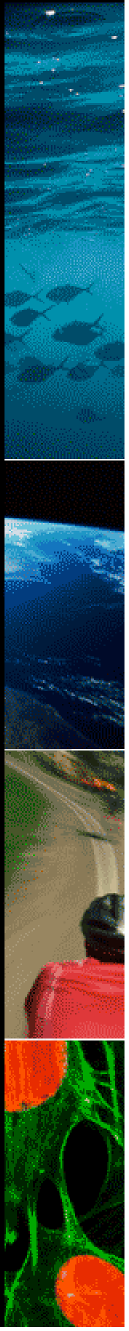
The Supercomputer Company

Eldorado

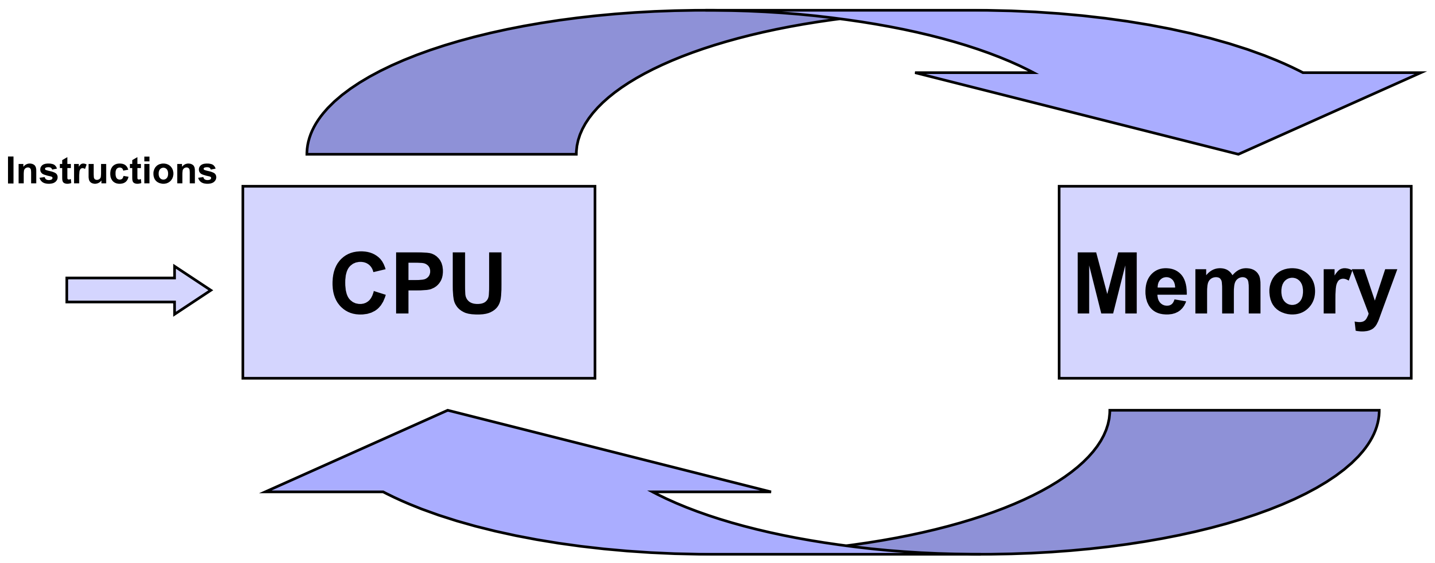
John Feo, David Harper,
Simon Kahan, Petr Konecny
Cray Inc.



Outline

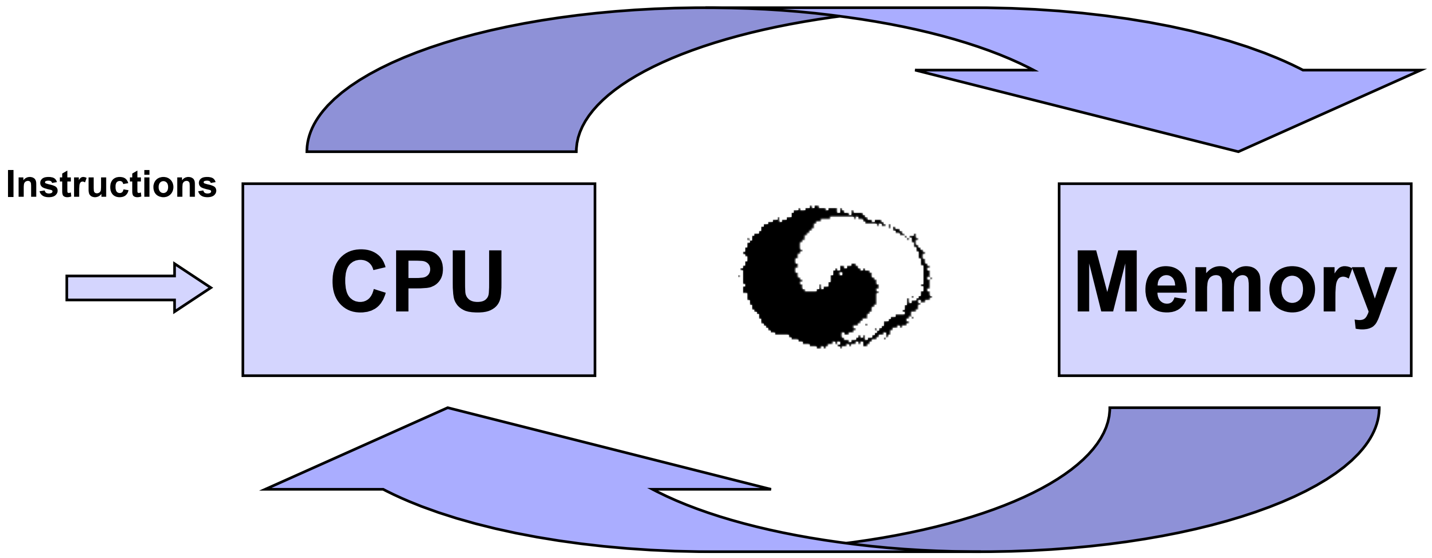
- This talk is introductory
  - Please read the paper, contact authors, for details
- High-level contextual motivation
  - Parallel programming is too hard
- Cray multithreading
  - Cray has been evolving a solution
- Eldorado
  - Cray's first **production** multithreaded computer
  - Narrowly focused with unmatched potency
- Roadmap
  - Future systems carry multithreading forward

~1970's Generic Micro-computer



Memory keeps pace with CPU.

~1970's Generic Micro-computer



Memory keeps pace with CPU.

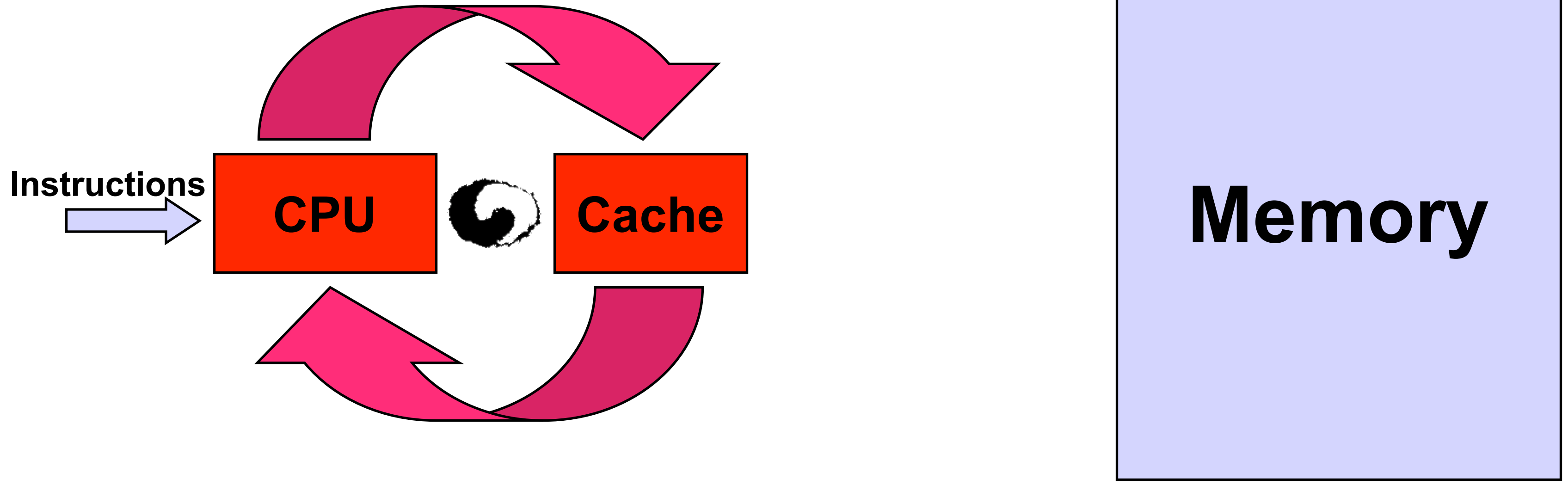
Everything was in Balance…

…but balance was short-lived: by year 2000,

- Processors had gotten much faster: over **3000**x.
- Memories hadn't: about **20**x faster while **1,000,000**x larger.
- Memory cannot keep up with processing.

