UMEÅ UNIVERSITET
Institutionen för Datavetenskap
Rapport obligatorisk uppgift

Systemnära Programmering (C) 7.5 p
5DV088

Obligatorisk uppgift nr 4

Namn	Jonas Sjödin
E-post	josj0105@student.umu.se
Datum	3 december 2017
Kursansvarig	Mikael Rännar
Handledare	Didrik Lindqvist, Jakob Lindqvist, William Viktorsson

1 Sökalgoritm

Sökfunktionen fungerar så att varje tråd går in i en oändlig loop där jag låser min globala mutex. Sedan körs en whileloop som körs så länge som stacken är tom. I den kollar jag om alla trådar är färdiga eller om alla trådar väntar. I detta fall är sökningen klar och tråden terminerar. Om inte detta uppfylls så är inte sökningen färdig än och tråden går in i else-satsen. Där ökar jag antalet väntande trådar med 1 och broadcastar på min mutex så att mutexen låses upp men tråden väntar på condition-variabeln. Denna släpps när ett nytt directory satts in i stacken och mutexen låses igen.

Sedan fortsätter sökningen med att det första värdet i stacken hämtas och mutexen låses sedan upp. Sedan öppnas directoriet som hämtats från stacken och den börjar läsas igenom. Om namnet inte är ”.”eller ”..så skapas en sträng som sätts till nuvarandedir + / + fil/mapp/länknamnet. Om det är en mapp så sätts den in i stacken. Sedan kollas det om den är av rätt typ (fil/mapp/länk) om den är det sätts den in i en lista. Sedan stängs directoriet och trådens antal lästa directories variabel ökas med 1.

2 Trådsäkerhet

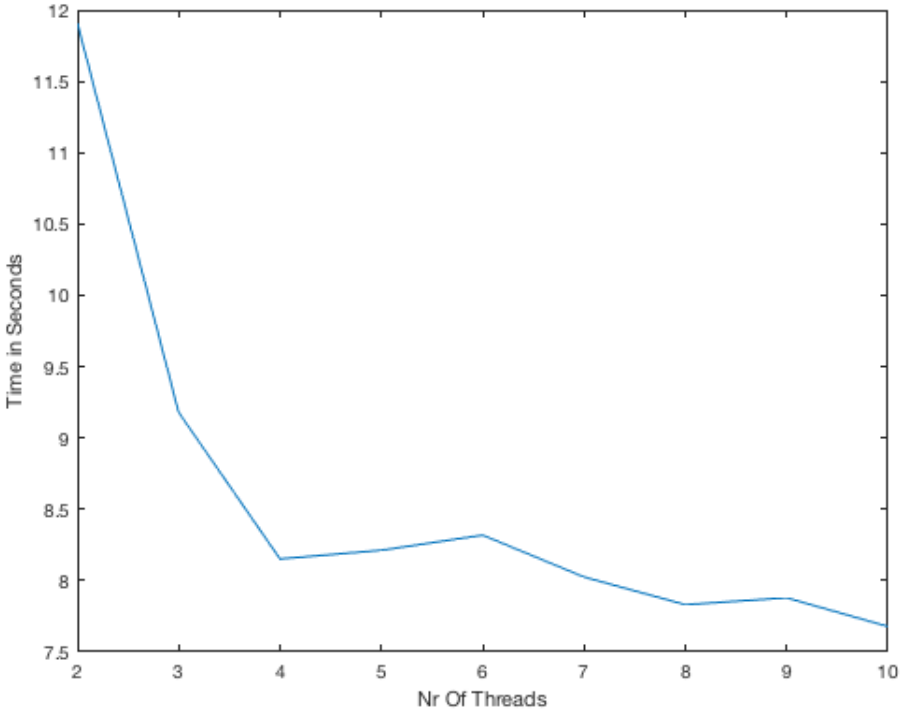
Min kodimplementation är trådsäker. I denna uppgift använder jag mig av en stack och en dubbellänkad lista där listan är trådsäker. Där är allting utom pop/remove och isempty trådsäkra. Detta är pga att de inte går att göra helt trådsäkra då de måste köras i samma sekvens i min implementation. Jag låser alltid min globala mutex innan jag kör I/O-anrop eller kör cond_broadcast och cond_signal. Detta gör jag även när jag ska använda min stack. När en lista ska användas låses listans egna mutex innan operationerna körs. Därför uppstår inga race-conditions i min kod.

3 Effektivitet

Detta test är utfört genom att söka efter namnet ’mfind’ i root-directory på min egen Lenovo YOGA 900isk med intel i7 som kör openSUSE. Testet är utfört 10 gånger får varje antal trådar och sedan beräknades medeltiden för varje antal trådar. Som grafen visar så går det snabbare ju fler trådar som används. Den stora skillnaden sker dock när man går från 2 till 4 trådar. Detta har förmodligen att göra med att min processor har 2 kärnor men 4 trådar. Sedan sker en liten förbättring ju fler trådar som används men den skillnaden är väldigt liten.

4 Reflektioner

I min implementation skriver jag inte ut startmappen om den har samma namn som sökordet då jag anser att det är onödigt då man redan söker från den mappen. Jag har implementerat en funktion som kollar dubletter och subdirectories i stacken innan sökningen startar. Om dessa hittas så tas de bort då jag anser att det är onödigt att söka igenom ett directory mer än en gång.


12
11.5
11
10.5
10
9.5
9
8.5
8
7.5
Time in Seconds
2
3
4
5
6
7
8
9
10
Nr Of Threads