CO405H

Computing in Space with OpenSPL

Topic 12: 3D Space

Oskar Mencer **Georgi Gaydadjiev**

Department of Computing
Imperial College London

http://www.doc.ic.ac.uk/~oskar/
http://www.doc.ic.ac.uk/~georgig/

CO405H course page: http://cc.doc.ic.ac.uk/openspl15/
WebIDE: http://openspl.doc.ic.ac.uk
OpenSPL consortium page: http://www.openspl.org

o.mencer@imperial.ac.uk g.gaydadjiev@imperial.ac.uk

Why Computing in Space?

- Reducing costs
- Dealing with insane amounts of data (Big Data)
- Increasing speed (latency and throughput)
- Supporting growth at a very large scale
- Increasing competitive advantage
- Doing something that could not be done before…

- Cloud Computing!!!

"In the cloud, nobody knows that you're a DFE…"

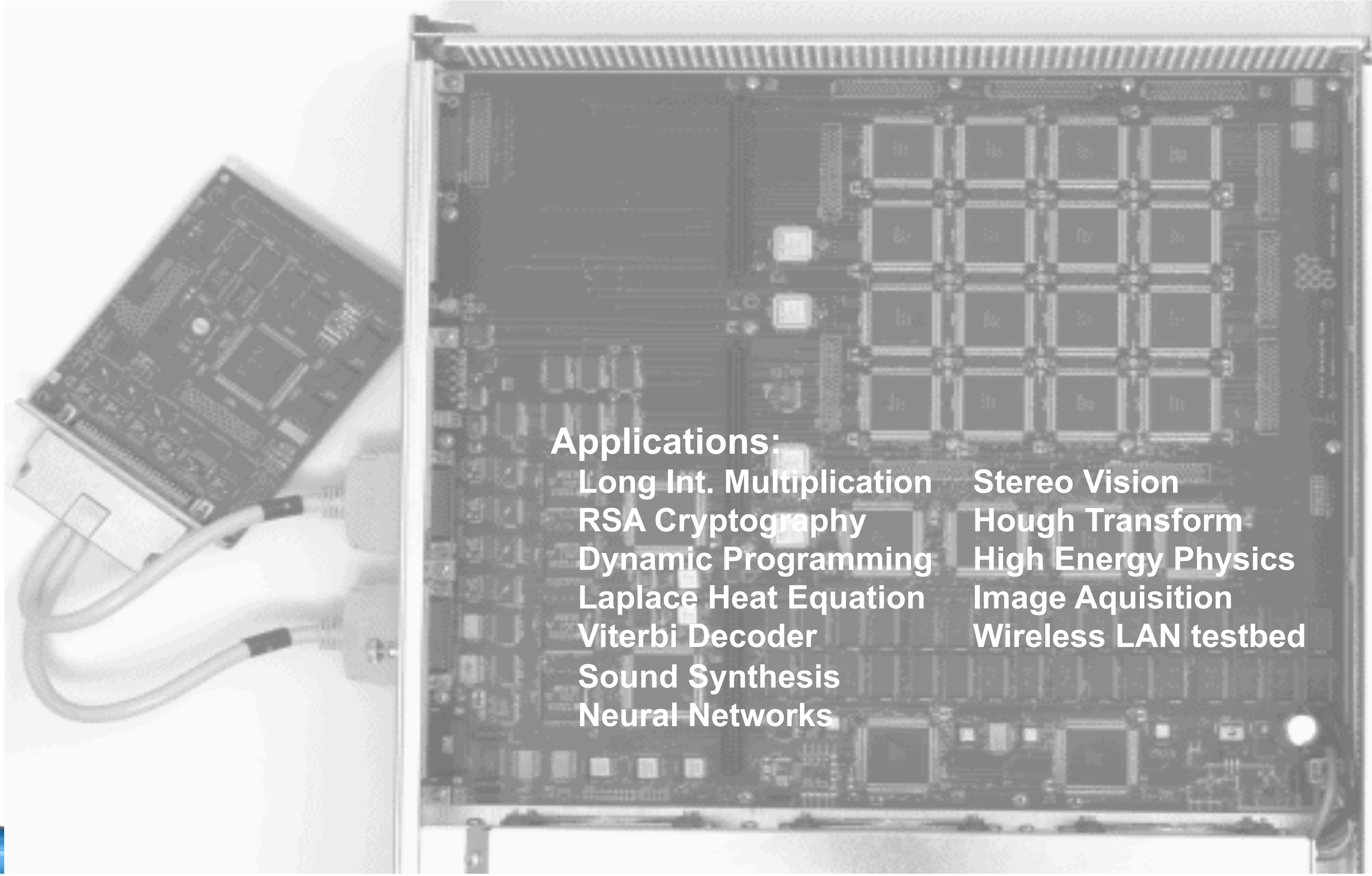
What is Computing in 3D Space

- **First, fix the size of the computer in 3D space**
- Then program a CPU machine and a Spatial Computer of the same size, and see the gain from computing in space

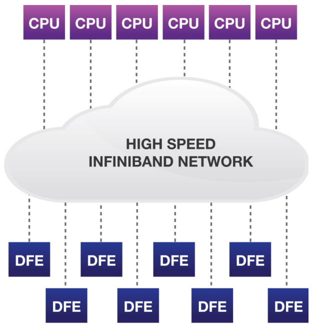
- Find a large enough dataset to fill the DFE machine

Computing in Space (1992)

Digital Equipment Corporation DEC PeRLe-1



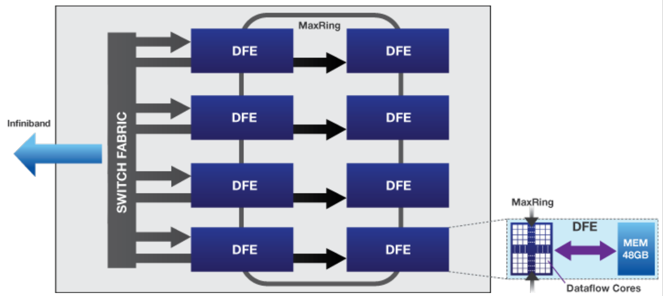
MPC-X 2012 Scalable Dataflow Computing

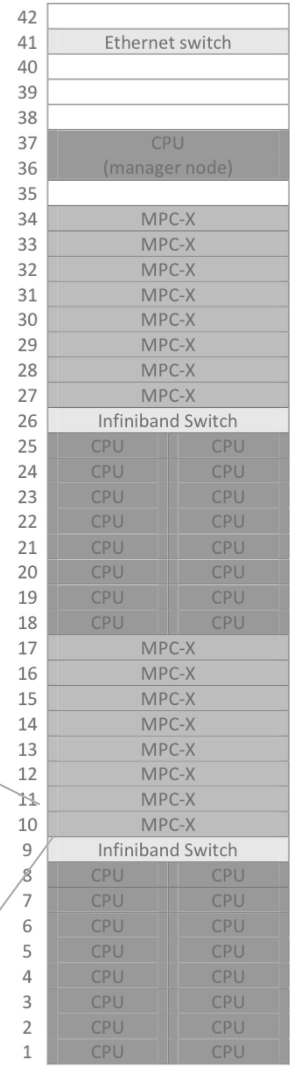


MPC-X Architecture
- Combine CPU and DFE nodes to handle unique compute challenges efficiently
- Use low-latency, high bandwidth Infiniband fabric for expandable compute
- Multi-Scale Cluster optimization balances resources at cluster, rack, node and DFE level

1U form "pizza box" form factor

MPC-X Node
- 112GB/s Infiniband Connectivity provides exceptional zero-copy bandwidth and latency
- 768GB DRAM to contain massive datasets
- 2GB/s MaxRing for intra-node communication
- PSU redundancy and 'lights out' management





OpenSPL enabled optimizations

Multiple scales of computing	Important features for optimization
complete system level	⇒ balance compute, storage and IO
parallel node level	⇒ maximize utilization of compute and interconnect
microarchitecture level	⇒ minimize data movement
arithmetic level	⇒ tradeoff range, precision and accuracy = discretize in time, space and value
bit level	⇒ encode and add redundancy
transistor level	=> create the illusion of '0' and '1'

And more, e.g., trade Communication (Time) for Computation (Space)

DSF Analytics - CME Group

www.cmegroup.com/trading/interest-rates/dsf-analytics.html

Designed for educational use only using Maxeler Technologies' curve construction methodology. This tool uses delayed data and displayed results are indicative representations only.

