Introduction to High Performance computing

Dr Etienne de Villiers

KEMRI Wellcome Trust Research Programme
Kilifi, Kenya

KEMRI | Wellcome Trust

Why High Performance Computing?

- Science and engineering needs
  - Replace complicated physical experiments by computer simulations
  - Evaluate more fine-grained models
- User requirements
  - Compute masses of individual tasks
  - Compute complicated single tasks
- Available computational power
  - Single workstation is not sufficient

What Is It?

- High Performance Computing (HPC)
  - Networking and Storage
    - Deals with high and highest performance computers, with high speed networks, and powerful disk and tape storage systems
  - Performance improvement
    - Compared to personal computers and small workstations:
    - Factor 100...10.000

What Do I Need?

- **Small scale high performance** computing
  - Cheapest version: use what you have
  - Workstations with disks and network
- A bit more expensive: buy PCs
  - E.g. 16 personal computers with disks and gigabit ethernet
  - It´s mainly a human resources problem
  - Network of workstations is time consuming to maintain
  - Software comes for free

What Do I Need...?

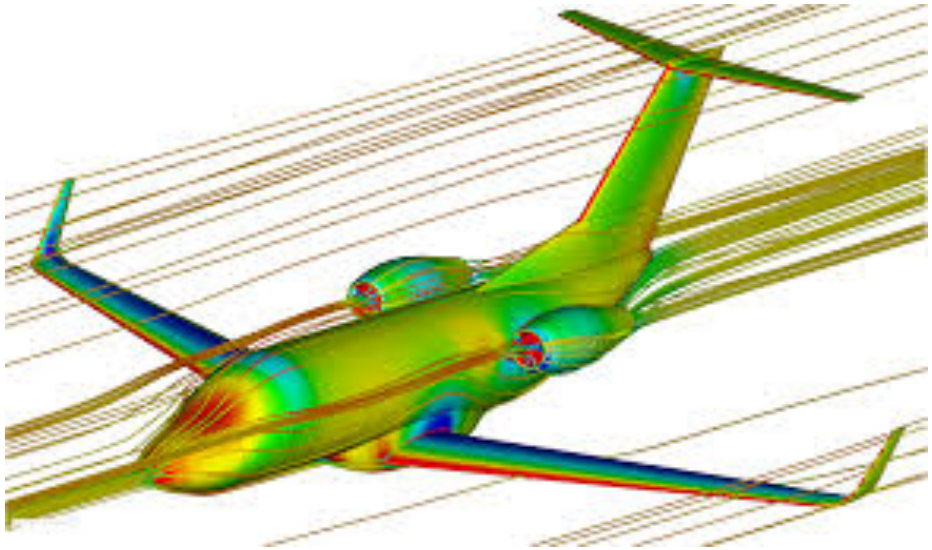
- **Large scale high performance** computing
  - Buy 10.000 PCs or a dedicated supercomputer
  - Buy special hardware for networking and storage
  - Add a special building
  - Add an electric power station

How Much Do I Have to Pay?

- Small scale (<64 nodes)
  - 1000€/node
- Medium scale (64-1024 nodes)
  - 2000€/node (multiprocessor, 64-bit)
  - 1500€/node for high speed network
  - 500€/node for high performance I/O
- Large scale (>1024 nodes)
  - Money for building
  - Money for power plant
  - Current costs range between 20...400 million Euros

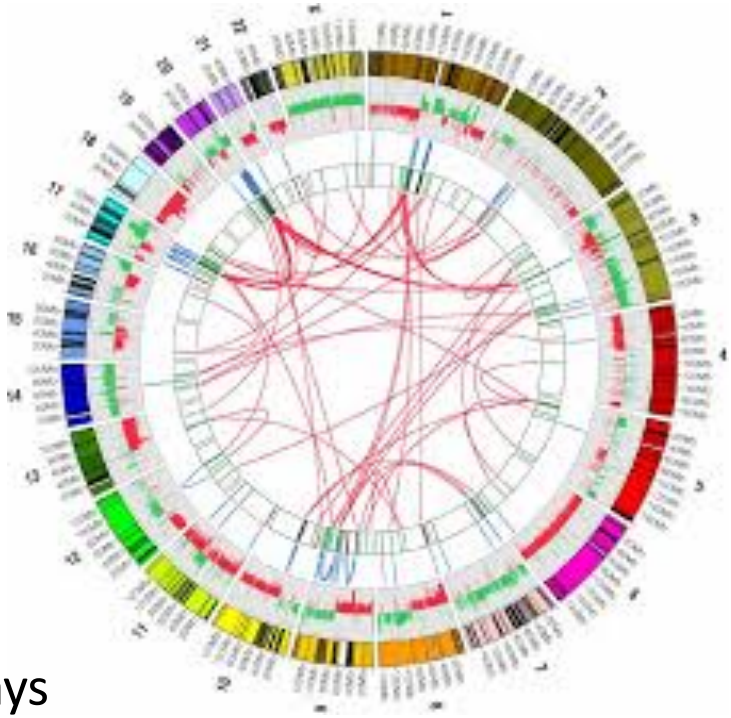
Application Fields

- Numerical calculations and simulations
  - Particle physics
  - Computational fluid dynamics
  - Car crash simulations
  - Weather forecast
  - Commercial database applications
- Non-numerical computations
  - Chess playing, theorem proving
  - Commercial database applications



Application Fields…

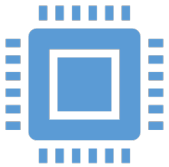
- All fields of Bioinformatics
  - Computational genomics
  - Computational proteomics
  - Computational evolutionary biology
  - …
- In general
  - Everything that runs between 1 and 10.000 days
  - Everything that uses high volumes of data



Measures

- Mega (10^6) – Giga (10^9) – Tera (10^{12}) – Peta (10^{15}) – Exa (10^{18})
- Computational Performance (Flop/s)
  - Flop/s = floating point operations per second
  - Modern processor: 3 GFlop/s
  - Supercomputer: 1.88 ExaFlop/s (factor 1,000,000,000)
- Network performance (Byte/s)
  - Personal computer: 10/100 MByte/s
  - Supercomputer networks: gigabytes/s

Measures...



Main memory (Byte)

Personal computer: 4 GByte

Supercomputer: 40 TByte

(factor 10,000)

Disk space (Byte)

Single disk 2004: 200 GByte

Supercomputer: PetaByte

(factor 1,000,000)

Tape storage (Byte)

Personal computer: 200 GByte

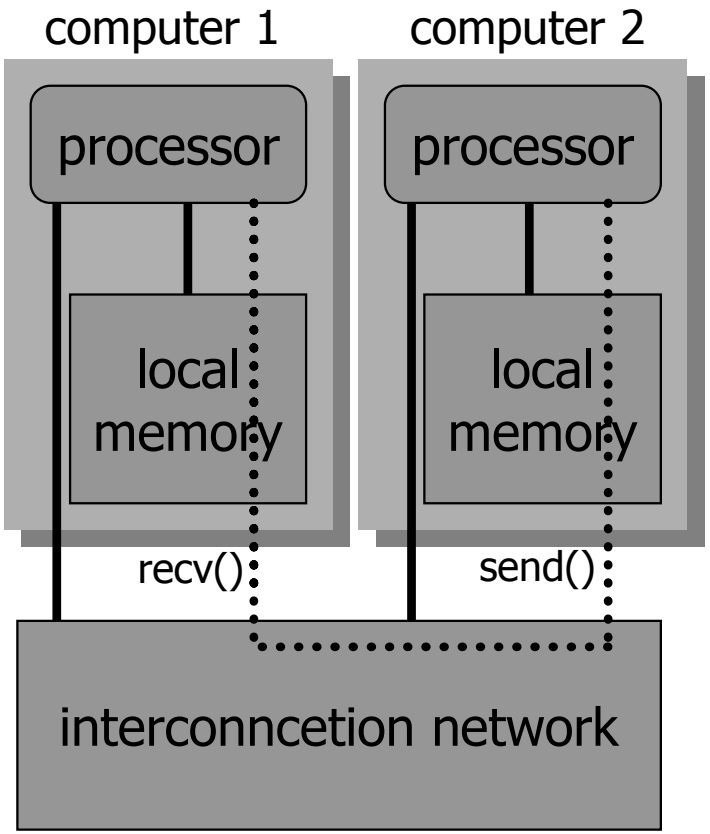
Supercomputer: 1.6 Pbyte

(factor 8,000)

Memory Architecture

- How is the main memory organized?
  - Distributed memory architecture
  - Shared memory architecture

Distributed Memory Architecture

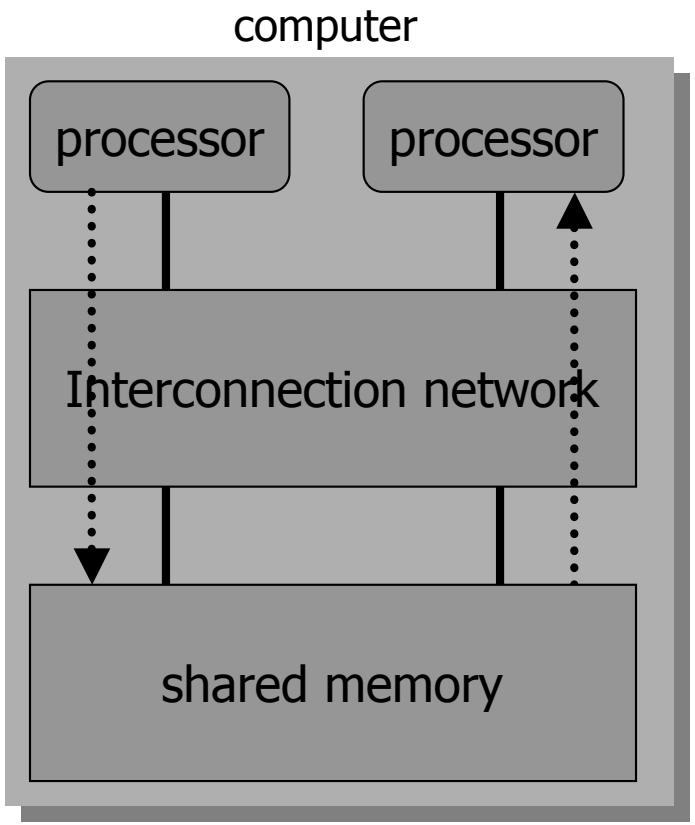


- Autonomous computers connected via network
- Processes on each compute node have access to local memory only
- Parallel program spawns processes over a set of processors
- Communication between computers (processes) via message passing
- Called: multi computer system

Distributed Memory Architecture...

