UMEÅ UNIVERSITET
Institutionen för datavetenskap
Erik Elmroth och Bo Kågström

Parallella beräkningar 2 – ht95
Tentamen
11 januari 1996

Tentamen på kursen Parallella Beräkningar II

torsdagen den 11 januari 1996
kl 9 - 15 i skrivsal 1

Tentamen kan maximalt ge 52 poäng. För betyget godkänd krävs 26 och för betyget väl godkänd 39 poäng. Poängtal anges inom parentes för varje uppgift. Uppgifterna är ej placerade i svårighetsordning.
Hjälpmedel:

- Miniräknare

Skrivsalen får lämnas tidigast en halvtimme efter att skrivningen påbörjats.

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 (2p + 2p + 2p + 2p + 1p = 9p)
Konstruera algoritmer för att utföra alla-till-alla utsändning (all-to-all broadcast) för *store-and-forward-nätverken* i uppgift **a** - **c**. (Varje processor skall skicka ett meddelanden av storlek m till alla andra processorer.) Beräkna även kommunikationstiden för varje nätverk.

a. En ring.

b. Ett 2-dimensionellt nät.

c. En hyperkub.

d. Antag att varje processor P_i äger ett tal x_i och att alla processorer behöver veta den totala summan $x_0 + x_1 + \ldots + x_{p-1}$. (Den kommunikationsoperation som krävs för detta kallas reduktion.) Beskriv hur algoritmen i uppgift **c** kan modifieras för att beräkna denna summa på en hyperkub. Ange den totala tiden som krävs för denna operation.

e. Visa hur algoritmen i uppgift **d** kan modifieras för att billigt utföra global synkronisering på en hyperkub.

Uppgift 2 (3p + 3p = 6p)
Låt W vara arbetsmängden då ett problem löses på en processor och låt $T_p(W_p, p)$ vara tiden då problemet av storlek W_p löses på p processorer. (Antag att en beräkningsoperation tar en tidsenhet, dvs problemstorleken W_p tar tiden W_p att lösa på en processor.) I följande uppgifter skall du undersöka relationen mellan isoeffektivitetsfunktionen och uppförandet för olika varianter av *skalad* uppsnabbning:

$$S = \frac{W_p}{T_p(W_p, p)}.$$

a. Antag att arbetsmängden per processor hålls konstant då p varieras, dvs $W_p = p \cdot W$. Vad krävs för att S skall växa linjärt med antalet processorer? Redogör för detta utifrån algoritmens isoeffektivitetsfunktion.

b. Man kan naturligtvis tänka sig en skalad uppsnabbning där problemstorleken skalas på annat sätt än ovan. Visa hur problemstorleken skall skalas i uttrycket för S för att den skalade uppsnabbningen skall växa linjärt med antalet processorer.

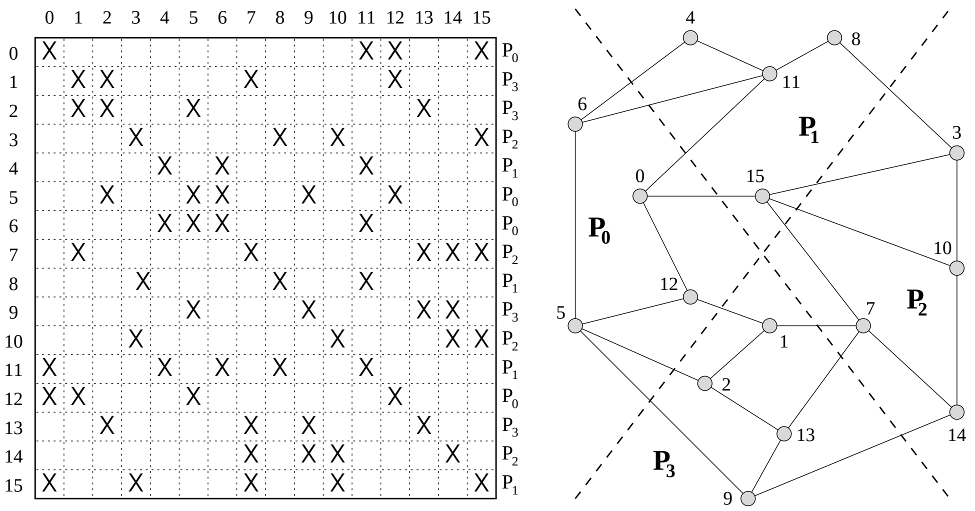
Uppgift 3 (5p + 3p = 8p)
Betrakta en parallell matris-vektor-multiplikation $y = Ax$ baserad på en matrispartitionering av en ostrukturerad gles matris A som i exemplet i figur 1. Antag att

- Matrisen A är $n \times n$ med i genomsnitt m nollskilda element per rad.
- Om en partition innehåller q stycken noder så ligger i genomsnitt $\sqrt{q}$ av dessa längs en partitionsgräns (dvs de har minst en grann-nod i en annan partition).
- Varje partition gränsar till högst c stycken andra partitioner, där c är en liten konstant.
- Partitioner som gränsar till varandra ägs av processorer som är fysiska grannar i nätverket.

