Parallel kompleksitetsteori og BSPlab m.m.

gjesteforelesning i fag 75330
Innføring i bruk av superdatamaskiner

- **Pensum:**
  - ***Evaluating Parallel Algorithms: Theoretical and Practical Aspects***, Lasse Natvig, dr.ing.avhandling NTH 1991:2 (ISBN 82-7119-253-1)
    - » Side 1-3, 13-47.
    - » noe allerede dekket (f.eks. Amdahls lov)
  - ***Computational Models for Parallel Computing and BSPlab,*** Lasse Natvig, Technical Report no. 2/98, 19 january 1998, 16 pages Department of Computer and Information Science (IDI), NTNU (kursorisk - dvs. støttelitteratur)
- **Innhold**
  - Innledning
    - » Motivasjon / Bakgrunn
  - Intro. til kompleksitetsteori
    - » O(n), P, NP, NPC m.m.
  - Modeller for parallelle beregninger
    - » generelle egenskaper
    - » ønskete egenskaper
  - Parallell kompleksitetsteori
    - » Nicks Class (NC)
    - » Iboende serielle problemer
  - Valiant's brobyggende BSP-modell
  - BSPlab

Motivasjon

- **Gapet mellom teori og praksis**
  - *"There are two main communities of parallel algorithm designers: the theoreticians and the practitioners. **Theoreticians** have been developing algorithms with narrow theory and little practical importance; in contrast, **practitioners** have been developing algorithms with little theory and narrow practical importance".* Clyde P. Kruskal in *Efficient Parallel Algorithms: Theory and Practice*, [Krus89].
- **(Massivt) parallelle maskiner tar over markedet som vektormaskiner hadde**
  - fig. 1.1 i [Natv91] (Hvor fort går det ?)
  - NTNUs CRAY er nå en parallell-maskin
- **Datamaskiners ytelse (regnekraft og lagerplass) øker ==> vi kan løse større problemer**
- **Større problemer øker relevansen til asymptotisk analyse og kompleksitetsteori**
- **Bruk av et stort antall prosessorer blir mer vanlig ==> algoritmer fra teoretisk databehandling (Eng. TCS) (f.eks. PRAM algoritmer) blir mere relevant**

Kompleksitetsteori

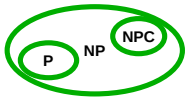
- **En kort og uformell introduksjon**
- **Beregningsmessig kompleksitet (Eng. computational complexity) vs. "beskrivelsesmessig kompleksitet (Eng. descriptional complexity)**
- **Klassifisering av problemer i kompleksitetsklasser**
- **Om behovet:**
  - *"The idea met with much resistance. People argued that faster computers would remove the need for asymptotic efficiency. Just the opposite is true, however, since as computers become faster, the size of the attempted problems becomes larger, thereby making asymptotic efficiency even more important."* John E. Hopcroft in his Turing Award Lecture *Computer Science: The Emergence of a Discipline* [Hopc87].

Kompleksitetsteori

- **Sentral bok:** Garey, Michael R. and Johnson, David S., **COMPUTERS AND INTRACTABILITY, A Guide to the Theory of NP-Completeness**, W. H. Freeman and Co., New York, 1979.
- **Fundamental kompleksitetsteori**
  - problem, problem-instans, (løsnings)algoritme
  - kompleksitetsfunksjoner
    - » tidskompleksitet
    - » plasskompleksitet
  - worst-case / average case / best case
  - ordensnotasjon / asymptotisk kompleksitet
  - øvre/nedre grense (Eng. 'upper/lower bound')
  - deterministiske og ikke-determinstiske algoritmer
  - polynomisk og eksponensiell kompleksitet
  - NP-komplette problemer
  - Parallel kompleksitetsteori
    - » polylogaritmisk tid og Nicks klasse (NC)
    - » P-komplette problemer (iboende serielle)
  - mange, mange andre kompleksitetsklasser