Please hover your mouse pointer over column titles and links for further information.

CME Ticker	Bloomberg Ticker	DSF Pricing					Timestamp
		Price	Coupon	PV01	NPV	Implied Rate	
T1UM4 2Y	CTPM4	100'057	0.750%	$19.97	$179.69	0.6600%	4:00:03 PM CT 4/4/2014
F1UM4 5Y	CFPM4	100'115	2.000%	$48.49	$359.38	1.9259%	4:00:03 PM CT 4/4/2014
N1UM4 10Y	CNPM4	100'225	3.000%	$90.16	$703.12	2.9220%	4:00:03 PM CT 4/4/2014
B1UM4 30Y	CBPM4	102'270	3.750%	$195.07	$2,843.75	3.6042%	4:00:03 PM CT 4/4/2014
T1UU4 2Y	CTPU4	100'085	1.000%	$19.93	$265.62	0.8668%	4:00:03 PM CT 4/4/2014
F1UU4 5Y	CFPU4	100'110	2.250%	$48.27	$343.75	2.1788%	4:00:03 PM CT 4/4/2014
N1UU4 10Y	CNPU4	101'125	3.250%	$89.55	$1,390.62	3.0948%	4:00:03 PM CT 4/4/2014
B1UU4 30Y	CBPU4	106'020	4.000%	$193.47	$6,062.50	3.6868%	4:00:03 PM CT 4/4/2014

Quotes and analytics are updated every 15 minutes.

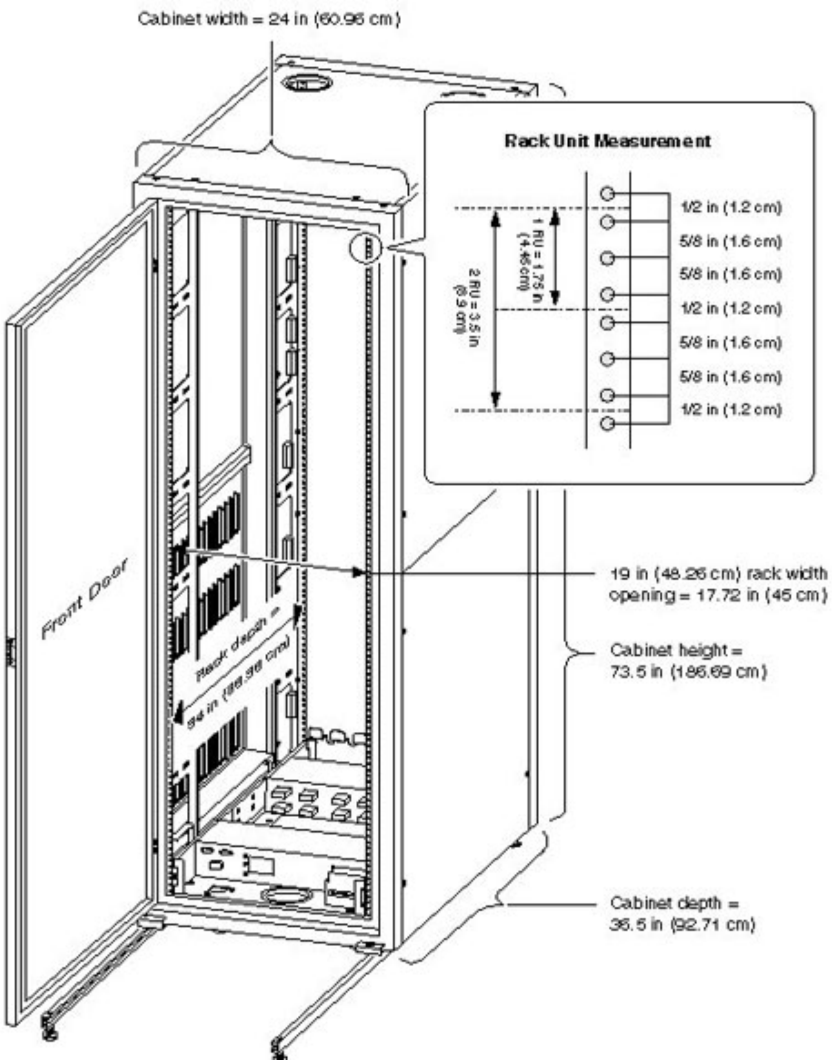
Analytics powered by Maxeler Technologies©

Instrument	CPU 1U-Node	Max 1U-Node	Comparison
European Swaptions	848,000	35,544,000	42x
American Options	38,400,000	720,000,000	19x
European Options	32,000,000	7,080,000,000	221x
Bermudan Swaptions	296	6,666	23x
Vanilla Swaps	176,000	32,800,000	186x
CDS	432,000	13,904,000	32x
CDS Bootstrap	14,000	872,000	62x

3D Spatial perspective on OPEX: *Measuring Rack Power at the Socket*

- Measurement includes:
  - Cooling, A/C
  - Power supplies
  - Rack level power distribution
  - Networking
  - Storage
  - Memory Chips
  - Compute Chips (CPUs and DFEs)
  - all the other stuff inside a rack that needs electricity
- Power per rack is nice but really we want *Useful Computations per Watt* of rack power consumption

1U definition



A rack holds computing units of 1U, 2U, 3U, 4U

1U = 19inch × 36.5inch × **1.75inch**
2U = 19inch × 36.5inch × **3.5inch**

Each compute unit has it's own power supply

1U CPU servers can have 1-2 mother boards
each with multiple CPU chips
each with multiple cores.

1U DFE boxes can have 8 DFEs or
[6 DFEs + 2 CPU chips], connected via Infiniband

Each 1U DFE box has 384GB of LMEM
and 48 independent memory channels.

Spatial perspective on OPEX:
Measuring the 3D Space for a program

1. Need a good reference point:

   Running multithreaded programs on multiple cores, how many cores are there in a 1U box? How many DFEs?

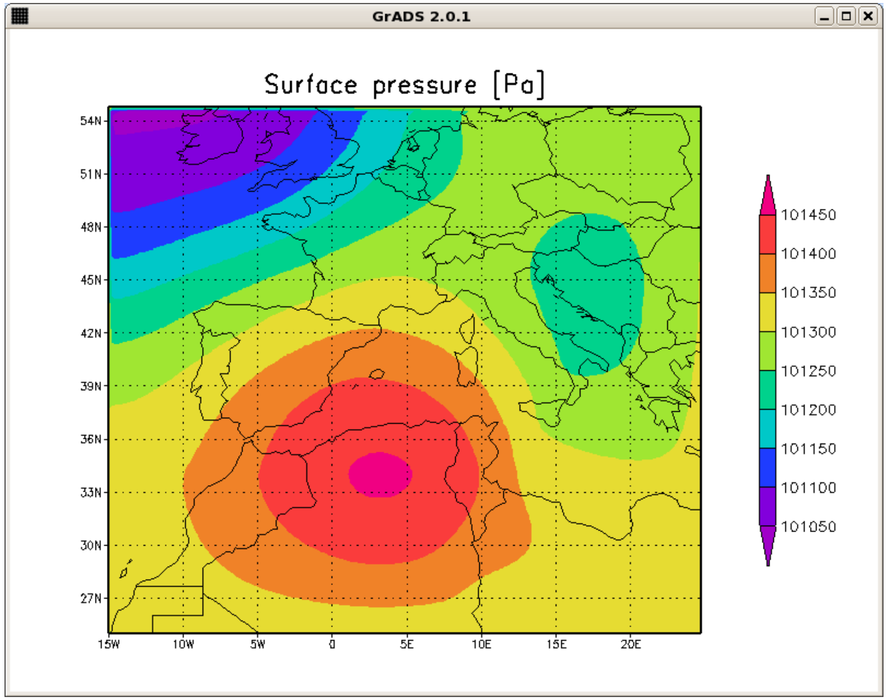
2. How many SysAdmins do you need per rack?