- Advantages
  - Good scalability: just buy new nodes
    - Concept scales up to 10.000+ nodes
  - You can use what you already have
  - Extend the system when you have money and need for more power
- Disadvantages
  - Complicated programming: parallelization of formerly sequential programs (including complicated debugging, performance tuning, load balancing, etc.)

Shared Memory Architecture



- Several processors in one box (e.g. multiprocessor motherboard)
- Each process on a processor sees complete address space
- Communication between processes via shared variables
- Called: multiprocessor system, **symmetric multiprocessing system** (SMP)

Shared Memory Architecture...

- Advantages
  - Much easier programming
- Disadvantages
  - Limited scalability: up to 64 processors
  - Reason: interconnection network becomes bottleneck
  - Limited extensibility
  - Very expensive due to high performance interconnection network

Hybrid Architectures

Use several SMP systems

- Combination of shared memory systems and distributed memory system

The good thing: scalable performance according to your financial budget

The bad thing: programming gets even more complicated (hybrid programming)

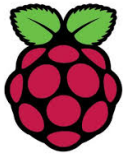
The reality: vendors like to sell these systems, because they are easier to build

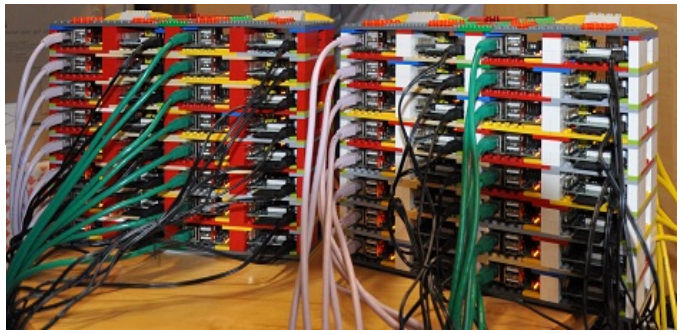
KWTRP and Pwani clusters are hybrid systems

Supercomputers vs. Clusters

- Supercomputers (Distributed/shared memory)
  - Constructed by a major vendor (IBM, HP, ...)
  - Use custom components (processor, network, ...)
  - Custom (Unix-like) operating systems

- Clusters (Network of workstations, NOWs)
  - Assembled by vendor or users
  - Commodity-of-the-shelf components (COTS)
  - Linux operating system

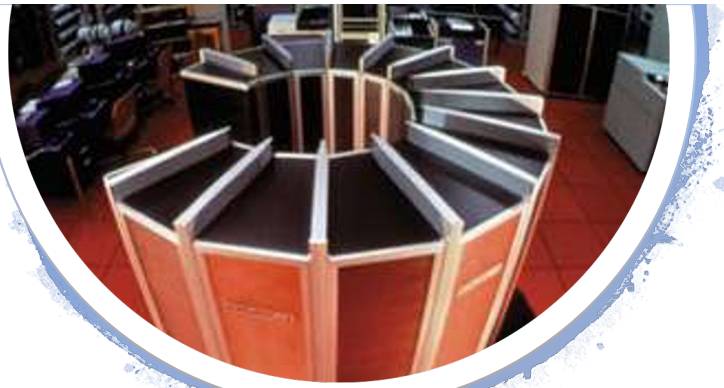




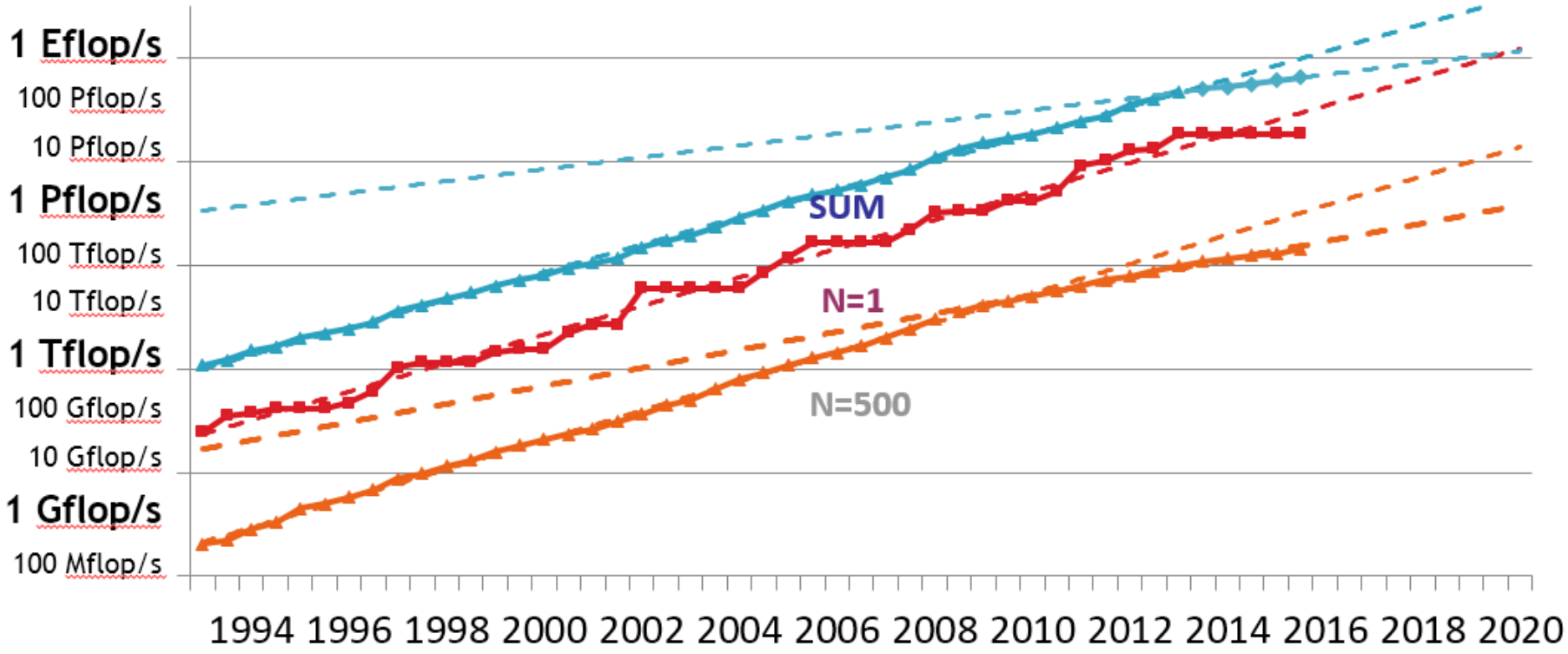
Supercomputers vs. Clusters...

- Supercomputers
  - Very expensive to buy
  - Usually high availability and scalability

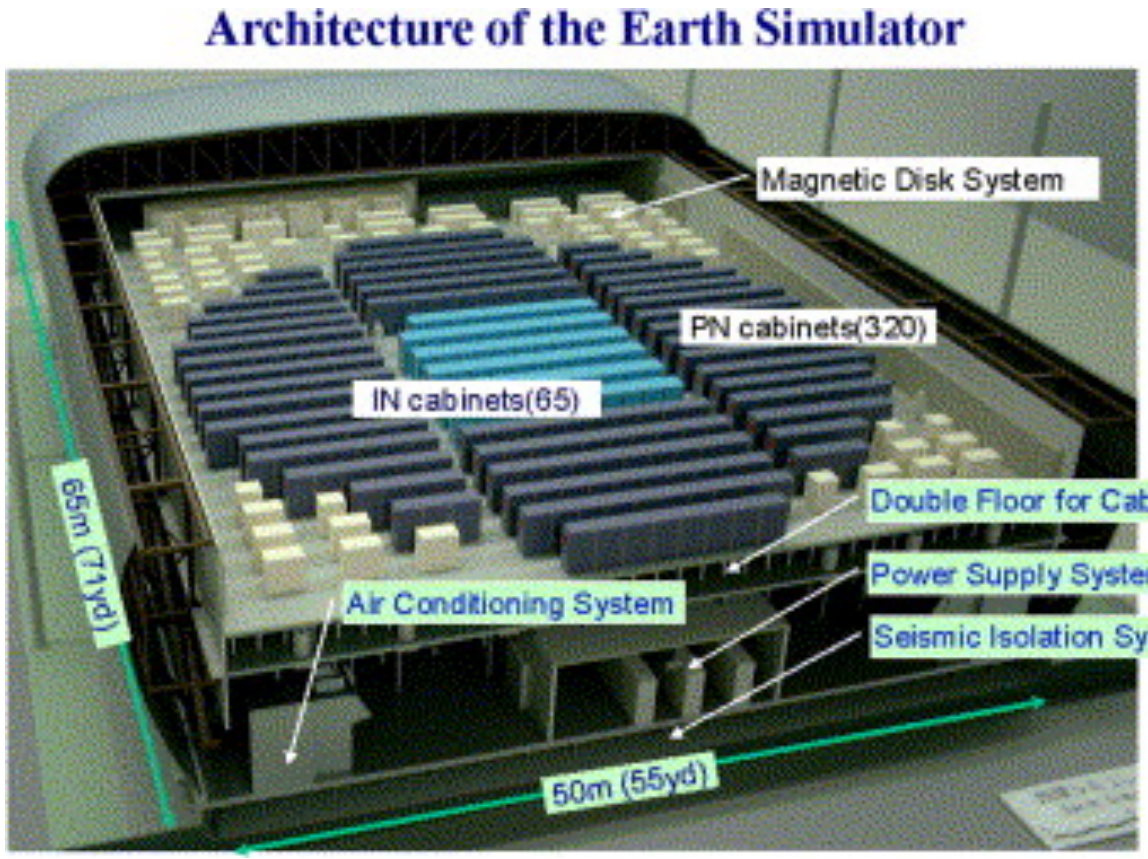
- Clusters
  - Factor 10 cheaper to buy, but:
  - Very expensive to own
  - Lower overall availability and scalability



