UMEÅ UNIVERSITET
Institutionen för datavetenskap
Erik Elmroth och Bo Kågström

Tentamen på kursen Parallella Beräkningar II

onsdagen den 11 januari 1995 kl 9 - 15 i skrivsal MIT

Tentamen kan maximalt ge 44 poäng. För betyget godkänd krävs 22 och för betyget väl godkänd 33 poäng. Poängtal anges inom parentes för varje uppgift. Uppgifterna är ej placerade i svårighetsordning.
Hjälpmedel:

- Miniräknare

Skrivsalen får ej lämnas förrän en halvtimme efter skrivningens början.

OBS!!!
Skriv namn på varje blad och börja varje uppgift på nytt blad. Var noga med att redovisa alla beräkningar. Markera på försättsbladet vilka uppgifter du lämnar in lösning till och lägg lösningarna i nummerordning.

Skriv TYDLIGT!

Lycka Till!

Uppgift 1 ($1 + 1 + 1 + 1 = 4$ poäng)
Beskriv kortfattat vad som avses med följande begrepp för parallella algoritmer.

a. Kornighet (*granularity*).

b. Grad av parallellitet (*degree of parallelism*).

c. Kostnadsoptimal (*cost optimal*).

d. Skalbarhet (*scalability*).

Uppgift 2 ($2 + 2 + 2 + 2 = 8$ poäng)
Konstruera algoritmer för 1-till-alla utsändning (*one-to-all broadcast*) av meddelande av storlek m för p processor i de tre *store-and-forward*-nätverken i deluppgift **a** - **c**, där varje länk är dubbelriktad. Beräkna även kommunikationstiden för varje nätverk. Antag att p är en potens av 2 och utnyttja lösningen i deluppgift **a** för att lösa deluppgift **b** och **c**.

a. En ring.

b. Ett 2-dimensionellt nät.

c. En hyperkub.

d. Om vi för en ring av processorer istället har ett *cut-through*-nätverk så kan algoritmen i deluppgift **a** förbättras. Konstruera en förbättrad algoritm och beräkna dess kommunikationstid.

Uppgift 3 ($2 + 2 + 2 = 6$ poäng)
Redogör kortfattat för hur följande tre i kursboken beskrivna strategier för lastbalansering i parallella algoritmer för djupet-först-sökning fungerar. Ange också en fördel och en nackdel med varje metod.

a. *Asynkron Round Robin.*

b. *Global Round Robin.*

c. *Random Polling.*

Uppgift 4 ($2 + 2 + 3 + 2 = 9$ poäng)
Matrismultiplikationen $C = C + AB$, där A, B och C är $n \times n$-matriser kan göras parallellt på en ring av p processorer avbildad på en p-processorers hyperkub. Antag att matriserna A och C i utgångsläget är partitionerade radvis med *block-striped partitioning* så att processor P_i håller radblock i och att B är partionerad kolumnvis med *block-striped partitioning* så att processor P_i håller kolumnblock i.

a. Konstruera en parallell algoritm för att utföra matrismultiplikationen på hyperkuben betraktad som en ring av processorer. Rita gärna figurer som visar att du förstått matrispartitioneringen.

b. Antag att en addition och en multiplikation tar en tidsenhet. Beräkna arbetet W ($= T_1$), tiden på p processorer T_p, uppsnabbningen S och effektiviteten E. Antag att t_s är uppstartningstiden och t_w är tiden per element (tal i matriserna) vid meddelandeöverföring.

c. Avgör om algoritmen är kostnadsoptimal (och bestäm i så fall för vilket förhållande mellan n och p det gäller) och beräkna dess asymptotiska isoeffektivitetsfunktion med avseende på kommunikation och parallellitet samt dess totala (*overall*) isoeffektivitetsfunktion.

d. I kursboken beskrivs Cannons algoritm för att utföra matrismultiplikationen på ett 2-dimensionellt nät av processorer. Med det 2-dimensionella nätet avbildat på en hyperkub med p processorer blir $T_p = \frac{n^3}{p} + 2\sqrt{p}t_s + 2t_w \frac{n^2}{\sqrt{p}}$ och algoritmens asymptotiska isoeffektivitetsfunktion $\theta(p^{3/2})$. Försök förklara vilka skillnader i algoritmerna som ger upphov till skillnaderna i isoeffektivitetsfunktionerna för de två algoritmerna.

