André

Introduktion

Vi i grupp 20 kom fram till att vi ville utöka laboration 5 som handlade om att undersöka om vilken av Collections.sort och en egenskriven sorteringsalgoritm presterar bättre. I vårt projekt valde vi ut att undersöka prestandan kring datastrukturerna Linked List och TreeSet med input av typen Integer om vilken som presterar bäst. Ni känner igen Linked List men TreeSet är helt enkelt ett träd där ordningen av elementen ligger som ett Set, alltså i stigande ordning enligt den naturliga ordningen där det inte finns dubbletter av något element som

BYT BILD

Vi har valt ut 4 frågeställningar som vi undersökte under projektet. Vi ska undersöka operationerna för tilläggning och sökning i datastrukturerna och därför har vi två frågeställningar där vi ska undersöka om mätningarna för dessa två operationer i Linked List och TreeSet överensstämmer med det teoretiska värdet. Vi tyckte att testerna kring att köra med och utan JIT-kompilatorn var intressanta i laboration 5 och därför har vi en frågeställning där vi ska jämföra om resultatet är likt det teoretiska även när JIT-kompilatorn är avstängd. Som sista frågeställning har vi att vi ska undersöka kring vilken datastruktur som presterar bäst vid given indata.

BYT BILD (Din tur Gustav)

Vår metod är kraftig inspirerad av lab 5 i det att vi valde att skriva ett java program som kör insättning och sökning i det två data strukturerna linked list och tree set 30 gånger och skriver ner tiden för varje operation. Vi använde oss av javas interna algoritmer sorterings och insättnings. Detta program körs på 3 olika indata bestående av 1 000, 10 000 respektive 100 000 osorterade heltal mellan 0 och 9999. Körningarna upprepas med JIT-kompilatorn avstängd på indata 1 000 och 10 000. 100 000 Hade tagit väldigt lång tid och kördes inte.
Datan importeras i R för att summera resultat och köra t-tester.

BYT BILD

Erik - resultat?

BYT BILD

Axel - snacka om potentiella felkällor /

- **Notera att för JIT undersöktes enbart 1000 och 10 000 då indata på 100 000 tog alldeles för långt tid för att exekvera**

Potentiella felkällor ish:

Kunde tagit mer hänsyn till invängningsförloppet samt jämviktområdet. Om uträkning i jämviktsområdet av medelvärdena görs kommer resultatet vara mer konstant då javas minnesmodell lär sig varje gång programmet exekveras och tiden kortas ner.
I vårt resultat används data från de första körningarna vilket ger ett högre medelvärde än det faktiska.

R-scriptet skulle också kunna kört säg 30 gånger och tagit medelvärde varje gång för bättre precision, men endast vid en mätning tog det väldigt lång tid för stor indata.
Vi kunde ju inte helt bestämma huruvida exekveringstiden tycktes följa den teoretiska tidskomplexiteten. Men där det nog var tydligast och fanns antydningar var när programmet kördes utan JIT, men där kunde vi bara göra mätningar för 1000 och 10.000