Viktige kompleksitetsklasser

Tradisjonell (seriell) kompleksitetsteori



P = klassen av problemer som kan løses i polynomisk tid vhja. en deterministisk algoritme

NP = klassen av problemer som kan løses av en ikke-deterministisk algoritme i polynomisk tid (bare deterministisk i eksponensiell tid)

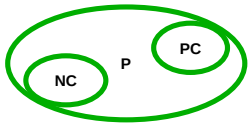
NPC = NP-komplette problemer, klassen av problemer i NP som er vanskeligst å løse i polynomisk tid

"NP-hard" problemer

NP-komplette problemer

- **Eksempel, ryggsekkproblemet**
  - Uformell definisjon:
    - » en mengde ting med kjent størrelse (volum), og en stor sekk av kjent størrelse
    - » hver ting har en verdi
    - » er det plass til et utvalg av tingene som har en samlet verdi større enn K?
- **Eksponensiell parallellitet**
  - NP komplette problemer kan løses i polynomisk tid hvis vi tillater bruk av et eksponensielt antall prosessorer
  - Eks. Ryggsekkproblemet
- **Den praktiske nytten av NP-kompletthet**

Parallell kompleksitetsteori
--- oversikt ---



P = klassen av problemer som kan løses deterministisk i polynomisk tid

NC = Nicks Class, problemer som kan løses i poly-logaritmisk tid ("ultraraskt") vha. et polynomisk ("realistisk") # prosessorer

PC = "de vanskeligste problemene i P m.h.p. "ultrarask" løsning med "realistisk" # prosessorer ("P-komplette")

Nick's Class

- **Nick's Class *NC* is the class of problems that can be solved in polylogarithmic time (*i.e.* time complexity $O(\log^k n)$ where k is a constant) with a polynomial number of processors (*i.e.* bounded by $O(f(n))$ for some polynomial function f, where n is the problem size).**
- ***NC* er robust**
- ***NC^1* og *NC^2***
- **Hva er ``rimelig" grad av parallellitet?**
- **Har teoretikerne fokusert for mye på *NC*?**
- **Eksempler**

P-komplette problemer

- **også kalt "iboende serielle problemer"**
- ***PC* is the class of *P*-complete problems. A problem X is *P*-complete if (a) X is member of *P, and* (b) For any other problem Y in *P* there exists a *NC*-reduction from Y to X.**
- **Litt om bevis av P-kompletthet**
  - ett første P-komplett problem
  - NC-reduksjon;
    A *NC*-reduction from A to B (written as "$A \propto_{\em NC} B$)"(Latex) is a (deterministic) polylogarithmic time bounded algorithm with polynomially bounded processor requirement for transforming (or reducing) any instance IA of A to a corresponding instance IB of B so that the solution of IB gives the required answer for IA.
  - Bevisførsel i praksis:
    If A is *P*-complete, B is member of P, and "$A \propto_{\em NC} B$"then B is *P*-complete.
  - CVP-problemet

Mer om P-komplette problemer

- **Eksempler**
  - Maximum nettverk flyt
  - Lineær programmering
  - dybde først søk
  - Unifisering
  - Spill med to spillere

http://www.cs.ualberta.ca/~hoover/P-complete/

- *Limits to Parallel Computation: P-Completeness Theory*, av Raymond Greenlaw, H. James Hoover og Walter L. Ruzzo [GHR95]
- Innholdsfortegnelse, forord, kap.-1 (Introduksjon), liste av problemer m.m.

P-komplette algoritmer

- **Noen algoritmer er mindre egnet for hurtig parallellisering**
- **Grådighets-algoritmer**
- **An algorithm A for a search problem is *P-complete* if the problem of computing the solution found by A is *P*-complete.**

Taksonomier

- **Flynn**
  - mest benyttet
  - SISD, SIMD, MISD, MIMD (antas kjent)
  - generelt ansett som for grov
    - » Duncans taksonomi er et svar på det, men er lite benyttet
- **Mange mange flere finnes**

Modeller for parallelle beregninger ("abstrakte maskiner")