3. Do you measure computations per rack or TeraBytes processed per rack per second?

<u>Evaluate performance by comparing 1U to 1U!</u>

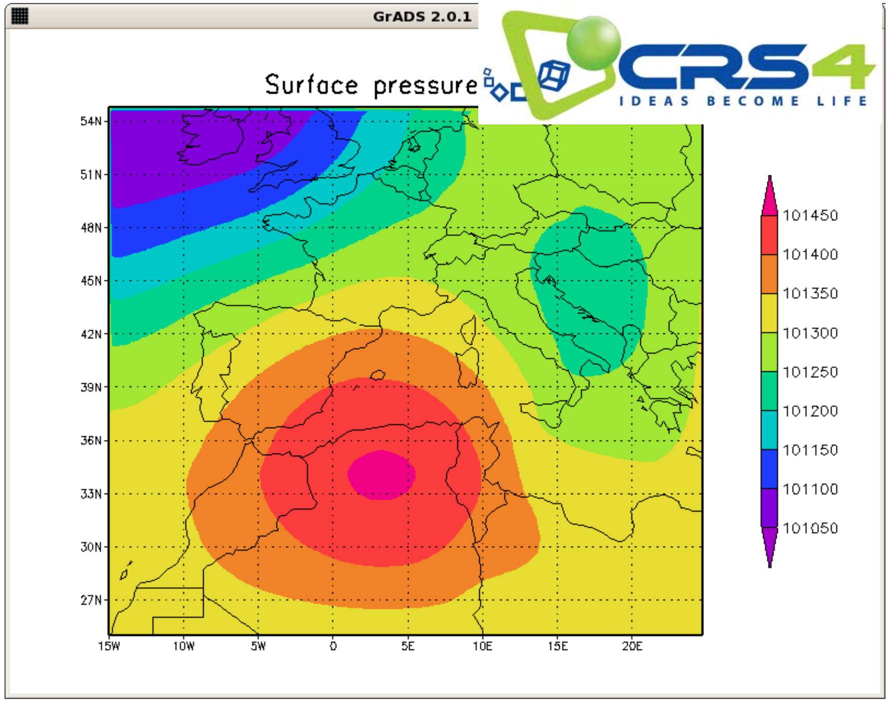
?

Imaging Platform Example: Weather



1U CPU Node

Wall Clock Time: 2 hours



1U Dataflow Node

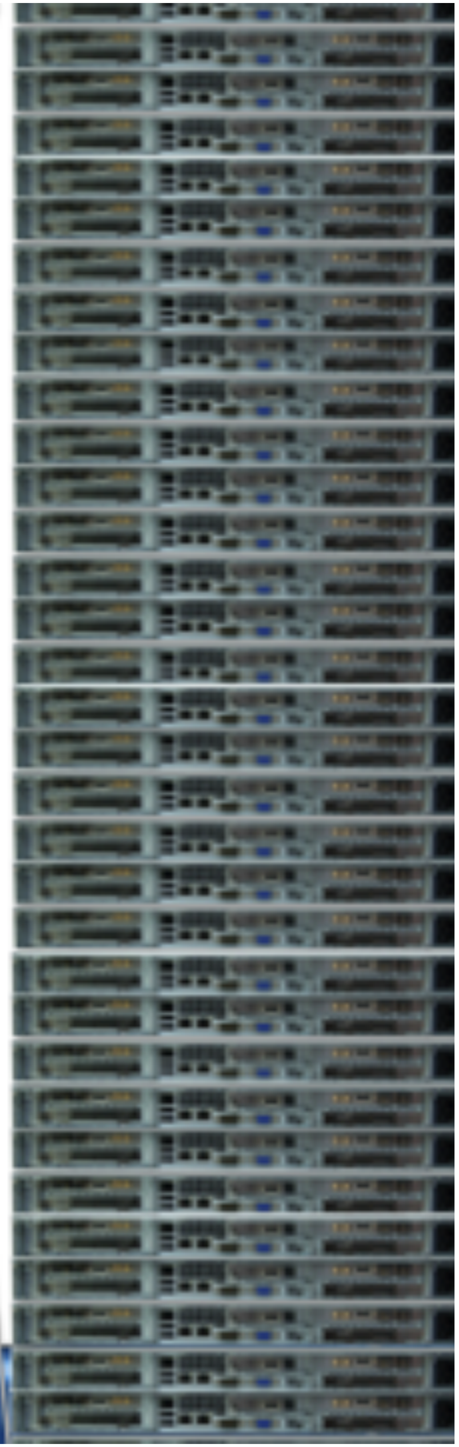
less than 2 minutes

Problem size: (Longitude) 13,600 Km x (Latitude) 3330 Km
Simulation of baroclinic instability after 500 time steps.

Acceleration of a Meteorological Limited Area Model with Dataflow Engines, D. Oriato, S. Tilbury (Maxeler), M. Marrocu, G. Pusceddu (CRS4), SAAHPC Conference, May 2012.

What does 60x mean?

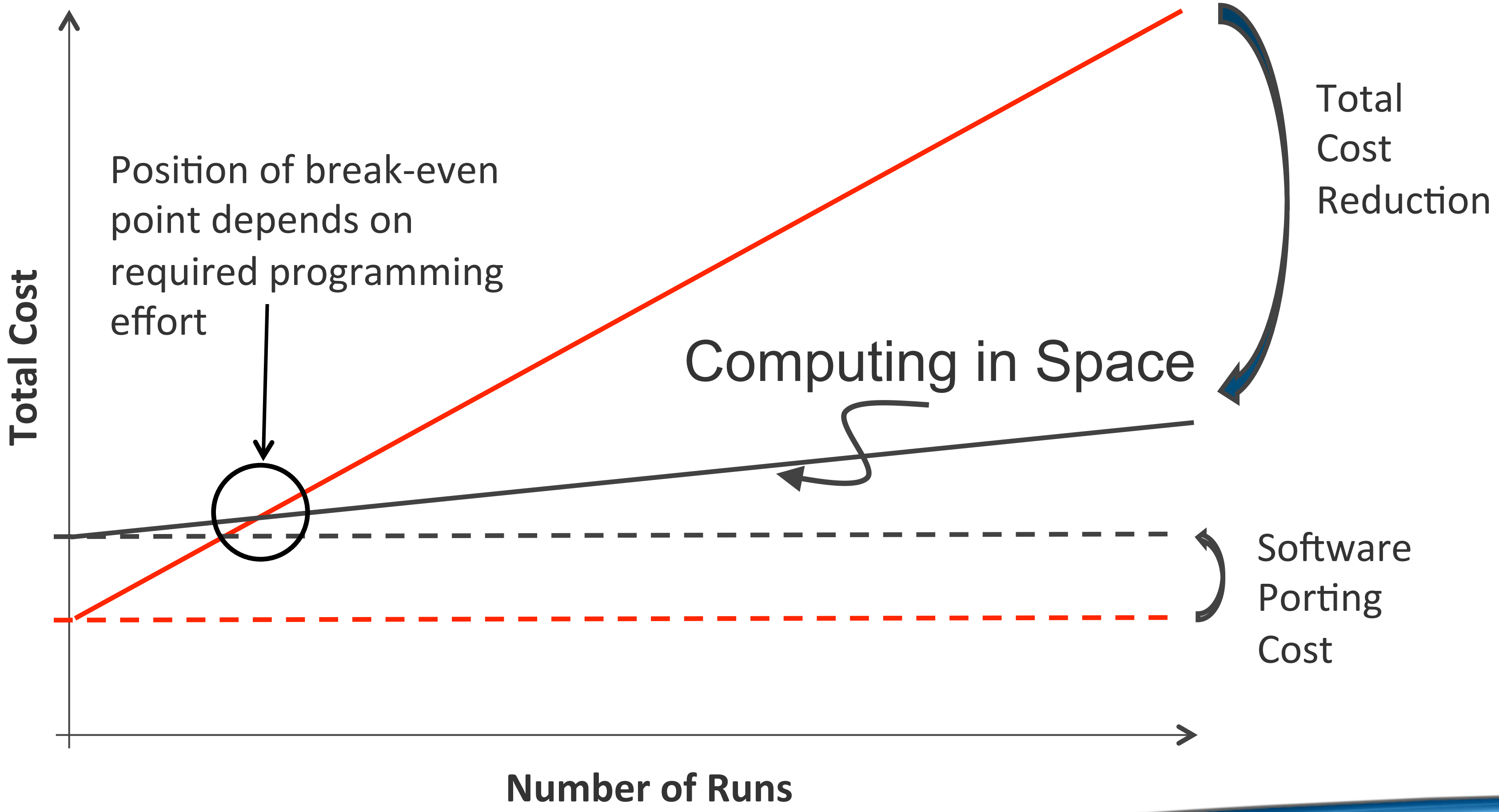
- Putting a supercomputer into a drone with 60x less space/power consumption.
- 6 days => 2.4 hours computing
- 10 images assembled
  => 600 images assembled
- 1 Mega Watt => 17 Kilo Watts
- Every $1M spent on
  buying a dataflow machine
  delivers power savings
  over 3 years
  worth more than $900K.



