CS475 Parallel Processing

Cost Optimality and Iso Efficiency

Wim Bohm, Colorado State University

Cost and Optimality

- Cost = $p.T_p$
  - p: number of processors
  - T_p: Time complexity for parallel execution
  - Also referred to as **processor-time product**
  - Time can take communication into account
    - Problem with mixing processing time and communication time
    - Simple but unrealistic:

operation: 1 time unit

communicate with direct neighbor: 1 time unit

- Cost optimal if Cost = $O(T_1)$

E.g. - Add *n* numbers on hypercube

- n numbers on **n** processor cube
  - Cost?, cost optimal?
  - assume 1 add = 1 time step
    1 comms = 1 time step
  - Assume the numbers are already distributed over the cube
- n numbers on **p** (<n) processor cube
  - Cost?, cost optimal? S(n)? E(n)?
  - Again, assume the numbers are already distributed over the cube

E.g. - Add *n* numbers on hypercube

- n numbers on **n** processor cube
  - Cost = O(n.log(n)), not cost optimal
- n numbers on **p** (<n) processor cube
  - $T_{\mathbf{p}}$ = n/p + 2.log(p)
  - Cost = O(n + p.log(p)),

    cost optimal if n = O(p.log(p))
  - S = n.p / (n + 2.p.log(p))
  - E = n / (n + 2.p.log(p))

E.g. - Add *n* numbers on hypercube

- n numbers on **p** (<n) processor cube
  - $T_p = n/p + 2.\log(p)$
  - Cost = $O(n + p.\log(p))$,

    cost optimal if $n = O(p.\log(p))$
  - $S = n.p / (n + 2.p.\log(p))$
  - $E = n / (n + 2.p.\log(p))$
  - Build a table: E as function of n and p
    - Rows: n = 64, 192, 512 Cols: p = 1, 4, 8, 16
    - larger n → higher E, larger p → lower E

$E = n / (n + 2.p.\log(p))$

p / n	1	4	8	16
64	1	64/(64+16) = **4/5**	64/(64+48) = 4/7	64/(64+128) = 1/3
192	1	192/(192+16)=12/13	192/(192+48) = **4/5**	192/(192+128) = 3/5
512	1	512/(512+16)=32/33	512/(512+48) = 32/35	512/(512+128) = **4/5**

Observations

- to keep E=80% when growing p, we need to grow n
- larger n → larger E
- larger p → smaller E

Scalability

- Ability to keep the efficiency fixed,
  when p is increasing, provided we also increase n
- e.g. Add n numbers on p processors (cont.)
  - Look at the (n,p) efficiency table
  - Efficiency is fixed (at 80%) with p increasing
    - only if n is increased

Quantified..

- Efficiency is fixed (at 80%) with p increasing only if n is increased
- How much?
- $E = n / (n + 2p\log p) = 4/5$

  $4(n + 2p\log p) = 5n$

  $n = 8p\log p$

  (Check with the table)

Iso-efficiency metric

- Iso-efficiency of a scalable system
  - measures degree of scalability of parallel system
  - parallel system: algorithm + topology

    + compute / communication cost model

  - Iso-efficiency of a system: the growth rate of problem size n, in terms of number of processors p, to keep efficiency fixed

    eg n = O(p logp) for adding on a hypercube

Sources of Overhead

- Communication
  - PE - PE
  - PE – memory
  - And the busy waiting associated with this
- Load imbalance
  - Synchronization causes **idle processors**
  - Program parallelism does not match machine parallelism all the time
    - Sequential components in computation
- Extra work
  - To achieve independence (avoid communication), parallel algorithms sometimes re-compute values