PROJECTED PERFORMANCE DEVELOPMENT



EarthSimulator / NEC

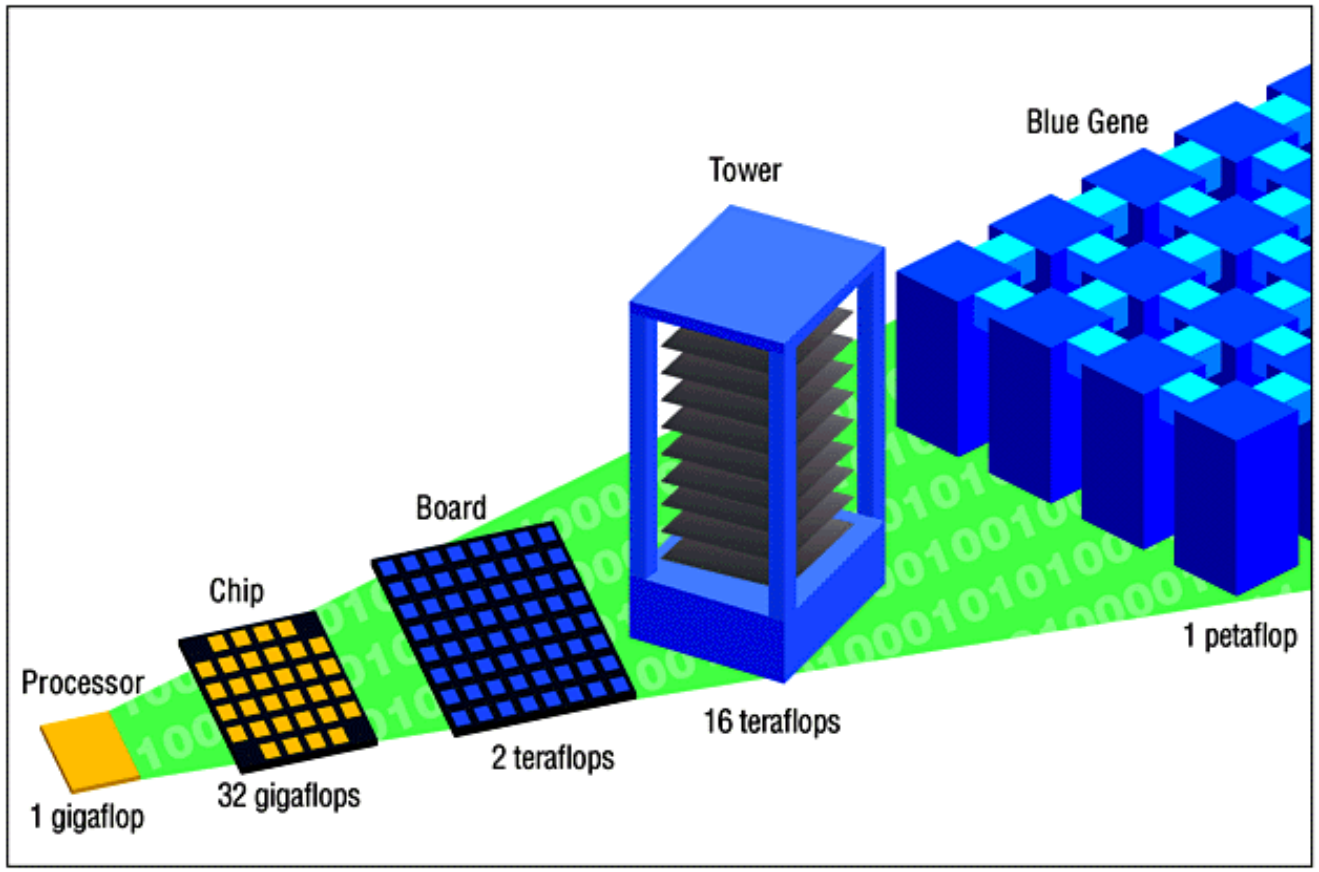


EarthSimulator / NEC

- 640 nodes x 8 processors
  = 5120 processors
- 200Mio USD for computer
- 200Mio USD for building and electric power station
- Building: 3250m2 earth quake protected
- Power: 7MW = 7,000 homes

- 10TByte main memory
- 700TByte disk
- 1.6PBytes tapes
- 83,000 copper cables
- 2,800km / 220t of cabling
- Application field: climate simulations

BlueGene / IBM



BlueGene / IBM

- 1PFlop/s
- Power consumption and floor space problems solved by new packing technology (less than 30 Earth- Simulators! ☺)

- Application fields:
  - Bioinformatics
  - *Ab initio* protein folding
  - Molecular dynamics on a millisecond to second time scale
  - Protein structure prediction

Summit (supercomputer)

- IBM latest supercomputer
- November 2018 is the fastest supercomputer in the world, capable of 200 petaflops
- First supercomputer to reach ExaFlops (exa 10^{18} operations per second) speed, achieving 1.88 ExaFlops during a genomic analysis

MareNostrum IV

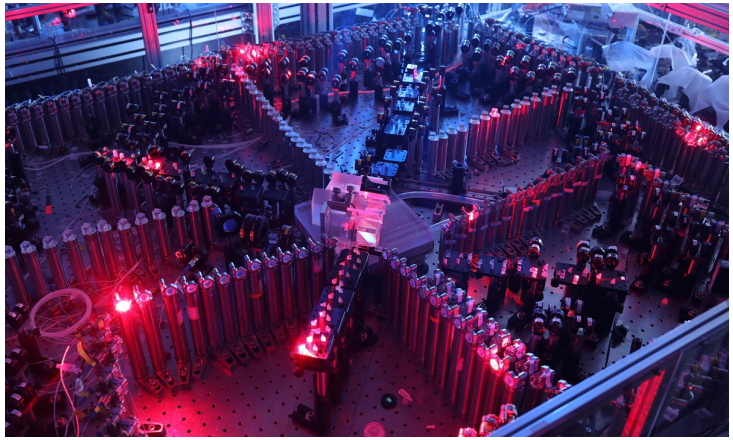
- European HPC in Barcelona
- Peak Performance of 11.15 Petaflops
- 384.75 TB of main memory
- 3,456 cores
- 14 Petabytes disk storage

EMBL – EBI HPC

- Based at Sanger Centre, Cambridge
- 330 TFLOPS
- 14,000 CPU cores
- 90 TB total RAM/memory
- 40 PB Data storage
  - doubling each 18 months

Quantum computing

- A **quantum** computer **works** with particles that can be in superposition. Rather than representing bits — such particles **would** represent qubits, which can take on the value 0, or 1, or both simultaneously
- Chinese quantum computer took 200 seconds to compute a problem that will take a regular supercomputer a staggering 2.5 *billion* years to complete



Jiuzhang light-based quantum computer prototype

Types of computing problems

Tightly Coupled	Embarrassingly Parallel
1 + 1 = X	1 + 1 = X
X + 2 = Y	2 + 2 = Y
Y + 3 = Z	3 + 3 = Z
N + Z = W	N + N = W

Generic bioinformatics use case

Embarrassingly Parallel

1 + 1 = X

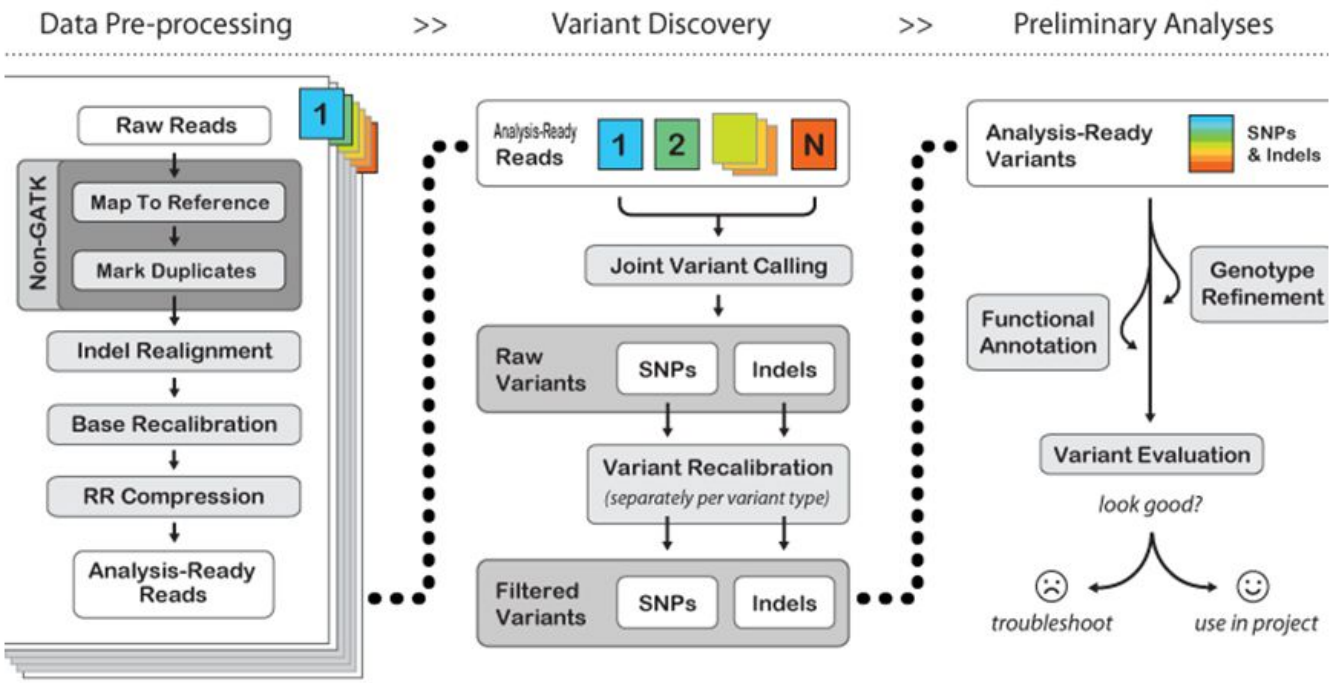
2 + 2 = Y

3 + 3 = Z

N + N = W

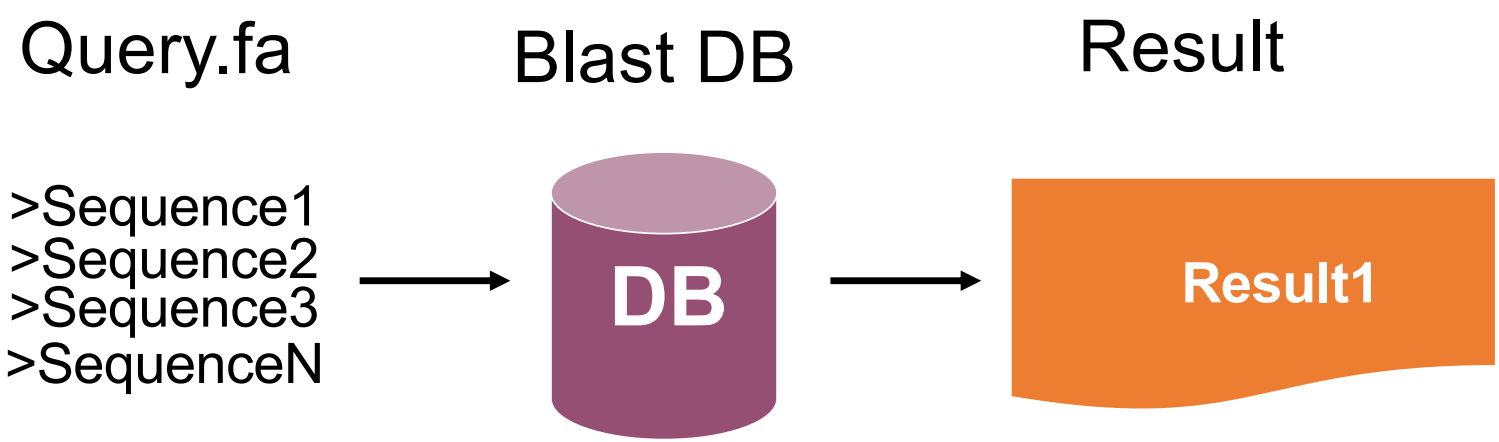
- Scientist has a problem
- She does not want or need to run ONE parallel application stretching across a 100 CPU cluster
- She wants to run one standalone program 100,000 or 1,000,000 times with slightly different input and output values