Economics of Computing Space

you got to have a lot of data….

Total Cost of Ownership = f(HW, SW, maintenance, power, real estate,…)



Total Cost of Ownership = CAPEX + OPEX

Capital Expenditure + Operating Expenditure

50x

- **50x** Speed-up per 1U server node
- **32** MPC-X Node Solution
- Equivalent to **1600** CPU-only Nodes
- **$3.2m** Operational cost savings over 3 years

30x

- **30x** Speed-up per 1U server node
- **40** MPC-X Node Solution
- Equivalent to **1200** CPU-only Nodes
- **$1.8m** Operational cost savings over 3 years

20x

- **20x** Speed-up per 1U server node
- **50** MPC-X Node Solution
- Equivalent to **1000** CPU-only Nodes
- **$1.7m** Operational cost savings over 3 years

40x

- **40x** Speed-up per 1U server node
- **32** MPC-X Node Solution
- Equivalent to **1280** CPU-only Nodes
- **$2.6m** Operational cost savings over 3 years

Example of Economics of Computing in Space

On the plus side:

1. Budget for buying a new machine is $2.8M
2. An x86 CPU-only machine for $2.8M has $1.3M annual electricity cost
3. A DFE machine for $2.8M creates $27K annual electricity cost

Given that the purchase budget is fixed, the decision to use DFEs saves $1.27M per year.

AND the DFE machine is a lot faster than the CPU machine with the same purchase price.

So by using DFEs, it is possible to get the advantage of saving operational costs AND at the same time get a machine that is a lot faster.

On the minus side:

Have to port applications to DFEs, and operational savings might not be attractive to everyone…

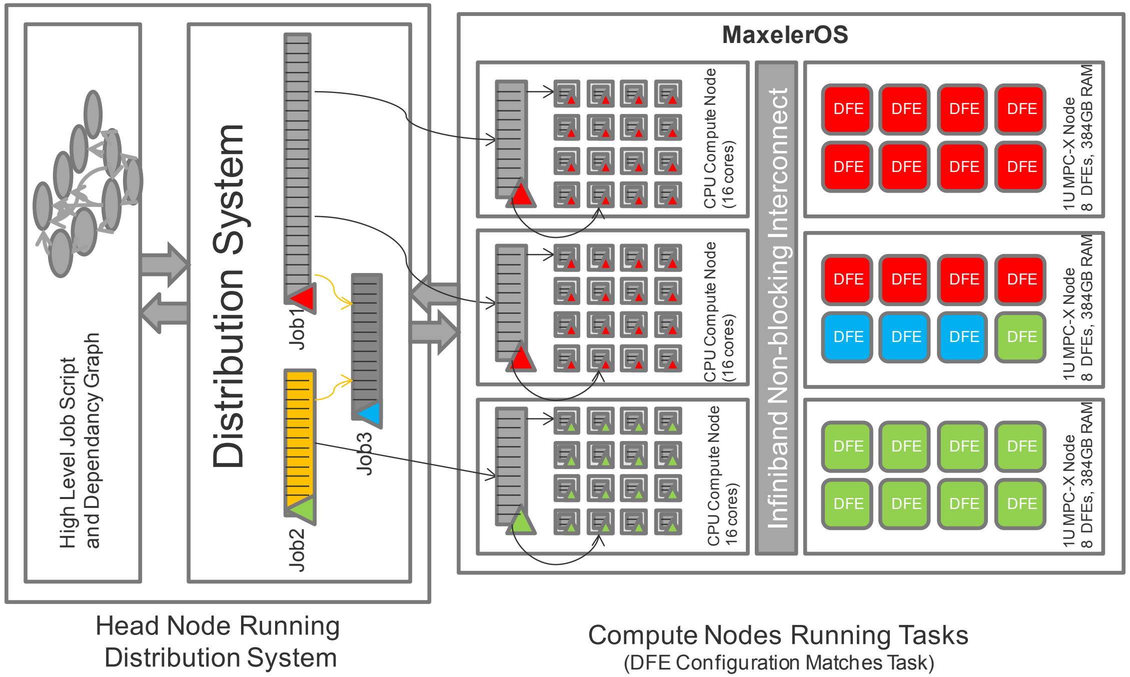
The management challenges

- Resource Utilization
  - CPU bottleneck may mean no work for DFE
  - Direct impact on <X>/performance metrics
- Varying workloads
  - Some applications may need many DFEs, others few or none
- Context switching
  - DFEs are *configured* for a particular task
  - Context switching is expensive and should be minimized

➡ Enable system to *adapt* to runtime workload

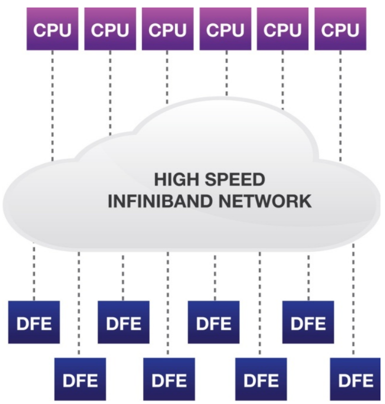
DFE Cluster Management

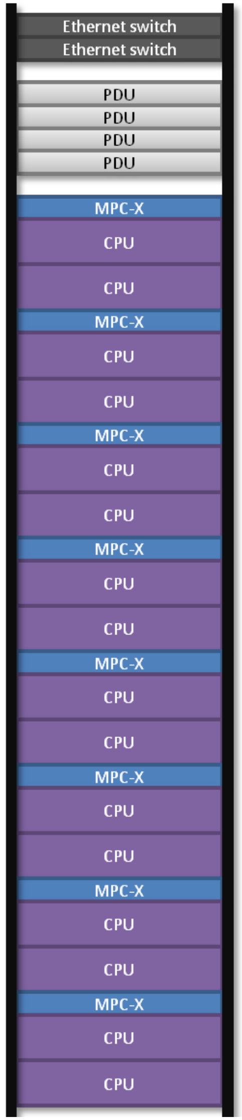
- ClusterMap job distribution for DFE clusters
- Optimized for distributing large numbers of small compute jobs with complex dependencies