- **Sekvensielle modeller**
  - RAM tilsv. Von Neumann
  - Enerådende for tradisjonelle datamaskiner (uniprosessorer)
- **Parallelle modeller**
  - Et stort antall finnes
    - ofte ulike navn på samme modell
    - ofte ulik tolkning/forståelse av samme modell
      - f.eks. flere kjente forskere/firfattere hevdet at PRAM er SIMD, det er feil !
  - Kan bygge bro mellom teoretisk databehandling og praktisk databehandling
  - Ingen etablerte standard-modeller for parallelle datamaskiner (multiprosessorer)
    - » Anslagsvis 100 ulike er foreslått og brukt
    - » Eksempler;
      - PRAM (Parallel Random Access Machine)
      - BSP (Bulk Synchronous Parallel Machine)
- **ABSTRAKSJON:**
  - Husk det essensielle, glem detaljer ("selektiv glemming")

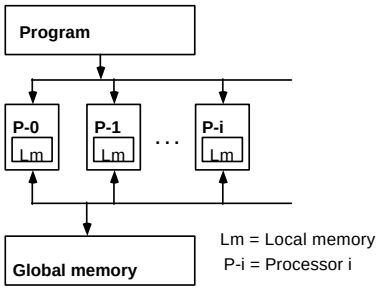
Abstrakte maskiner (Eng.: computing model)

- **Hensikt:**
  - abstraksjon forenkler
  - **økt viktighet pga. økt kompleksitet**
    - » ved konstruksjon, sammensetting (integrasjon) og bruk
  - standardisering for algoritmeanalyse
  - felles platform / forståelse
    - » samarbeid
    - » flyttbarhet
    - » utveksling av resultater
    - » undervisning
  - vurdering av ytelse
    - » analyse
    - » prediksjon
      - f.eks. detaljerte modeller som gir ytelsen til et system som funksjon av sentrale arkitektur-parametre
      - ekstrapolering forutsetter skalerbarhet

PRAM-modellen

- **Fortune & Wyllie 1978**
  - dominerende modell for forskning og beskrivelse av parallelle algoritmer
  - generelt kritisert som for urealistisk
- **Arkitektur**
  - (Uendelig mange) synkrone prosessorer med hver sin program-teller (dvs. synkron MIMD)
  - Et (uendelig) stort globalt lager
    - » uniform og "billig"aksess
  - (små lokale lager)
- **Varianter;**
  - eksklusiv/samtidig lesing/skriving gir 4 varianter
  - EREW PRAM, den svakeste
  - **CREW PRAM**, den mest brukte (den sterkeste veldefinerte)
  - ERCW PRAM
  - CRCW PRAM, den sterkeste, finnes i en rekke undervarianter
    - » Common / Arbitrary / Minimum / Sum / Priority / Garbage

The PRAM Model



- **Fortune and Wyllie 1978**
- **Synchronous processors**
- **Shared memory with ?R?W-access**
- **Unit cost instructions (also glob.mem.access)**
- **one program, BUT MIMD**

Egenskaper ved gode modeller "Properties of good models"

1) easy to understand
2) easy to use
3) well defined
4) general
5) performance representative
6) realistic
7) durable
8) mathematically convenient

Model tradeoffs ?

- **These are conflicting requirements:**
  - Easy programming vs. efficient execution
  - General purpose model vs. special purpose model
  - Easy to use vs. mathematical convenience
  - Easy to analyse vs. easy to build

BSP programmering

- **SPMD**
- **Parallell slakk ("overskuddsparallellitet")**
  - parallellitet i et program (# logiske prosessorer) dividert på parallellitet i maskinen (# fysiske prosessorer tilgjengelig)
- **Kommunikasjons slakk**
  - "behov for lokale data dividert på behov for globale (ikke-lokale) data"
- **Skalerbare, effektive og flyttbare parallelle algoritmer er mulig vhja. parametrisering:**

```
BSP-ALGORITHM KjempeSmart (n, p, l, g)
BEGIN
 ...
END;
```

