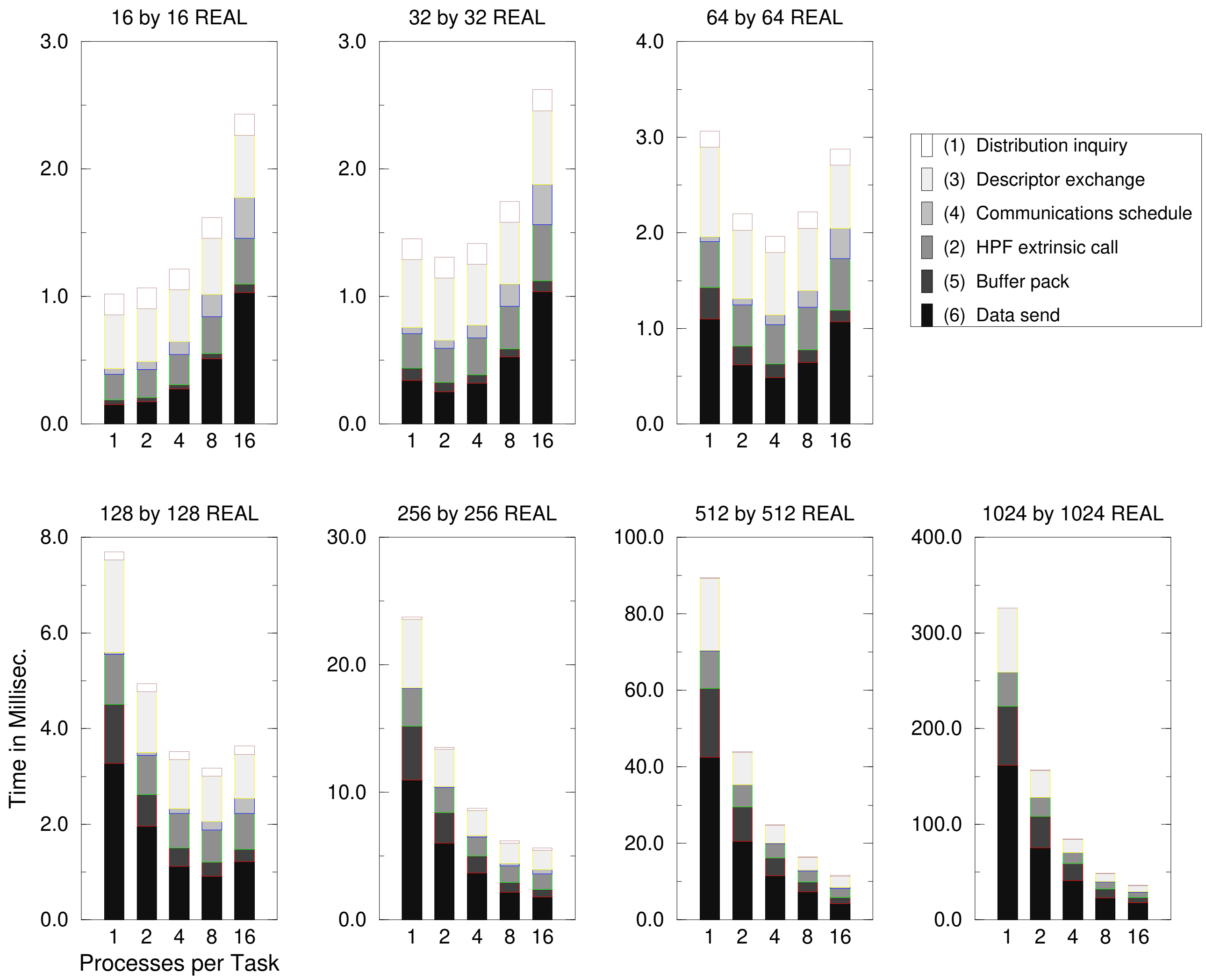

Figure 1: EPS figure for ping-pong times.

1 Introduction

Figure 1 shows the EPS figure for our ping-pong bar chart, generated using **xmgr**. This plotting program is the Motif version of *xvgr*, and is the only version supported now by the author, so it is at a later version. **xmgr** does a better job of shading stacked barcharts, generates EPS, is supported here within MCS, and has documentation on the Web, so I'm doing all the SC96 paper's plots using it. The conversion of the ping-pong plot from *xvgr* to **xmgr** was painless as far as differences in functionality are concerned, but I still ended up spending a lot of time reformatting some things by hand in a trial-and-error process, because the X display doesn't match the PS output precisely, and because the bounding box in the EPS output appears to include an undesirable border.

