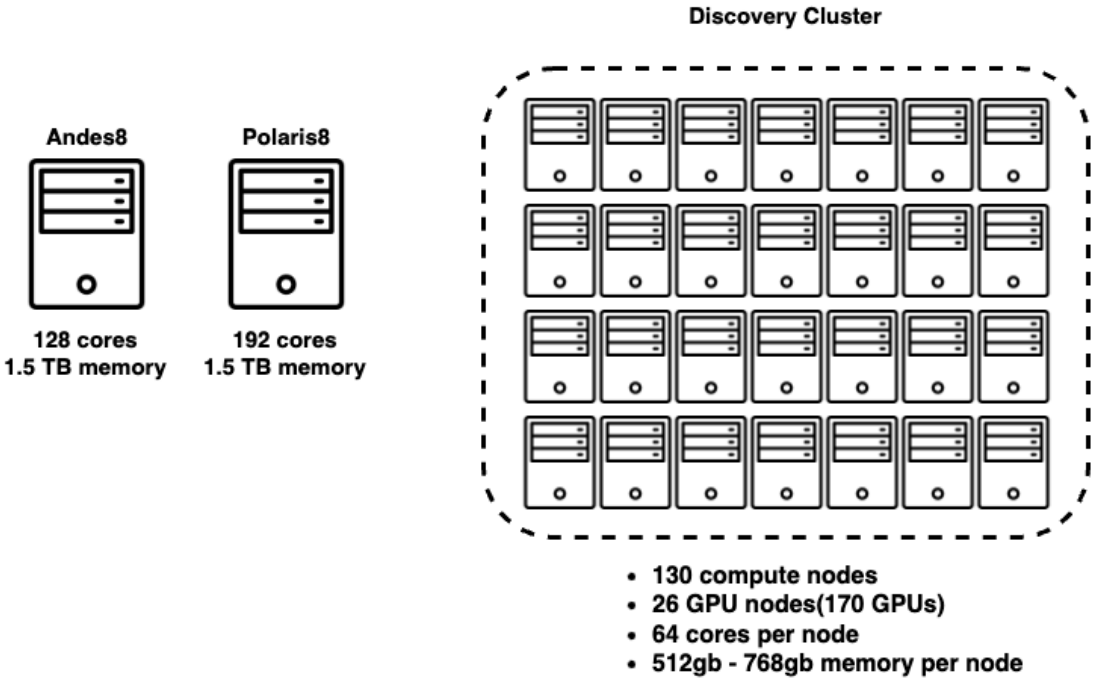
What is HPC?

What is HPC?

> High Performance Computing (HPC) generally refers to the practice of aggregating computing power in a way that delivers much higher performance than what a typical desktop computer or workstation can offer. This enhanced capability is used to solve large problems in science, engineering, or business.

At Dartmouth, we have access to both large-scale shared memory platforms (Andes and Polaris) and a cluster (Discovery) with many nodes, or a set of individual machines connected through a high-bandwidth network.



Remember that the HPC resources at Dartmouth are shared resources across the entire research community. Being conscious of this aspect is especially important when on Andes, Polaris, or on the login node, Discovery.

Andes and Polaris are designed for interactive use, similar to how one typically uses a computer. Being a responsible user on these systems involves carefully monitoring your processes to ensure you're not consuming an excessive amount of CPU resources or memory.

On the other hand, Discovery is primarily utilized for batch-scheduled jobs, which means submitting jobs to a scheduler. When you log into Discovery, you're on the login node. This node serves as a place for users to submit and monitor their jobs. It's suitable for minor tasks like compiling code or monitoring jobs, but it's not intended for testing or running user programs. The login node is not designed for computationally intensive tasks and can become overwhelmed if used for such purposes. This can lead to system-wide issues, such as other users being unable to submit jobs to the cluster.

Why do we use HPC?

> An important concept to understand in HPC is the difference between a shared memory model and a distributed memory model. Standard systems like laptops and desktops utilize a shared memory model, where all components of a single system share the memory. In contrast, HPC often employs a distributed memory model, where the computational load is distributed across multiple compute nodes, each using its shared memory.