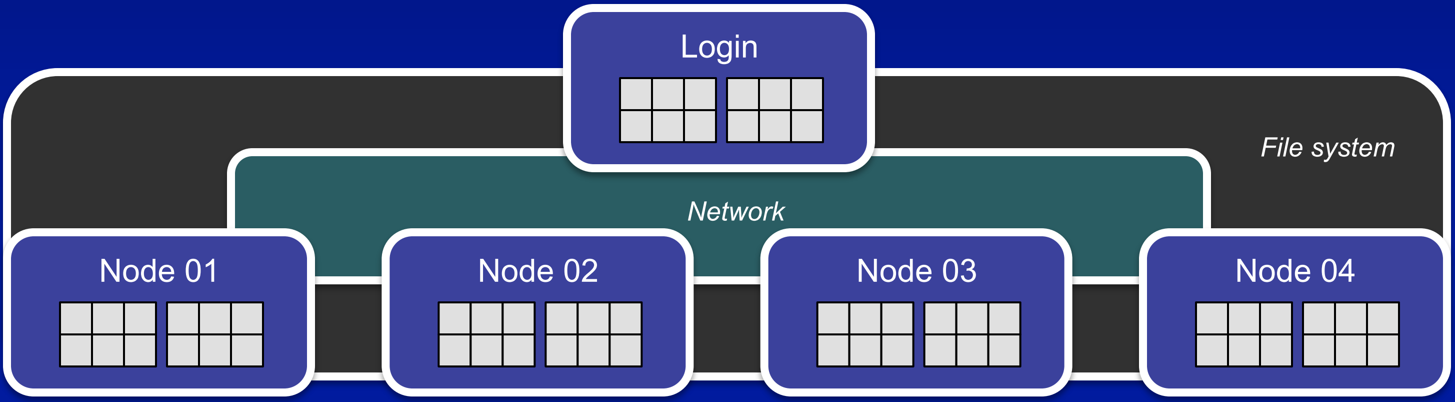
Simplified Supercomputer Architecture Slide

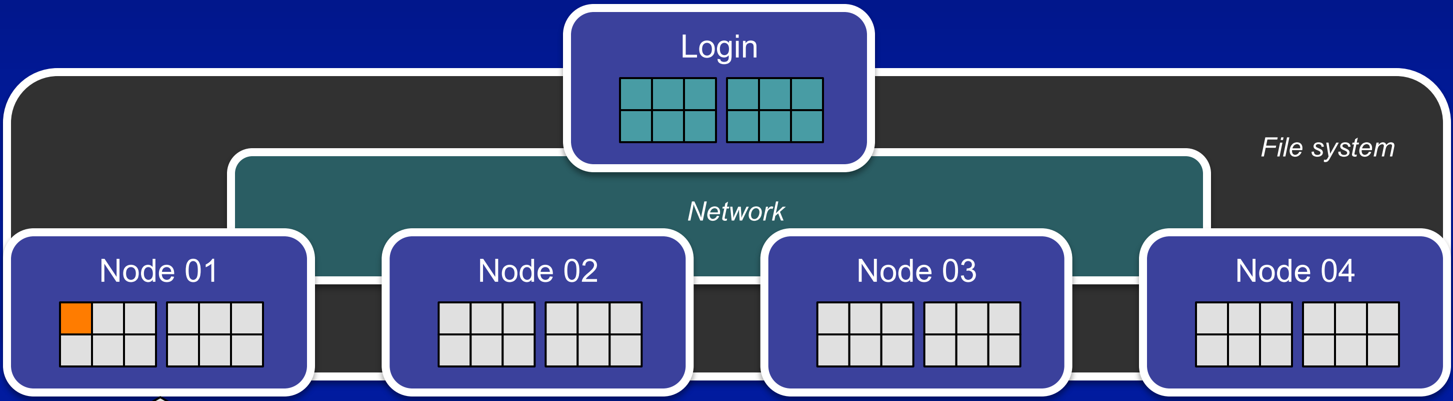


4 nodes; 48 cores; Global file system; Connected by network; Controlled by Login node

