What is Tasking?
- Define units of serial work (tasks) of an application and dependencies among the tasks
- Problem-centric decomposition vs. data-centric (SPMD)

On-Node
- Tasks do not migrate across nodes
- Tasks coordinate with MPI to communicate across nodes
- Further decompose the domain of an MPI rank and assign tasks to the second-level decomposition

Why use tasks?
- Separates exposing parallelism (programmer) and mapping parallelism onto the architecture (runtime)
- Transparently adjust to the level of concurrency on the node, including heterogeneous accelerators
- Load balancing is natural by migrating tasks

Primary Challenges
- How to express and submit tasks?
  - fork/join
  - Dynamic Task Graph
- How to encode dependencies?
  - Dataflow: inferred from task access mode to data
  - Control flow: barriers, sync, futures
- How to schedule tasks onto the hardware efficiently?

Modern Runtimes Example

MiniFutures: A new task-based mini-application
- Proxy for unstructured finite element method
  - Discretization, element analysis, assembly
  - Pre-conditioned conjugate gradient solver
  - Gauss-Seidel preconditioner
- Runtime agnostic
  - Goal is to evaluate modern runtimes
  - Study how to design future task-based applications