Homework 6. Languages for faster text searching

[131 home > Homework]

Background

You’re working for Dudelsack, a data analytics firm that deals with large sets of large log files. A set might contain 100 million files, and a file might have a few lines or a billion lines or more. Lines are almost always less than 256 bytes each, though a few can be quite long (as much as a GiB). Although your company has indexing techniques for many of your searches, some of them boil down to running the ‘grep’ command with a regular expression, searching for any match in any line of any file of a set.

Although you can use GNU `grep -r` for this, you’d like something faster. One problem you notice is that `grep -r` uses only sequential code; if `grep` were multithreaded then it could search files in parallel, or parts of a single large file in parallel, and could perhaps get results faster that way, though you would have the issue of outputting lines in the correct order.

GNU `grep` is written in C, but your company is willing to rewrite it in some other language if that would make it easier to parallelize in this way. Your boss asks you to investigate the possibility of using Rust. (Other employees will get different languages.)

Do some research on the chosen language and its support software as a potential platform. Your research should include an examination of the language and system documentation to help determine whether it would be effective. We want to know whether the language supports the proposed application well.

Evaluate the features of the proposed language. What are the biggest strengths and weaknesses you anticipate when using it for this application?

Write an executive summary that gives the language’s strengths and weaknesses, along with problems for this application. Your summary should focus on the technology’s effects on security, ease of use, flexibility, generality, performance, and reliability; the idea is to explore the most-important language-relevant technical challenges in doing the proposed app.

The summary should be suitable for Dudelsack’s software executives, that is, for readers who have some expertise in software, and who know C and C++ reasonably well, along with all the languages already covered in this class.

The summary should be in 10-point font or larger and should be at most two pages. You can put references and appendixes in later pages, if you can’t get under the page limit: the appendixes can contain any source code or diagrams that don’t otherwise fit. Please keep the resources for written reports and oral presentations in mind, particularly its rubrics and its advice for citations to sources that you consulted.

You are allowed, indeed encouraged, to use generative AI like ChatGPT to write your summary. However, you must cite this source just like any other source, and must put into an appendix a complete log of your session (both prompts and results) with ChatGPT or similar service.

For the purpose of this assignment, assume the latest stable version of the language; you need not worry about portability to earlier versions. Specify the version number in your summary.

Submit

Submit a file `hw6.pdf` containing your summary.
