ROBERT GORDON UNIVERSITY ABERDEEN

Assessment - Coursework

Academic Year	**2023-2024**
Semester	**Resit**
Module Number	**CM3038**
Module Title	**Artificial Intelligence for Problem Solving**
Assessment Method	**Coursework**
Deadline (time and date)	**19/07/2024 16:00**
Submission	**Assessment Dropbox in the Module Study Area in CampusMoodle.**
Word Limit	**Report: 1500 words and 3 A4 pages**
Use of Generative Artificial Intelligence (AI) text	**IS NOT authorised**
Module Co-ordinator	**Dr. Kit Ying Hui**

What knowledge and/or skills will I develop by undertaking the assessment?

You will be able to:

- Use the given library in Java/Python to implement the A* algorithm that solves a specific problem.
- Derive a heuristic function to use in the A* algorithm for the given problem domain.
- Justify admissibility of the derived heuristic function.

On successful completion of the assessment students will be able to achieve the following Learning Outcomes:

LO3: Design and develop an intelligent game-playing system using a suitable search strategy.

Please also refer to the Module Descriptor, available from the module Moodle study area.

What is expected of me in this assessment?

Task(s) - content

This is an individual assessment.

You will implement the A* search algorithm to solve a puzzle using the given *aips* library in Java or Python. You also need to submit a brief report to describe the heuristic function you use, and justify its admissibility.

What is expected of me in this assessment?

The following shows the relevance of course materials to this assessment:

Task	Topic
A* search	Lectures and labs on Informed searches
Heuristic function in A* search	Lectures and labs on Informed searches
Justification of heurisitic function admissibility	Lectures on Information searches, and admissibility of heuristic function
Use of aips library	All labs using the *aips* library, in particular, those on A* search.

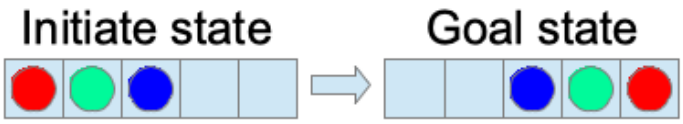
Task(s) - format

1. **Aim**

   This assessment examines the student's ability to solve a search problem using the A* algorithm.

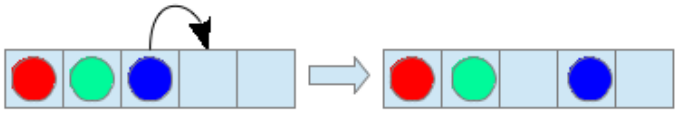
2. **The Problem Domain**

   The *Sliding Tile Puzzle* consists of red, green, and blue tiles arranged on a linear game board with some empty spaces. Given the initial state of the board, the aim of the game is to arrange the tiles into a target pattern. For example:

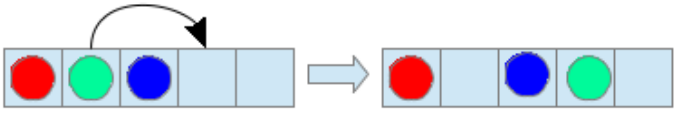


   Depending on its colour, a tile can move in different ways:

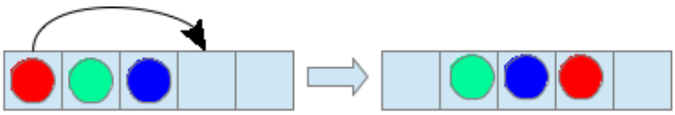
   - A tile of any colour can slide into an adjacent empty space.



   - A tile of any colour can jump over 1 tile into an empty space.



   - Only a red tile can jump over 2 tiles into an empty space. This is NOT allowed for green and blue tiles.



   - No tile can jump over more than 2 tiles, or any empty space.

   The cost of a move is calculated according to the following rules:

   - Sliding a tile to the adjacent empty space has a cost of 1.0.

- Jumping over a red tile costs 4.0, jumping over a green costs 3.0, and jumping over a blue costs 2.0.
- The cost of jumping over multiple tiles (a max of 2) is the sum of the costs of jumping over those tiles individually.