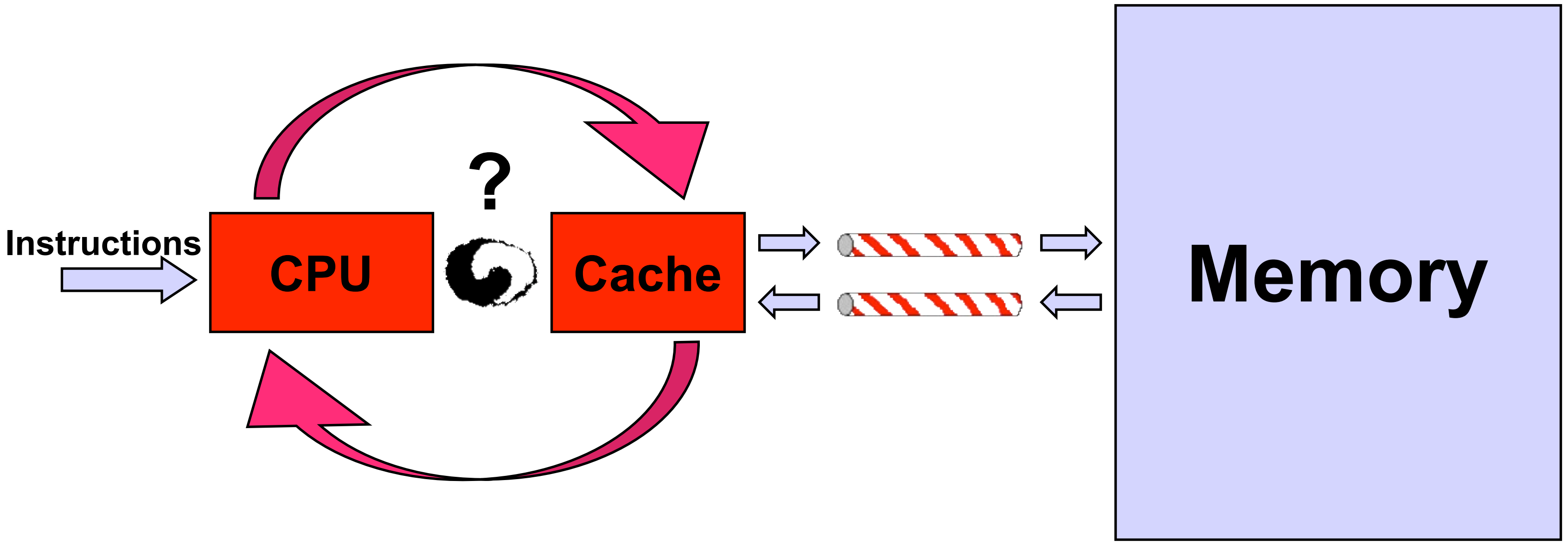
Meanwhile, a band-aid solution evolved, groping for balance as it slipped away…