NGS data processing is embarrassingly parallel

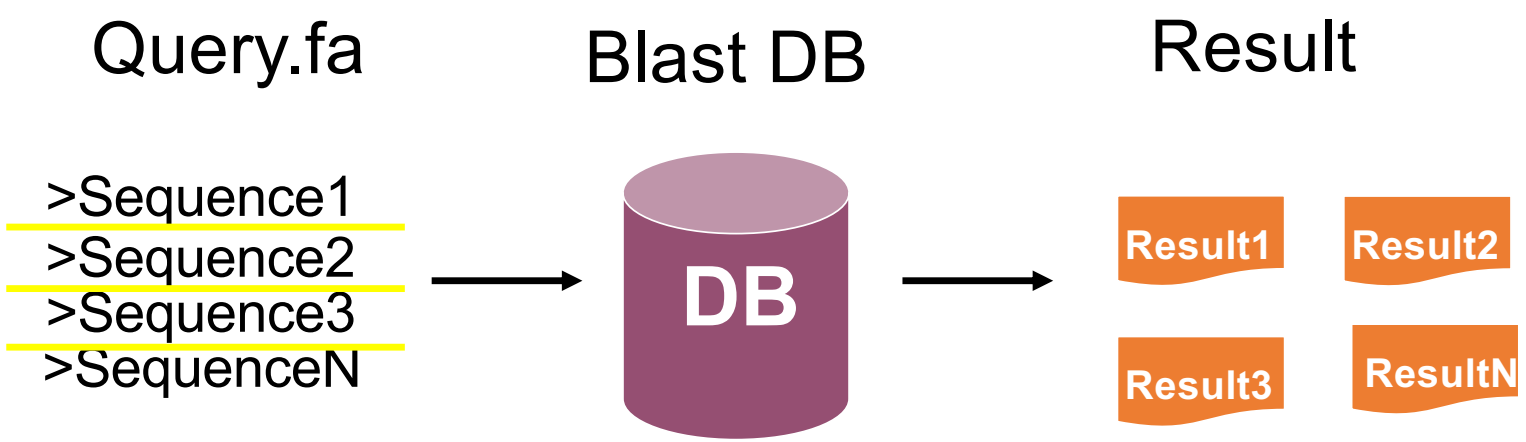


Taken from: http://www.broadinstitute.org/gatk/guide/best-practices

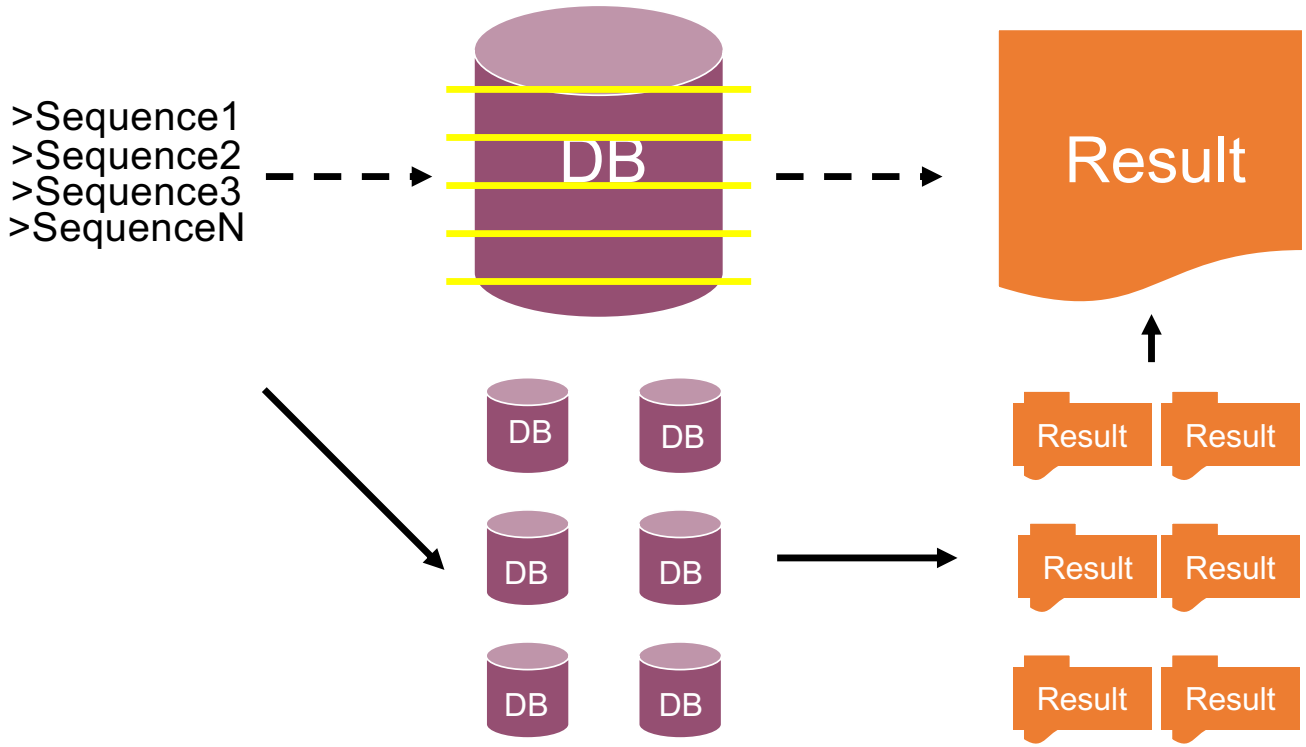
Example: BLAST use case



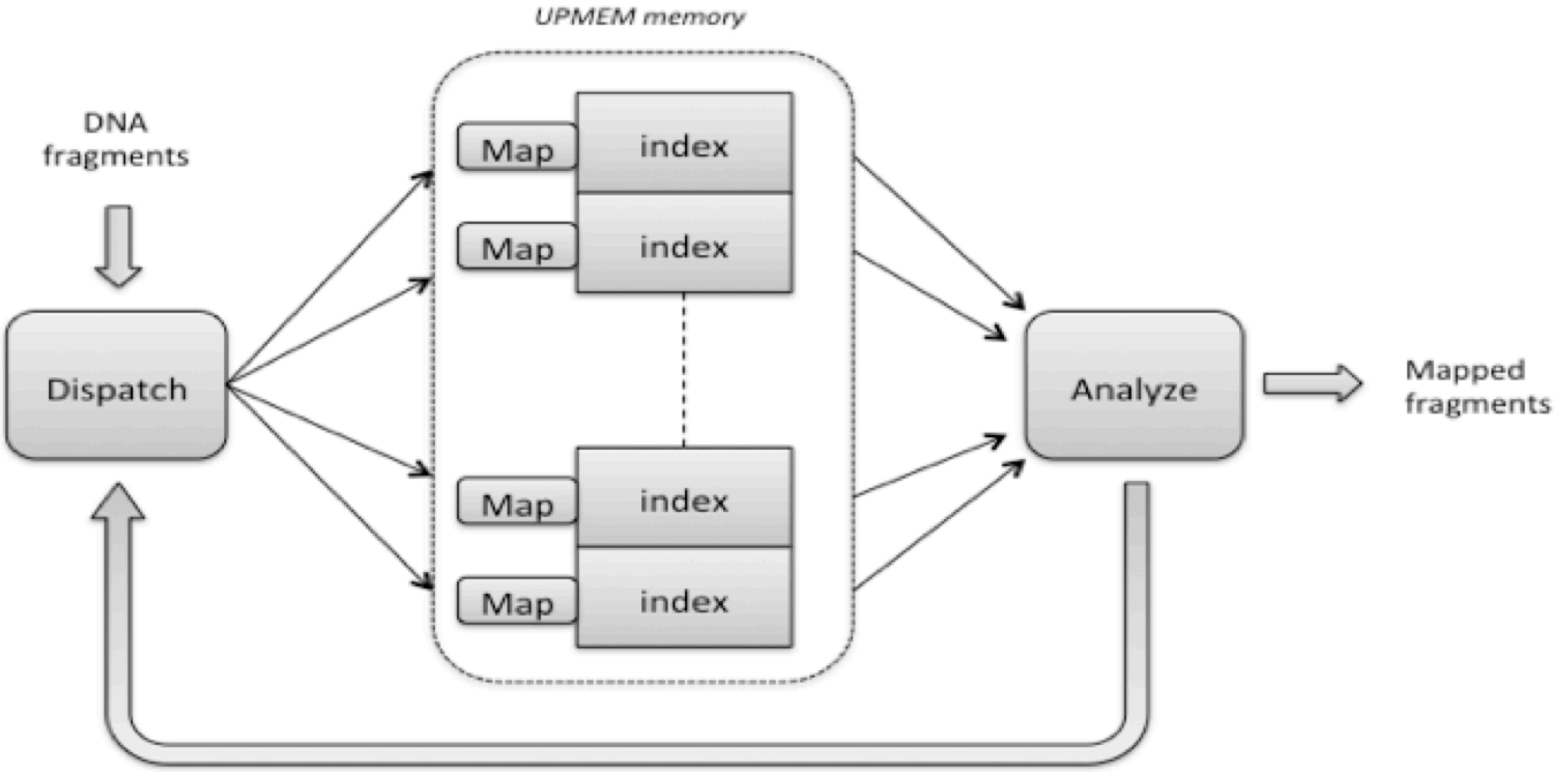
Split BLAST by query sequence



Split target databases & by query



Mapping DNA sequences to reference



Embarrassingly parallel tasks lend themselves well to "high performance computing"

Parallelization Paradigms...