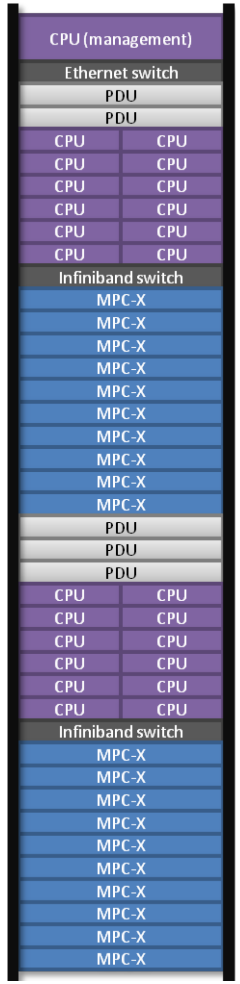
Rack-level optimization

- Optimized to balance resources for particular application challenges
- Flexible at design-time and at run-time





48U seismic imaging cluster



42U in-memory analytics cluster

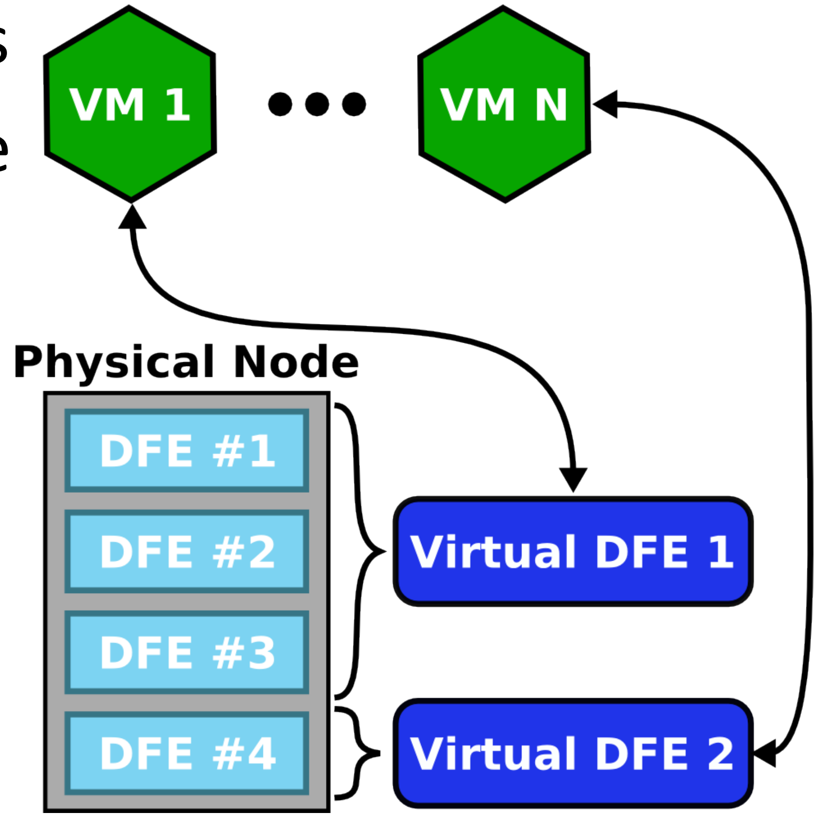
Using a single DFE with SLiC

- **S**imple **Li**ve **C**PU Interface:
  MaxCompiler-generated + fixed software functions for interacting with DFEs

```
//include auto-generated AVG_* prototypes
#include “AVG.h”

// Load .max-file onto any (“*”) available engine
max_file_t *avg_maxfile = AVG_init();
max_engine_t *eng1 = max_load(maxfile, “*”);

// Set-up and execute an “action”
float *x = <relevant data>, *y = <relevant data>;
AVG_action_t a;
a.instream_x = x;
a.outstream_y = y;
a.total_items = n;
AVG_run(eng1, &a);

// Shutdown
max_unload(eng1);
```

Virtual DFEs

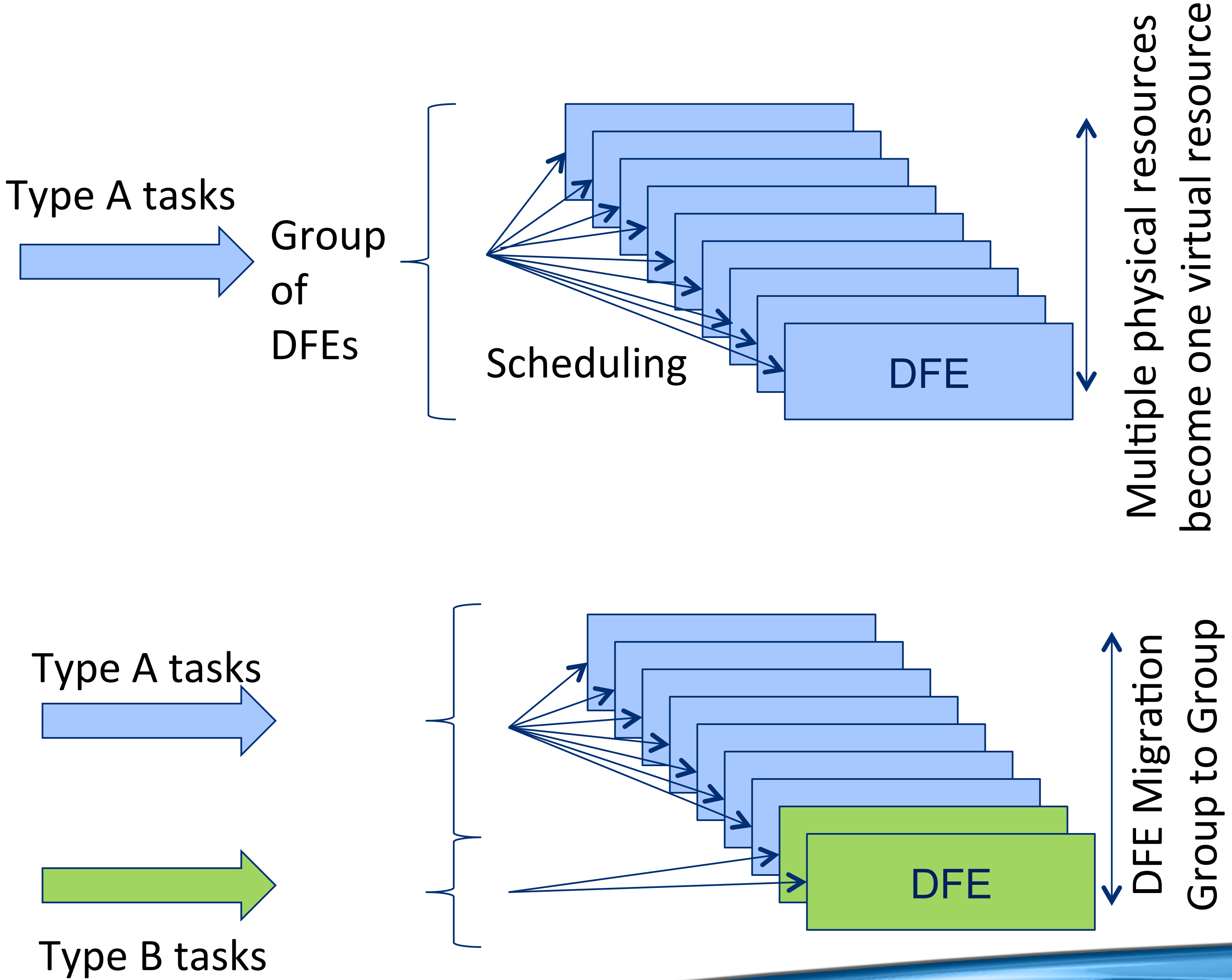
- Aggregate zero or more physical DFEs into *virtual DFEs*
- Multiple CPU clients can share virtual DFEs
  - Minimize reconfiguration
  - Allow sharing of datasets
- Every thread/process wishing to use the same group uses a common “tag”
- SLiC automatically allocates DFEs for group

