CS314 Lab Report
Assignment 2

Aayush Vats CS22BT001, Jai Sharma MC22BT009, Anvay Jaykar CS22BT023

September 2, 2024

0.1 Introduction

This report explains how process management was used to search for patterns in a file, broken down into three parts. The project shows how processes are created, communicate with each other, and end, all done in a C++ environment.

0.2 Part I: Single Process Pattern Search

0.2.1 Objective

To search for a pattern within a specific portion of a file using a single process.

0.2.2 Implementation

- File: `part1_searcher.cpp`
- The program searches a file segment and prints whether the pattern is found.

0.2.3 Results

- Output when the pattern is found: `[66641] found at 64520807`
- Output when the pattern is not found: `[66713] didn't find`

0.3 Part II: Multi-Process Pattern Search

0.3.1 Objective

To search for a pattern using multiple processes, each handling different chunks of the file.

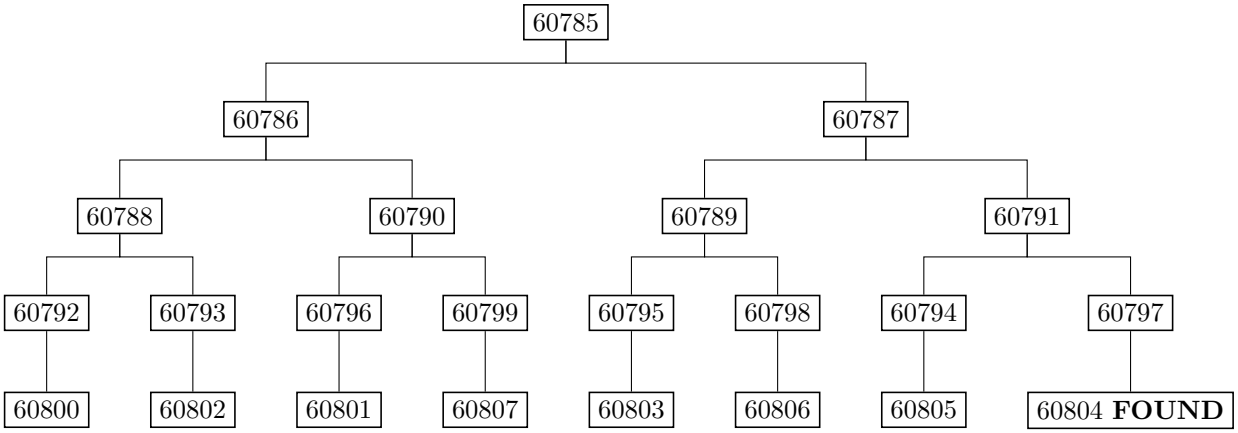
0.3.2 Implementation

- File: `part2_partitioner.cpp`
- Each process forks additional child processes if the file segment exceeds a defined chunk size.

0.3.3 Results

The process tree generated during execution is shown in Figure below.

0.3.4 Tree diagram



0.4 Part III: Optimized Multi-Process Search with Early Termination

0.4.1 Objective

To optimize the search by terminating unnecessary processes once the pattern is found.

0.4.2 Implementation

- Files: `part3_searcher.cpp`, `part3_partitioner.cpp`
- Processes send termination signals to other processes once the pattern is found.

0.4.3 Results

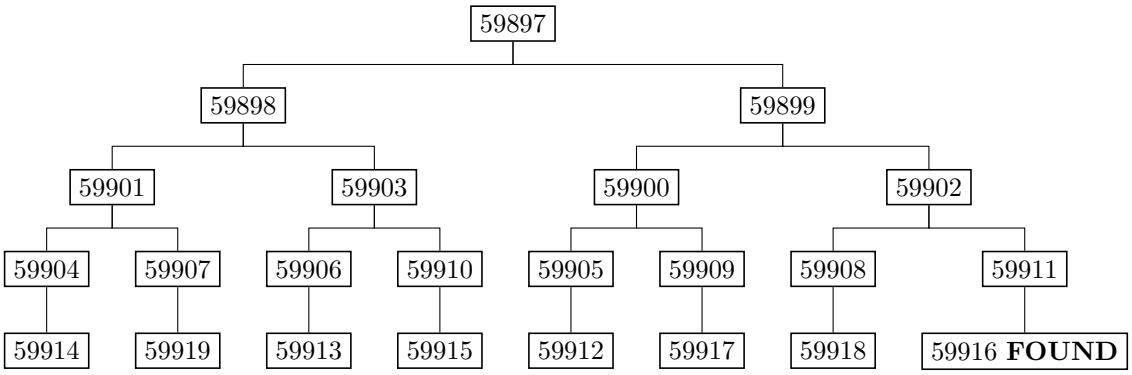
The following output was observed:

```
 [59897] start position = 0 ; end position = 67108863
[59897] forked left child 59898
[59897] forked right child 59899
[59899] start position = 33554432 ; end position = 67108863
[59898] start position = 0 ; end position = 33554431
[59899] forked left child 59900
[59898] forked left child 59901
[59899] forked right child 59902
[59898] forked right child 59903
[59901] start position = 0 ; end position = 16777215
[59900] start position = 33554432 ; end position = 50331647
[59903] start position = 16777216 ; end position = 33554431
[59901] forked left child 59904
[59902] start position = 50331648 ; end position = 67108863
```

```
[59900] forked left child 59905
[59903] forked left child 59906
[59902] forked left child 59908
[59901] forked right child 59907
[59900] forked right child 59909
[59903] forked right child 59910
[59902] forked right child 59911
[59905] start position = 33554432 ; end position = 41943039
[59906] start position = 16777216 ; end position = 25165823
[59904] start position = 0 ; end position = 8388607
[59905] forked searcher child 59912
[59906] forked searcher child 59913
[59910] start position = 25165824 ; end position = 33554431
[59904] forked searcher child 59914
[59911] start position = 58720256 ; end position = 67108863
[59910] forked searcher child 59915
[59909] start position = 41943040 ; end position = 50331647
[59908] start position = 50331648 ; end position = 58720255
[59907] start position = 8388608 ; end position = 16777215
[59911] forked searcher child 59916
[59909] forked searcher child 59917
[59908] forked searcher child 59918
[59907] forked searcher child 59919
[59916] found at 64520807
[59914] received SIGTERM
[59913] received SIGTERM
[59919] received SIGTERM
[59911] received SIGTERM
[59906] received SIGTERM
[59909] received SIGTERM
[59916] received SIGTERM
[59905] received SIGTERM
[59902] received SIGTERM
[59907] received SIGTERM
[59908] received SIGTERM
[59899] received SIGTERM
[59903] received SIGTERM
[59904] received SIGTERM
[59900] received SIGTERM
[59897] received SIGTERM
[59901] received SIGTERM
[59910] received SIGTERM
[59898] received SIGTERM
[59915] received SIGTERM
[59918] received SIGTERM
[59912] received SIGTERM
```

```
[59917] received SIGTERM
```

0.4.4 Tree diagram



0.5 Conclusion

This project successfully used process management for pattern searching. Early termination made the process more efficient by stopping unnecessary searches.