COMP0005 Algorithms
Academic Year 2024-25

Group Coursework: Experimental Evaluation of Search Algorithms

Released: March 6th, 2025
Submission deadline: March 24th, 2025 @ 4pm GMT

The problem.

The problem of "*searching an element in a set*" has been studied for decades in computer science; several algorithms and data structures have been developed to tackle the problem, each with different characteristics, for example in terms of their *time and space complexity*. Some of the proposed solutions have the exact same *theoretical* complexity, for example:

	Data Structure	**Space** Complexity	**Time** Complexity for **Search** Op (worst case)
[1]	2-3 Tree	*O(n)*	*O(log n)*
[2]	AVL Tree		
[3]	LLRB BST		
[4]	B-Tree		
[5]	Scapegoat Tree		

However, *empirically* they may exhibit rather different behaviours depending on, for example, how big the set is, how the elements in the set are distributed, in what order they are inserted in the data structures, and in general the "hidden constants" that are not captured by asymptotic growth (big Oh) analysis.

The task.

Your task for this coursework is twofold:

1) to create an experimental evaluation framework and use it to comprehensively evaluate three data structures of your choice out of the five listed above. The 2-3 Tree and the LLRB BST have been covered during lectures; for this coursework, you are expected to learn at least a third one on your own (among AVL Tree, B-Tree and Scapegoat Tree);
2) to thoroughly and critically discuss the circumstances under which some data structures are better suited than others (among the three you have chosen to compare).

For this coursework, assume the elements in the set to be strings; furthermore, assume that your set grows over time but does not shrink – i.e., you must implement (and evaluate) both *search* and *insert* operations, but there is no need to support a *delete* operation. The data itself is not provided; rather, as part of your experimental framework, you are expected to generate it, so to thoroughly test your chosen data structures (e.g., as your set grows in size and varies in content distribution).

You are expected to implement the search data structures and algorithms, and the experimental framework, from scratch. Do not import any library other than `timeit`, `random`, `string` and `matplotlib`. If in doubt about what you can and cannot use, ask in Moodle "Ask a question" forum. Your experiments must be conducted on your local machine (not on the cloud) using Jupyter Notebook with Python 3.11.

Submission instructions.

Using the Moodle course website, each group must submit exactly two files:

- A Jupyter notebook `grp-`[your group number]`-0005code.ipynb` containing the group's implementation of all search data structures and algorithms, and experimental framework. This notebook must follow the structure of the skeleton code provided. Do not submit the synthetic data you created to experimentally compare the performance of the algorithms; rather, your notebook should contain code to generate it. You may use the Python `random` and `string` packages for this purpose. Use the Python `timeit` library to measure the execution time of your implementation (do not include the time to generate test data in your measurements). Use Python `matplotlib` to visualise your experimental results.

- A PDF document `grp-`[your group number]`-0005report.pdf` of maximum 5 pages (font style: Arial or Times New Roman, font size: 12pt), where you present the results of your experimental evaluation of the three algorithms under consideration. Start the report with a brief overview of the experimental framework you have designed and report the hardware/software characteristics of the platform you have used for experimentations. Then thoroughly and critically discuss the performance results of each of the three algorithms under study. Conclude with a comparative assessment of the selected algorithms and discuss the circumstances under which some algorithms are better suited than others. As part of the report, you are also expected to provide a concise assessment of the contribution made by each team member towards the coursework submission. A template with a recommended report structure is provided.

Submission deadline: Monday, March 24th, 2025 @ 4pm GMT.

Assessment.

This coursework represents 20% of the final mark for COMP0005 Algorithms. Submissions will be evaluated in terms of the following marking criteria:

- Quality of your code (i.e., is your implementation *correct*? Is your code *readable, well documented, reusable*? Is it time and space *efficient*?) [40 points]
- Empirical analysis (e.g., is your experimental framework comprehensive? Have edge-cases been analysed? Have suitable stress-test conditions been experimented with? Are the results of your experimentation correct? Are they thoroughly discussed?) [60 points]

Marks for each criterion will be assigned following the UCL CS Grade Descriptor (criteria 4, 5 and 6).

Note: if a group is experiencing lack of engagement from some members, please report this to the module lead immediately. Students who fail to make a substantial contribution to the group coursework will receive a zero mark.

Academic Integrity.

UCL has a very clear position about academic integrity – what it is, why it is important, and what happens if you breach it. Make sure you have read UCL position on this matter before attempting this coursework.