Simplified Cache-based Computer



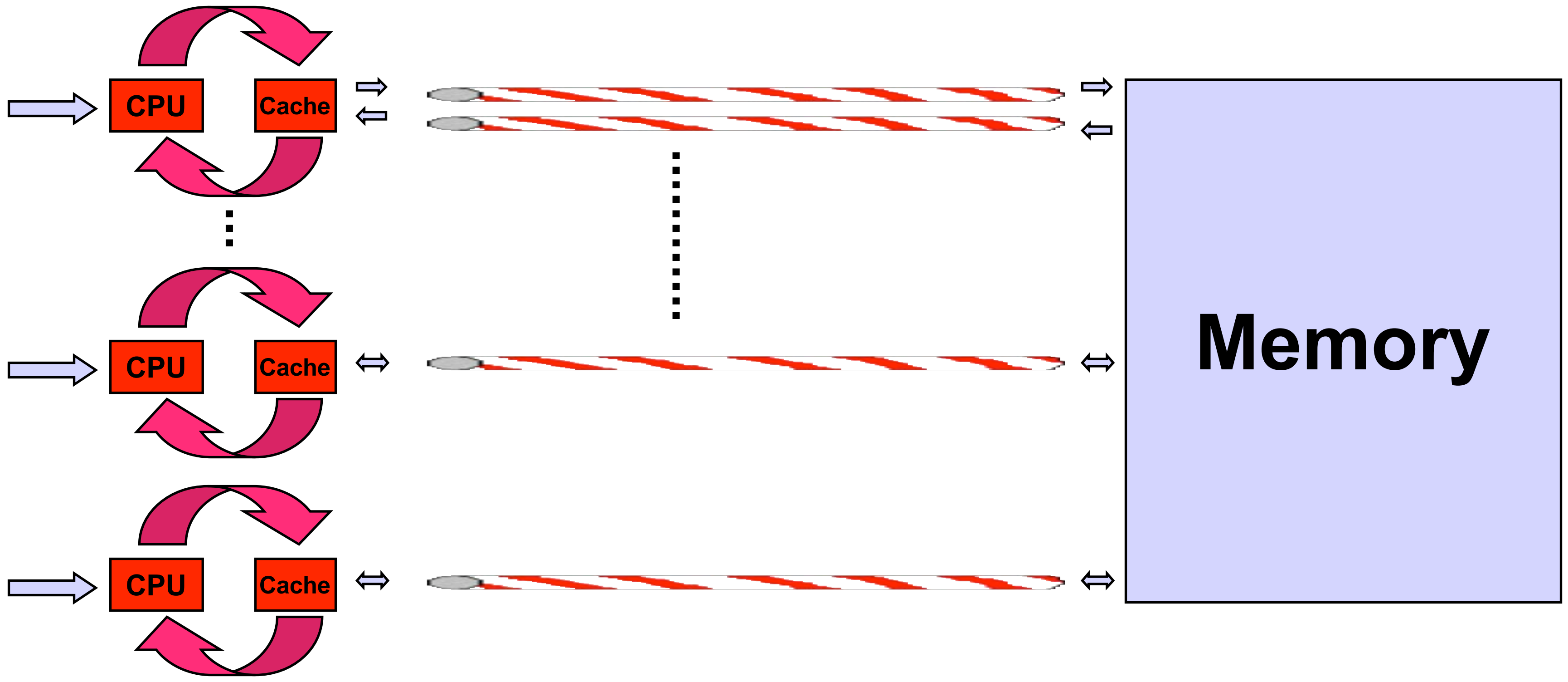
When you're hot, you're hot…

Simplified Cache-based Computer



…and when you're not,
you're sucking through a straw

20th Century Parallel Computer

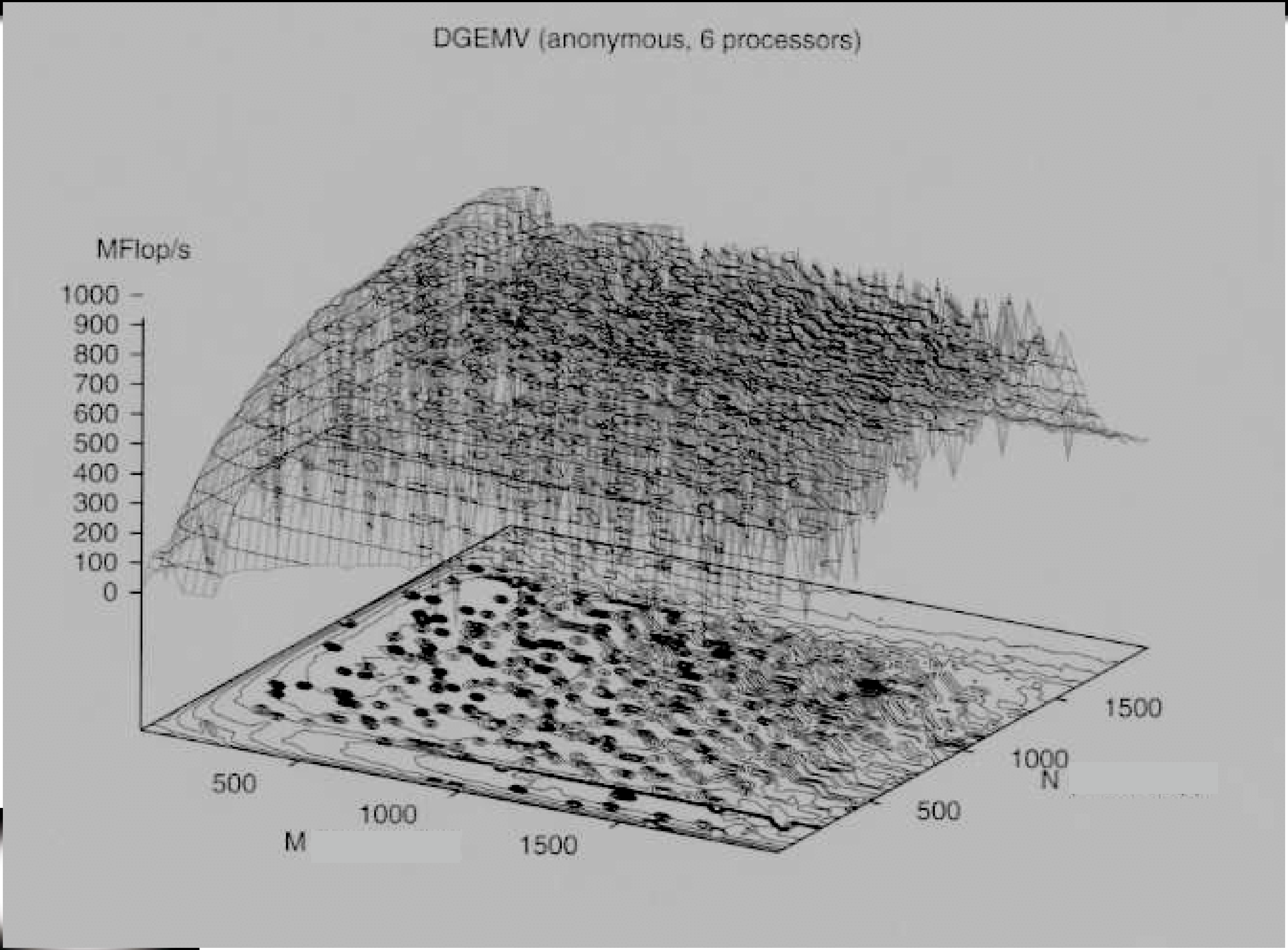


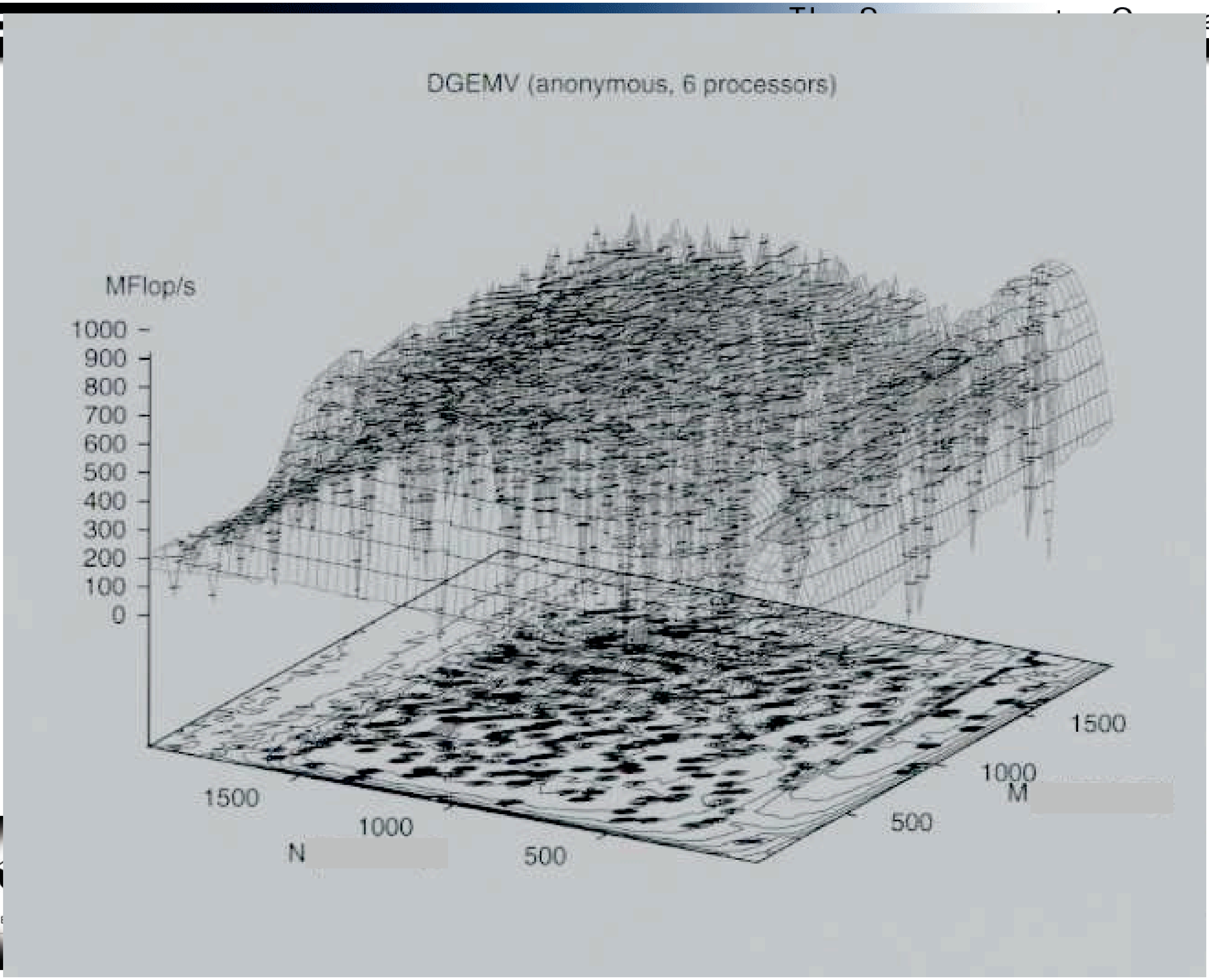
Avoid the straws and there are no flaws.

(And good luck to you.)

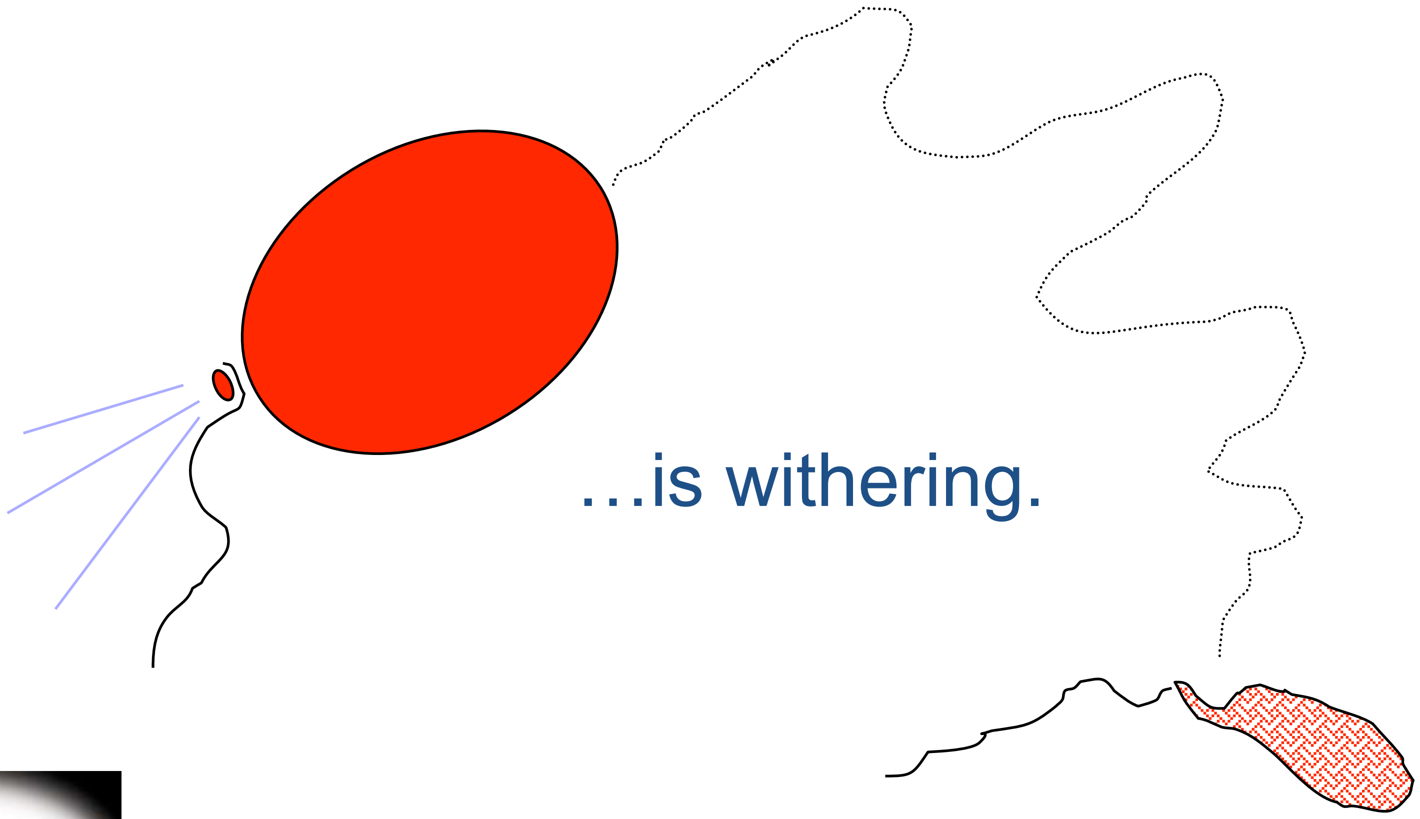
Programming 101: Rules for Performance Programming

- **Avoid modifying shared data**
- **Access data in address order**
- **Avoid indirection, linked data-structures**
- **Partition into independent computations**
- **Avoid synchronization**
- **Place data near processor doing computation**
- **Avoid conditional branching**
- **Do not expect predictable performance!**


DGEMV (anonymous, 6 processors)
MFlop/s
1000
900
800
700
600
500
400
300
200
100
0
500
1000
1500
M
500
1000
N
1500

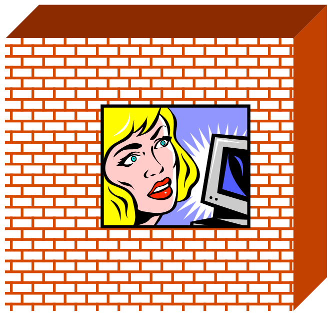

DGEMV (anonymous, 6 processors)
MFlop/s
1000
900
800
700
600
500
400
300
200
100
0
1500
1000
500
N
1500
1000
500
M

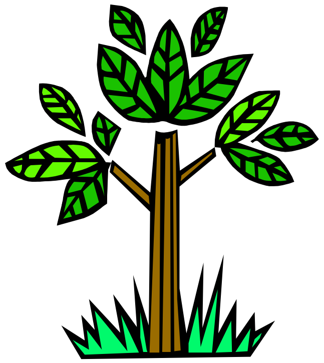
The space of feasible scalable solutions…



…is withering.

Programmers boxed-in by performance constraints…

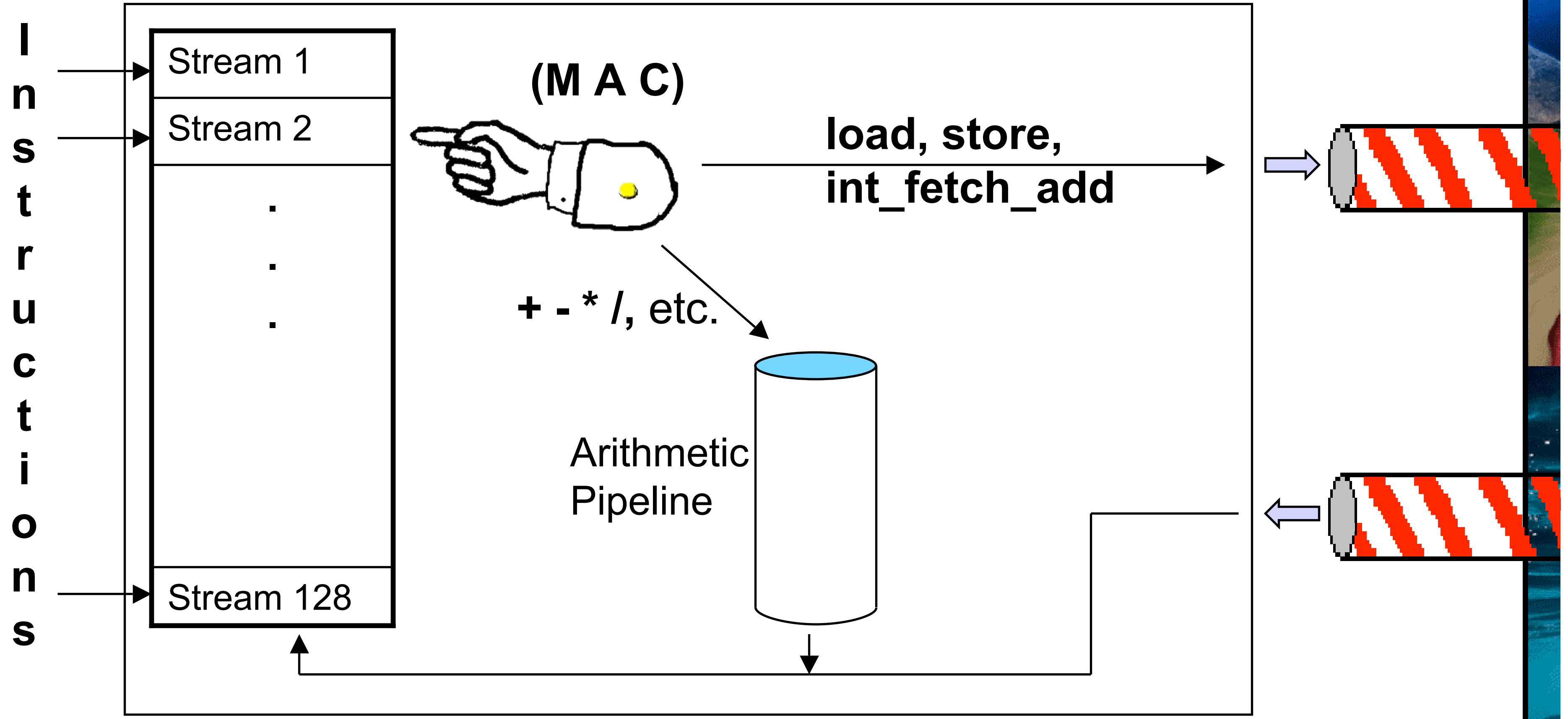




Computer science offers abundance of algorithmic methods;
conventional computing technology renders many of them useless.

