Assignment 2 report for CSCI P536

Saurabh Mathur

September 19, 2018

Concurrency Design

My implementation of pargrep is designed using the producer-consumer model. Concretely, threre is one the main thread which is the producer and t child threads which are the consumers. The producer reads from the input file or standard input line by line and pushes each line along with its line-number to a shared queue. Each consumer removes a line and its line-number from the queue and searches for the word in the line. Then, the consumer checks if all lines before it have been printed, by comparing the value of a shared variable to the current item's line number. If they have not been printed, the thread yields to the other threads. Once the thread prints its line, it increments the shared variable.

The queue is a ring buffer implemented using an array. Access to the queue is synchronized using a mutex and two semaphores. The queue implementation resides in `ring_buffer.c` and `ring_buffer.h`

I chose the producer consumer model for my implementation as it allows clear separation of concerns and reduces the chances of concurrency related bugs. The implementation does not make any assumptions about the input being a file on disk or about the RAM of the system on which it is being executed. The only limiting factor in this regard is the line number which is stored as long long int. As a result, the maximum number of lines an input file can have would be $2^{63} - 1$.

Performance Comparision

I used the `enwik8` 100MB file from the hutter prize (http://prize.hutter1.net/) for performance testing.

The time taken in seconds by pargrep and grep along with the line count in the output is given in table 1.

Word	pargrep (t=1)	pargrep (t=4)	pargrep (t=8)	grep	Line count
you	1.165	1.850	5.691	0.6	6570
system	4.991	5.663	5.591	0.952	10950
federation	1.110	1.604	5.766	0.096	396

Table 1