Ukeoppgave IN3030/IN4330 – v2025

Eric Jul etter opplegg av *Arne Maus*

Ved gruppetimene vil hjelpelærerne hjelpe med parallelprogrammering i Java. Disse ukeoppgaver er til at øve at lage simple Java programmer med tråde. De er ikke obligatoriske, MEN det bliver nemmere å lage Oblig 1, hvis du gjør det.

1. Hei til verden

Det første alle skal gjøre er å lage et program som genererer like mange tråder k (i tillegg til main-tråden) som det er kjerner på den maskinen du sitter på. Hver tråd skal ha en lokal variabel 'ind'. (ind = 0,1,..,antallTråder-1)

I metoden 'run' skal du plassere først denne setningen:

a) System.out.println("Traad nr: " + ind+" sier hei");

og så følgende to setninger i en try{..} catch blokk:

```
b) try {  Thread.sleep(1000);
          System.out.println("Traad nr: " + ind+" sier hei etter å ha ventet ett sekund") ;
   } catch (Exception e) { return;}
```

Kjør programmet flere ganger – både a) og b) – og se etter om setningene kommer i 'riktig' rekkefølge hver gang. Blandes utskriften?

2. Finn maksimalverdien i en array

Vi skal nå parallellisere problemet med å finne det største elementet i en heltallsarray a[] av lengde n - og skrive: 'int finnMax(int a[]) {..};

Arrayen vi skal først lage en slik array med tilfeldig innhold med klassen Random i pakken: java.util hvor man nytter metoden nextInt(n) for å trekke elementene i a[].

Vi skal nedenfor teste denne for n = 100, 1000, ..., 10 millioner.

a) Lag først en sekvensiell løsning og ta tiden på denne med System.nanoTime() da i nanosekunder og skriv ut tiden i millisekunder. Del nanosekundene med 1000000.0 for å få millisekunder som en double-variabel. Lag en tabell over resultatene.

b) Parallell løsning:
Vi skal så se på to måter å parallellisere dette problemet hvor vi har k tråder:

b.1 Først deler du opp arrayen slik at tråd 0 får de første n/k elementene, tråd 1 har de neste n/k elementene,.. osv., og den siste tråden får resten av arrayen.

b.2 Deretter deler vi arrayen a[] slik at tråd 0 tester element: 0, k, 2k, 3k,..osv. Tråd 1 tester element nr. 1, k+1, 2k+1, ... Tråd 2 tester element 2, k+2, 2k+2,... Husk å teste slik at man ikke går ut over enden av arrayen.

Vi ser at i begge tilfeller skal trådene oppdatere en felles variabel kalt f.eks. globalMax. Her kan du teste ut to strategier:

b.3 Bruk en synchronized metode som alle trådene kaller for hvert element de har og som oppdaterer globalMax hvis det nye elementet er større enn gammel verdi av globalMax.

b.4 La alle trådene ha hver sin variabel localMax som de bruker til å finne max i sin del av arrayen, Tilslutt kaller hver tråd da bare en gang på en synchronized metode (samme for å sette golbalMax i b.3) som da finner ut hvilken av trådenes lokale max som var størst.

Lag en liten rapport om tidsforbruket med de 4 mulige varianter av algoritmen + tidsforbruket til den sekvensielle algoritmen hvor du også regner ut speedup S. Skriv også om det synes mulig å få S >1 for noen av disse fire alternativene. Kommenter hvorfor noen av algoritmene evt. er spesielt raske eller spesielt treige.

3) Hvis tid: Løs problemet å lage en parallell versjon av summen av elementene i en array

$$Sum = \sum_{i=0}^{a.length-1} a[i]$$

etter samme mal som oppgave 2 . Lag en liten rapport med kjøretidene. Trekker du samme konklusjoner som ved finnMax-problemet?