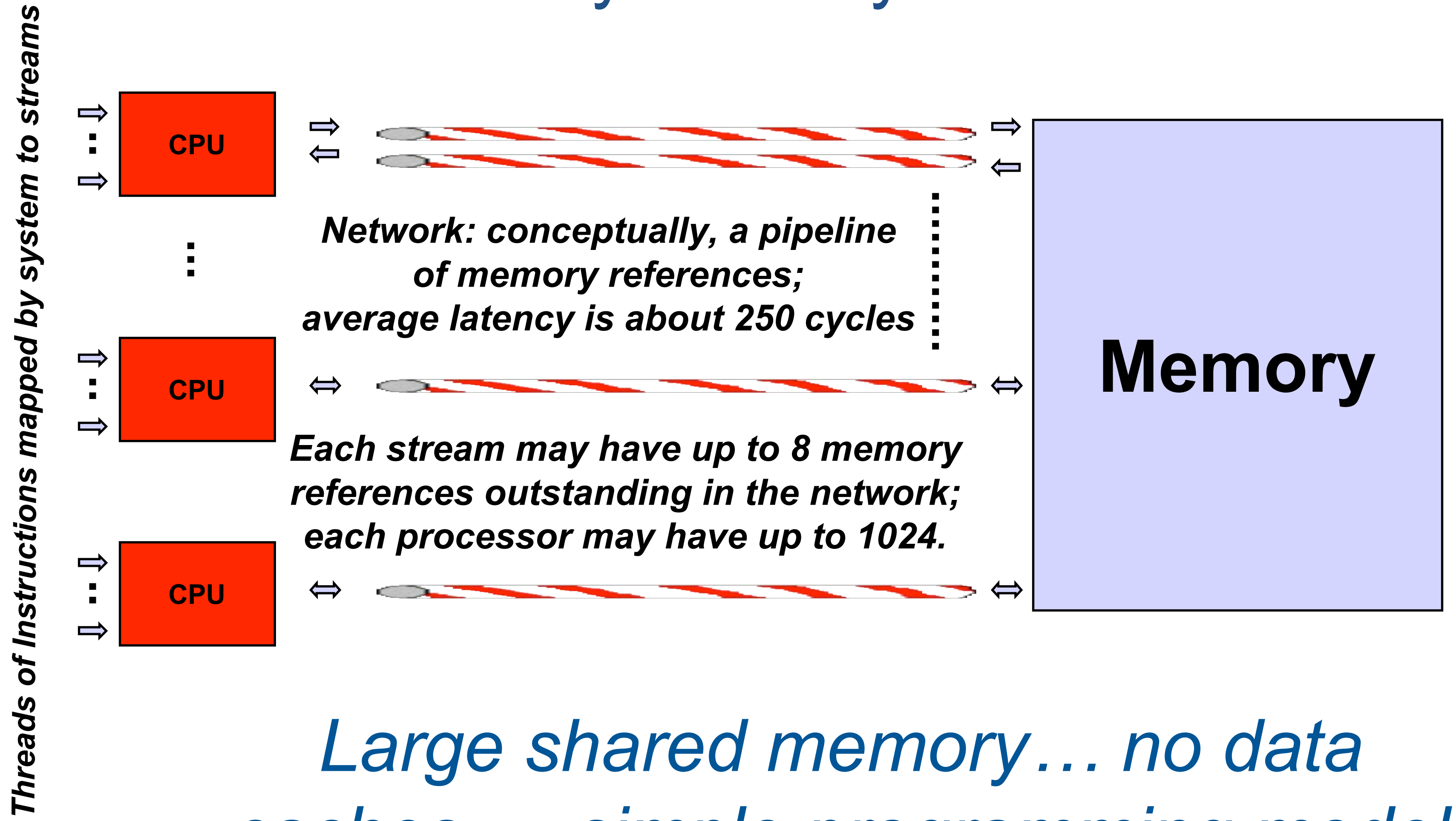
Cray Multithreading serves to open the box.

Cray MTA Processor



With enough active streams, an instruction may be executed at every clock cycle, thus ***masking latency to distant memory****…*

Cray MTA System



Network: conceptually, a pipeline of memory references; average latency is about 250 cycles

Each stream may have up to 8 memory references outstanding in the network; each processor may have up to 1024.

Large shared memory… no data caches… simple programming model

MTA Basic Features

- Flat shared memory
  - Virtual addresses hashed per word to physical locations
  - No data placement concerns
- Lightweight synchronization
  - int_fetch_add operation…

    `iglobal = iglobal + 3` requires one mem op
  - Full-bit on every location …
    - Producer-consumer semantics: two mem ops

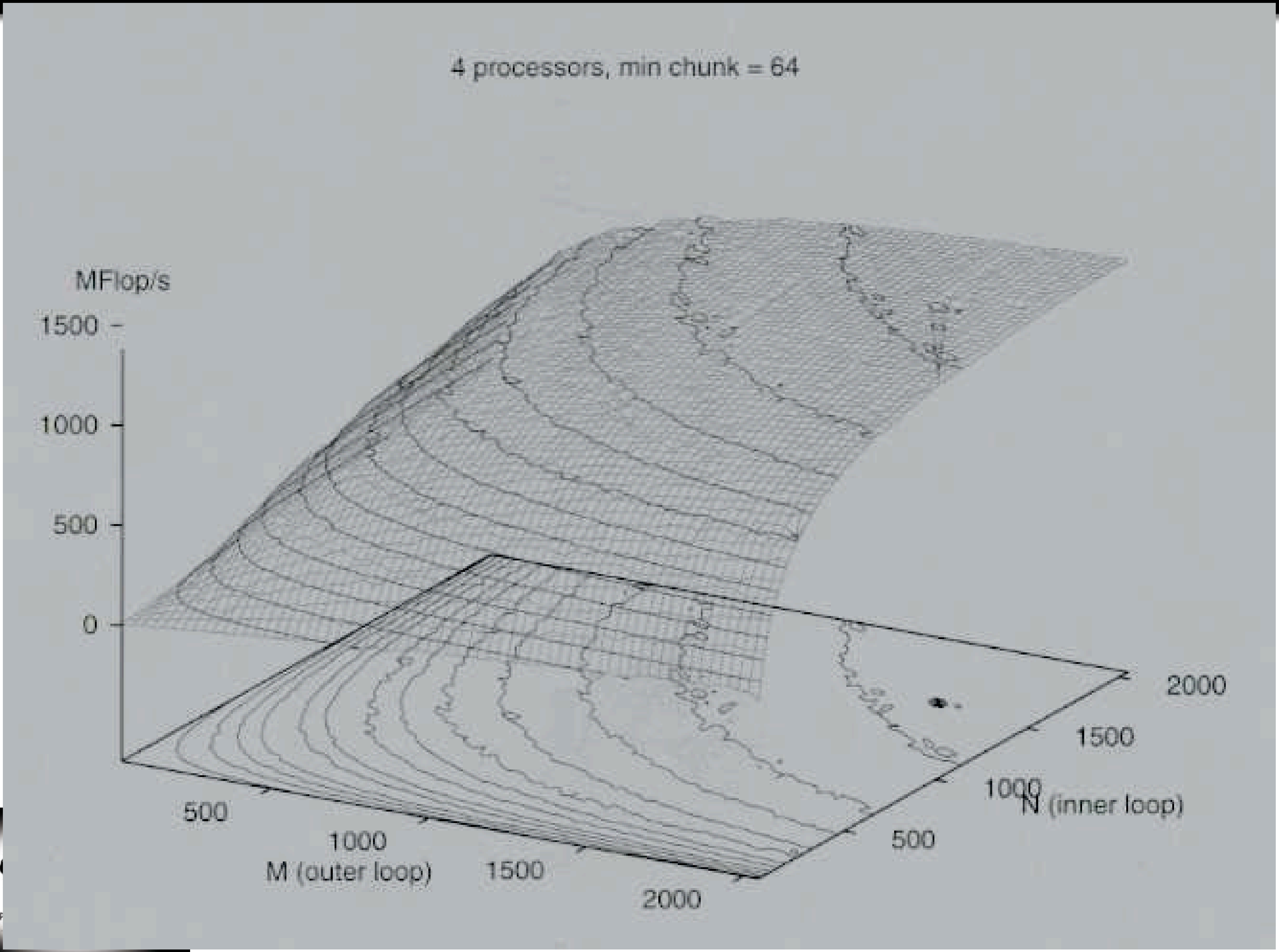
    `A[I] = A[I] + 0.5` requires two mem ops
- Thread-centric, not processor-centric, programming
  - Programmer expresses parallelism;
  - Compiler and runtime manage it
- Parallelizing compiler for C, C++, Fortran
  - Nested parallelism supported to any depth
  - Recursive parallelism supported via future construct

Programming 101: Rules for Performance Programming

- Av ying shared data
- in address order
- d ind n, linked data-struc
- tition into ndent computat
- oid synchroni
- ce data near proc oing co tion
- d conditional branc
- D xpect predictable pe e!

