PaRSEC is a generic framework for architecture aware scheduling and management of micro-tasks on distributed many-core heterogeneous architectures. Applications we consider can be expressed as a Direct Acyclic Graph of tasks with labeled edges designating data dependencies. DAGs are represented in a compact problem-size independent format that can be queried on-demand to discover data dependencies in a totally distributed fashion. PaRSEC assigns computation threads to the cores, overlaps communications and computations and uses a dynamic, fully-distributed scheduler based on architectural features such as NUMA nodes and algorithmic features such as data reuse.

PaRSEC can accept input in multiple forms that enable task based execution:
- Serial programs can use the prototyping Insert Task interface to submit tasks.
- Developers can describe the Dataflow representation of a program as a Parameterized Task Graph (PTG).
- Serial programs with Affine loops can be automatically converted to a PTG.
- Third party, user defined DSLs can be compiled into DAGs of tasks.

This information along with the Data distribution, the Application code & Codeletes, the Runtime and relevant libraries are linked by the system compiler to generate the executable that will run on a heterogeneous distributed memory supercomputer.

PaRSEC expresses the Directed Acyclic Graph (DAG) of the Dataflow of a program using the Parameterized Task Graph (PTG) – a symbolic, problem size independent representation. As a result, at runtime, successors and predecessors of any given task can be evaluated independently, without exploring portions of the DAG pertaining to tasks localized on other nodes. Furthermore, the whole DAG is never unfolded, and only the set of locally active tasks resides in the memory at any given time.