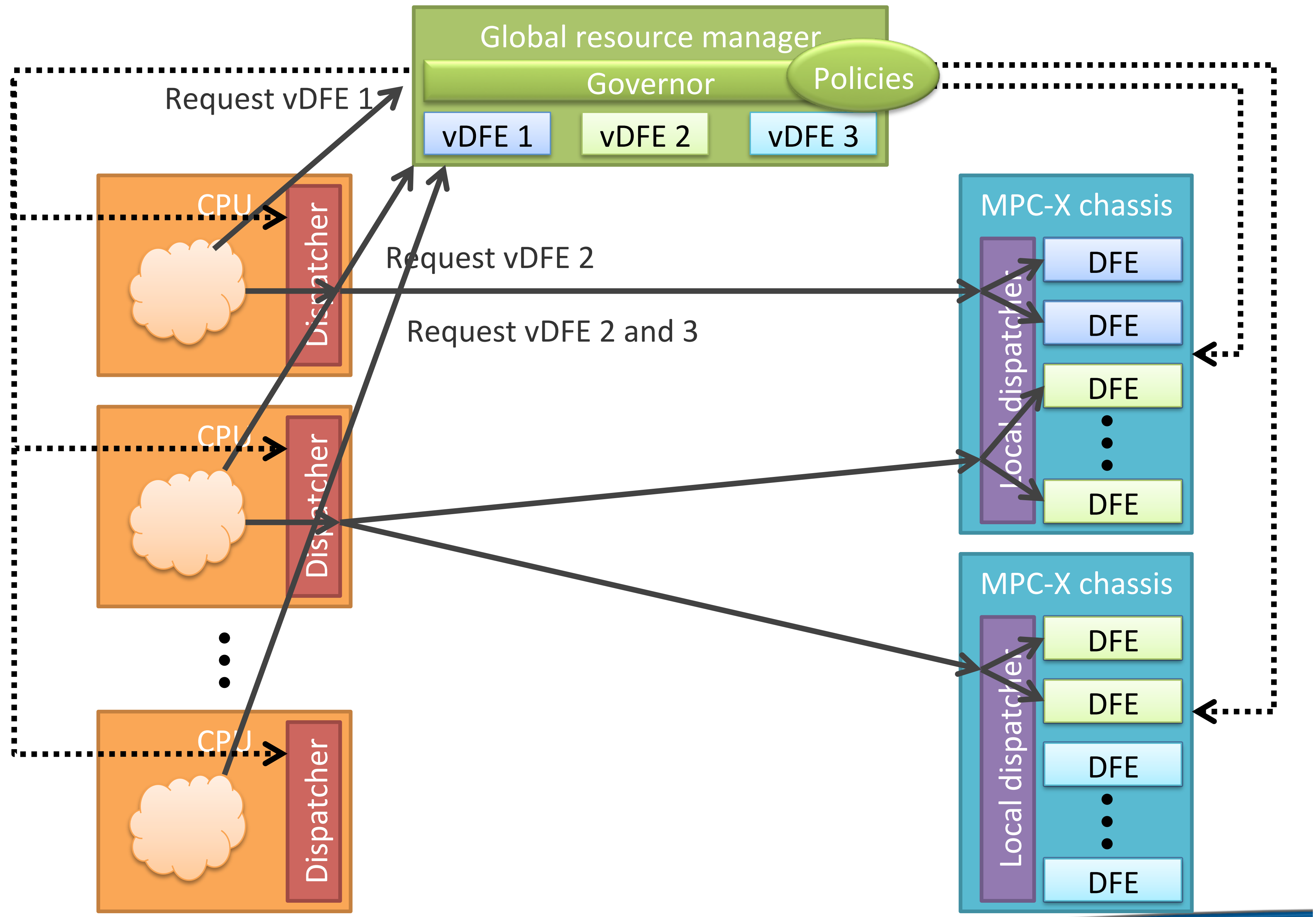


Accessing a Virtual DFE group

```
max_file_t *maxfile = AVG_init();

// Load a named group of engines that will be shared with other processes.
// All processes in group must make same call.
int group_size = 10;
max_group_t *grp =
        max_load_group(maxfile, MAXPROP_SHARED, “tag@*”, group_size);

// ... sometime later ...

// Set-up actions as normal and then either
AVG_action_t *act1 = <relevant data>, *act2 = <relevant data>;
if (condition) { // Lock DFE to preserve state
        max_engine_t *eng = max_lock_any(grp1);
        AVG_run(eng, &act1);
        AVG_run(eng, &act2);
        max_unlock(eng1);
} else { // DFE run atomically (fast)
        AVG_run_group(grp, &act1);
        AVG_run_group(grp, &act2); // no state preserved between calls
}
```

Virtual/Physical Resource Balancing





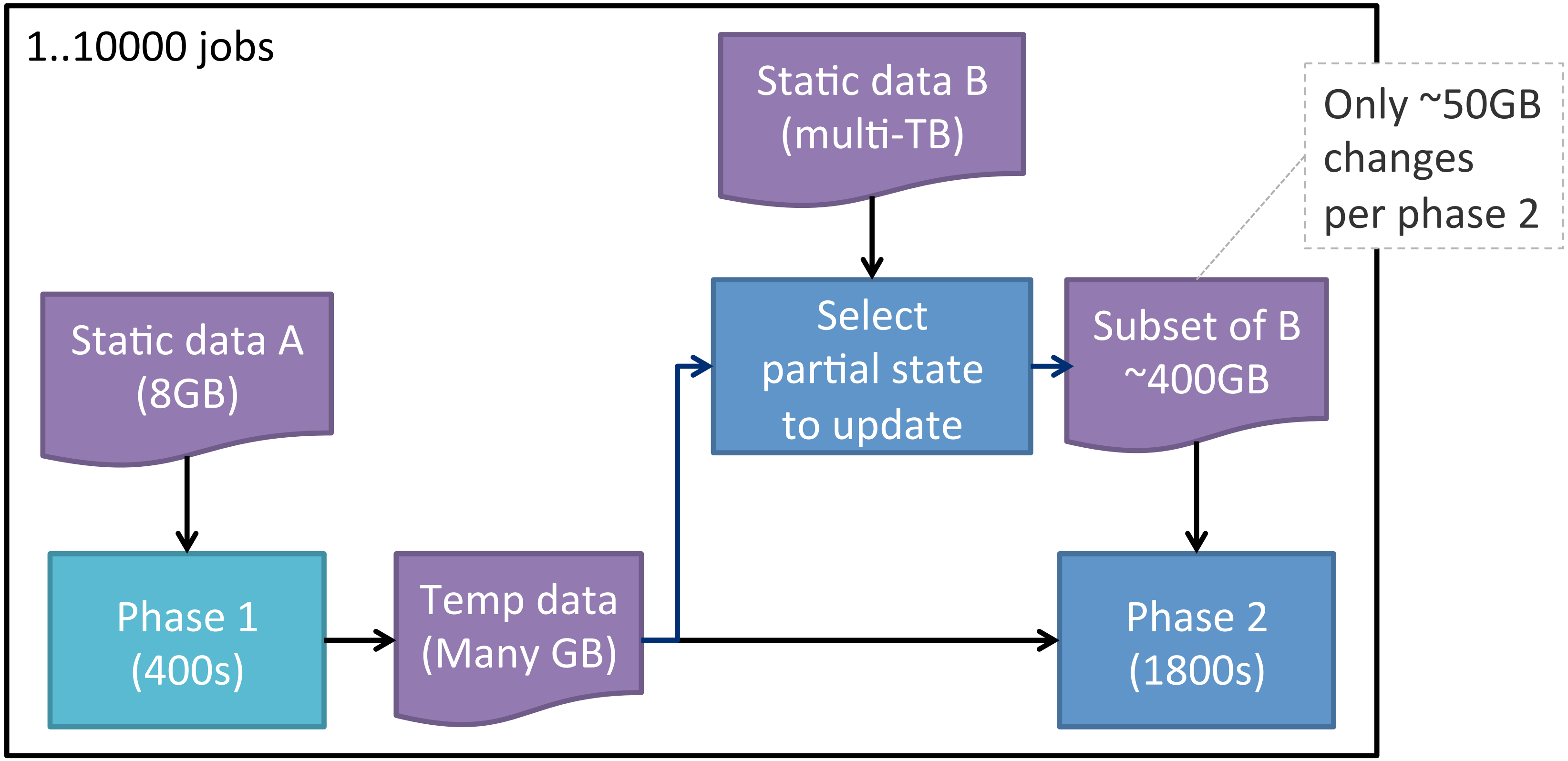
Global resource manager
Governor
Policies
vDFE 1
vDFE 2
vDFE 3
Request vDFE 1
Request vDFE 2
Request vDFE 2 and 3
CPU
Dispatcher
CPU
Dispatcher
CPU
Dispatcher
MPC-X chassis
Local dispatcher
DFE
DFE
DFE
DFE
MPC-X chassis
Local dispatcher
DFE
DFE
DFE
DFE