According to these rules, the costs of the three moves illustrated above are 1.0, 2.0 and 5.0 respectively.

3. Resources

Download the file `CM3038 C2 resource.zip` from Campus Moodle. In the zip file, you will find the following:

- Java source of the `aips` library.
- Javadoc documentation of the Java `aips` library.
- A Python module implementing the `aips` library.

You must not change any files in the given library.

- Any Java code should be put into a different package, outside the library.
- Any Python code should be put into a separate file, not inside the library files.

4. The Task

Your task is to develop a program that finds the optimal solution to the *Sliding Tile Puzzle* using the A* algorithm. You must implement the program and submit a written report.

4.1 The Program

The requirements:

- Your program must be written in Java or Python.
- It must implement the A* search algorithm.
- It takes an initial state and a goal state as input parameters.
- It finds and prints the solution with the lowest cost, or "no solution" if none can be found.
- Add appropriate annotations to explain your program code.
- Your solution must use the given Java or Python `aips` library.
  - *You must not change anything in the given libraries.*
  - *A solution that does not use the given libraries will be assigned a grade F automatically.*

In the basic problem:

- The board can be of any size.
- There are 0 or more red, green and blue tiles.
- There is only 1 single empty space.
- *Your highest grade is confined to a B if your program can only solve the basic problem.*

In the general problem:

- All constraints of the basic problem hold, except there must be 2 or more empty spaces.
- *Your program must be able to solve the general problem if you want to achieve an A grade.*

The following is an optimal solution to a basic problem with 1 empty space:

```
Problem: RGGB- -> -BGGR
Node explored: 131 Cost: 16.0

RGGB-

Move tile from position 2->4 (jump over 1) cost: 2.0
RG-BG

Move tile from position 0->2 (jump over 1) cost: 3.0
-GRBG

Move tile from position 1->0 (slide tile) cost: 1.0
G-RBG

Move tile from position 3->1 (jump over 1) cost: 4.0
GBR-G

Move tile from position 4->3 (slide tile) cost: 1.0
GBRG-

Move tile from position 2->4 (jump over 1) cost: 3.0
GB-GR

Move tile from position 0->2 (jump over 1) cost: 2.0
-BGGR
```

The following is an optimal solution to a general problem with multiple empty spaces:

```
Problem: R-GG-B- -> B-GG-R-
No. of nodes explored: 1000
Node explored: 1775 Cost: 17.0

R-GG-B-

Move tile from position 5->4 (slide tile) cost: 1.0
R-GGB--

Move tile from position 2->1 (slide tile) cost: 1.0
RG-GB--

Move tile from position 0->2 (jump over 1) cost: 3.0
-GRGB--

Move tile from position 2->5 (jump over 2) cost: 5.0
-G-GBR-

Move tile from position 4->2 (jump over 1) cost: 3.0
-GBG-R-

Move tile from position 2->0 (jump over 1) cost: 3.0
BG-G-R-

Move tile from position 1->2 (slide tile) cost: 1.0
B-GG-R-
```

4.2 **The Report**

Your report, with a maximum of 1500 words and 3 A4 pages, must have the following content:

- An explanation of your heuristic function.
  - Describe how the heuristic function value is computed from a state.
  - Justify the admissibility of your heuristic function.
    - **Do not just claim that your heuristic function is admissible. You must give a logical argument to justify its admissibility.** i.e. Given any possible state, it will

What is expected of me in this assessment?

never over-estimate the cost to reach the goal. Saying that you have "relaxed some game rules/constraints" is not enough as a justification.

- Examples of running your program on different problem instances.
  - Give actual output of your program. You can copy-and-paste the program output into the report.
  - You should include different problem cases to show that your program is capable of handling tricky situations.
- Clearly cite and reference all resources that you have used.
  - This includes any idea/inspiration/program code that you have copied, adapted, referenced, or used in your implementation. **Failing to do so may result in the accusation of academic misconduct.**
  - You can ignore/omit the `aips` library as you must use it.

Note: RGU has a policy where exceeding the word limit by 10% will result in a grade penalty. You can find out more information at the following link: Assessment Word Limit Statement.