High Performance Fortran (HPF) provides a portable, high-level notation for data-parallel algorithms. An HPF computation has a single-threaded control structure, global name space, and loosely synchronous parallel execution. Many problems requiring high performance imple-

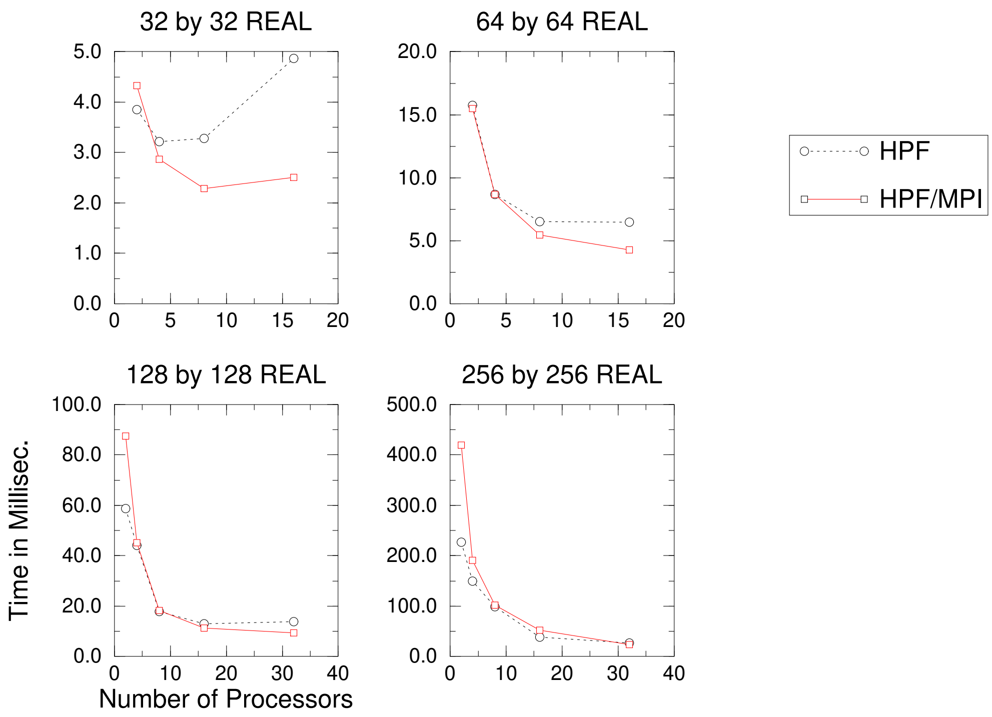

Figure 2: Execution times for 2D FFT application using both pure HPF and HPF/MPI, for varying input/operator array sizes.

mentations can be expressed succinctly in HPF.

This is an extraneous reference to the 2D FFT plot, which is Figure 2.

In brief, the contributions of this paper are as follows:

1. The definition of a novel parallel programming model in which group-oriented communication libraries are used to coordinate the execution of process groups corresponding to data-parallel tasks.

2. A demonstration that an HPF/MPI instantiation of this model can be defined and constructed, and that this instantiation allows the range of problems efficiently expressible in HPF to be extended considerably without excessive conceptual or implementation complexity. This is an extraneous reference to the 2D Convolution plot, which is Figure 3.

3. The definition, demonstration, and evaluation of novel techniques for implementing the communication structures incorporated in our HPF/MPI library, for integrating MPI library calls into HPF programs, and for exploiting information provided by MPI communication calls to improve communication performance.

Given the apparent deficiencies of both compiler-based and language-based approaches, it is interesting to consider alternative, simpler instantiations of the mixed task/data-parallel model. We propose an approach based on task-parallel coordination libraries that support communication and synchronization among data-parallel tasks. In this approach, multiple data-parallel tasks are created, either externally to the data-parallel language or internally using a mechanism akin to task regions. Each task executes a data-parallel program. Communication operations among two or more tasks are implemented as collective communication operations