PETROGLYPHS TO PETAFLOPS
CUG 2005

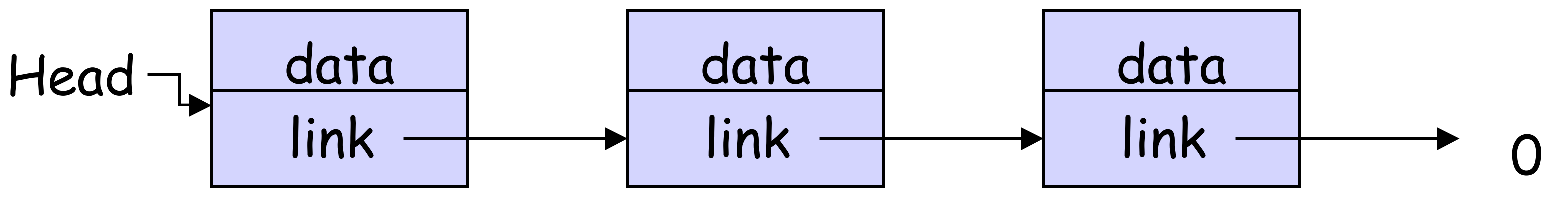
One Rule: ***Be parallel or die!***


4 processors, min chunk = 64
MFlop/s
1500
1000
500
0
500
1000
M (outer loop)
1500
2000
500
1000
N (inner loop)
1500
2000

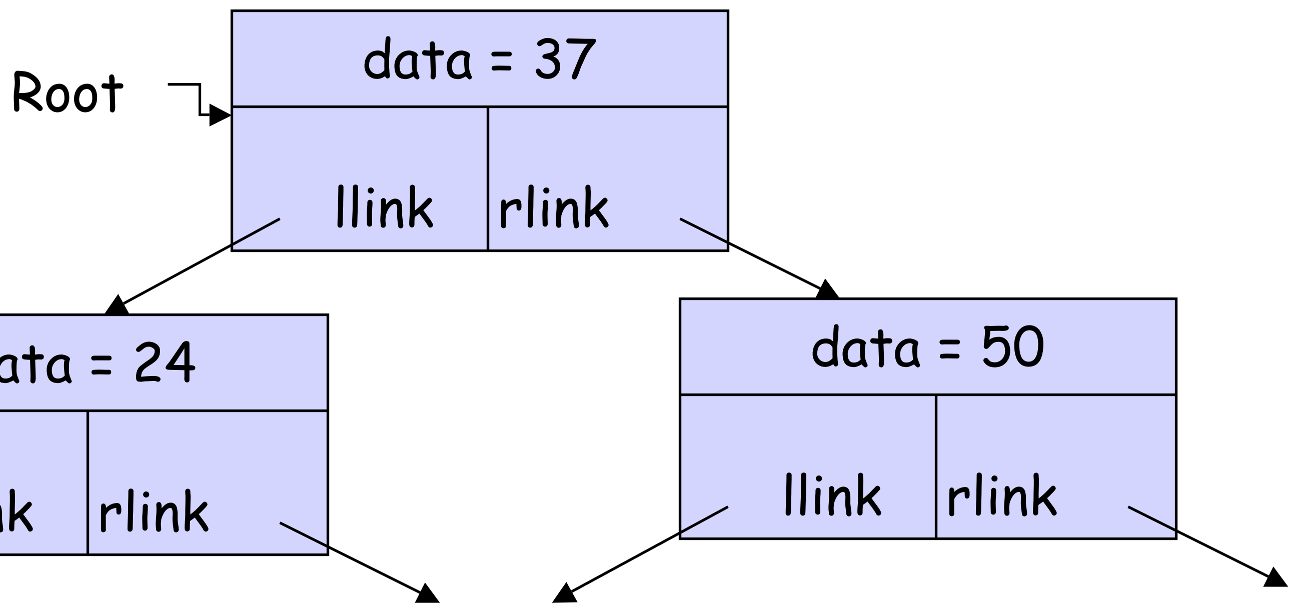
Linked Data Structures

{ C, C++, Objects, … }

Linked List:



Binary Tree:



A Basic Linked List Benchmark

- “Count the number of instances of some **target** value found in **N** randomly constructed lists of length **N**.”
- Representative of codes that do pointer chasing.
- Simple enough to analyze.

Linked-List Kernel

```
struct List {
  List *link;
  int data;
};
int count_data(List **ListHeads,
               int num_lists, int target) {
  int i; int sum = 0;
  for (i = 0; i < num_lists; i++) {
    List *current = ListHeads[i];
    while (current) {
      if (current->data == target) sum++;
      current = current->link;
    }
  }
  return sum;
}
```

Canal Output

```
                 | int count_data(List ** ListHeads,
                 |                int num_lists, int target) {
                 |   int i; int sum = 0;
                 |   for (i = 0; i < num_lists; i++) {
      2 P        |     List *current = ListHeads[i];
      3 PX       |     while (current) {
      2 P        +
      3 PX:$     |       if (current->data == target) sum++;
  ** reduction moved out of 1 loop
      3 PX       |       current = current->link;
                 |     }
                 |   }
                 |   return sum;
                 | }
```

Canal remarks

```
Loop   2 in count_data at line 10 in region 1
       In parallel phase 1
       Dynamically scheduled

Loop   3 in count_data at line 12 in loop 2
       Loop summary: 2 memory operations,
                     0 floating point operations
                     4 instructions, needs 68 streams for full utilization
                     pipelined
       1 instructions added to satisfy dependences
```

Performance Prediction

- Canal claims 4 instructions per inner loop body.
- MTA-2 peak is 220M instructions per processor per second.
- Searching N lists of N items using P processors
  => (4*N*N inst's)/(P*220M inst's/sec)
    = (N*N/55M sec) / P
- N = 5000   => (25/55 sec)/P = ~.45 sec /P
  N = 10000 => ~1.8 sec /P

Performance of Linked-List Kernel

System	Time for N=5,000	Time for N=10,000
SunFire 880MHz, 32GB	9.3 seconds	107
Intel Xeon 2.8GHz (32bit), 4 GB	7.15	40
Cray MTA-2/**1** 220MHz, 40 GB	**0.485**	**1.98**
Cray MTA-2/**10**	0.0525	.197

- N is # of lists and length of each list.
- MTA-2 performance is order of magnitude better per processor and scales predictably both with processors *and* problem size.

Sweet-spot comes at a price:

- Only 1 Cray MTA-1 was ever sold (~1995).
- Only 2 Cray MTA-2's were sold (~2002).
  - Largest was 40 processors, 160GB
- Price-performance for dense linear algebra?
  - 220 MHz. => 660 MFlop/s for $125,000
  - About 250:1…
- Single threaded execution is ~3-5 Mips.
- No current commercial applications.

High price relative to general purpose performance meant volume was too limited to justify investment in software.

CRAY The Supercomputer Company

Eldorado: MTA-3

CRAY USER GROUP INCORPORATED

Overview

- Eldorado is a peak in the North Cascades.
- Also, Cray project name for the MTA-3 with goals to:
  - Reduce cost
  - Increase scale
  - Minimize development risk
  - Preserve the good
- Eldorado strategy:
  - Hardware infrastructure reuses Red Storm, XT3.
  - Processor, programming model and software almost identical to MTA-2.

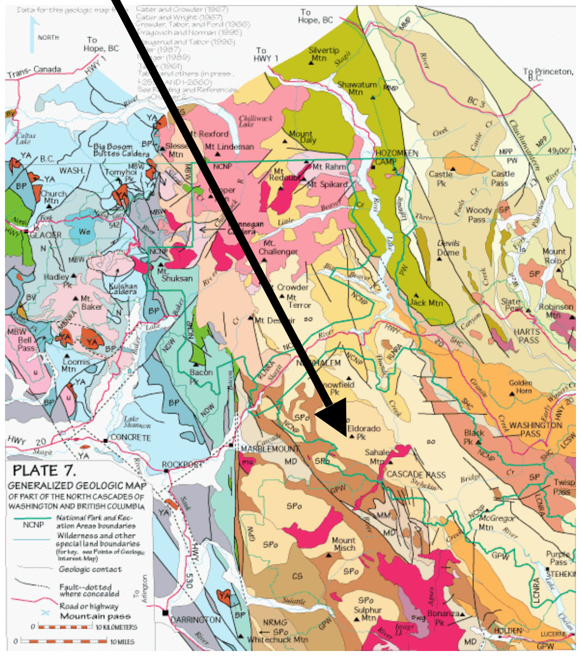
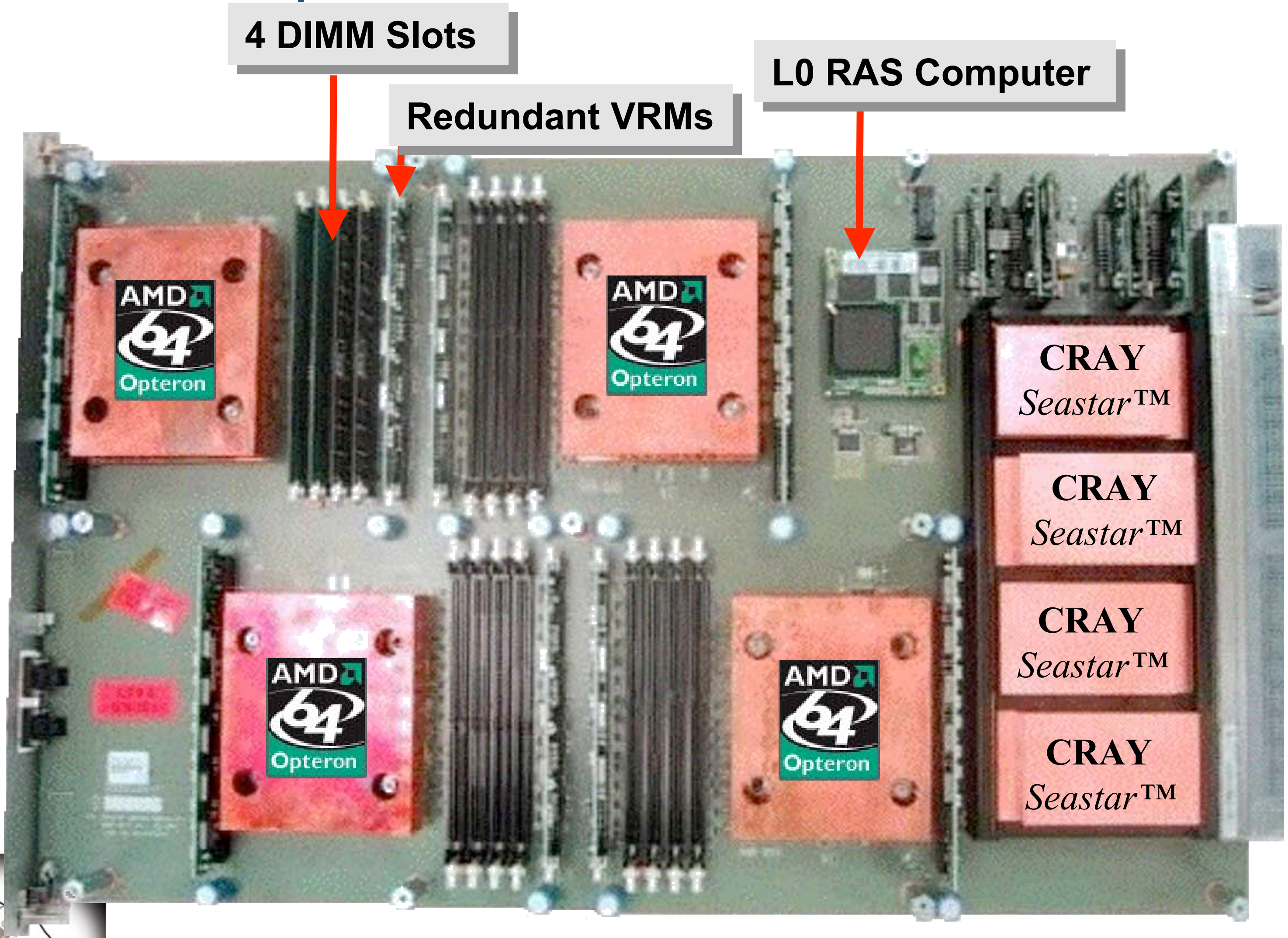