5 Deliverables

Submit to the Campus Moodle drop-box. Your submission must include:

- Your program code which is sufficient to run your program.
  - You do not need to include the given `aips` libraries.
  - You must include other third-party libraries used.
  - **DO NOT upload your work to any public repository. (e.g. Github)**
- A report in PDF format. See section 4.2 for report content.

How will I be graded?

- A grade will be provided for each criterion on the feedback grid which is specific to the assessment. The overall grade for this assessment will be calculated using the algorithm below.
- This assessment contributes 50% to your overall module grade. Your grade for the examination (C1) and the coursework (C2) will be combined using the grid below to give your final module grade. You require an overall grade D or more to pass this module.

A	At least 4 subgrades at Grade A-or-above, and at least 6 subgrades at Grade B-or-above, and normally all subgrades at Grade C-or-above. Solution must be able to solve "general problem" to get a Grade A.
B	At least 4 subgrades at Grade B-or-above, and at least 6 subgrades at Grade C-or-above, and normally all subgrades at Grade D-or-above. Solution that cannot solve "general problem" is limited to a highest grade of B.
C	At least 4 subgrades at Grade C-or-above, and at least 6 subgrades at Grade D-or-above.

How will I be graded?	
D	At least 4 subgrades at Grade D-or-above, and at least 6 subgrades at Grade E-or-above.
E	At least 4 subgrades at Grade E-or-above.
F	Failing to achieve at least 4 subgrades at Grade E-or-above.
NS	Non-submission.

*If the word count is above the specified word limit by more than 10% or the submission contains an excessive use of text within tables, the grade for the submission will be reduced to the next lowest grade.

		Component 1 (Exam)						
		A	B	C	D	E	F	NS
Component 2 (Coursework)	A	A	A	B	B	C	E	NS
	B	A	B	B	C	C	E	NS
	C	B	B	C	C	D	E	NS
	D	B	C	C	D	D	E	NS
	E	C	C	D	D	E	E	NS
	F	E	E	E	E	E	F	NS
	NS	NS	NS	NS	NS	NS	NS	NS

ROBERT GORDON UNIVERSITY ABERDEEN

Grading grid

GRADE	A	B	C	D	E	F
DEFINITION / CRITERIA (WEIGHTING)	**EXCELLENT** Outstanding Performance	**COMMENDABLE/VERY GOOD** Meritorious Performance	**GOOD** Highly Competent Performance	**SATISFACTORY** Competent Performance	**BORDERLINE FAIL**	**UNSATISFACTORY** Fail
Program Functionality **(4 subgrades)**	General problem capable with no major weakness. Implemented all required functionalities correctly & efficiently. (Examples of major weaknesses: inadmissible heuristic, incorrect equality check of state, missing/invalid action generation, incorrect action cost, poor in design/implementation.)	Basic problem only with no major weakness, or general problem capable with 1 major weakness.	Basic problem only with 1 major weakness, or general problem capable with 2 major weaknesses.	Basic problem only with 2 or more major weaknesses, or general problem capable with 3 or more major weaknesses. Correct solution to basic problem.	Partially implemented some basic functionalities but contains significant errors. (e.g. Incorrect result even for basic problem, or always returns "no solution")	No/minimal attempt. (e.g. Program does not compile, or senseless design, or has no clue what is happening.)
Report **(2 subgrades)**	Clear description of heuristic function. Clear justification of heuristic function admissibility. Full citation and references.	Description of heuristic function is mostly clear. Justification of heuristic function admissibility but with a minor flaw. Proper citation and references with minor weakness.	Sufficient explanation of heuristic function. Justification of heuristic function admissibility with a major flaw. Proper citation and references with major weakness.	Barely sufficient description of heuristic function. Justification of heuristic function admissibility is logically unsound. Citation and references mostly missing.	Only description of heuristic function or justification of admissibility exists. The other one is completely missing. No citation and references of used resources.	No description of heuristic function and no attempt to justify admissibility. No citation and references of used resources.
Program Annotations & Test Runs **(1 subgrades)**	Annotations are clear, concise and appropriate. Annotations effectively help the understanding of code. An extensive collection of examples that demonstrate all expected operations, including extreme situations.	Annotations are clear, concise and appropriate but with minor weakness. Annotations help the understanding of code. A good collection of examples that demonstrate all expected operations. May have redundancy/omission.	Annotations are mostly clear, concise and appropriate but with major weakness. Examples demonstrate most of the expected operations.	Some annotations given, which may not help the understanding of code. Examples demonstrate some of the expected operations.	Minimal annotation. Minimal test runs which demonstrate a minority of the expected operations.	No annotations No examples.