.... desverre ikke så lett å lage slike

BSP World Wide og BSPlib

- **BSP Worldwide**
  - forening for BSP interesserte
  - koordinering av forskning
  - ca. 140 medlemmer
  - **forslag til standard bibliotek for BSP-programmering; BSPlib**
- **BSPlib funksjonalitet:**
  - Starte og stoppe program, allokering av prosessorer (4)
  - Forespørsler (3)
    - » # prosessorer tilgjengelig, # i bruk, prosess-identifikasjon (nr), tid
  - Synkronisering (1)
  - "Direct Remote Memory Access"(DRMA) (4)
    - » metode for å overføre data mellom prosessorer (registrering, skriv, les)
  - "Bulk Synchronous Message Passing"(BSMP) (5)
    - » alternativ metode
  - Høy-ytelse kommunikasjon (3)
    - » pekerutveksling istf. kopiering m.m.

\+ høyere-nivå funksjoner (f.eks. kringkaste)

"Parameters.dat" (eksempel / utdrag)

```
////////////// General parameters
// Network, NOW, Simple, Null or Bus
[BSPMachine] = Null
...
//Print a dot after each super step?
[ProgressIndicator] = Off
...
//The simulator machine CPU speed compared with
   a 100 MHz 486
// Used to set the overhead sizes
[BytesInGetMessage] = 8
[BSMPMessageMarkSize] = 2
...
[NumberOfProcessors] = 10
...
/////// Parameters for the network machine
//Network topology:
//TwoDTorus, TwoDMesh, TwoDMeshHP, ThreeDMesh,
   (HyperCube)
[Network_Topology] = TwoDMeshHP
...
//Network size
//Used for TwoDTorus, TwoDMesh, ThreeDMesh
[Network_Xsize] = 4
[Network_Ysize] = 4
...
```


Videre arbeid (Superdat & Par.alg))

- **Algoritmeutprøving på BSPlab**
  - Er de (teoretisk) effektive BSP-algoritmene også effektive i praksis?
  - Hvor mye er skjult i kompleksitetskonstanter?
  - Tungregning
  - **Faget superdatamaskiner**
  - **CRAY T3E**
- **BSPlab som PRAM simulator**
  - parallelle algoritmer / IMF

Mere om BSP og BSPlab

- **BSP Worldwide**
  - www.bsp-worldwide.org/bspwwact.htm
- **Ravi Palepus BSP-pekere**
  - www.scs.carleton.ca/~palepu/BSP.html
  - Oxford
  - Harvard
  - m.fl
- **BSPlab**
  - www.idi.ntnu.no/bsplab

Referanser

[GHR95] *Limits to Parallel Computation: P-Completeness Theory*, av Raymond Greenlaw, H. James Hoover og Walter L. Ruzzo [GHR95], Oxford University Press, 1995, ISBN 0-19-508591-4 Se også http://www.cs.ualberta.ca/~hoover/P-complete/

[GJ79] Garey, Michael R. and Johnson, David S., *COMPUTERS AND INTRACTABILITY, A Guide to the Theory of NP-Completeness*, W. H. Freeman and Co., New York, 1979.

[Hopc87] Hopcroft, John E., *Computer Science: The Emergence of a Discipline,* Communications of the ACM, vol. 30, no 3, mar, 1987, pages198-202, Turing Award Lecture.

[Krus89] Kruskal, Clyde P.*Efficient Parallel Algorithms: Theory and Practice*, in Proceedings of the NSF - ARC Workshop on Opportunities and Constraints of Parallel Computing, side 77-79, 1989.

[Natv90] Natvig, Lasse*Logarithmic Time Cost Optimal Parallel Sorting is Not Yet Fast in Practice!*, in Proceedings of SUPERCOMPUTING'90, New York, November 1990, pages 486-494

[Natv91] *Evaluating Parallel Algorithms: Theoretical and Practical Aspects* , Lasse Natvig 1990, dr.ing.avhandling NTH 1991:2, ISBN 82-7119-253-1,