Pin Finder

Daniel A. Ellsworth
Mathematics and Computer Science
Colorado College
Colorado Springs, CO, USA
dellsworth@coloradocollege.edu

Allen Malony
Computer Science
University of Oregon
Eugene, OR, USA
malony@cs.uoregon.edu

***Abstract*—A basic assignment that has connections to parallel programming, security, and performance topics.**

***Index Terms*—parallel programming, parallel processing, curriculum development**

I. Assignment Context

Pin Finder was initially developed as the first parallel programming lab in a 10 week shared memory parallel programing course at the University of Oregon in 2015 [1], [2]. The course used Intel's complier technologies and Structured Parallel Programming [3] as the textbook. Labs in this course were gamified by placing students in the context of developers at the Office of Strategic National Alien Planning (OSNAP). Each lab was 50 minutes and would start with a brief reminder of a parallel pattern from the textbook, an OSNAP business problem, and serial code in C that would slowly solve the problem. Students would spend roughly 45 minutes working on a solution with a grad student to help with questions.

The Pin Finder assignment was also used at Colorado College in a Fall 2017 parallel programming course [4] based on the University of Oregon course. Courses at Colorado College are 18 days long, meeting for 3 hours of class each day. During the parallel patterns portion of the course, the last hour of class was used as a lab. No grad student support was available and the OSNAP gamification was mostly removed for this delivery.

II. Sample Lab Prompt

OSNAP security policies demand workers use an 8 digit pin that may not be written down, cannot be reset, and is changed daily to access secured resources. Executive management has requested a pin recovery tool since the pins are frequently forgotten. To be compliant with organizational security policies, no system or person may keep pins in plain text or reversibly encrypted. A hashed pin has no confidentiality requirements according to OSNAP policies.

1) User carries the hashed pin
2) User enters the hash when the pin is needed
3) Software hashes all pins
4) Software return a matching pin

Serial code has been provided that supports this procedure, but it is too slow. How much faster can you make this using parallelism?

Development at the University of Oregon supported by an Intel IPCC Grant.

III. Discussion Topics

The lab can be connected to several different individual or class discussion topics based on instructor and student interests.

A. Parallelism

The lab is placed early in the course when the map pattern is being discussed. In my deliveries correctly using a parallel-for is sufficient to produce a passing lab solution. For the right pins, even with a naive parallel-for, good speed-up can be observed on hardware with low core counts. Placing the parallel-for is relatively easy however lack of care in variable handling results in errors due to race conditions.

B. Performance

Depending on where the pin is located in the search space, the observed performance improvement of parallel-for over the serial loop varies since the serial code uses early loop termination. This provides an opportunity to talk about the need to be aware of how parallelism is implemented, overheads, and care in design when deploying parallelism. In the Spring 2017 delivery at the University of Oregon a leader board of student submissions was used to encourage students to optimize their code [5].

More performant solutions comparative to the serial solution will involve students considering how the search is conducted and how the threads interact. Students might try a serial loop around the parallel region, a shared variable to flag to skip work loop bodies after a solution has been found, changing how the search space is allocated across threads, or a parallel pattern other than map.

C. Security

Hashing is one-way but collisions must be considered. Hashed passwords are better than plaintext but are not sufficient if the hash is not properly protected. Password crackers, like the simple one being built in this lab, can recover a working password from a hash sniffed over the network or exfiltrated from a database given sufficient resources. Adding blocks to blockchains, via proof of work, frequently involves a similar problem to the pin finder.

IV. Common Stuck Points

The lack of guidance around construction of the final solution invites creativity in the problems students encounter.

A. Finding a Parallel Pattern

This assignment has been used when the map pattern is being discussed and most students therefore gravitate toward a parallel-for. Students that are unsure where they might start should be encouraged to think about what the while loop in the serial code is doing (iterating over a range) and to modify the code to use a for-loop.

B. Test Iteration Time

Using a large PIN space enables even naive solutions to achieve good observed speedup for some PINs on low core counts however using the full search space makes testing slow. Students that are waiting a long time for tests should be encouraged to think about wether the whole search space is needed during testing.

C. Wrong/no PIN Returned

The serial code uses variables declared outside of the loop. Of specific interest is a character buffer used in converting pin numbers to corresponding strings for hashing. Naively adding parallelism results in a race condition where the wrong pins will be checked, resulting in no matches, or the returned "matching" pin will be incorrect. Students might be reminded of race conditions from the lecture/textbook or asked to consider how the shared memory model interacts with threads.

REFERENCES

[1] [Online]. Available: http://ipcc.cs.uoregon.edu/curriculum.html

[2] [Online]. Available: https://classes.cs.uoregon.edu/14S/cis410parallel/lab5.php

[3] M. McCool, J. Reinders, and A. Robison, *Structured Parallel Programming: Patterns for Efficient Computation*, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012.

[4] [Online]. Available: http://cs.coloradocollege.edu/ dellsworth/2017b2/

[5] [Online]. Available: https://classes.cs.uoregon.edu/17S/cis431/lab4.php