We consider a parallel computational model that consists of $P$ processors, each with a fast local ephemeral memory of limited size, and sharing a large persistent memory. The model allows for each processor to fault with bounded probability, and possibly restart. On faulting all processor state and local ephemeral memory is lost, but the persistent memory remains. This model is motivated by upcoming non-volatile memories that are as fast as existing random access memory, are accessible at the granularity of cache lines, and have the capability of surviving power outages. It is further motivated by the observation that in large parallel systems, failure of processors and their caches is not unusual.

Within the model we develop a framework for developing locality efficient parallel algorithms that are resilient to failures. There are several challenges, including the need to recover from failures, the desire to do this in an asynchronous setting (i.e., not blocking other processors when one fails), and the need for synchronization primitives that are robust to failures. We describe approaches to solve these challenges based on breaking computations into what we call capsules, which have certain properties, and developing a work stealing scheduler that functions properly within the context of failures. The scheduler guarantees a time bound of $O(W/P_a + D(P/P_a)\cdot \text{ceil}(\log 1/f \cdot W))$ in expectation, where $W$ and $D$ are the work and depth of the computation (in the absence of failures), $P_a$ is the average number of processors available during the computation, and $f \leq 1/2$ is the probability that a capsule fails. Within the model and using the proposed methods, we develop efficient algorithms for parallel sorting and other primitives.