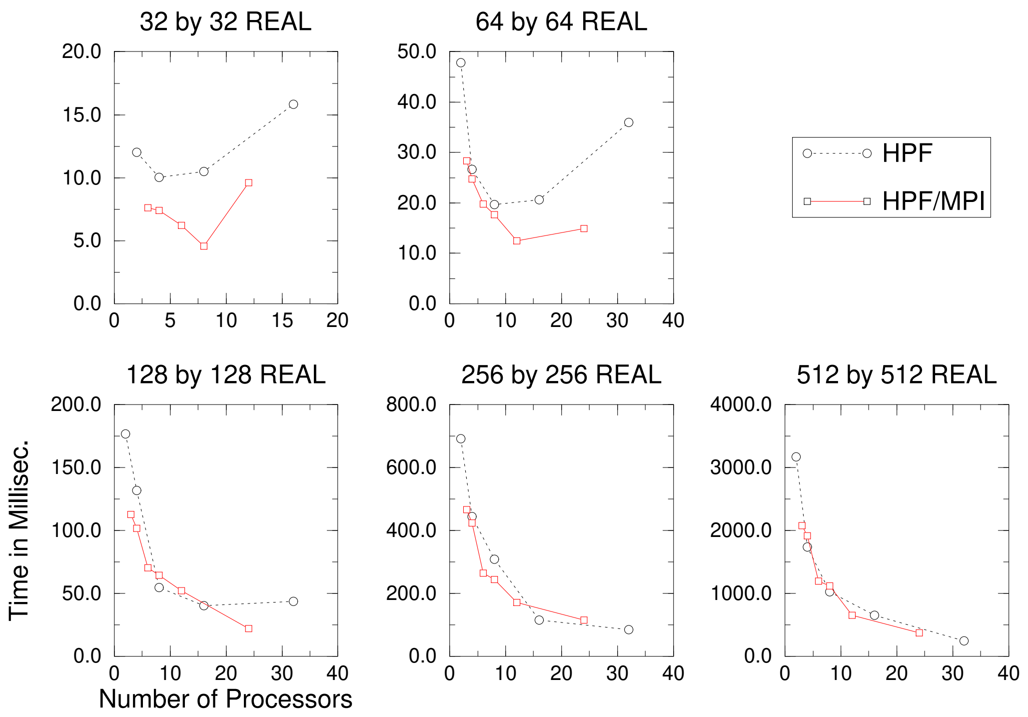

Figure 3: Execution times for 2D Convolution application using both pure HPF and HPF/MPI, for varying input/operator array sizes.

between the sets of processes constituting the tasks involved (see Figure **??**). These collective communication operations can be performed by libraries designed to be called from data-parallel programs via a mechanism such as the HPF extrinsic procedure. This yields a group-oriented communication model with some similarities to other systems supporting process groups, but distinguished by a focus on collective communications between groups. This is an extraneous reference to the Multiblock plot, which is Figure 4.

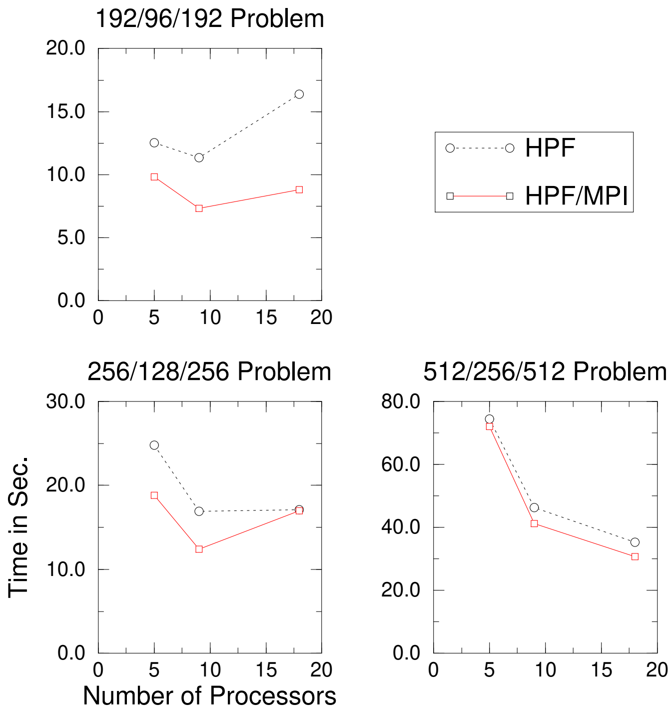

Figure 4: Execution times for Multiblock application using both pure HPF and HPF/MPI, for varying sizes of a simple 3-block geometry.