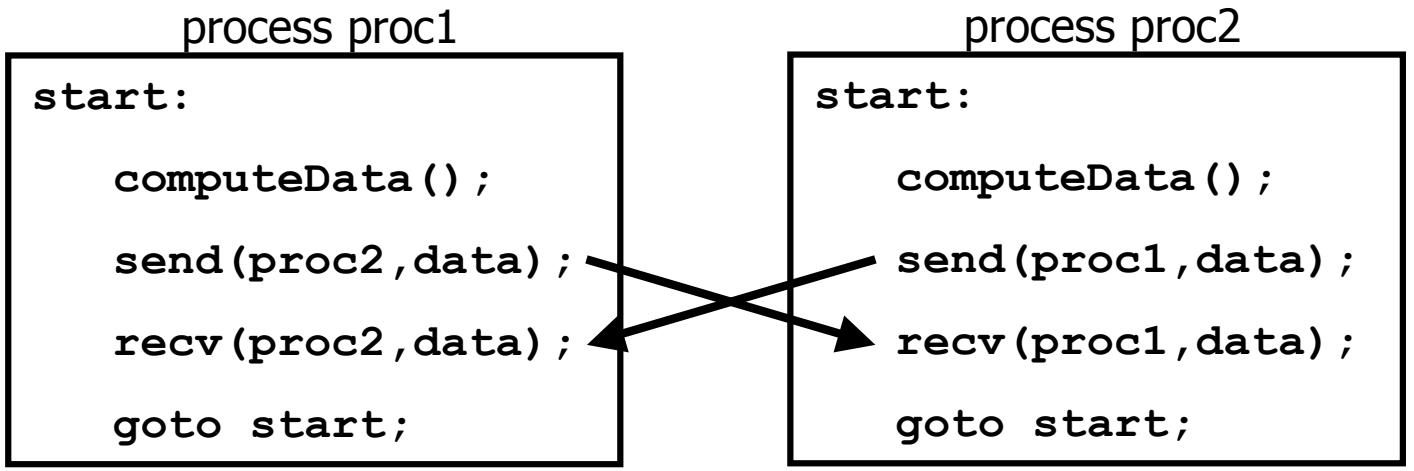
- Category 2: non-trivial parallelism
  - Usually: data partitioning
    - Data is partitioned amongst the processes
    - Each process computes results
    - During computation processes have to coordinate with each other: done by communication
    - A selected process organizes input/output
  - Example: Molecular dynamics simulations or genome assembly
  - Only for experienced programmers

Parallelization Support

- Parallelization via automatically parallelizing compilers
  - Complicated, seldom, inefficient, not scalable
- Parallelization for shared memory architectures
  - Language support via OpenMP
  - Only for small node numbers
- Parallelization for distributed memory architectures
  - Done manually
  - Decorate program with message passing calls

Message Passing

- Basic principle: use send and receive library calls to transfer data



Message Passing...

- Message passing standard:

  MPI (Message Passing Interface)
- Conceived in the mid 90s
  - Language bindings: C, C++, Fortran (others available)
- Divided into two parts
  - MPI: just message passing
  - MPI-2: process management, input/output, ...
- Available implementations
  - Vendor versions on all major supercomputers
  - Open source versions openMPI
- ABySS *de novo* sequence assembler uses openMPI

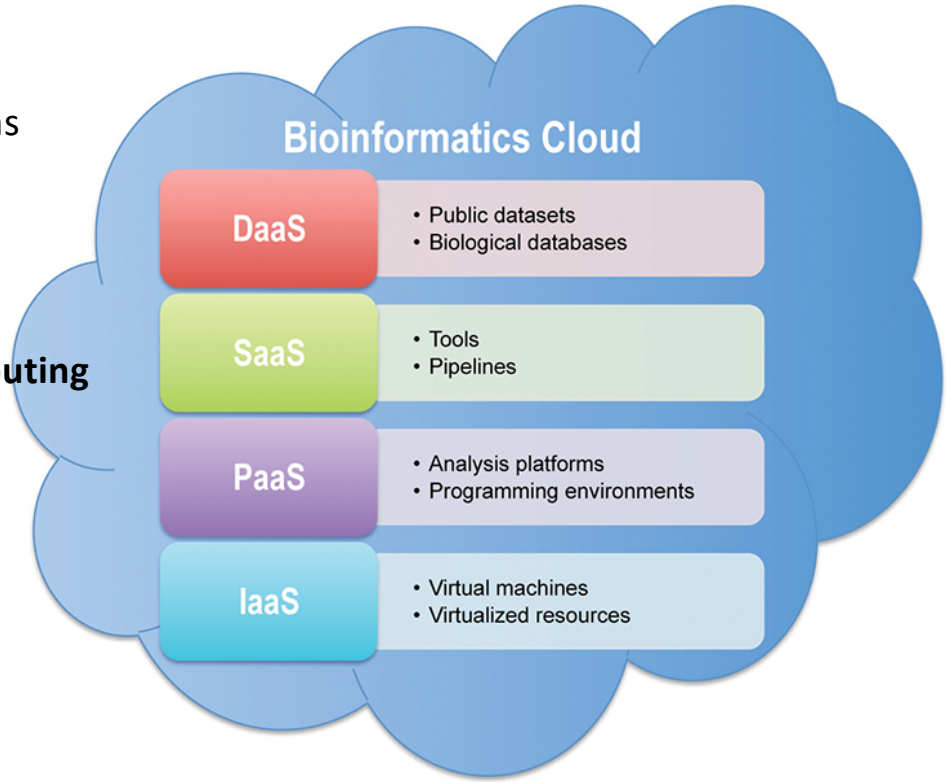
Cloud Computing

The Pros: Benefits and Advantages of Cloud Computing

- Provides Cost Savings For Individuals and Organizations
- Location Independence Through Remote Access
- Scalability and Flexibility of IT Capabilities
- All the Benefits of Outsourcing

The Cons: Limitations and Disadvantages of Cloud Computing

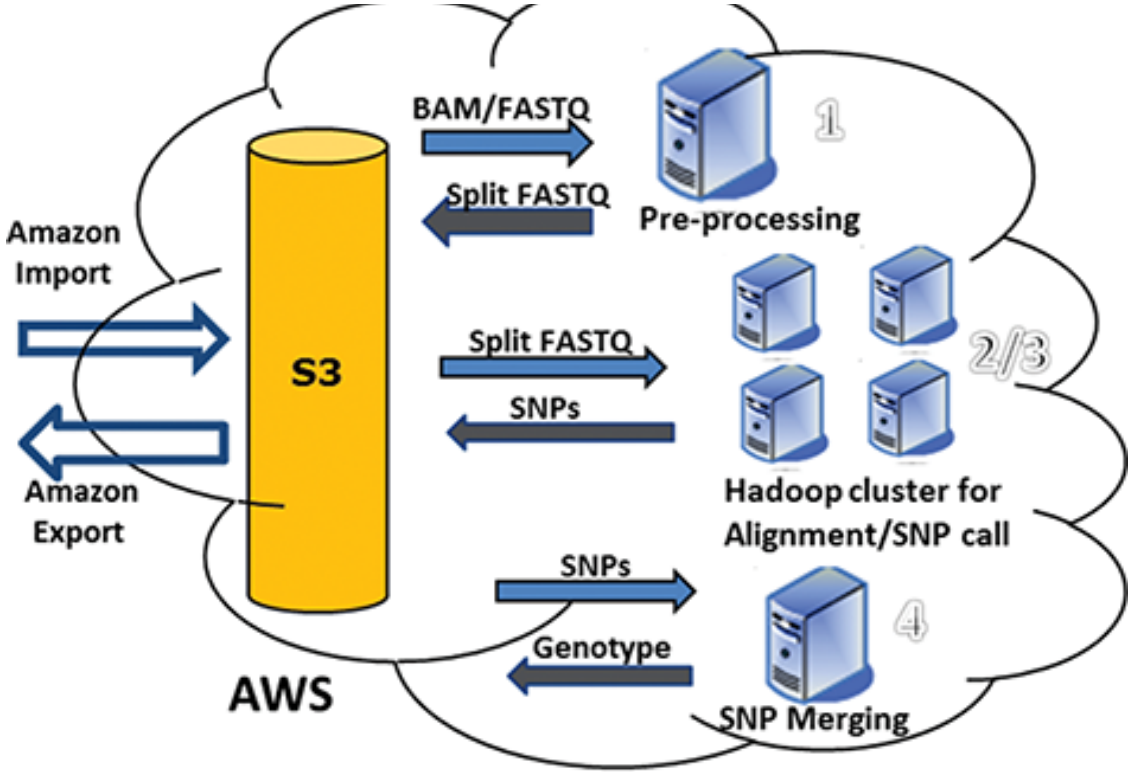
- Lack of Full Control Over IT Capabilities
- Issues Concerning Security and Privacy
- Dependence on Internet Connectivity
- Data motion and storage in the cloud can become very expensive



Rainbow AWS cloud

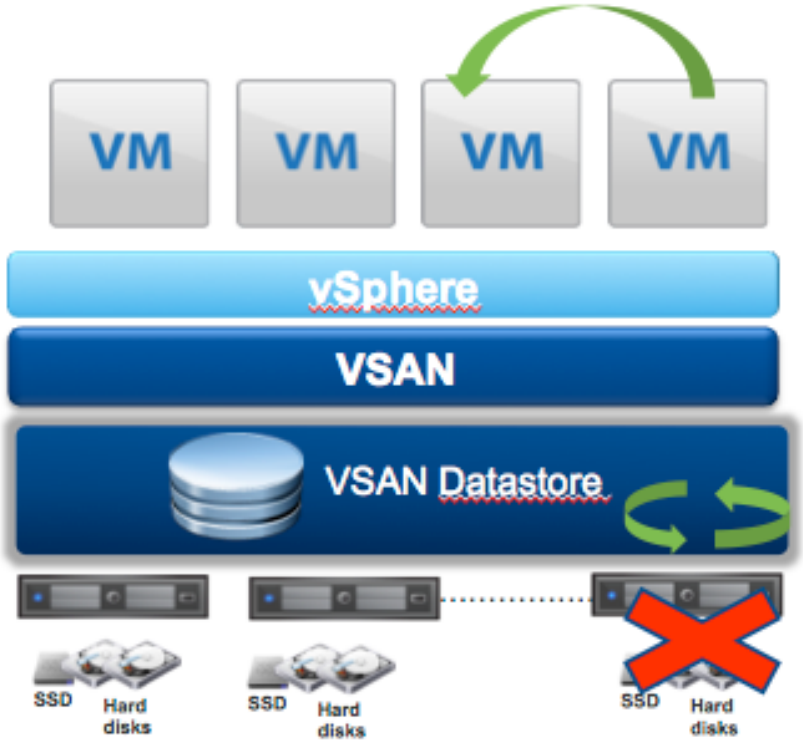
- **Rainbow: a tool for large-scale whole-genome sequencing data analysis using cloud computing**

- Detect SNPs in WGS data from a single subject.