PLATE 7.
GENERALIZED GEOLOGIC MAP OF PART OF THE NORTH CASCADES OF WASHINGTON AND BRITISH COLUMBIA
Eldorado Pk

Red Storm

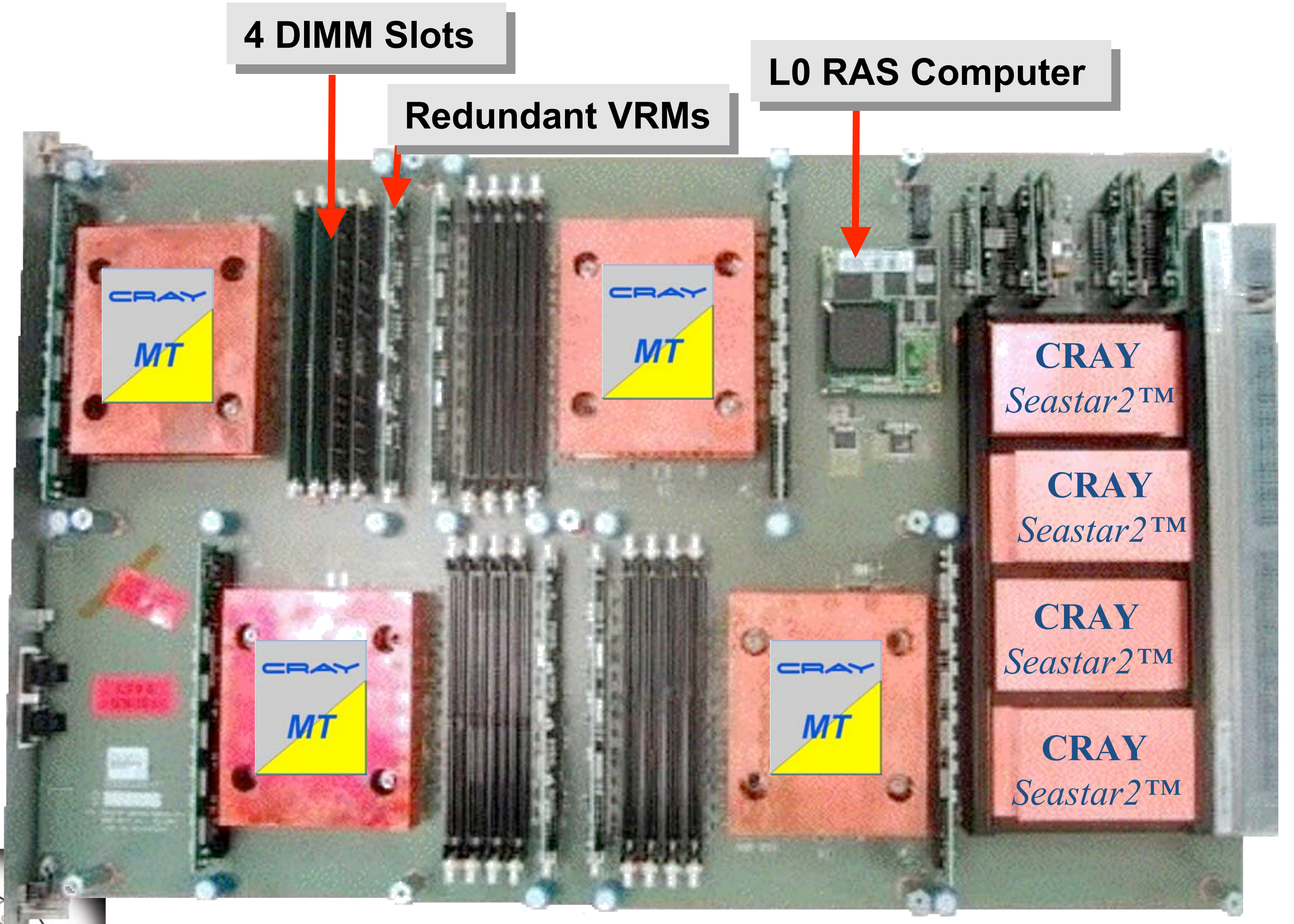
- Red Storm consists of over 10,000 AMD Opteron™ processors connected by an innovative high speed, high bandwidth 3D mesh interconnect designed by Cray (Seastar).
- Cray is responsible for the design, development, and delivery of the Red Storm system to support the Department of Energy's Nuclear stockpile stewardship program for advanced 3D modeling and simulation.
- Red Storm uses a distributed memory programming model (MPI).

Red Storm Compute Board



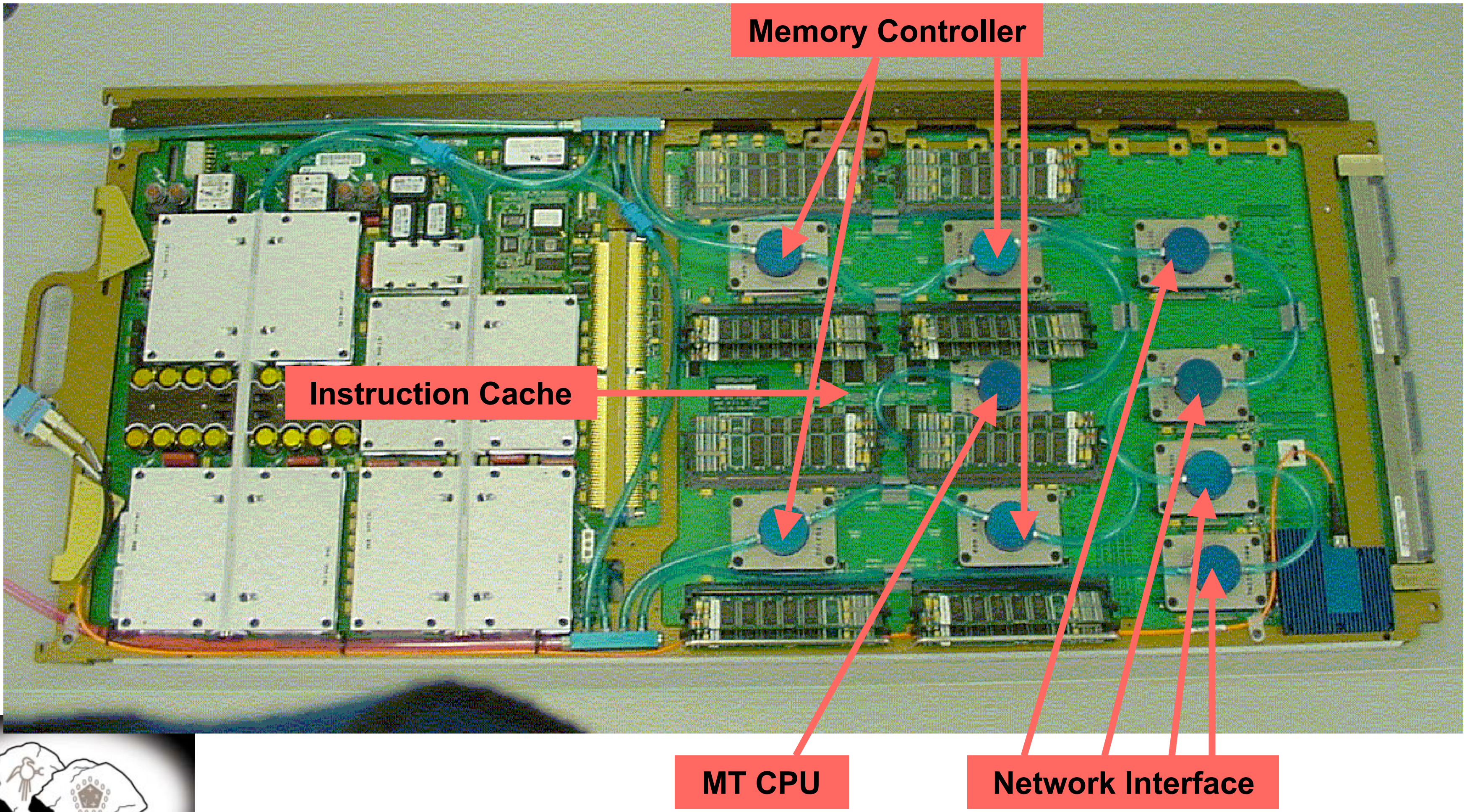
Just a few chips away…

Eldorado Compute Board (4 processors, 16-64GB)

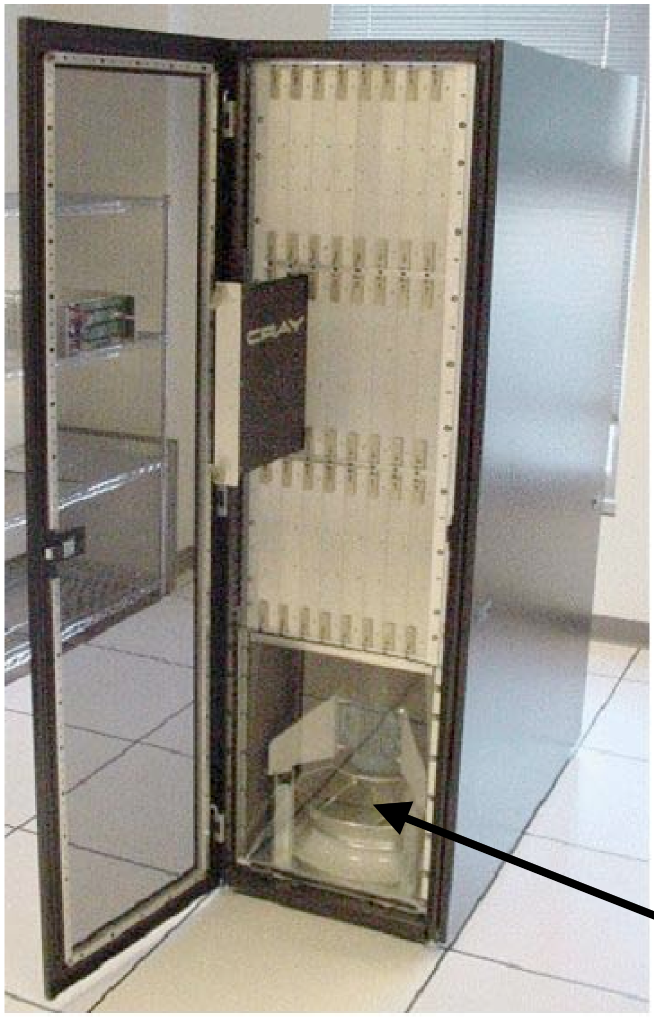


…is an MTA!

