Transpositional Communication Group

Problem

Parallelization is required on a mesh or butterfly data layout where row-wise decomposition fits naturally but column-wise communication is needed in the subsequent phases. [define what it means to be column elements. (1,4,8,12…)] The algorithm requires computation with elements regarding in the same row for a while, and then move on to the elements in the columns which demands use of communication. There is certain benefit obtained by working with row elements

1) locality
2) communicator definition.
3) underlying network topology.

…

than having column communications.

Driving Forces

1. Row computation must be exploited when possible.
2. Must keep the overhead of the communication within the original overhead.

Solution

One could approach this problem with rectangular mesh or appropriate butterfly structure. Large communication overhead that comes from the column communication will results in slower overall performance. Instead of fitting the communication to the data layout we change the data layout for elimination of future column communications. Namely, we can transpose the data topology so that the column elements become the present row elements. Work with the row elements, and upon need, transpose the layout again in the subsequent phases. Note that we still pay the price of communicating in the transposition. But the benefit obtained from processing by the row will outweigh the transposition cost (in fact it must).

Difficulty

Comparing the benefit raised by the row computation with the transposition overhead.

Related Pattern

Mesh, Butterfly

Example

Transpositional FFT, Shear Sort