MPI-3 Helper Threads

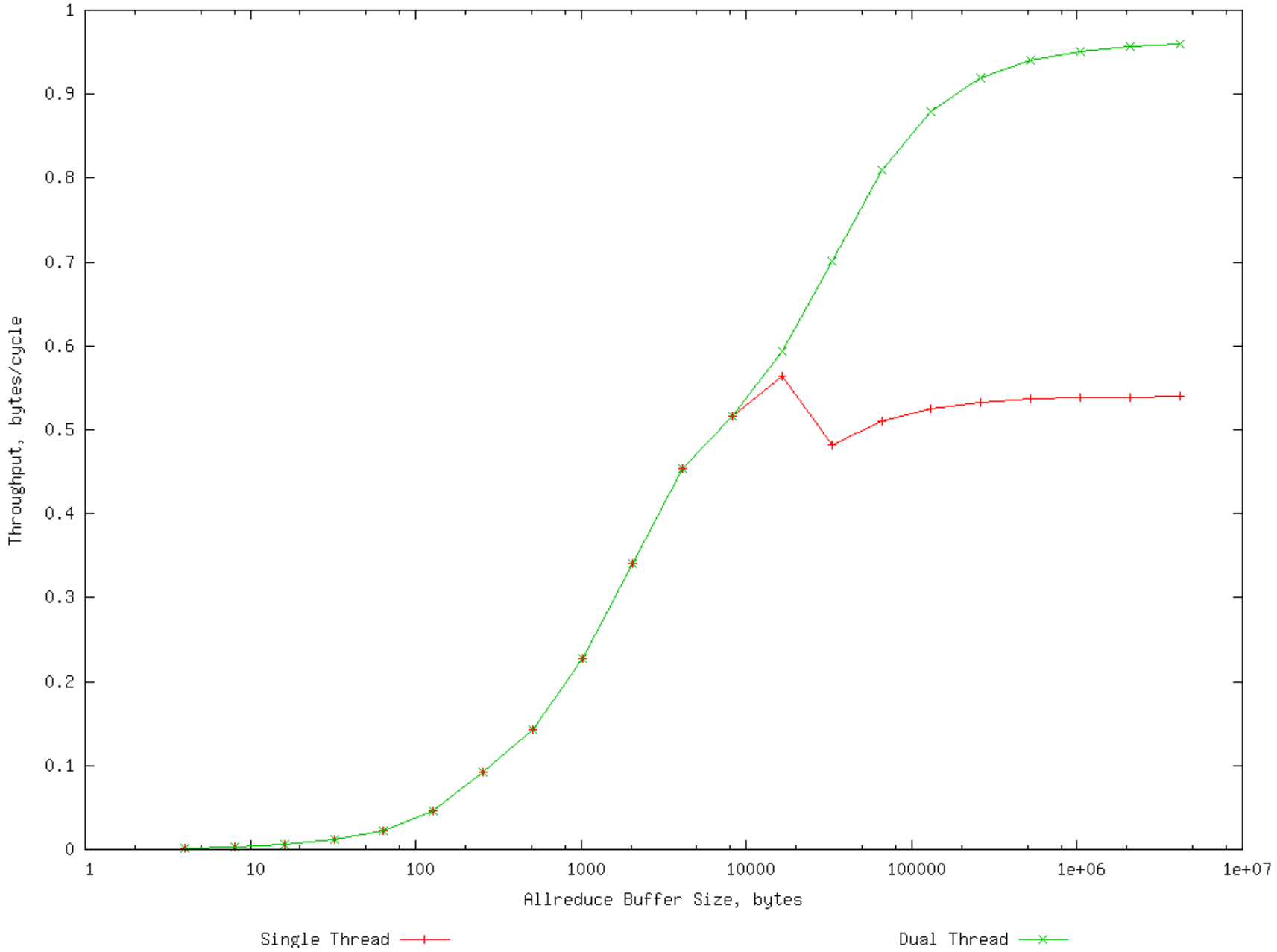
Douglas Miller, IBM

dougmill@us.ibm.com

MPI+OpenMP programming

- Many MPI+OpenMP applications use OpenMP parallel regions for computation, but call MPI in FUNNELED/SERIALIZED mode
  - In such cases the remaining cores that are not making MPI calls are possibly idle (except when there is no strong synchronization between threads)
- In many cases, being able to use multiple threads within the MPI stack is helpful
  - To drive the network at full performance
  - To perform internal MPI processing such as datatype processing
- But the MPI implementation cannot spawn its own threads
  - Difficult to identify whether the application threads are “active” or not

Collective Communication Performance on BG/P



Proposed Solution: Helper Threads

- Idea is that the application is allowed to “hand over” its threads to the MPI stack when its not using them

```
#pragma omp parallel num_threads(N) {
      /* ... other thread setup */
      t = omp_get_thread_num();
      MPI_Helper_team team;
      MPI_Helper_team_create(0, omp_get_num_threads(), &team);
      /*
       * some computation may occur here...
       *
       * then a communications phase begins:
       */
      MPI_Helper_join(team);

      if (t == 0) {
            MPI_Allreduce(...);
      }
      MPI_Helper_leave();
      /*
       * more computation and/or communication
       */
      MPI_Helper_team_destroy(&team);
      /* ... other thread tear-down */
}
```

MPI stack's perspective

- MPI_Helper_join() can possibly wait for all threads to join in
- At MPI_Helper_leave(), the MPI stack can block all threads waiting for work
  - When there is no more work, all threads exit