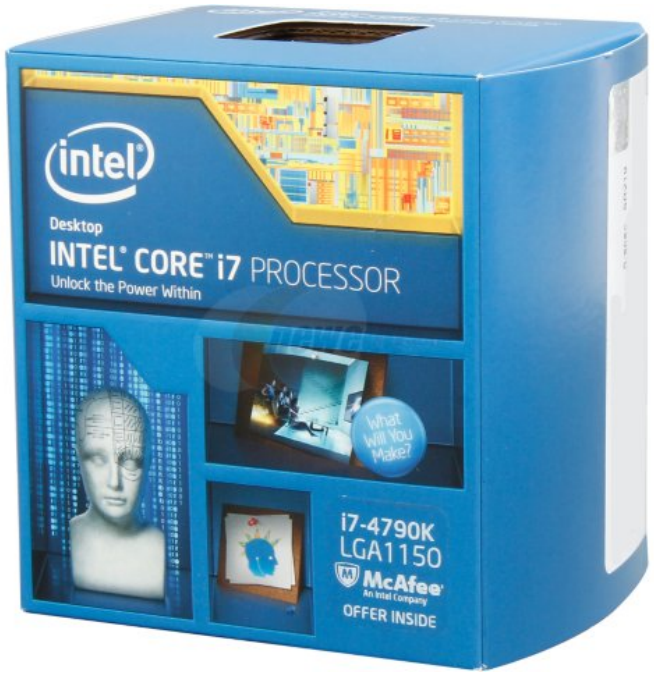
Hyper Converged Infrastructure

- Is an IT framework that combines **storage**, **computing** and **networking** into a single system.
- Run on commodity hardware.
- The future?



What is available on your desktop?

- Intel Core i7 4790 (Haswell) processor
- 4 cores at 4.0 GHz
- 8 double precision floating point operations per second (FLOP/s) – full fused multiply add instructions (FMAs)
  - 2 floating point units (FPUs) per core
  - 16 FLOP per cycle
- 64 GFLOP/s per core theoretical peak (4GHz)
- 256 GFLOP/s full system theoretical peak (4 cores)



High Performance Compute Clusters

- Group of servers connected with a dedicated high-speed network

- KWTRP HPC
  - 120 cpu/nodes
  - 2.5 GHz, 12 core Haswell server processors
  - 40 GFLOP/s per node theoretical peak
  - 4.8 TFLOP/s total system theoretical peak

- PUBReC HPC
  - 16 cpus/nodes
  - 3.3 GHz, 8 core Haswell server processors
  - 40 GFLOP/s per node theoretical peak
  - 0.3 TFLOP/s total system theoretical peak

KEMRI | Wellcome Trust

Pwani UNIVERSITY

High Performance Linux cluster Job management

Unmanaged

Managed

Computational Job

- A job is a computational task
  - that requires processing capabilities (e.g. 64 nodes) and is subject to constraints (e.g. a specific other job must finish before the start of this job)

- The job information is provided by the user
  - resource requirements
    - CPU architecture, number of nodes, speed
    - memory size per CPU
    - software libraries, licenses
    - I/O capabilities
  - job description
  - additional constraints and preferences

- The format of job description is not standardized, but usually very similar

Resource Management on HPCs

- HPCs are usually parallel computers or large-scale clusters
- The local resource management systems (RMS) for such resources includes:
  - configuration management
  - monitoring of machine state
  - job management

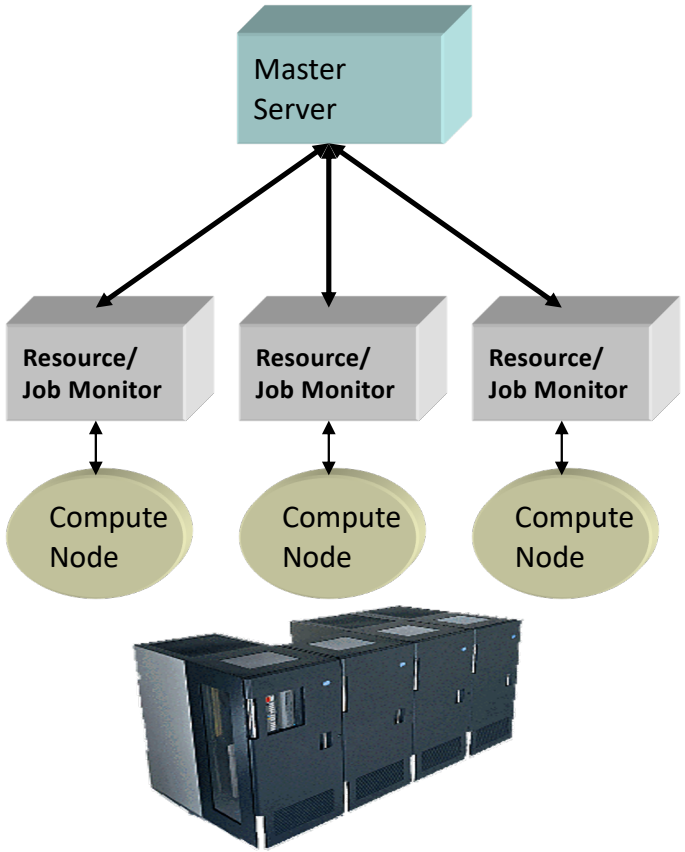
- There is no standard for this resource management.
- Several different proprietary solutions are in use.
- Examples for job management systems:
  - SLURM, PBS, Condor

HPC Management Architecture in General

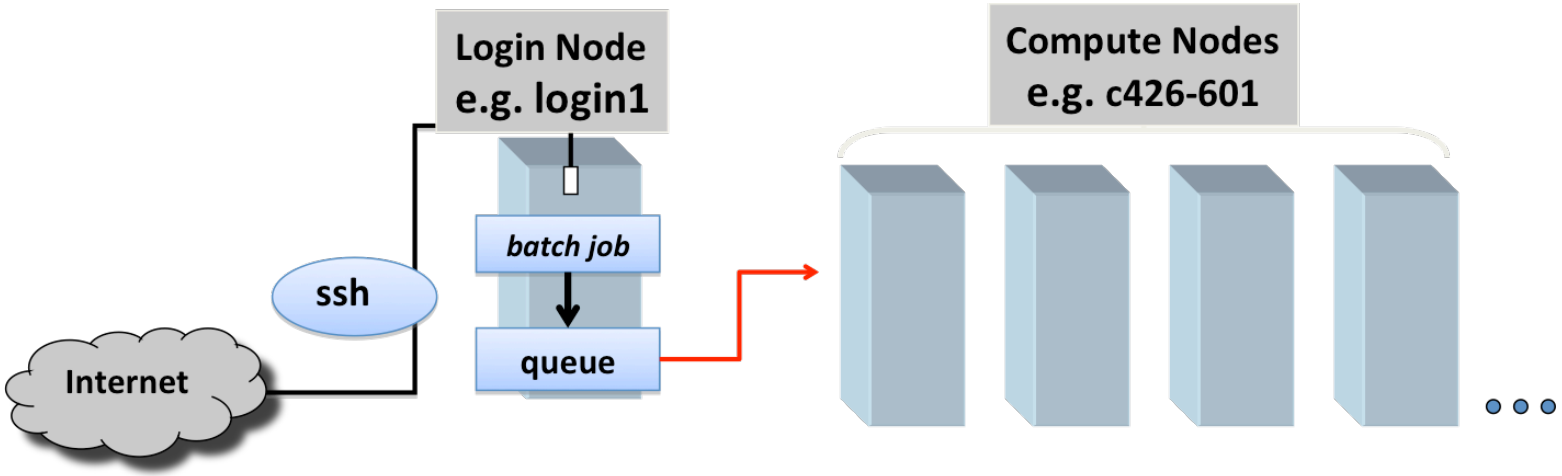
Control Service
Job Master

Resource and Job Monitoring
and Management Services

Compute Resources/
Processing Nodes



Job Submission with SLURM



SLURM workload manager

- Slurm is an open source, fault-tolerant, and highly scalable cluster management and job scheduling system for large and small Linux clusters
- Slurm has three key functions:
    1. Allocates exclusive and/or non-exclusive access to resources (compute nodes)
    2. Provides a framework for starting, executing, and monitoring work processes on the set of allocated nodes
    3. Arbitrates contention for resources by managing a queue of pending work

SLURM commands

- **sinfo -** reports the state of partitions and nodes managed by SLURM
- **smap** - reports state information for jobs, partitions, and nodes
- **sbatch** - is used to submit a job script for later execution
- **squeue** - reports the state of jobs or job steps.
- **srun** - is used to submit a job for execution in real time
- **scancel** - is used to stop a job early.

SLURM commands ….