MTA-2 Module (1 processor; 4GB)

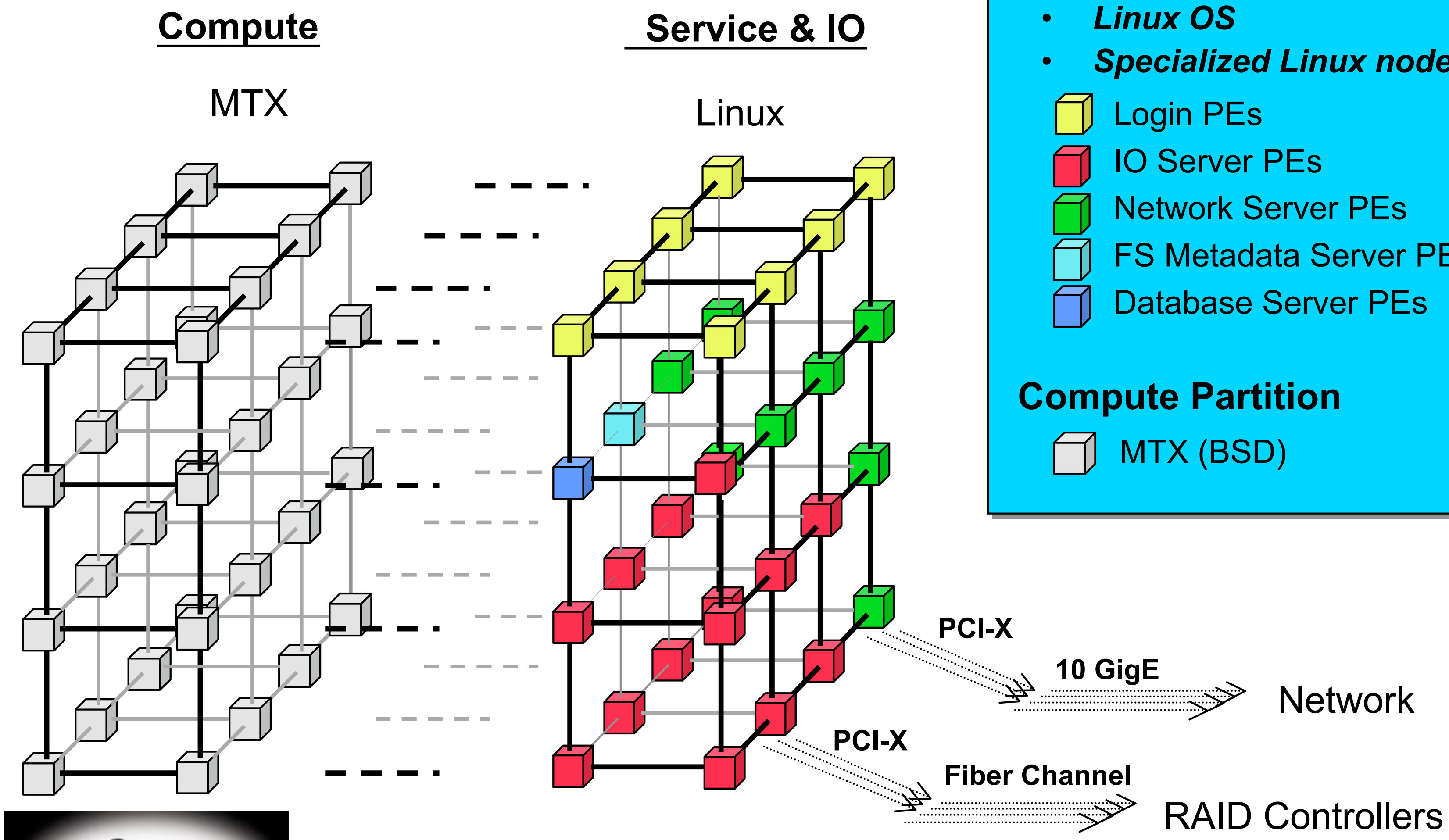


~~Red Storm Cabinet~~ Eldorado



Much simpler cooling system, too!

Eldorado System Architecture



Service Partition

- ***Linux OS***
- ***Specialized Linux nodes***

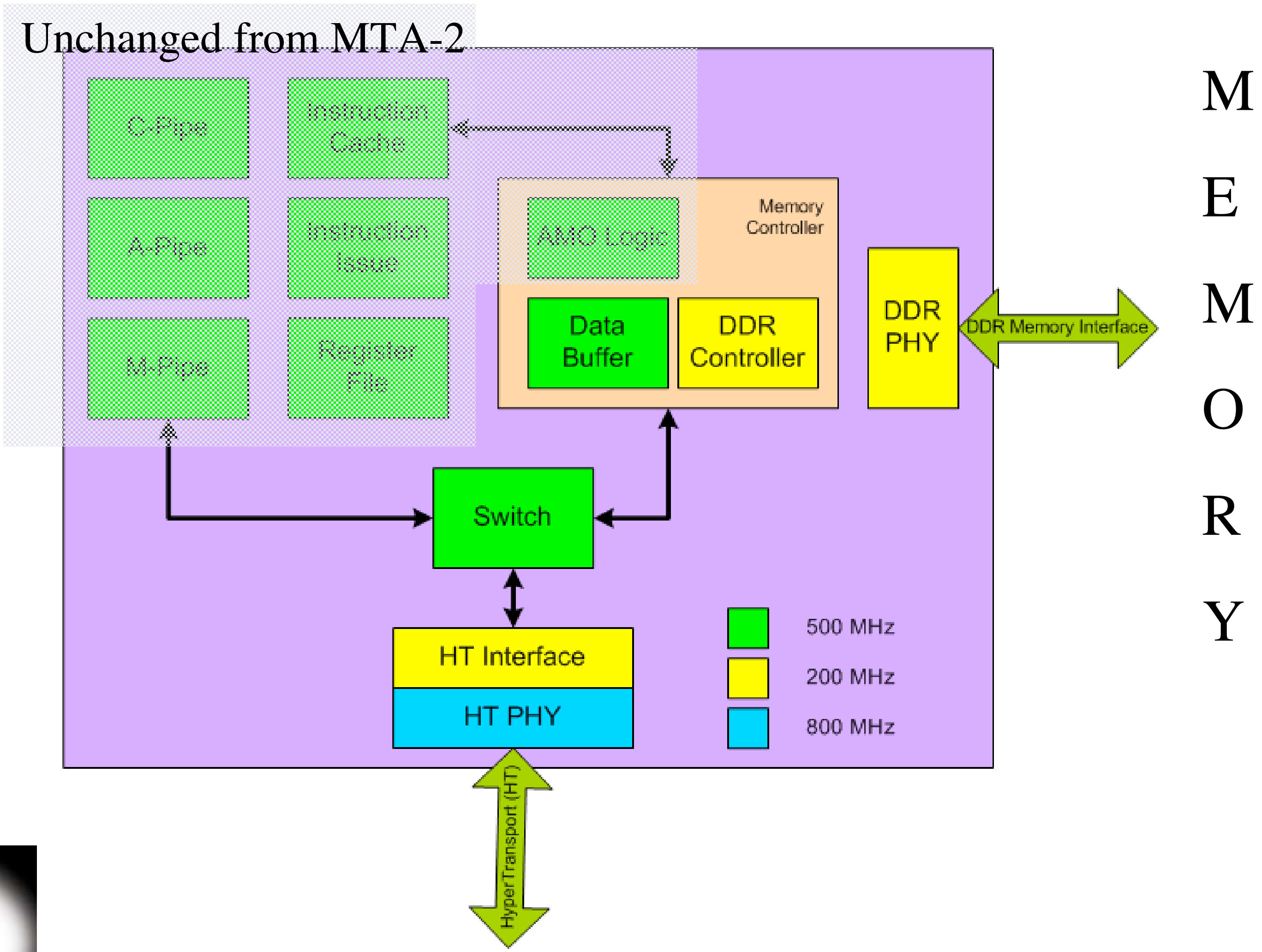
🟨 Login PEs
🟥 IO Server PEs
🟩 Network Server PEs
🟦 FS Metadata Server PEs
🟦 Database Server PEs

Compute Partition

⬜ MTX (BSD)

PETROGLYPHS to PETAFLOPS CUG 2005

Eldorado CPU/Mem Ctlr/Network ASIC



N E T W O R K (via SeaStar2 router)

Eldorado’s MT memory structure

- Partitioned per processor at boot time
  - Global memory – (e.g., what malloc returns)
    - Inaccessible to Opterons
    - Globally addressable by MT processors
    - Scrambled in blocks of 8 words
      - As on MTA-2, uniformly utilizes network & memory
  - Local memory – programmer generally oblivious to this
    - Accessible to Opterons via portals
    - Globally addressable by MT processors
    - Used to accelerate stack & local runtime data structures
- 128kB Buffer (cache) at memory module – 8 words per line
  - Mitigates bandwidth limitation & latency of DIMMs
  - Processor can access buffered memory every cycle

MTA-2 / Eldorado Comparisons

	MTA-2	Eldorado
CPU clock speed	**220 MHz**	**500 MHz**
Max system size	**256 P**	**8192 P**
Max memory capacity	**1 TB (4 GB/P)**	**128 TB (16 GB/P)**
Peak memory bandwidth (per processor)	**Stride 1: 4.0 GB/s** **Random: 4.0 GB/s**	**Stride 1: 6.4 GB/s** **Random: 0.8 GB/s** **Random from buffer: 4.0 GB/s**
TLB reach	**128 GB**	**128 TB**
Network topology	**Modified Cayley graph**	**3D torus**
Network bisection bandwidth	**3.5 P GB/s**	**15.36 $P^{2/3}$ GB/s**
Network injection rate	**220 M Ops/P**	**Variable (next slide)**

