Hennessy & Patterson [HP90]
Kap. 10: Future Directions

- **John Hennessy & David Patterson; *Computer Architecture, A Quantitative Approach,* [HP90].**
- **Kap. 10: Future Directions, Innhold**
  - "Speed-of-light-argument"
  - SIMD og MIMD maskiner
  - MIMD med sentralisert el. distribuert lager
  - Framtidens kompilatorer
  - Eks. Sequent Symmetry multiprosessor
  - Evolusjons-revolusjons-spekter

SIMD-maskiner vs. MIMD-maskiner

- **SIMD-maskiner (repetisjon)**
  - Sentralisering av kontrollenhet reduserer kostnaden ved en generell multiprosessor
  - Lagrer programmet bare et sted, dvs. billigere enn [vanlig] MIMD
  - Sentral avveining: Få kraftige eller mange "svake" prosessorer
- **MIMD-maskiner**
  - Høy maksimal ytelse, høy pålitelighet og tilgjengelighet gjennom redundans
  - Man har lenge søkt etter bedre arkitekturer
  - Delt lager eller meldingsutveksling
  - Akseptert i dag som mer effektive enn SISD for tidsdeling
  - Få applikasjoner kjører raskere på MIMD enn SISD !?

Hvorfor så få effektive MIMD applikasjoner?

- **Parallellitet -> koordinering -> kommunikasjon -> overhead**
- **Programmereren har til nå måttet vite mye om HW**
  - vanskeligere programmering
  - mindre portabel kode
  - *... also makes portable parallel programs rare. Though* this may lessen over time, it is now the biggest challenge facing computer science
- **Eksplisitt meldingsutveksling gir vanskeligere programmering**
- **Amdahls lov; seriell del av algoritme må gjøres minst mulig**
- **Vanskeligere for stort antall prosessorer**
- **AAA: vi må forstå applikasjoner, algoritmer og arkitektur**
- **Veier mot "El Dorado", Fig. 10.1**
  - mot høyre lettest for programmerere
  - oppover lettest for maskinkonstruktører

Delt lager eller distribuert lager? / Opprydding

- **Opprydding:**
  1) sentralisert lager el. distribuert lager?
  2) delt adresserom el. flere private adresserom?
- **Disse to kriteriene er ortogonale, dvs. 4 muligheter finnes**
- **Debatter hver av de to spørsmålene, ikke sammenblandet**
- **Delt adresserom**
  - har implisitt kommunikasjon , ved vanlig lageraksess
  - noen slike maskiner har også eksplisitt kommunikasjon
    - ulike former, f.eks. SEND/RECEIVE
- **Private adresserom**
  - må ha eksplisitt kommunikasjon
- **Fig. 10.2**
- **Distribuert lager begrenser ikke antall prosessorer, dvs. mer skalerbart**
- **Sentralisert lager gir enklere programmering**

Spesial prosessorer (koprosessorer)

- **Økt utbredelse p.g.a.**
  - bedre DAK-verktøy
  - økt kapasitet pr. krets
- **Digitale signal prosessorer (DSP)**
- **Systoliske array**
  - trend mot færre, kraftigere og programmerbare celler (PE´er)
- **Konsument markedet**

Framtidens kompilatorer for framtidens maskiner

- **Plassering og kopiering av data for hurtigere aksess og mindre kommunikasjon**
  - Prosessor-lager-"gapet" vokser
  - Plasser kode og data -> større treffrate i hurtigbuffer
  - Bedre "prefetching"
  - Bedre høynivå-språk, f.eks. array som egen datatype, f.eks. FORTRAN-90 (8X)
    - kompilator kan fordele data effektivt
    - lettere enn SISD
      - slipper å programmere løkker for å behandle tabeller
- **Utnyttelse av parallellitet**
  - Inne i prosessor, f.eks. superskalare prosessorer
  - Vektorisering
- **Vanskelig område**
  - Har holdt på siden 1975
  - Spesielt "dusty deck"

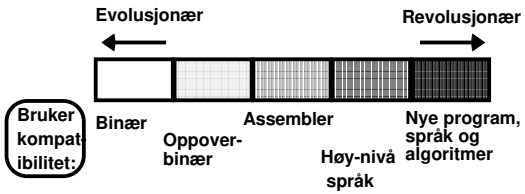
Eksempel; Sequent Symmetry Multiprosessor

- **Fornyet interesse for multiprosessorer på 80-tallet**
  - mikroprosessorer på felles buss
  - flyttbare OS
- **Sequent corporation**
  - 1984 Balance 8000, 2-12 NS 32032, cache coherence, WT
  - 1986 Balance 2100, opp til 30 prosessorer
- **1986->**
  - Symmetry, 80386 prosessorer som er 3-400 % raskere
  - 8 x større hurtigbuffer
  - fra tversigjennomskriving (WT) til utsatt tilbakeskriving (WB)
  - dobbelt buss-bredde
  - dobbelt antall lager-banker
  - Fig. 10.5
- **WB gir lavere buss-belastning enn WT**
  - Fig. 10.6

Vanlige feil og misforståelser

- **Målt eksekveringstid = "*de eneste helt sikre mål*"**
- **Ekstrapolering av lineær "speedup" kan være galt el. misvisende**
- **Observert lineær "speedup"**
  - viser god skalerbarhet men ikke nødvendigvis god ytelse
  - sammenlikning av eksekveringstid
  - sammenlikne de beste algoritmer på hver maskin
- **Si hva du gjør !!!**
  - relativ hastighetsøkning (samme program)
  - sann hastighetsøkning (de beste program)
- **Amdahls lov gjelder fortsatt ! men, skalering av problemet vil ofte føre til at andel av seriell kode synker**
  - Jfr. kap. 3 i [Hwa93]

Evolusjon - revolusjon - spekteret i datamaskinarkitektur



- **Brukerkompatibilitet**
  - påvirker hvor realistisk/kostbar en innovasjon er
  - **evolusjonær** vil si i stor grad usynlig for bruker
  - **revolusjonær** vil si i stor grad synlig for bruker
  - **Innovasjoner kan klassifiseres etter hvor lavt nivå en har brukerkompatibilitet på**

- **Tidsdelt MIMD**
  - kjører mange uavh. program samtidig
- **"Latency-MIMD"**
  - kjører en jobb hurtigere

Historiske perspektiver, referanser

- **Parallellitet omtalt siden Menabrea 1842 !**
- **Illiac IV**
  - meget mislykket som maskin -> forsinket senere SIMD prosjekter
  - men, vellykket som forskning
  - *"... the sin is not in having a novel architecture that is not a commercial success; the sin is in not quantitatively evaluating the strengths and weaknesses of the novel ideas"*
- **Enorme mengder informasjon å forholde seg til**
- **Gordon Bells framtids-spådommer:**
  - 1) 1 TeraFlop "sustained" i 1995
  - 2) flere MFLOPS selges i superdatamaskiner med < 100 datastrømmer enn i superdatamaskiner med > 1000 datastrømmer.