TACC

Serial Tasks

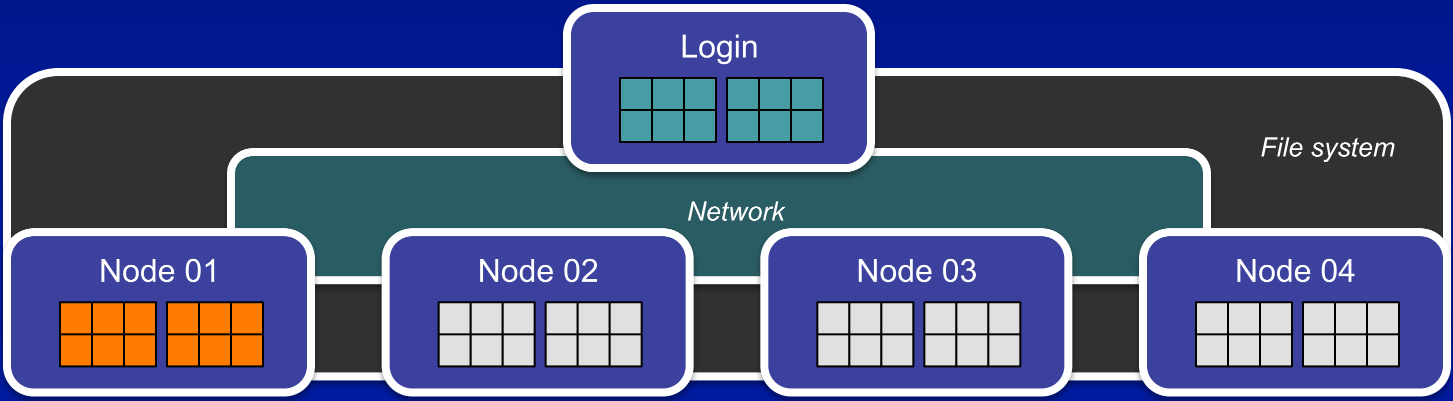
SAMtools, Perl or Python scripts, R scripts,
Other non-parallel code…



Other 11 CPUs sit idle
Task takes longer
You are charged for all
12 CPUs

On-node Parallelism

BWA, Bowtie, Velvet, etc. use pthreads or OpenMP to implement ‘threads”



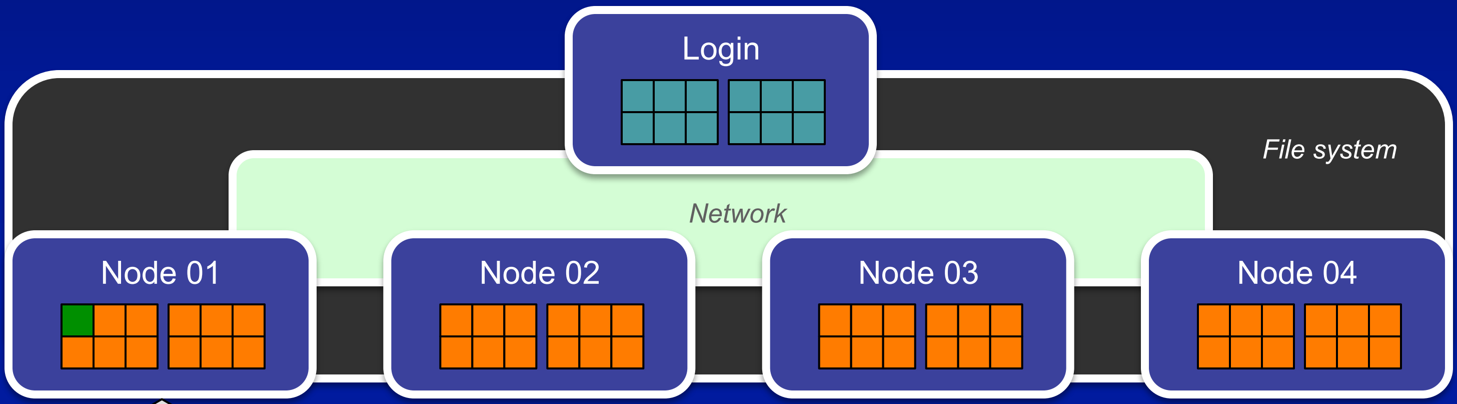
Makes use of all CPUs
Your task can run up to 12x faster*

Caveats

1. Do you have enough RAM to run all 12 threads?
2. Is the code actually more efficient with additional ‘threads’?
3. What happens if you need more than 12 CPUs?