Uppgift 5 (3 + 4 + 1 = 8 poäng)

Betrakta ett $\sqrt{n} \times \sqrt{n}$-nät som i figur 1 för fallet $n = 16$ (figuren visar fallet $n = 64$). Nätet kan exempelvis utgöra den diskretiserade representationen av en yta som vi vill beräkna temperaturen på. Nätets punkter (noder) representerar då de punkter på den yta där vi skall beräkna temperaturen. Om vi antar att temperaturen i varje punkt endast beror direkt av temperaturen i grann-punkterna så kommer temperaturen i punkt i direkt att bero på temperaturen i punkt $i - \sqrt{n}$, $i - 1$, i, $i + 1$ och $i + \sqrt{n}$ (för de fall där vissa av dessa punkter ej finns, dvs randpunkterna, skall de helt enkelt utelämnas). Dessa beroenden för fallet $n = 16$ kan på matrisform representeras av matrisen A i figur 2.

Antag att vi vill beräkna en vektor x där x_i representerar temperaturen i punkt i och där x är lösningen till $Ax = b$. Detta kan exempelvis göras iterativt med Gauss-Seidels metod där $x^{(k)}$ är den approximativa lösningen beräknad i iteration k på följande sätt för en godtycklig matris A:

for $i = 0,\ n - 1$

$$x_i^{(k)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=0}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n-1} a_{ij} x_j^{(k-1)} \right)$$

I en gles matris är naturligtvis många av a_{ij}-elementen lika med 0, så att alla dessa beräkningar inte behöver utföras.

a. I en iteration av Gauss-seidels iteration med A som i figur 2 finns det stora begränsningar för parallellisering. Visa vilka punkter som kan beräknas parallellt under en iteration för fallet $n = 16$.

b. Visa hur man med hjälp av röd-svart omordning av nätets punkter (*Red-Black ordering*) kan omordna ekvationerna så att det är möjligt att beräkna flera värden samtidigt (för fallet $n = 16$) och ange vilken matris som erhålls (med samma notation som i figur 2) samt vilka värden som kan beräknas parallellt och när synkroniseringar behövs.

c. Beskriv kortfattat hur principen med röd-svart omordning kan generaliseras till att gälla andra typer av glesa matriser.

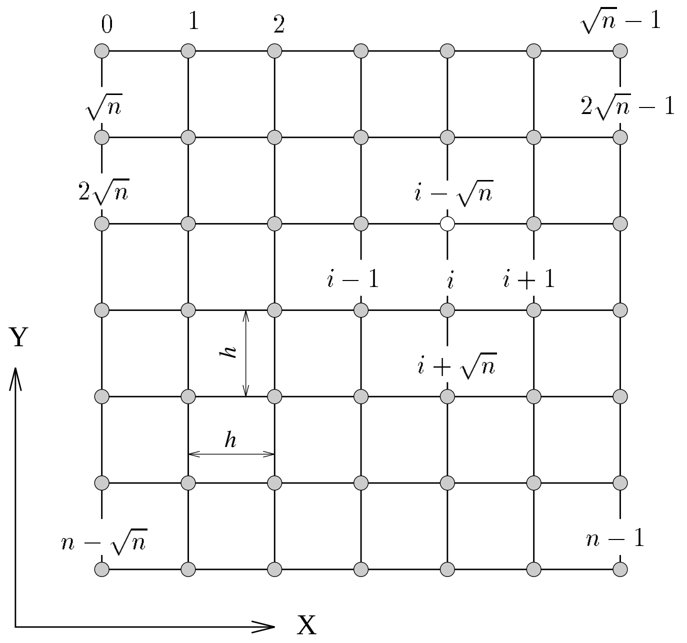

Figure 11.8 A $\sqrt{n} \times \sqrt{n}$ grid with natural ordering of grid points.

Figure 1: Ett kvadratiskt n-punkters nät med naturligt ordnade punkter.

