MyFind Protocol: Parallel File Search

Introduction: The myfind utility, coded in C++, offers a parallel file search feature. It optimizes search speeds in vast directories by parallelizing the search for multiple filenames.

Key Parallelization Concepts:

- *Forking Child Processes:* For each file the user wishes to search, the main program forks a child process. This way, every filename is searched in parallel. fork() is the core function that allows for this parallelization. Upon calling fork(), a duplicate process is created, making it feasible to run the search function concurrently for different filenames.
- *Inter-Process Communication (IPC) via Pipes:* With multiple child processes running in parallel, there's a need to communicate the results back to the parent process. For this, the program uses UNIX pipes. Pipes offer a mechanism for processes to communicate with each other by writing to and reading from a shared buffer. In this implementation, child processes write their results (found files' paths or error messages) to the pipe, which the parent process then reads.
- *Synchronized Output:* Even though the processes run concurrently, there is a guarantee of synchronized output to the standard output. This is achieved by reading from the pipe after waiting for each child process to finish its execution using waitpid(). This ensures that full lines are always readable without intermixing.

Synchronization:

- *Avoiding Zombie Processes:* To prevent defunct or "zombie" processes, the parent process waits for each child to complete using the waitpid() function. This ensures that the parent process collects the exit status of the child, thus avoiding any zombie processes.
- *Pipe Synchronization:* Pipes inherently provide a mechanism for synchronized read and write operations. The child processes write their results into the write end of the pipe. Each message is followed by a newline character to delimit different messages. The parent reads from the read end of the pipe. Each read operation fetches the result until it encounters the newline character, ensuring full lines are read.

Features and Functionalities:

- *Recursive Search (-R flag):* The -R option allows the program to search recursively through all subdirectories of the given search path.
- *Case-insensitive Search (-i flag):* The -i option makes the search case-insensitive, so that the program finds the desired file regardless of whether its name was typed in on lower or upper case .
- *Multiple File Search:* The program supports searching for multiple files in parallel. The number of files to search for is flexible, and each file search spawns a separate child process.