Asynchronous PDE Solver for Computing at Extreme Scales

Konduri Aditya
Department of Aerospace Engineering
Texas A&M University
College Station, Texas 77843
Email: akonduri@tamu.edu

Diego A. Donzis
Department of Aerospace Engineering
Texas A&M University
College Station, Texas 77843
Email: donzis@tamu.edu

Torsten Hoefler
Department of Computer Science
ETH Zurich
8092 Zurich, Switzerland
Email: htor@inf.ethz.ch

***Abstract*—Scalability of codes at extreme scales is a critical issue of current relevance. At extreme levels of parallelisms, communication between processing elements (PEs) could represent a substantial running time, resulting in substantial waste in computing cycles. We investigate a novel approach based on common finite-difference schemes for partial differential equations (PDEs) in which computations are done locally with values already available at each PE, without waiting for updated data from neighboring PEs. No synchronization among cores is enforced and computations proceed regardless of messages status. This drastically reduces processor idle times, improving computation rates and scalability. We show that accuracy of common numerical schemes is reduced when asynchronicity is present. We derive new schemes that can maintain the accuracy under asynchronous conditions. These new schemes are implemented and tested. Performance is evaluated through statistics of different measures of asynchronicity. A new implementation of RMA communications is shown to provide significant advantages.**

I. Introduction

Many natural phenomena and engineering systems can be described with partial differential equations (PDEs). These equations are often nonlinear for which analytical solutions are typically not known. Hence, they are solved using numerical simulations. A considerable number of problems are complex in nature and demand massive computations with extreme levels of parallelism. With current petascale systems, simulations are routinely being done on hundreds of thousands of processing elements (PEs) and the trend has been to increase the degree of parallelism with advances in parallel architecture and software. Performance studies of these simulations show that data synchronization across PEs can take up significant amount of the total computing time, resulting in waste of compute cycles and severely affecting the scalability of the codes at large scales. With computation rates of PEs being much faster than the communication of data over the network, synchronized communication is likely to be the major bottleneck in future exascale. This calls for a need to relax synchronization of data not just in communication algorithms, but also at a mathematical level.

An extensive work on asynchronous iterative methods for solving linear systems is present in the literature [1]-[2]. Unfortunately, these methods cannot relax the data synchronization involved at every time step in solving PDEs. There have been efforts [3] to develop asynchronous solvers, but were restricted to only parabolic PDEs.

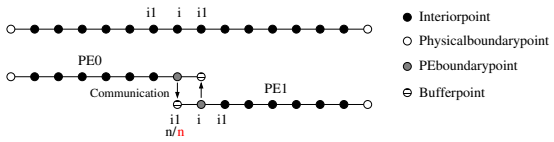

Fig. 1. 1D computational domain

In this work, we present a novel asynchronous computing strategy based on finite difference schemes that relaxes data synchronization and significantly improve the scalability of PDE solvers at large scales.

II. Concept

To illustrate the concept let us consider the problem of 1D heat conduction. This phenomena is governed by

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}. \tag{1}$$

A simple finite difference scheme that uses first order forward difference in time and second order central difference in space gives the following algebraic equation.

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \alpha \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2} + O(\Delta t, \Delta x^2) \tag{2}$$

For a set of initial and boundary conditions, this equation is computed at all the points in the discrete 1D domain (see[1] Fig. 1) to obtain time evolved solution. When the simulation is carried out on a parallel architecture with distributed memory, the computational domain is divided among the available PEs. The computation of spatial derivatives at the PE boundary points (see Fig. 1) need values from neighboring PEs, which are communicated over the network into the buffer points. Therefore, PEs are typically forced to wait for the communication of data across the network to finalize at every timestep.

We relax this waiting time by not enforcing the synchronization of data at the buffer points among PEs and letting the computations proceed regardless of the status of communication messages. This means that the solution at point i in Fig. 1 will be computed to the equation,

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} \approx \alpha \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^{n^*}}{\Delta x^2}, \tag{3}$$

where, depending on the communication time, the value of n^* can be n, $(n-1)$, $(n-2)$, etc. The effect of using $n^* < n$

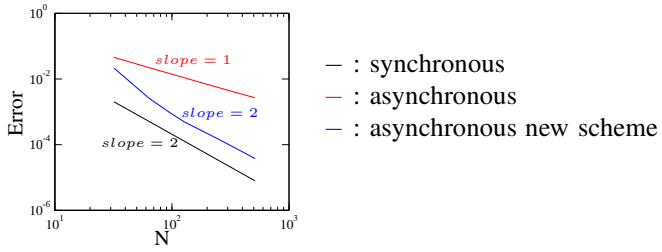

Fig. 2. Plot of accuracy of numerical scheme

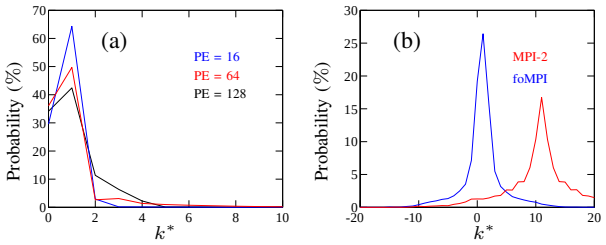

Fig. 3. Probability distribution of k^*

on the properties of the numerical scheme, such as stability, consistency and accuracy, is studied in [4]-[5], where it is shown that accuracy will drop to first order when $n^* < n$.