- En processor som äger nod i i grafen äger initialt element x_i och skall efter beräkningarna äga element y_i.

a. Ge en beskrivning på hög nivå av en effektiv parallell algoritm för matris-vektor-multiplikationen $y = Ax$ som utnyttjar denna partitionering. (Du kan naturligtvis inte tala om exakt vilka processorer som skall kommunicera med varandra eftersom du inte känner till vilka beroenden som finns i den aktuella matrisen, så algoritmen måste beskrivas på en högre nivå än så.) Härled ett uttryck för exekveringstiden för algoritmen. Var noga med att förklara vilka operationer i algoritmen som ger upphov till de olika delarna av T_p.

b. Är algoritmen skalbar? Härled i så fall dess isoeffektivitetsfunktion.



(a) A 16 x 16 symmetric random sparse matrix (b) The associated graph and its four partitions

Figure 11.12 A 16×16 unstructured sparse matrix with symmetric structure and its associated graph partitioned among four processors.

Figur 1: En symmetrisk matris, dess grafrepresentation samt dess partitionering på fyra processorer.

Uppgift 4 (6p)
Förklara och relatera begreppet datalokalitet (*data locality*) till moderna högpresterande datorsystems minneshierarki. Förklara begreppet grad av parallellitet (*degree of concurrency*). Beskriv i stora drag hur en parallell algoritms prestanda beror på båda dessa begrepp. Illustera gärna med något exempel.

Uppgift 5 (4p + 5p + 4p = 13p)
I denna uppgift ska vi betrakta parallelldatorsystem med ett gemensamt minne (*shared adress space*). En PRAM-modell (*parallel random access machine*) är en modell av ett parallelldatorsystem där alla processorer exekverar samma instruktioner styrda av en central

kontrollenhet (i SIMD-stil). Tiden för att läsa, skriva eller utföra en beräkningsoperation är konstant. I underklassen CREW-PRAM (*concurrent-read, exclusive-write* PRAM) kan alla processorer samtidigt läsa samma minnesposition men två processorer kan ej skriva till samma minnesposition samtidigt.

a. Beskriv en algoritm som multiplicerar två $n \times n$ matriser A och B på en CREW-PRAM så att $C = A \cdot B$.

Ledning: Organisera n^2 processorer i ett logiskt 2-dim nät av dimension $n \times n$. Processor P_{ij} beräknar (i, j)-elementet av C.

b. Betrakta ett parallelldatorsystem med gemensamt minne med n^2 processorer. Antag att varje processor har ett lokalt minne, och att A_{ij} och B_{ij} finns lagrade i processor P_{ij}s lokala minne. Vidare ska P_{ij} beräkna C_{ij} och lagra elementet i det lokala minnet. Antag att det tar t_{nonlocal} tid att utföra en läs- eller skrivoperation i det gemensamma (icke-lokala) minnet, och t_{local} tid för motsvarande operationer i det lokala minnet. Förklara vad som normalt gäller i praktiken för kvoten $t_{\text{nonlocal}}/t_{\text{local}}$. Härled ett uttryck för den parallella exekveringstiden av algoritmen i b-uppgiften på denna parallelldator, innefattande både kostnader för aritmetik och kommunikation. Är den parallella algoritmen kostnadsoptimal (*cost optimal*)?

c. Modifiera algoritmen och analysen i b-uppgiften givet att parallelldatorsystemet endast har $p < n^2$ processorer. Hur mycket data måste processor P_{ij}s lokala minne minst rymma?

Uppgift 6 (7p + 3p = 10p)

Givet en sekvens $A =< a_1, a_2, \ldots, a_n >$ ges en delsekvens genom att ta bort några element ur A. (M.a.o. behöver delsekvensens element inte vara konsekutiva element från A, men de måste vara i samma ordning som i A.) Exempelvis är $< b, c, e >$ en delsekvens av $< a, b, c, d, e, f >$ medan $< b, c, g >$ och $< c, b, e >$ inte är det. Den längsta gemensamma delsekvensen av två sekvenser $A =< a_1, a_2, \ldots, a_n >$ och $B =< b_1, b_2, \ldots, b_m >$ är den längsta sekvens som är delsekvens av både A och B.

Låt $F[i, j]$ vara längden av den längsta gemensamma delsekvensen av $< a_1, a_2, \ldots, a_i >$ och $< b_1, b_2, \ldots, b_j >$. Då är $F[n, m]$ längden av den längsta delsekvensen av A och B. $F[n, m]$ kan härledas från

$$F[i,j] = \begin{cases} 0 & \text{om } i = 0 \text{ eller } j = 0 \\ F[i-1, j-1] + 1 & \text{om } i, j > 0 \text{ och } a_i = b_j \\ \max\{F[i, j-1], F[i-1, j]\} & \text{om } i, j > 0 \text{ och } a_i \neq b_j \end{cases}$$

a. Konstruera en effektiv parallell algoritm för att beräkna $F[n, m]$ på ett två-dimensionellt nät med p processorer där $m, n > p$. (För att undvika krångel med specialfall kan du själv sätta och ange förutsättningar på relationer mellan p, m och n.)

b. Härled uttryck för totala exekveringstiden T_p, uppsnabbningen S och effektiviteten E för algoritmen. Är algoritmen kostnadsoptimal?