- sinfo -lNe

```
NODELIST        NODES  PARTITION  STATE      CPUS  S:C:T  MEMORY TMP_DISK WEIGHT AVAIL_FE REASON
pubrec-bkup-01      1    batch    completing* 8    1:8:1      1        0      1   (null) none
pubrec-cls-01       1    debug*   idle        8    1:8:1      1        0      1   (null) none
pubrec-cls-01       1    batch    idle        8    1:8:1      1        0      1   (null) none
```

SLURM Queue Structure

Pubrec

PARTITION	TIMELIMIT	PURPOSE
debug*	5-23:59:00	interactive and/or development
batch	infinite	normal production

KWTRP

PARTITION	TIMELIMIT	PURPOSE
debug*	5-23:59:00	interactive and/or development
batch	infinite	normal production
longrun	infinite	queue for long-running jobs
highmem	infinite	large memory

Job Submission with SLURM

- The user “submits” the job to the RMS e.g. issuing

  “**`sbatch jobscript.pbs`**”

- The user can control the job
  - sbatch: submit
  - sinfo: poll status information
  - scancel: cancel job
- It is the task of the resource management system to start a job on the required resources
- Current system state is taken into account
- Can also submit interactive jobs
  - reserve CPU and memory for use

Example: SLURM Job Description

- Simple job script:

```
#!/bin/bash #
#SBATCH -n 10 # Number of cores
#SBATCH -p longrun
#SBATCH --mem-per-cpu 1024
#SBATCH --mail-user=evilliers@kemri-wellcome.org


#Commands to run. Replace example below with your arguments and
variables.


module load python/3.4.3
module load blastall/2.5.6


blastall –I query_seq.fa -d nr_database -p blastp
```

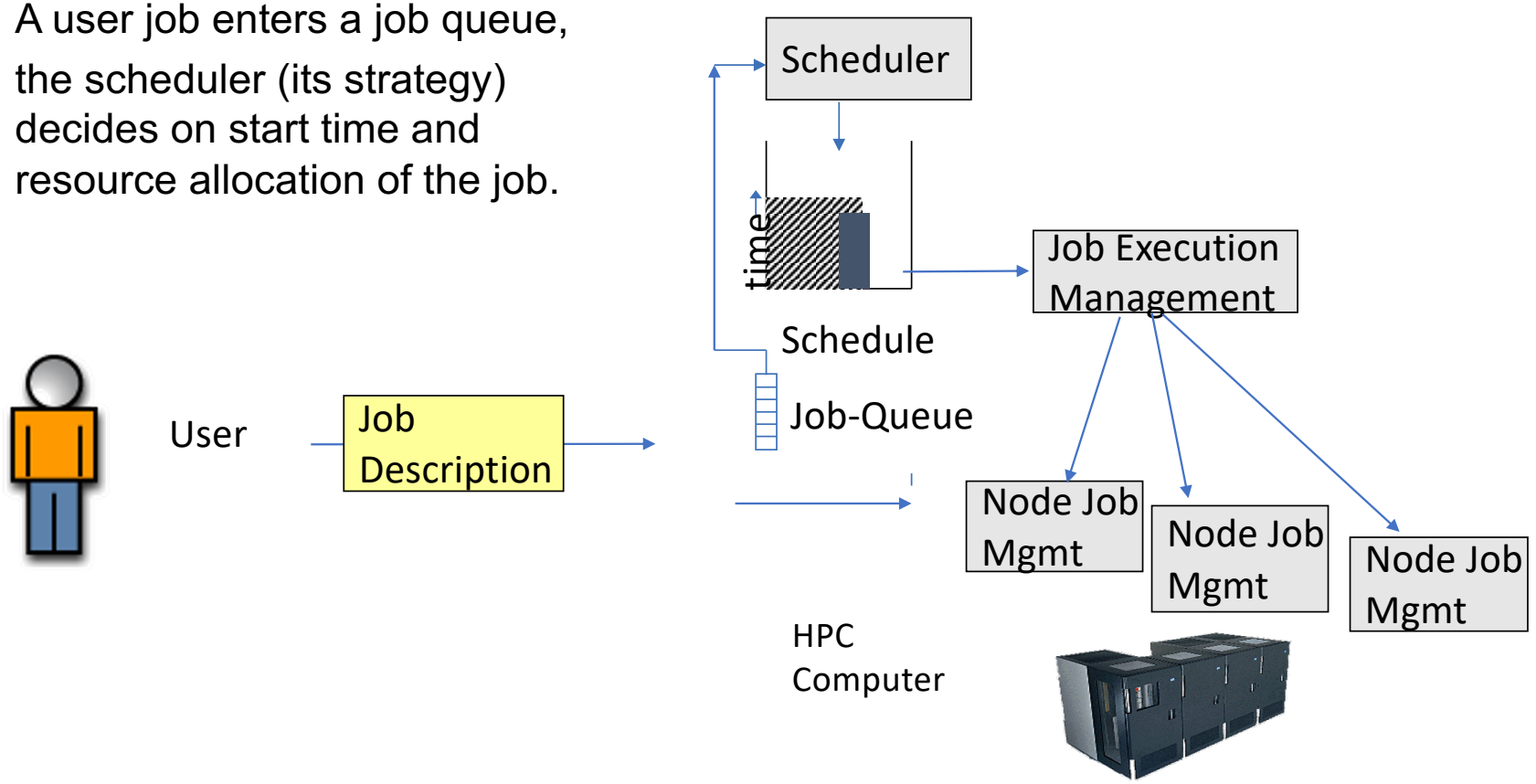
whole job file is a shell script

information for the RMS are comments

the actual job is started in the script

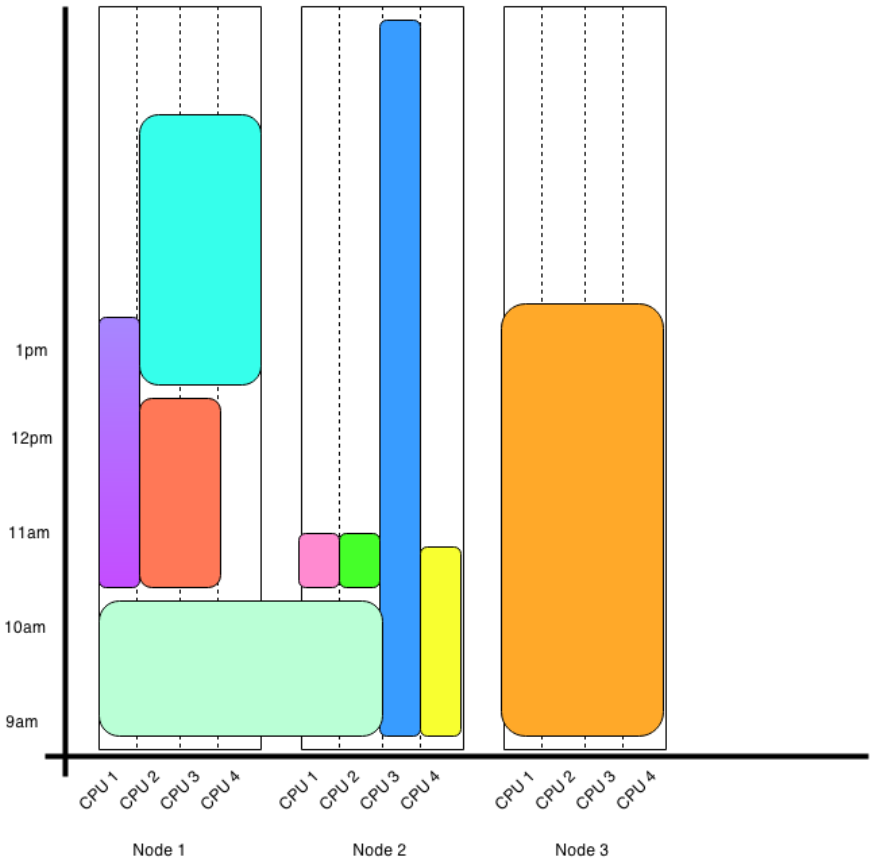
Job Steps

- A user job enters a job queue,
- the scheduler (its strategy) decides on start time and resource allocation of the job.



SLURM Job Scheduler

- **Priority factors**
  - age: how long the job has been waiting in the queue,
  - fair share: a measure of past usage of the cluster by the user,
  - size: the number of CPUs a job requests



Execution Alternatives

Time sharing:

- The local scheduler starts multiple processes per physical CPU with the goal of increasing resource utilization.
  - multi-tasking
- The scheduler may also suspend jobs to keep the system load under control
  - preemption

Space sharing:

- The job uses the requested resources exclusively; no other job is allocated to the same set of CPUs.
  - The job has to be queued until sufficient resources are free.

Job Classifications

- Batch Jobs vs interactive jobs
  - batch jobs are queued until execution
  - interactive jobs need immediate resource allocation
- Parallel vs. sequential jobs
  - a job requires several processing nodes in parallel

- The majority of HPC installations are used to run batch jobs in space-sharing mode!
  - a job is not influenced by other co-allocated jobs
  - the assigned processors, node memory, caches etc. are exclusively available for a single job.
  - overhead for context switches is minimized
  - important aspects for parallel applications

Preemption

- A job is preempted by interrupting its current execution
  - the job might be on hold on a CPU set and later resumed; job still resident on that nodes (consumption of memory)
  - alternatively a checkpoint is written and the job is migrated to another resource where it is restarted later

- Preemption can be useful to reallocate resources due to new job submissions (e.g. with higher priority)
- or if a job is running longer then expected.

Job Scheduling

- A job is assigned to resources through a scheduling process
  - responsible for identifying available resources
  - matching job requirements to resources
  - making decision about job ordering and priorities

