Project 8: Page Replacement

Due date: Midnight of Wednesday Apr 20, 2022

Project objective:

- You will implement a simulation of page replacement algorithms
- Your simulation will compare page faults when using different algorithms
- Your simulation will highlight the importance of locality of reference

Generate Reference Strings:

Write two functions to generate a reference string of page requests, one function generates a string with locality and the other generates a string without locality of reference. Locality if reference has two components:

- Temporal locality: if a process accesses an item in memory, it will tend to reference the same item again soon.
- Spatial locality: if a process accesses an item in memory, it will tend to reference an adjacent item soon.

The function that generates a reference string WITHOUT locality could follow the following logic:

1. The function takes length as an argument
2. Create an empty results list
3. While the length of result is less than the desired reference string length, generate a random number between 0 and the desired length of the list, and add it to results.
4. Return results

The function that generates a reference string WITH locality could follow the following logic:

1. The function takes length as an argument
2. Create an empty results list
3. Loop from zero to length, with i as index of current iteration
4. Generate a random number between -3 and 3, and add it to i
5. Append the result of the addition to the results list
6. Make sure that the generated reference is above zero and less than the length
7. Return results

Page Replacement Simulation:

Using a frame size of 5, create a simulation to compare the number of page faults for the following page replacement algorithms:

1. First-in First-out
2. Least Recently Used
3. Optimal Page Replacement

Make sure you use the same reference string with the three algorithms, to guarantee a fair comparison. In addition, repeat each algorithm 5 times with a new reference string of size 100 in each time. Use the reference string with locality for this part of the simulation. Generate a plot to compare the average number of page faults for each algorithm. Your plot should show which algorithm has the best performance (closest to optimal).

The second part of your simulation focuses on measuring the effect of locality on each algorithm. Therefore, you will compare the average number of page faults of each algorithm when given a reference string WITH locality compared to a reference string WITHOUT locality. Repeat each algorithm 5 times comparing its performance in the two conditions (with locality and without locality). Each reference string should be of size 100. Generate a plot to compare the average number of page faults for each algorithm in each of the two conditions. Your plot should show which algorithm is more dependent on locality and which isn't.

Report:

Write a separate Google Doc report and organize it as follows, maintaining a coherent document that includes screenshots and text to communicate the objective of your project:

1. Abstract: A brief summary of the project (including findings), in your own words. This should be no more than 150 words. Give the reader context and summarize the results of your assignment.

2. Results: A section that goes over the code you implemented and the performance of each synchronization solution.

3. Discussion: A section that interprets and describes the significance of your findings focussing on the results.

4. Extensions: Describe any extensions you undertook, including text output, graphs, tables, or images demonstrating those extensions.

5. References/Acknowledgements.

Extensions:

You can be creative here and come up with your own extensions. I will suggest the following ideas:

- Write your own page replacement algorithm and compare it to the three algorithms.
- Implement **Least Frequently Used (LFU)** Algorithm and compare it to the three algorithms.

- Implement **Most Frequently Used (MFU)** Algorithm and compare it to the three algorithms.
- Create a visualization of each algorithm like we did in slides.
- Create a simulation of **Belady's Anomaly**, highlighting with figures how adding frames can sometimes degrade performance.

Project submission:

- Add all your files (except *.ipynb_checkpoints* and *__pycache__*) to your ***project_8*** directory on Google Drive.
- Copy the rubric from Moodle to the project directory after you review it.