Architecture key points

- Action dispatch is fast & distributed
- Three levels of load balancing:
  - Actions dispatched to which MPC-X
  - Actions dispatched to which DFE
  - Physical DFEs (re)allocated between virtual DFEs
- Cluster configuration can be continually adapted at runtime to meet latest resource demands

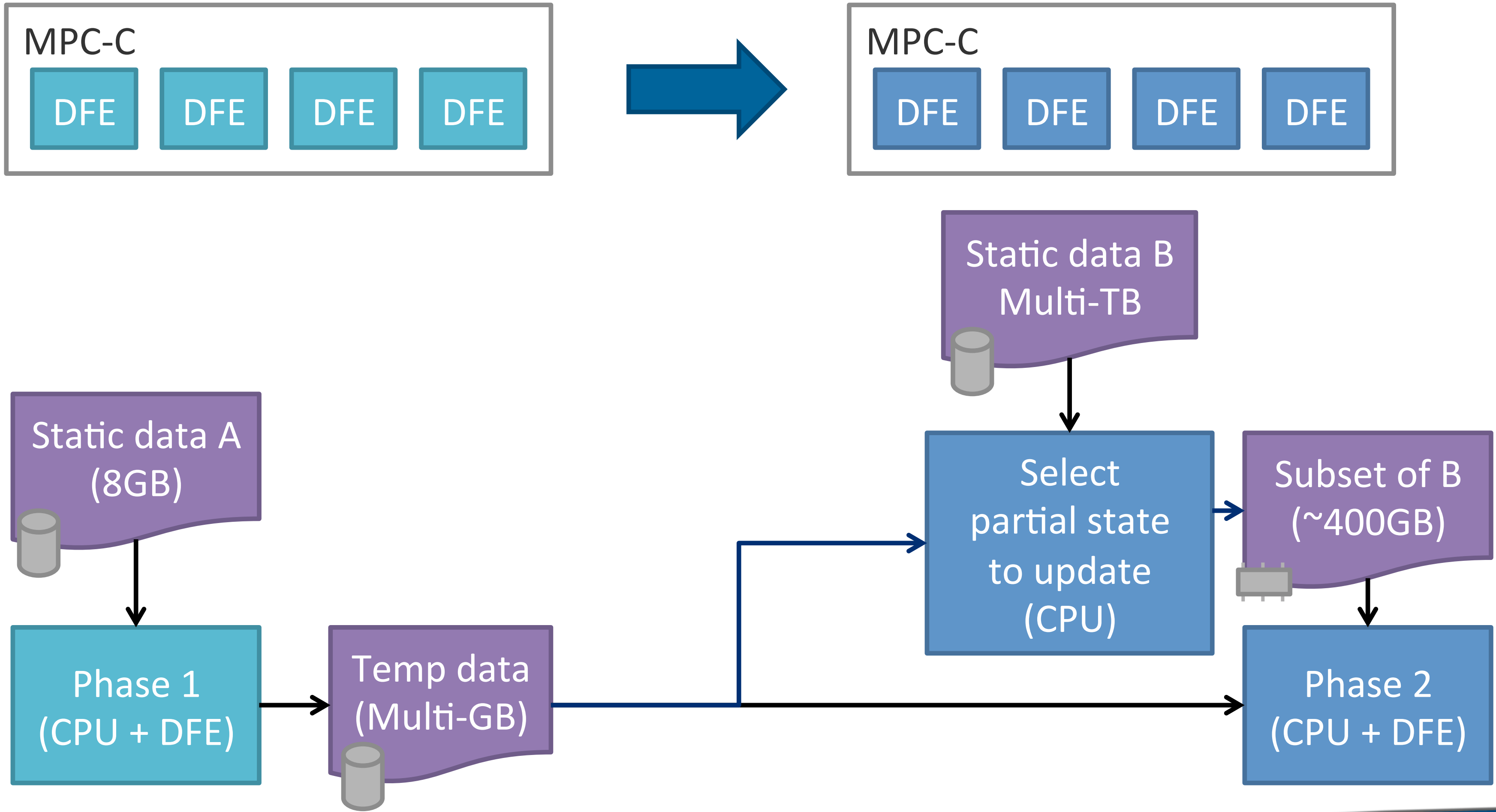
Case Study: Geoscience Application

- Original runs on high end 16-core x86 servers
- Highly optimized production code

