When using in-memory caching, we started to notice occasional non-deterministic fluctuations of the Map function execution times in some of the tasks. These slow tasks, even though few, affect the performance of the computation significantly because the execution time of a whole iteration is of course dependent on the slowest task of the iteration. Even though Twister4Azure supports duplicate execution of the slow tasks, duplicate tasks for non-initial iterations are often more costlier than the total execution time of a slow task that uses data from a cache, as the duplicate task would have to fetch the data from the Azure Blob Storage. We were able to narrow down the cause of the fluctuation anomaly to the use of large amount of memory, including the in-memory data cache, within a single .NET process. We switched to use of memory-mapped files, which can be shared across multiple processes and can be used to facilitate inter-process communication.

A key idea of our research is a Map-Collective model supported by Twister4Azure where each architecture is supported by an appropriate implementation of each collective operation. Figure 12 shows the results for MDS using an AllGather iterative MapReduce Collective primitive similar to the MPI AllGather communication primitive. The AllGather primitive broadcasts the Map Task outputs to all the computational nodes, and assembles them together in the recipient nodes and schedules the next iteration or the application. Usage of the AllGather primitive in MDS BCCalc computation eliminates the need for reduce, merge and the broadcasting steps in that particular computatation. In addition to improving the performance, this primitive also improves the usability of the system as it eliminates the overhead of implementing reduce and/or merge functions. Communication primitives also allow us to optimize the given operation transparently to the users. Figure 12 presents an execution trace of a computation that utilized both memory mapped files and the AllGather primitive.



*Figure 12: Execution traces of MDS iterative MapReduce computations using Twister4Azure using AllGather primitive together with memory mapped file caching. The left column shows the execution time of tasks in each iteration, which consist of two distinct steps. The right column shows the number of active map tasks of the computation at a given time with one second intervals. The gaps between the computations represent the overhead of task scheduling, reduce task execution, merge task execution and data broadcasting.*