Analyzing the truncation error of these schemes with $n^* < n$, it is possible to derive families of new difference approximations with any accuracy. For example, the following expression for second derivative will maintain second order accuracy even with a delay in communication.

$$\begin{aligned}\frac{\partial^2 u}{\partial x^2} &= \frac{1}{2\Delta x^2}[u^n_{i+3} - 2u^n_{i+2} + u^n_{i+1} \\ &+(k^*+1)(u^{n-k^*}_{i-1} - 2u^{n-k^*}_{i-2} + u^{n-k^*}_{i-3}) \\ &-k^*(u^{n-k^*-1}_{i-1} - 2u^{n-k^*-1}_{i-2} + u^{n-k^*-1}_{i-3})] + O(\Delta x^2)\end{aligned}$$

Results on accuracy from simulations are shown in Fig. 2.

III. Computing Algorithm

The relaxation of synchronization at a mathematical level essentially results in a computation-communication overlap at all times. This requires the use of non-blocking communication to move data across the network. With the current Message Passing Interface (MPI) standard, this can be achieved from two-sided non-blocking Send/Recv or one-sided remote memory access (RMA) methods. A schematic of these communication strategies is shown in the poster. As it not necessary to synchronize the data, participation of both source and target PEs is not necessary, which allows us to use one-sided communications because only the source PE carries out the communication (using MPI_Put). This avoids expensive overheads typical of two-sided communications.

In order to maintain accuracy under asynchronous computing, the numerical schemes need stencil values at multiple time level in the buffer. The schematic in the poster also provide these implementation details. An outline of the requirements and the algorithm is given in *Algorithm* section of the poster.

IV. Results

Simulations of a convection-diffusion (Burgers') equation are used to study the performance of the new asynchronous schemes. An important parameter in this regard is the distribution of k^* which determines the absolute error in the solution due to time asynchronicity of some values in the domain [5]. Fig. 3a shows that with an increase in the number of PEs, the most probable value of k^* decreases and the distribution becomes wider. This can be attributed to the increase in the number of messages at every timestep, resulting in communication delays. In Fig. 3b, we compare the probability distribution of k^* with MPI-2 and foMPI (fast one-sided MPI, [6]) implementations. Clearly, foMPI provides a much lower value of the mean of k^* (referred as k^*_{mean} in the poster) that results in better parallel performance and accuracy of the solution. This is seen in the strong-scaling plot in Fig. 4, where we show time per step versus PE count. Clearly, the scaling of asynchronous schemes is nearly ideal, well beyond the range in which synchronous schemes scale reasonably well.

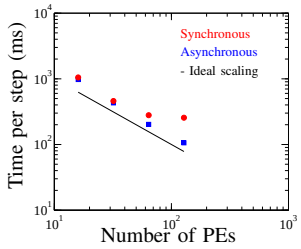

Fig. 4. Graph of strong scaling on hopper

V. Conclusion

We have presented an asynchronous computing method that has the potential to eliminate virtually all communication overheads and improve scalability of PDE solvers at large scale. As the accuracy of regular finite difference schemes drops, we have derived new schemes that are robust to asynchronous computing. Preliminary results suggest that the method has potential to provide extreme scalability at future exascale.

Acknowledgment: We are grateful to NERSC for computing time on Hopper and support.

References

[1] D. P. Bertsekas and J. N. Tsitsiklis, "Parallel and distributed computation," 1989.

[2] A. Frommer and D. B. Szyld, "On asynchronous iterations," *Journal of computational and applied mathematics*, vol. 123, no. 1, pp. 201–216, 2000.

[3] D. Amitai, A. Averbuch, M. Israeli, and S. Itzikowitz, "On parallel asynchronous high-order solutions of parabolic pdes," *Numerical Algorithms*, vol. 12, no. 1, pp. 159–192, 1996.

[4] A. Konduri and D. A. Donzis, "Asynchronous computing for partial differential equations at extreme scales," in *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:*, pp. 1443–1443, IEEE, 2012.

[5] K. Aditya and D. A. Donzis, "Asynchronous numerical schemes for partial differential equations," *Manuscript under preparation*.

[6] R. Gerstenberger, M. Besta, and T. Hoefler, "Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One Sided," in *IEEE/ACM International Conference on High Performance Computing, Networking, Storage and Analysis (SC13)*, p. Accepted, 2013.