Eldorado Scaling

Example Topology	8x8x8	12x12x8	12x12x16	16x16x16	20x24x16
Processors	**512**	**1152**	**2304**	**4096**	**7680**
Memory capacity	**8 TB**	**18 TB**	**36 TB**	**64 TB**	**120 TB**
Sustainable remote memory reference rate (per processor)	**90 M/s**	**60 M/s**	**45 M/s**	**45 M/s**	**30 M/s**
Sustainable remote memory reference rate (aggregate)	**46 G/s**	**69 G/s**	**103 G/s**	**184 G/s**	**230 G/s**
Rate relative to 8x8x8	**1.00**	**1.50**	**2.24**	**4.00**	**5.00**

HPC Challenge Benchmarks: Preliminary Results

Computer System	#Procs	Global HPL GFLOP/s (per proc)	Global PTRANS GB/s	Global Random Access GUPS/s	Global FFT GFLOP/s	EP-STREAM Triad (per proc) GB/s	EP-DGEMM GFLOP/s (per proc)
MTA-2 Seattle	1	.18	. 538	.041	.182	1.59	--
MTA-2	10	.18	5.29	.405	1.90	1.57	.417
Eldorado (estimates)	256	1	61	12	50	4	1
Cray X-1 Oak Ridge	252	9.4	96.1	.00438	No Data	21.7	No Data
AMD Opteron 1.4GHz	128	.2526	3.247	no data EP: .00569	No Data	1.629	No Data
IBM eServer °	256	4.2	23.7	.000086	10	6.43	17.98

° **Base runs; optimized results are not published.**

Even a small Eldorado system will be the GUPS champ

Code length for benchmarks w/o library calls

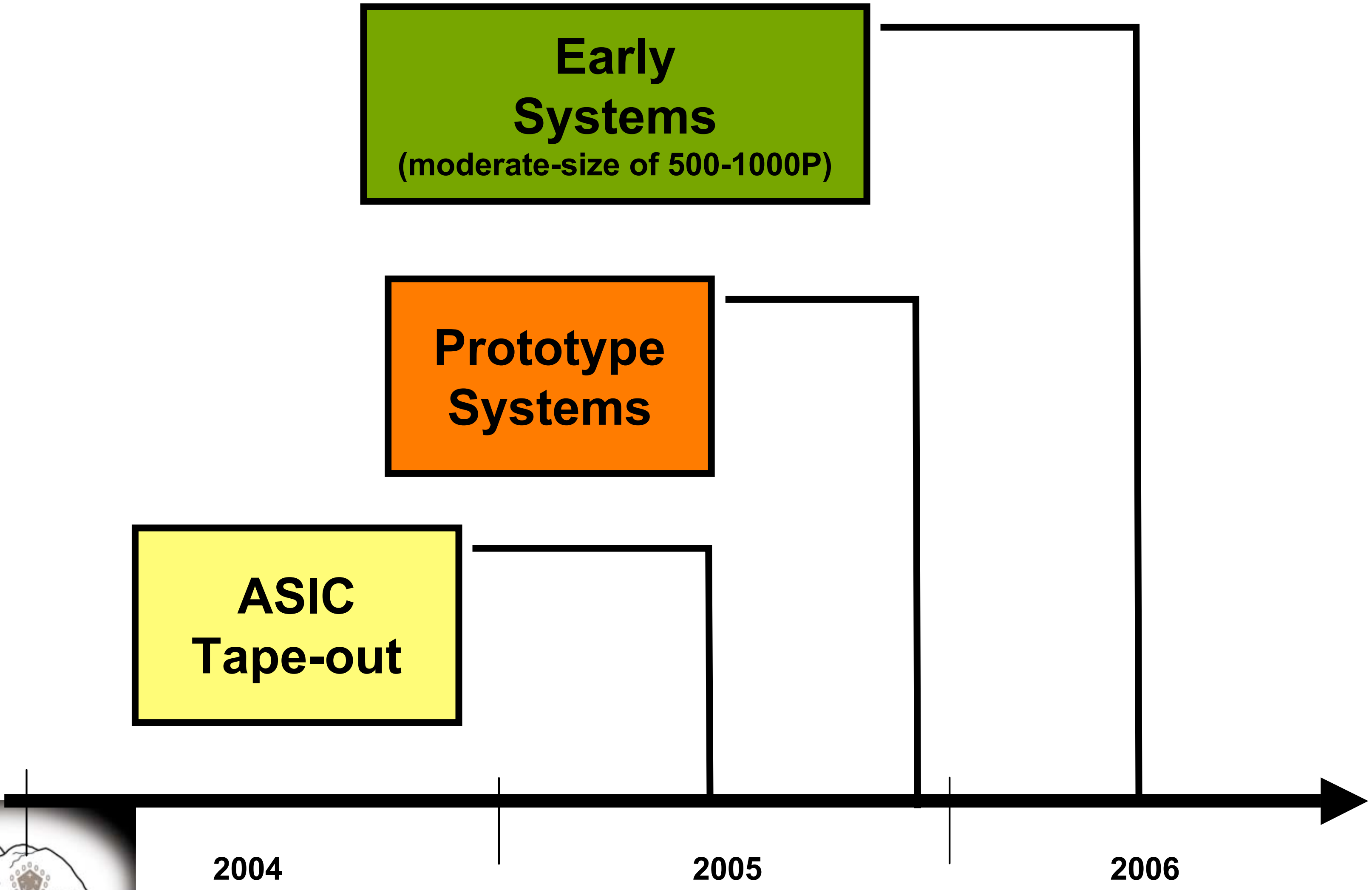
- HPCC MPI implementation:

RandomAccess	635
STREAM	509
PTRANS	923
Total	2067 lines

- MTA-2 shared memory implementation:

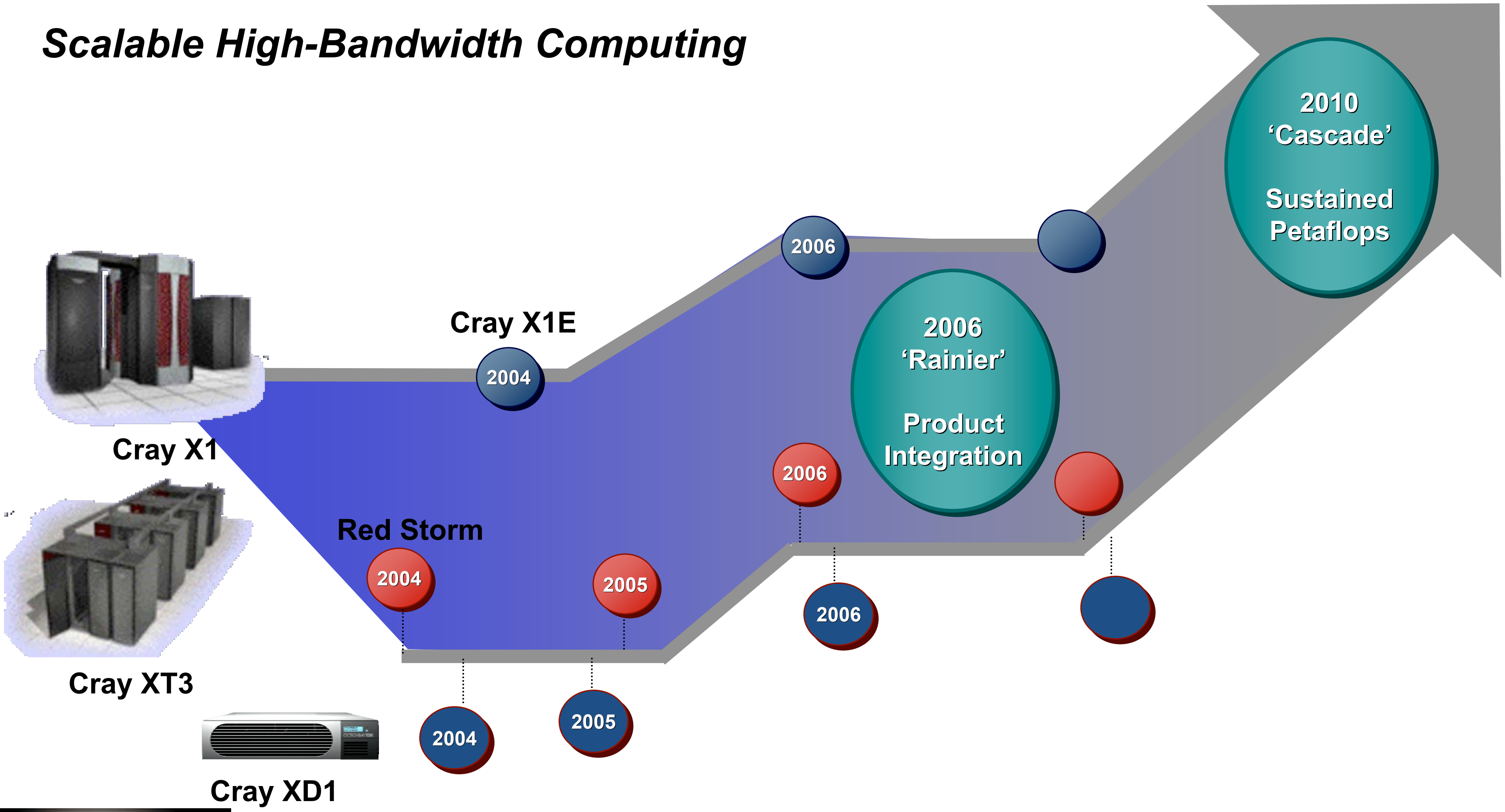
Total (for the three)	288 lines

Eldorado Timeline



Integrated computing

Scalable High-Bandwidth Computing



Eldorado Summary

- Unmatched features
  - 128TB globally addressable shared memory
  - Parallelism => Scaling, even when pointer chasing
  - Easy MTA programming model, simple code
  - Automatic parallelizing compiler
- Leverages Red Storm / XT3
  - Chip swap: distributed memory ➔ shared memory!
  - 10x less costly than MTA-2
  - Scales-up to over 8000 processors
- Focused on specialized, mission critical applications
  - Available mid-2006 at scale
- Major component of future computing by 2010
  - Multithreading is becoming mainstream
  - Integration into Cray's Cascade system as LWP

Meanwhile, we encourage you to learn more!