Coursework received late, without valid reason, will be regarded as a non-submission (NS) and one of your assessment opportunities will be lost.

What else is important to my assessment?

What is the Assessment Word Limit Statement?

It is important that you adhere to the Word Limit specified above. The Assessment Word Limit Statement can be found in Appendix 3 of the RGU Assessment Policy. It provides detail on the purpose, setting and implementation of wordage limits; lists what is included and excluded from the word count; and the penalty for exceeding the word count.

What's included in the word count?

The table below lists the constituent parts which are included and excluded from the word limit of a Coursework; more detail can be found in the full Assessment Word Limit Statement. Images will not be allowed as a mechanism to circumvent the word count.

Excluded	Included
Cover or Title Page	Main Text e.g. Introduction, Literature Review, Methodology, Results, Discussion, Analysis, Conclusions, and Recommendations
Executive Summary (Reports) or Abstract	Headings and subheadings
Contents Page	In-text citations
List of Abbreviations and/or List of Acronyms	Footnotes (relating to in-text footnote numbers)
List of Tables and/or List of Figures	Quotes and quotations written within "…"
Tables – mainly numeric content	Tables – mainly text content
Figures	
Reference List and/or Bibliography	
Appendices	
Glossary	

What are the penalties?

The grade for the submission will be reduced to the next lowest grade if:

- The word count of submitted work is above the specified word limit by more than 10%.
- The submission contains an excessive use of text within Tables or Footnotes.

What else is important to my assessment?

What is false authorship?

"The University defines this as the practice of submitting work where the student is not the author of that work. The 'false authorship' may relate to the student engaging with a third party and/or software tool to complete an assessment, either in part or whole. This may include work produced by, but not attributed to: another student, an essay mill, a family member or friend, a tutoring service or the unauthorised use of Artificial Intelligence (AI) software. It may also include payment, or other favours, though this will not always be the case. It may relate to any form of assessment or conducting research" (RGU 2023).

What is plagiarism?

Plagiarism is "the practice of presenting the thoughts, writings or other output of another or others as original, without acknowledgement of their source(s) at the point of their use in the student's work. All materials including text, data, diagrams or other illustrations used to support a piece of work, whether from a printed publication or from electronic media, should be appropriately identified and referenced and should not normally be copied directly unless as an acknowledged quotation. Text, opinions or ideas translated into the words of the individual student should in all cases acknowledge the original source" (RGU 2023).

What is collusion?

Collusion is defined as "two or more students working together, without the prior authorisation of the Course Leader, tutor or supervisor, to produce the same piece of work, and then attempting to present this work as their own" (RGU 2023).

For further information please see Academic Integrity.

What if I'm unable to submit?

- The University operates a Fit to Sit Policy which means that if you undertake an assessment then you are declaring yourself well enough to do so.
- If you require an extension, you should complete and submit a Coursework Extension Form. This form is available on the RGU Student and Applicant Forms page.
- Further support is available from your Course Leader.

What additional support is available?

- RGU Study Skills provide advice and guidance on academic writing, study skills, maths and statistics and basic IT.
- RGU Library guidance on referencing and citing.
- The Inclusion Centre: Disability & Dyslexia.
- Your Module Coordinator, Course Leader and designated Personal Tutor can also provide support.

What are the University rules on assessment?

The University Regulation 'A4: Assessment and Recommendations of Assessment Boards' sets out important information about assessment and how it is conducted across the University.