UMEÅ UNIVERSITET
Institutionen för datavetenskap
Elmroth, Kågström

Design och analys av algoritmer
för parallelldatorsystem – vt2000
Laboration 1
26 januari 2000

# Parallell matrismultiplikation

Sista inlämningsdag: 2000-02-17.

Laborationen, som får lösas individuellt eller parvis, består av två delar som tillsammans behandlar analys och implementation av en valfri algoritm för matrismultiplikation på IBM SP-systemet. Det står fritt att välja en av följande algoritmer: en enkel blockalgoritm (avsnitt 5.4.1 i kursboken), Cannons algoritm (avsnitt 5.4.2) och Foxs algoritm (avsnitt 5.4.3). I laborationen läggs stor vikt vid analys av algoritmens prestanda.

Uppgifterna skall dokumenteras väl och alla resultat skall följas av diskussioner som visar att du förstår resultaten. Även implementationen skall dokumenteras väl och testkörningar som visar att programmet räknar rätt skall bifogas. Ange sökväg till körbara filer och ge instruktioner till hur programmet körs.

En laboration som inlämnas i tid ger 1–2 bonuspoäng till tentamen. En förutsättning för att få bonuspoäng är naturligtvis att den inlämnade laborationen är tillräckligt bra för att ge betyget G eller O.

## Implementation av algoritm för matrismultiplikation

Implementera algoritmen för matrismultiplikation modifierad så att den beräknar $C = \beta C + \alpha AB$, där $A$ (av dimension $m \times k$), $B$ ($k \times n$) och $C$ ($m \times n$) reella matriser och $\alpha$ och $\beta$ är reella tal. Implementationen skall vara konstruerad för SP-systemet betraktat som ett 2-dimensionellt nät med $p$ processorer. Ni kan förutsätta att det 2-dimensionella processornätet är kvadratiskt. Förutsätt också att matriserna $A$, $B$ och $C$ är kvdratiska av dimensioner som går ”jämnt ut” på antalet processor i aktuell dimension av processornätet samt att varje matriselement samt $\alpha$ och $\beta$ är av typen `double`. För att minska kodningsarbetet finns skelett till program tillsammans med Makefile på ~elmroth/PB2/Lab1. Det är naturligtvis tillåtet att modifiera dessa program.

Observera att för de tre första algoritmerna skall inledningsvis nod $(i, j)$ (eller nod $(i-1, j-1)$ för den som hellre vill ha det så) hålla blocken $A_{ij}$, $B_{ij}$ och $C_{ij}$. Dvs, för samtliga algoritmer gäller att all omdistribuering av data skall utföras i samband med matrismultiplikationen.

För att erhålla bättre prestanda på varje enskild processor skall de lokala matrismultiplikationerna utföras genom anrop till rutinen `DGEMM` i det seriella ESSL-biblioteket.

## Analys av matrismultiplikationsalgoritmen

Gör tester och redovisa tider, Mflops, uppsnabbning och effektivitet för följande problemstorlekar: $m = n = k = 128, 256, 512, 1024$ och 2048. (Mflops är prestandan mätt i miljoner flyttalsoperationer per sekund). Notera att att ni för att få bra resultat på en processor för den största problemstorleken måste köra på en nod med mer än 128 MB minne. Det ska dock vara en 120 MHz-processor för att resultaten ska kunna nyttjas som utgångspunkt för beräkning av uppsnabbning och effektivitet på större antal processorer.

I boken finns ett uttryck för tiden $T_p$ på $p$ processorer. Modifiera detta uttryck så att en addition och en multiplikation vardera räknas som en operation, dvs tillsammans utgör de två operationer. Tag experimentellt fram ett approximativt värde på tiden $t_a$ för att utföra en flyttalsoperation utifrån testerna på 1 processor och det minsta kvadratiska problemet. Sätt in aktuella värden för kommunikationstider på SP-systemet i uttrycket och beräkna teoretiska resultat för samma problem som tidigare har testkörts (redovisa även här tider, Mflops, uppsnabbning och effektivitet, samt förklara skillnader mot de experimentella resultaten).

Uttnyttja ert detaljerade uttryck för $T_p$ för att göra följande skalbarhetsanalys. Utgå från den verkliga exekveringstiden på 1 processor för den minsta problemstorleken. Beräkna hur stort problemet måste vara för $p = 4, 16$ och 64 för att effektiviteten $E = 0.95$ skall erhållas. (Avrunda problemstorlekarna så att matriserna går "jämnt ut" på det aktuella processornätet.) Gör tester på dessa problemstorlekar och redovisa resultatet och försök förklara eventuella avvikelser.

När effektivitetsjämförelser görs på olika problemstorlekar beräknas den som

$$E_p = \frac{\text{Mflops}(p)}{p\text{Mflops}(1)}$$

där Mflops$(i)$ är beräknad på aktuell problemstorlek för $i$ processorer.