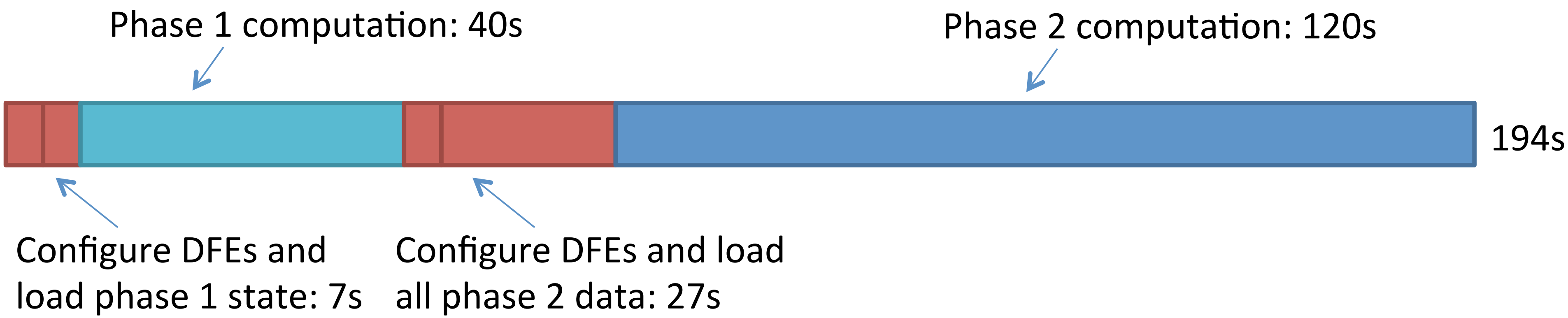


Dataflow Implementation: MPC-C

- Reload DFEs between phase 1 and phase 2



Runtime impact: MPC-C with 4 DFEs



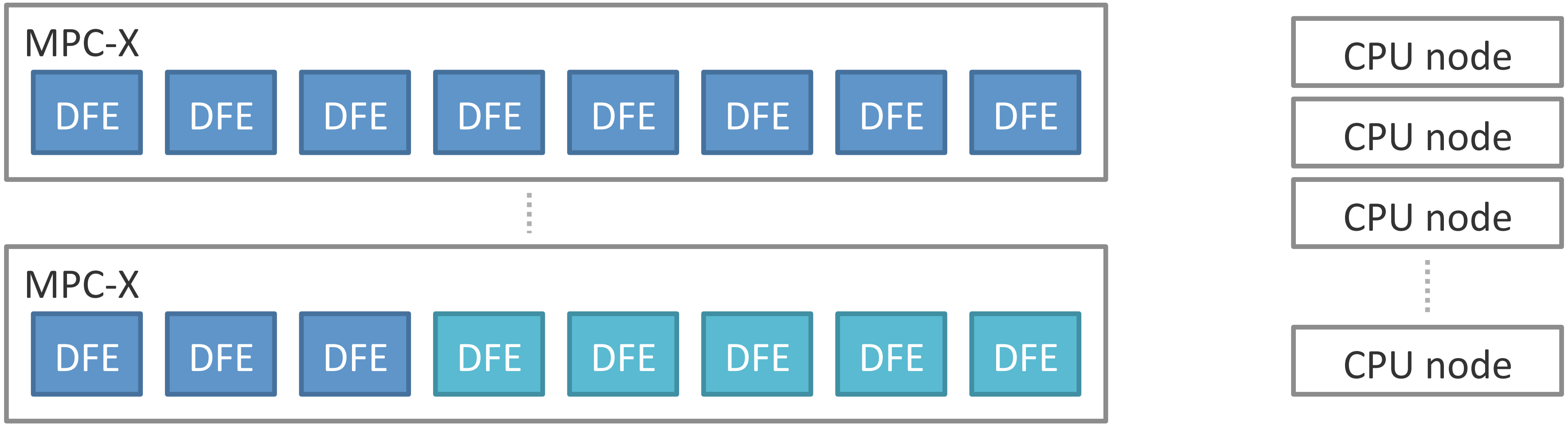
- DFEs are under-utilized
- Disk + CPU limited
- 17% of time in initialization

	DFE Potential	Actually achieved
Phase 1	19x	10x
Phase 2	20x	15x
Overall	~20x	**11x**

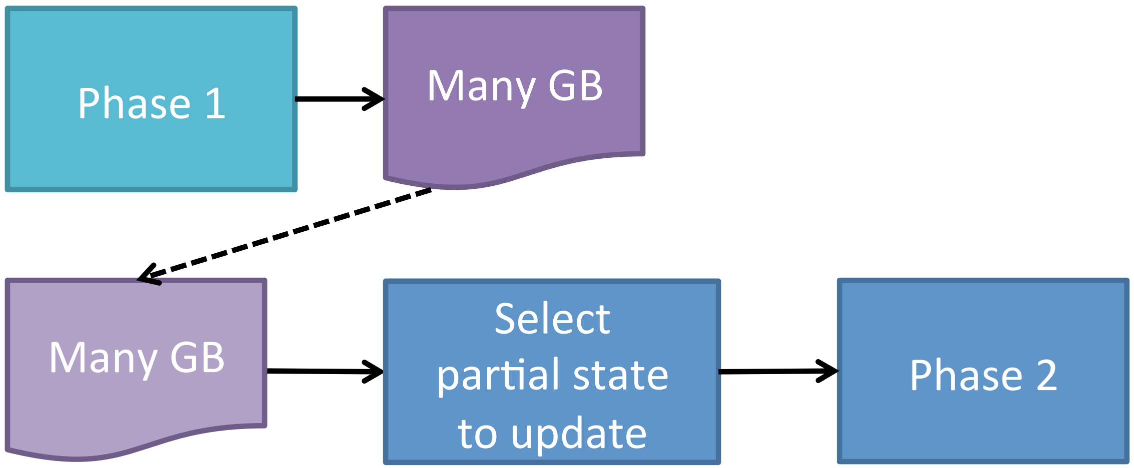
DFE under-utilization

- Increase CPU power per node?
  - 20% faster clock frequency possible but would still be limiting and costs a lot of power
- Increase disk speed per node?
  - Already using high speed disks, not possible to add more drives in the form factor
- Switch to SSDs
  - Continuous write-read cycle poor for SSD endurance
- Decrease number of DFEs per node
  - "Ideal" number of DFEs differs for different phases & based on individual job characteristics

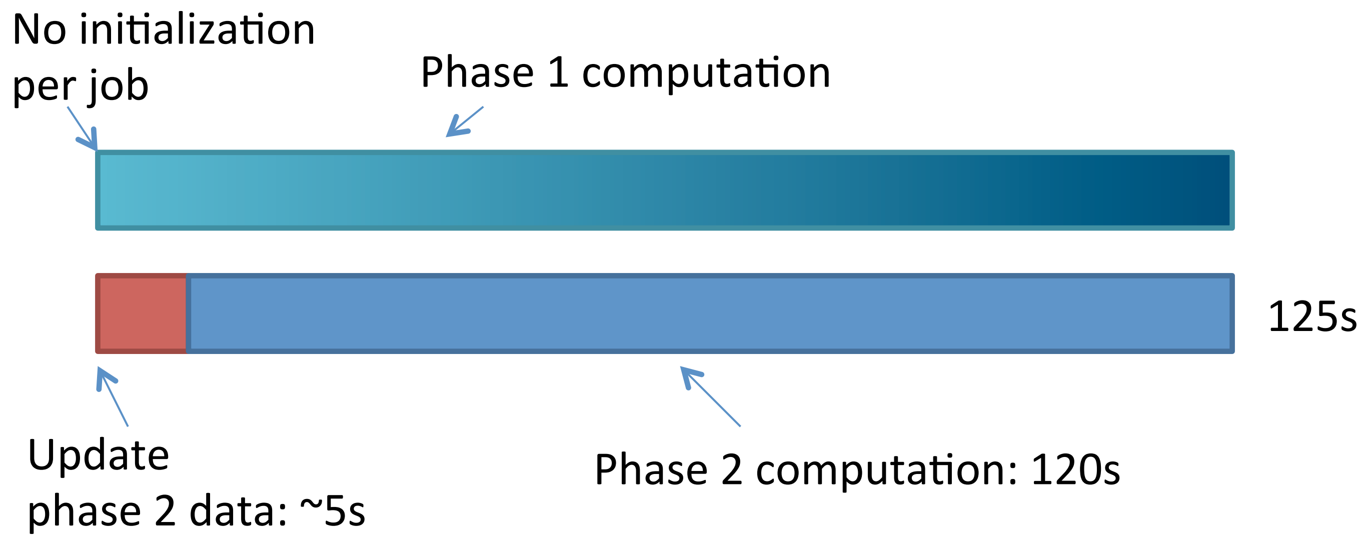
Dataflow Implementation: MPC-X



- Don’t reconfigure DFEs
- Allocate N DFEs per CPU node for Phase 2
- Allocate M DFEs per *cluster* for Phase 1
- *Vary N, M as workload characteristics change*
- Overlap Phase 1 of job J with Phase 2 of job J-1



Runtime impact: MPC-X



- Performance set by speed of phase 2 computation, phase 1 requires only ~0.6 DFE
- More CPUs accompany each MPC-X, but each node is less expensive (less memory needed)

CPU nodes per MPC-X	2.2
Speedup per MPC-X	**38.9x**

Summary

- Optimizing utilisation is key to maximizing performance
- Sharing physical resources can improve runtime
- Intelligent resource balancing can maximize overall system output
- Intelligent task dispatching should consider resource heterogeneity