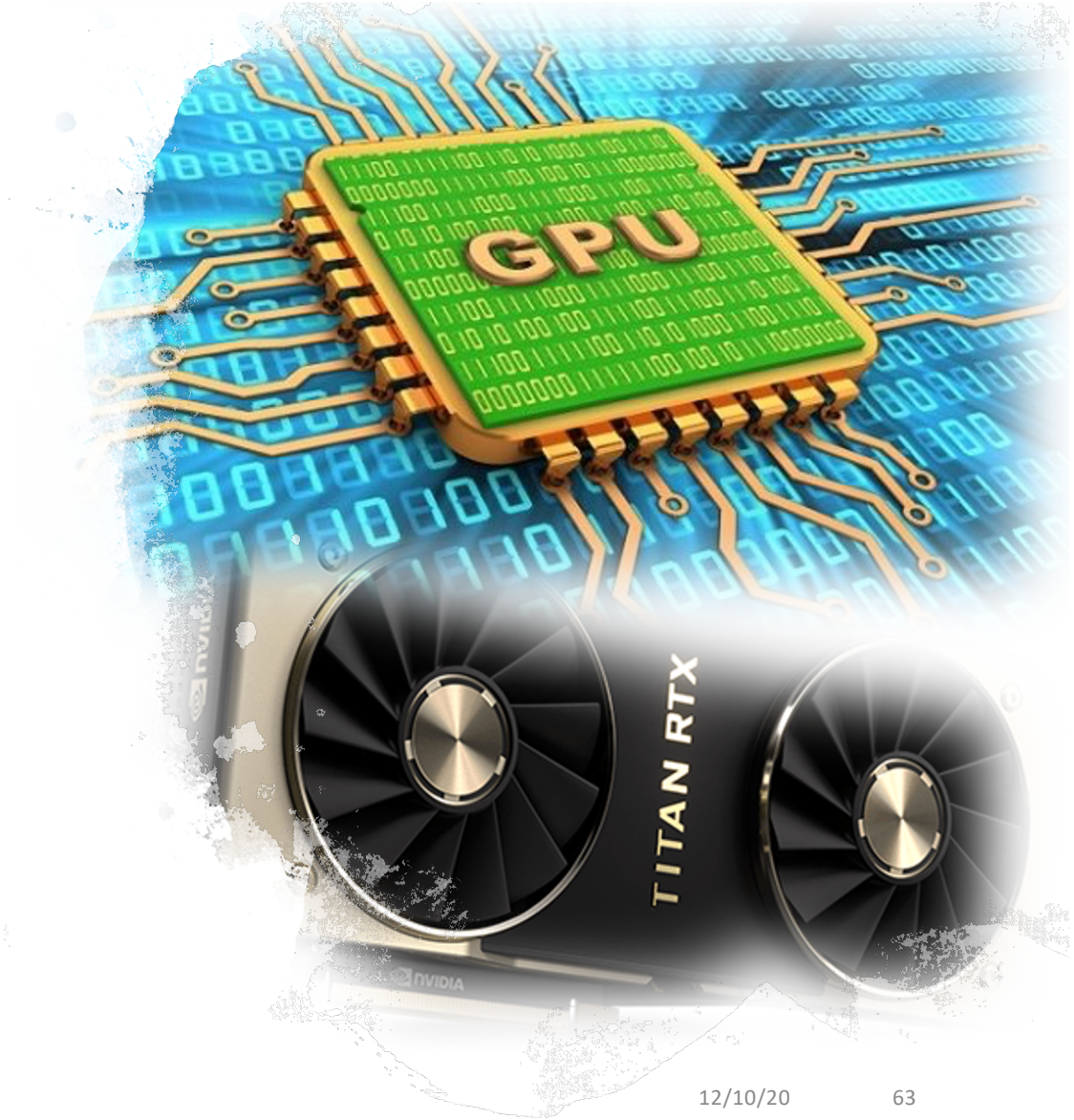
- HPC resources are typically subject to high utilization
- Therefore, resources are not immediately available and jobs are queued for future execution
  - time until execution is often quite long (many production systems have an average delay until execution of >1h)
  - jobs may run for a long time (several hours, days or weeks)

Accelerators and Many Integrated Cores

- Typical CPUs are not very energy efficient
  - Meant to be general purpose, which requires lots of pieces to the chip
- Accelerators package much more FLOP capabilities with less energy consumption
  - Not exactly general purpose
  - Always requires more work from the programmer
  - Performance improvements and application porting may take significant effort

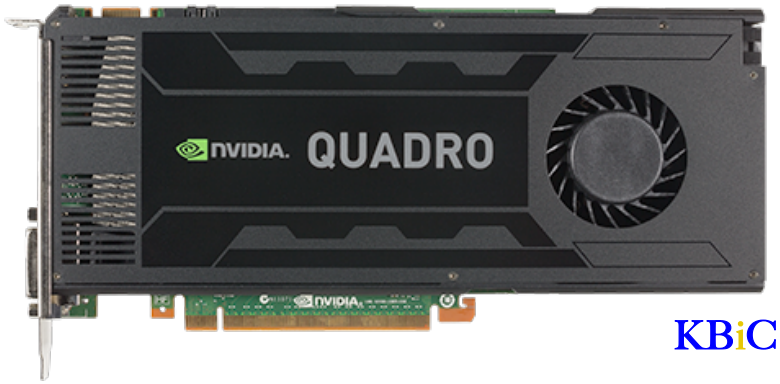
GPU Accelerators

- A graphics processing unit (GPU) as accelerators
  - NVIDIA GPUs can be used to perform calculations
  - Latest Turing generation offer 130 TFLOPs in a single package



GPU enabled software

- NVIDIA quadro K4200 GPU module (1344 CUDA Cores, 4GB GPU Memory) on KWTRP HPC
  - **CUDASW++:** Smith-Waterman protein database search to perform sequence searches 10x-50x faster than NCBI BLAST
  - **BarraCUDA**: bwa based short read aligner
  - **MUMmerGPU**: high-throughput parallel pairwise local sequence alignment program
  - **Guppy:** Oxford nanopore base calling



KBiC

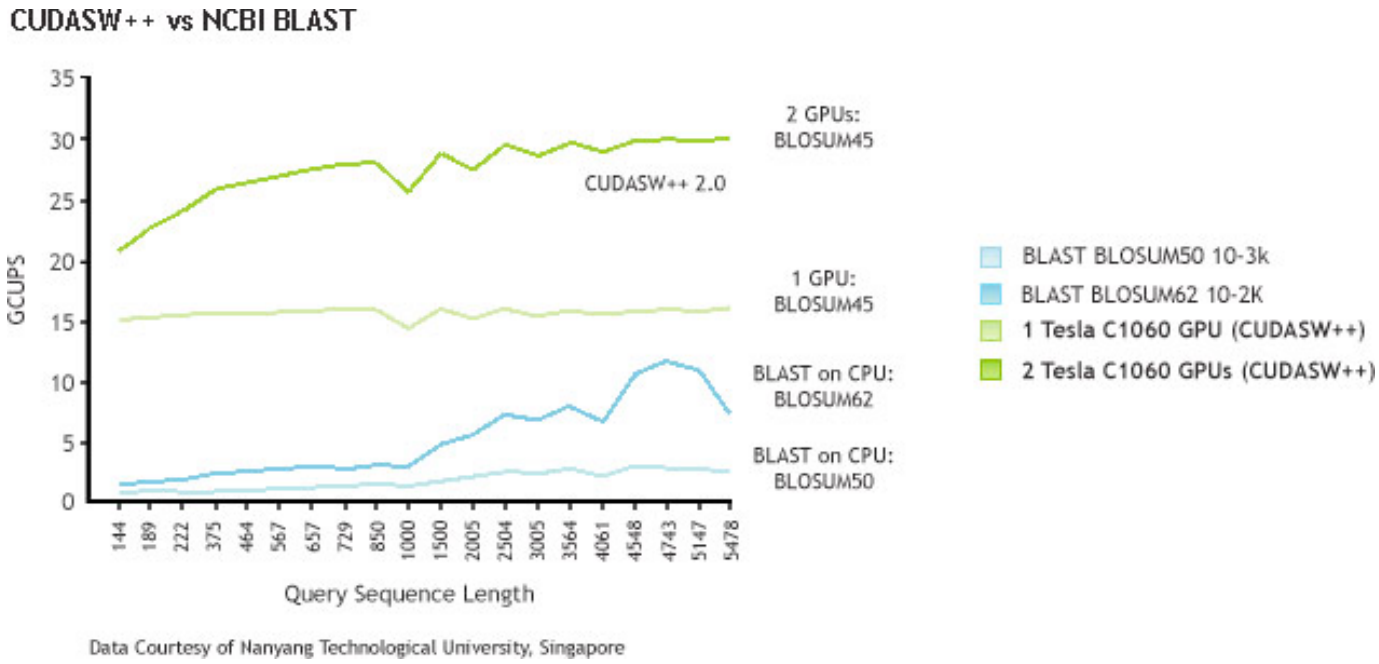
cudasw.sbatch

```
#!/bin/env bash
#SBATCH --job-name=cudasw
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=20
#SBATCH -p longrun
#SBATCH -w keklf-cls01
#SBATCH --gres=gpu:1
#SBATCH --mail-user evilliers@kemri-wellcome.org
#Command to run.

module load cudasw/v3.1.1
cudasw -query Queries/all.fasta -db uniprot_sprot.fasta -use_single 1
```

KBiC

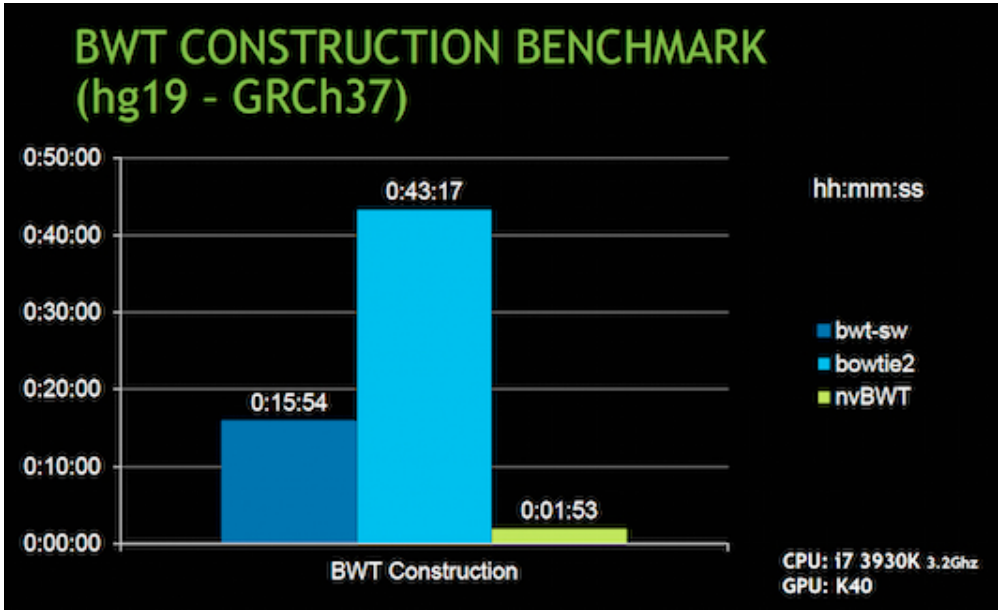
GPU-enabled Smith-Waterman



CUDASW++ compared to BLASTP

KBiC

GPU accelerated Bowtie2



- BWT-SW and Bowtie2 compared to nvBWT on human reference genome build *hg19*.
- nvBWT about 8X and 21X faster than BWT-SW and Bowtie2, respectively.

KBiC

Pubrec HPC server

16 CPUs
64 GB RAM
8 TB Disk storage

8 TB backup storage server

Centos 7 operating system
SLURM job scheduler

Access to Pubrec HPC

- Users access the Pubrec HPC server through ssh connection to their user accounts.
- User account will only be activated after a short induction training session.
- Only you have access to your directory and workspace.
- Currently hard disk space usage is not restricted.
  - However we will regularly check for duplication of large files and remove them.
    - Use soft links:

```
ctx.112.fastq -> /home/data/iontorrent/ctx_plasma/ctx.112.fastq
```

KBiC


Thank you