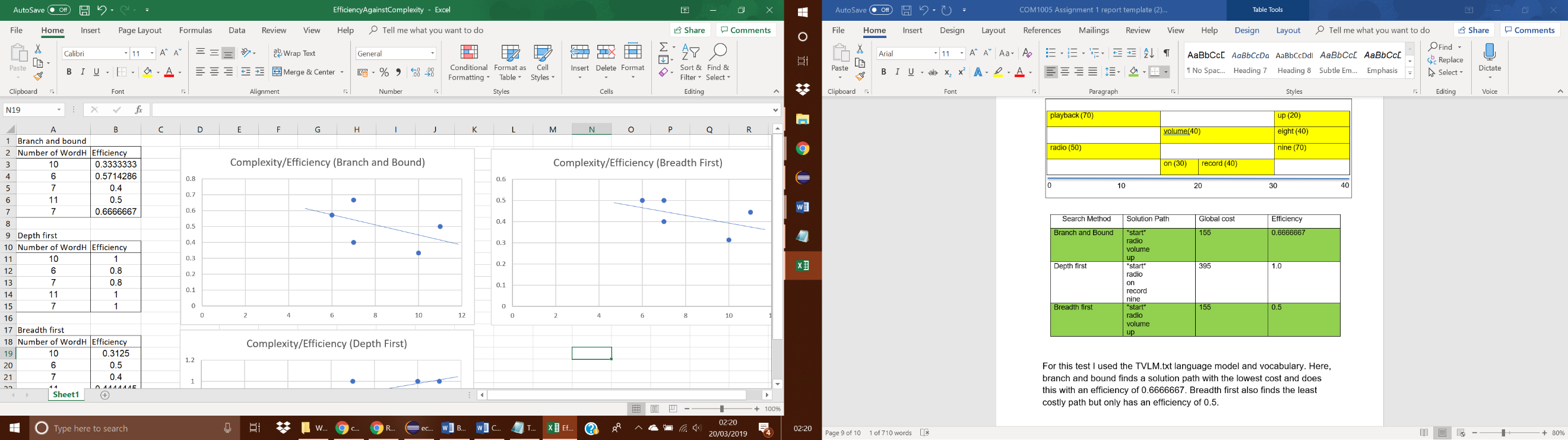
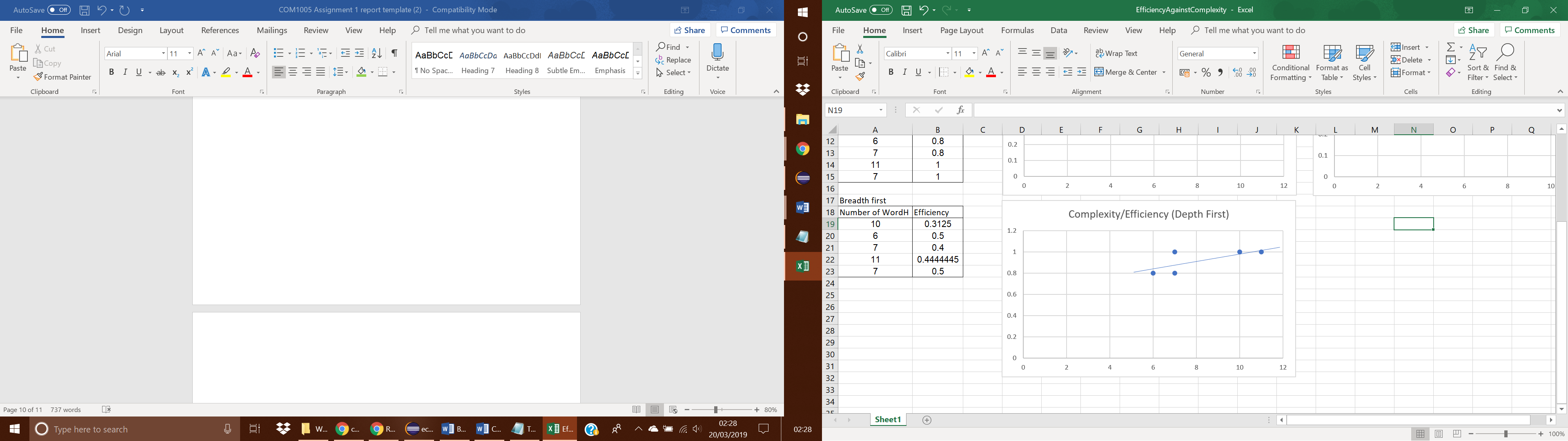
What my results show

From my results, I can see that branch and bound finds the solution path with the lowest cost every time which supports the argument that it is an admissible algorithm. Breadth first finds the least costly path some of the time but often at a lower efficiency.

