DRIVING FORCE?

(MSP Mtg Models Dec. 1992)

Software — Model — Hardware

Architecture-dependent software ← Message passing Paradigm (cf OCCAM) ← Distributed Memory Architectures 1980's

Functional programming (Implicit parallelism) 15 years of research → Graph reduction or data flow → Scalable parallel architecture ??

Sequential Computing is based on a model

SOFTWARE → High level languages, Architecture-independent software

VN MODEL → {Analysis of algorithms, Complexity Theory, Performance prediction

HARDWARE → {Forms for technological evaluation, Various efficient processor designs

danger of too much attention to current model

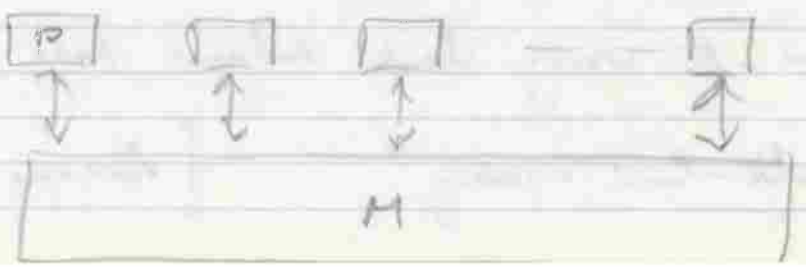
Parallel computing — mainstream? parallel software industry?

Goals for general-purpose computing

Scalable parallel performance + architecture & software independent

1. Idealised parallel computing
2. Special-purpose parallel computing
3. General-purpose ———

1. Idealised



Synchronous current read

Issues for concurrent write

common, nondeterministic, priority
value? one succeeds chosen one succeeds

PRAM 1-level memory
every memory location is equally far from a processor

Real difficulty not operation, but getting data in right place

communication or concurrent access

NC theory NC — $O(\log^k n)$ time on a poly no of processors

Is everything in P in NC. i.e. do all poly time problems have fast parallel algorithms?

P-complete problems

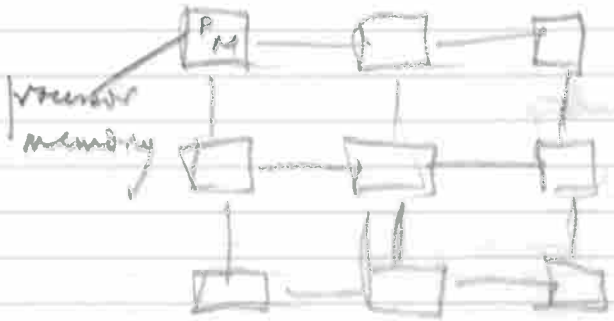
Linear programming
Circuit value problem

General methods for PRAM

Prefix computation, Tree restructuring, List ranking etc

Purely theoretical ⟶ realistic engineering issues

2. General-Purpose Parallel Computing

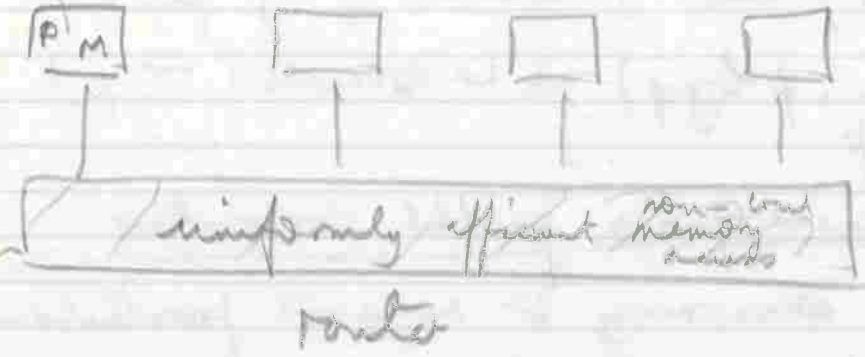


Distributed memory
Algorithmic efficiency depends on exploitation of network locality

Have to worry about how data is transported ← TRANSPUTER VLSI chip

"Multi-level memory" [threads + memory alloc regions

V. efficient



2-level memory

Virtual shared memory
Single shared address space

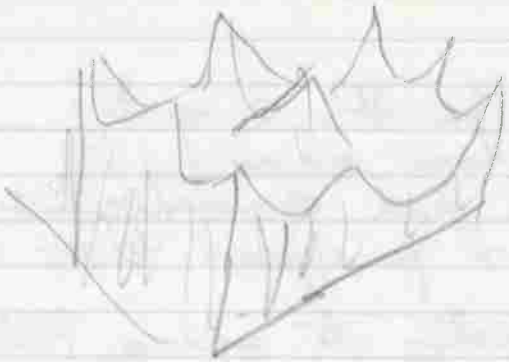
Box in a network e.g. torus wrapped mesh

Routing box must be able to realize any permutation efficiently

Randomized routing (Theory end of 70s + Practice)

Phase 1 send to intermediate node by shortest path

Phase 2 forward to destination by shortest path



randomized routing
evens out the
congestion points here.

Let h-relation denote problem where each processor sends h packets & receives h.

1-relation can be solved on n-node hypercube (CC etc.
in $O(\log n)$ steps (Valiant 1980)

⇒ PRAM algorithms with communication bandwidth can be implemented on hypercube with no overhead.
$\log n$-relation - - - (upfal 1984)

VALIANT

PRAM $(T, P) \Rightarrow$ HYPERCUBE $(\log p, p)$

PRAM $\left(\frac{T}{\log p}, p \log p\right) \Rightarrow$ HYPERCUBE (T, p) UPFAL

Key ideas for general-purpose parallel computing

Find grain of concurrency & maximize parallelism

Parallel slackness, latency tolerance, multithreading
(fast context switching)

Fast high-capacity routing networks, communication bandwidth

Uniform memory access.

Problem: retaining locality not a good thing to do!

Hash the address space to distribute memory references randomly

gets around number of threads accessing same pt.

good in theory & practice (PRAM)

BSP model

Bulk-synchronous parallel computing
- p pairs processor memory pairs
- point to point communication network
- mechanism for efficient barrier synchronization

Issue: is it cost-effective to address PRAM implementation at processor time
McColl + Valiant think not — Leiserson 'BSP.

PRAM ($g = 1$ no of instruction cycles to realize perm.
$g = 2000$ Paragon.
$g = 80, 90$?

Challenges

Architecture

Processor design to ~~support~~ v large number of threads
fast context switching, message handling, address

Improved networks

grand challenge reduce latency l & g — lots of ops for every non-local memory access

New optics — "granularity" every op. atomic

~~Fault tolerance~~

IO.

Algorithms

Algorithms & complexity theory for BSP

Effect of hashing

Sparse, irregular matrices & graphs.

Candidates for GPPC

PRAM, SIMD, dataflow, message passing,
tuple space, graph reduction,

"Maximum range of possibilities occur at outset — restriction — endless variations on a few surviving models."

BURGESS SHALE — S.J. GOULD.

parallel in computation ⟷ (fallacy) parallelism in machine

goal of ~~auto~~ parallelisation — 1 processor (magic) — opposite extreme

consider — get more than you want in parallel computation — "parallel slackness"