MPI Parallel Tasks

ABySS, MAKER, RaXML, etc. use MPI
(Message Passing Interface)



Login

File system

Network

Node 01

Node 02

Node 03

Node 04

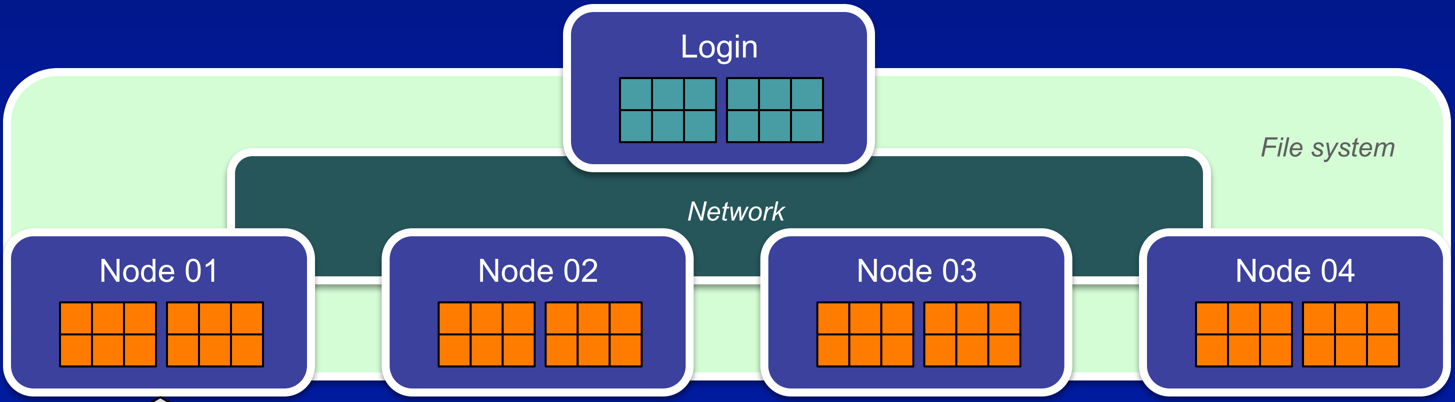
You can make use of all available CPUs
Tasks can run up to 48x faster!

Caveats

1. Really good networking is needed because tasks communicate with one another
2. Algorithm must be written to explicitly make use of MPI. Most bioinformatics code is NOT ☹

Simple Parallel Tasks

Any code where you can arbitrarily split the work across workers can be adapted to the ***TACC Parametric Launcher***

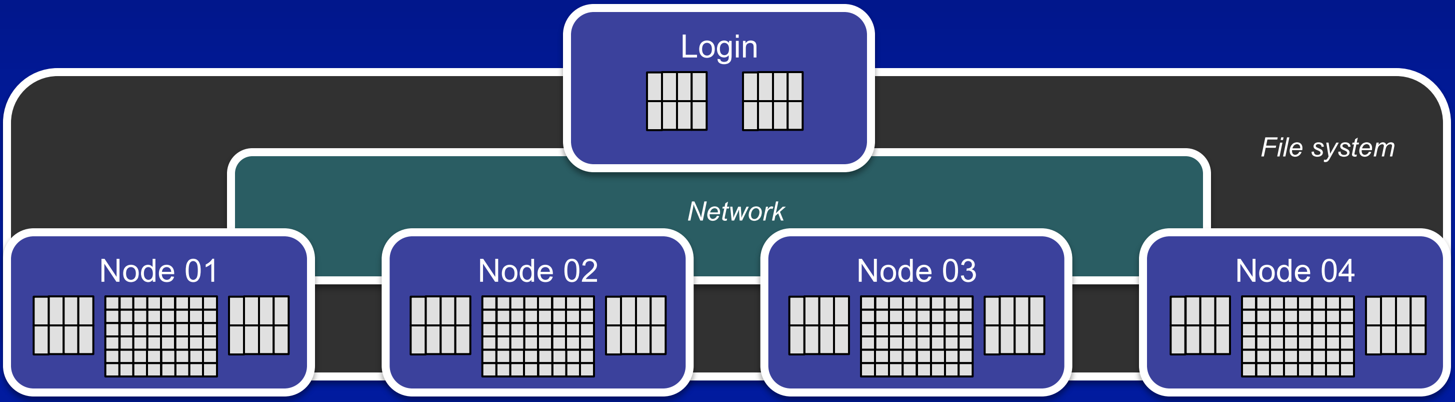


You can make use of all available CPUs Tasks can run up to 48x faster!

Caveats

1. Shared file system is needed for this approach
2. You have to figure out how to shoe-horn your algorithm into using the Launcher. Luckily, we have some recipes to share!

Why Does Parallelism Matter?



TACC Stampede, Jan 2013
(Not to scale)
Notice anything… different?