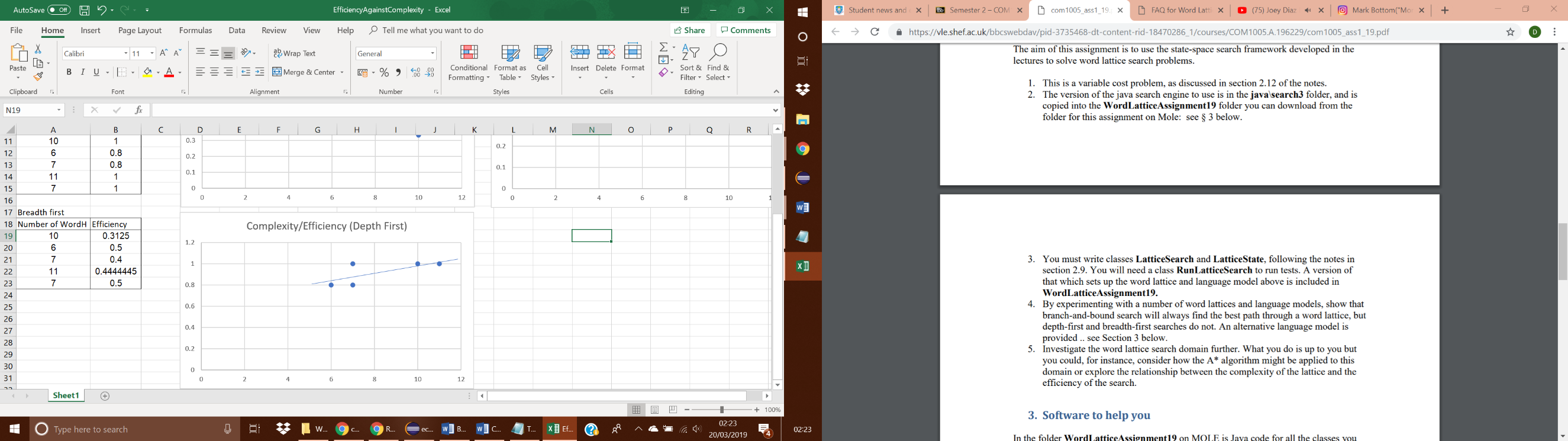
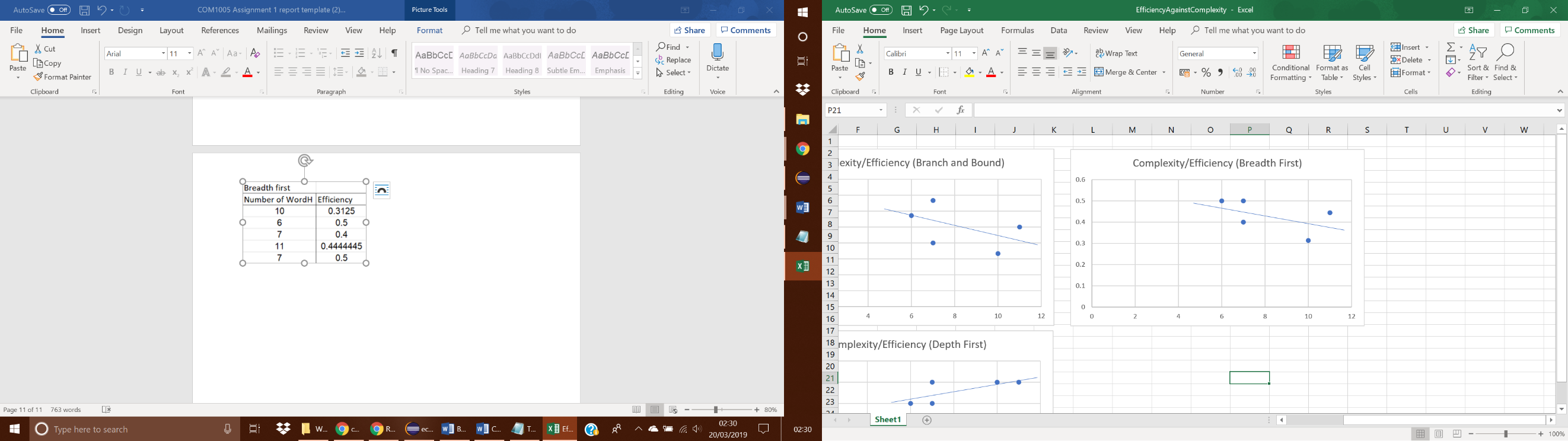
Investigating the relationship between complexity of the word lattice and the efficiency



These results came from my previous experiments and they show that the efficiency decreases as the complexity of the word lattice increases when using branch and bound.



These results came from my previous experiments and they show that the efficiency increases as the complexity of the word lattice increases when using Depth first.



These results came from my previous experiments and they show that the efficiency decreases as the complexity of the word lattice increases when using Breadth first.