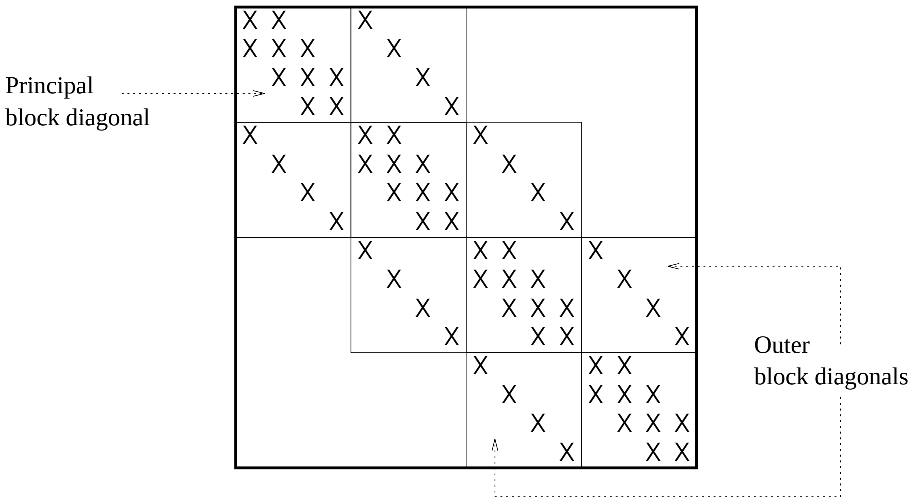

Figure 11.7 A 16×16 block-tridiagonal matrix. The nonzero elements are represented by the symbol $\times$. Zeros are not shown.

Figure 2: Blocktridiagonal 16×16-matris från nät som i figur 1.

Uppgift 6 $(3 + 4 + 2 = 9$ poäng)

Exekveringstiden för den seriella algoritmen i figur 3 för beräkning av en n-punkters FFT är $t_a n \log n$. I kursboken beskrivs en parallell variant av denna algoritm som kallas *Binary-Exchange*-algoritmen och lämpar sig väl för implementation på en hyperkub.

a. Beskriv kortfattat *Binary-Exchange*-algoritmen för beräkning av en n-punkters FFT på en hyperkub med p processorer (då p ej nödvändigtvis är lika med n). Använd gärna en kombination av text och figurer för att beskriva algoritmen.

b. Härled uttryck för den parallella exekveringstiden T_p, den parallella uppsnabbningen S och effektiviteten E för algoritmen i **a**, då $t_a = 1$, $t_s = 0$ och $t_w = 0.5$ (beräkningstid, uppstartningstid respektive per-ord-tid på samma sätt som i kursboken). Bestäm isoeffektivitetsfunktionerna med avseende på parallellitet, kommunikation och totalt för $E = 0.75$ (genom att bestämma isoeffektivitetsfunktionen så att E är exakt lika med 0.75, utan att förkorta några konstanter). (Tips: $\log x^y = y \log x$).

c. Upprepa isoeffektivitetsberäkningarna för $E = 0.6$. Kommentera resultatet.

```
procedure ITERATIVE_FFT(X, Y, n)
begin
    r := log n;
    for i := 0 to n − 1 do R[i] := X[i];
    for m := 0 to r − 1 do          /* Outer loop */
        begin
            for i := 0 to n − 1 do S[i] := R[i];
            for i := 0 to n − 1 do  /* Inner loop */
                begin

/* Let (b_0 b_1 ··· b_{r−1}) be the binary representation of i */

                    j := (b_0 ... b_{m−1} 0 b_{m+1} ··· b_{r−1});
                    k := (b_0 ... b_{m−1} 1 b_{m+1} ··· b_{r−1});
                    R[i] := S[j] + S[k] × ω^(b_m b_{m−1} ··· b_0 0 ··· 0);
            endfor;   /* Inner loop */
    endfor;           /* Outer loop */
    for i := 0 to n − 1 do Y[i] := R[i];
end ITERATIVE_FFT
```

Program 10.2 The Cooley-Tukey algorithm for one-dimensional, unordered, radix-2 FFT. Here $\omega = e^{2\pi\sqrt{-1}/n}$.

Figure 3: Seriell algoritm för beräkning av n-punkters FFT.